Improving Continuous Japanese Fingerspelling Recognition with Transformers: A Comparative Study against CNN-LSTM Hybrids

Akihisa Shitara†*, Yuhki Shiraishi*
†Graduate School of Library, Information, and Media Studies, University of Tsukuba, Japan
Email: theta-akihisa@digitalnature.slis.tsukuba.ac.jp
*Faculty of Industrial Technology, Tsukuba University of Technology, Japan
Email: yuhkis@a.tsukuba-tech.ac.jp

***Abstract*—To achieve smooth communication between d/Deaf and hard of hearing (d/DHH) and hearing people, we have developed a continuous Japanese Fingerspelling (JF) recognition system using sensor gloves and deep learning. We have selected a light and inexpensive sensor glove adapted for the system's daily use. In our prior system using a machine learning model that combines Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM), despite achieving an average micro F-measure of 92.1% across the 76 JF characters, we reported the average macro F-measure of only 64.7%. Two problems cause this issue: distinguishing between static and dynamic fingerspellings, and the decreased recognition rate due to the large number of instances "ϕ" (the transition movements characters). Therefore, we conducted a quantitative evaluation using the CNN-LSTM combined machine learning model as a baseline to verify whether the Transformer Encoder could improve JF recognition rates. Consequently, for the 76 JF characters, the average micro and macro F-measures were 93.8% (0.2) and 77.4% (1.0), respectively.**

Keywords–Deaf and hard of hearing; Sign language; Sensor glove; Recognition.

I. INTRODUCTION

To achieve smooth communication between d/Deaf and Hard of Hearing (d/DHH) and hearing people, we have developed a continuous Japanese Fingerspelling (JF) recognition system using sensor gloves and deep learning [1] [2]. However, a machine learning model that combines Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) [2] despite achieving the average micro F-measure of 76 JF characters was 92.1%, we reported the average macro F-measure of only 64.7% (Figure 1). Two problems cause this issue: distinguishing between static and dynamic fingerspellings, and the decreased recognition rate due to the large number of instances "ϕ" (the transition movements characters). However, in the current research community on sign language recognition, a machine learning model based Transformer [3] is often used. Thus, we conduct a comparative analysis by a quantitative evaluation using the CNN-LSTM combined machine learning model as a baseline to verify whether the Transformer Encoder could improve JF recognition rates. Additionally, in the current research community on fingerspelling recognition and sign language recognition, the small size of the dataset is listed as an issue. To the best of our knowledge, no reports have verified whether there are individual differences among signers expressing continuous fingerspellings. Thus, we also compare and analyze the impact of individual differences among signer data by conducting cross-validation evaluations selecting training data.

The remainder of this study is organized as follows. In Section II, we describe related studies on fingerspelling recognition and sign language recognition. In Section III, we describe a comparative analysis by a quantitative evaluation using the CNN-LSTM combined machine learning model as a baseline. In Section IV, we describe a comparative analysis learning model as CNN-Transformer Encoder as a baseline. In Section V, we discuss the results and the limitations of the work. Finally, in Section VI, we provide some concluding remarks and suggest some avenues for future research.

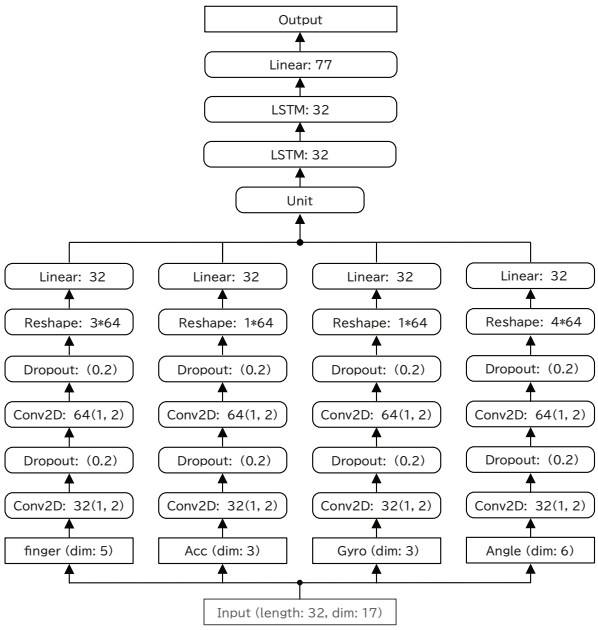

Figure 1. Architecture of the CNN-LSTM combined machine learning model.

II. RELATED WORK

Previous research on fingerspelling and sign language recognition has proposed two types of sensors for recognizing a series of operations in fingerspelling and sign language: contact-type sensor gloves and non-contact-type cameras.

A. Image recognition

Several methods have been proposed for recognizing hand shapes based on processing images of fingerspelling as captured by cameras.

As example of a fingerspelling recognition method, Mukai et al.'s method [4] used a classification tree and machine learning based on a support vector machine to classify individual

images; it targeted 41 immobile characters in Japanese Sign Language (JSL) resulted in an average recognition accuracy of 86%. Hosoe et al.'s method [5] used deep learning for recognition and achieved a recognition rate of 93%, but only for static fingerspellings. Jalal et al.'s method [6] achieved a recognition rate of 99% for American Sign Language (ASL) images based on a deep learning algorithm for static fingerspellings (i.e., excluding "J" and "Z").

However, the recognition accuracy could not be considered as sufficient for practical recognition in JF. Additionally, relatively few recognition results have been reported for dynamic fingerspellings (i.e., fingers moving when expressing a character). For example, in Kondo et al.'s study [7] of dynamic fingerspellings in JSL, the identification of hand shapes was performed using a kernel orthogonal mutual subspace method from images of hand regions obtained from distance images, and the classification of movements was performed using decision trees based on center-of-gravity coordinates. These results yielded a 93.8% identification rate. However, the recognition accuracy was insufficient for the practical recognition required for JF.

Furthermore, examples of machine learning models based on Transformers for sign language recognition include the machine learning model SignAttention [8] targeted Greek Sign Language, the machine learning model [9] using the American Sign Language dataset How2Sign [10], the machine learning model [11] using the German Sign Language dataset RWTH-PHOENIX-Weather [12] [13] and other examples such as the machine learning model [11]. Moreover, as a result survey a related research on sign language recognition [14], it is reported that CNN, LSTM, and Transformer are used in many research, and as reported in the research that surveyed the State-Of-The-Art (SOTA) in sign language recognition [15], it is reported that Transformer is used in many cases.

On the other hand, there is also research [16] using combination spatial-temporal modules and Multi-Layer Perceptron (MLP), and Takayama et al.'s model [17] using combination Spatial Temporal Graph Convolutional Networks [18] and Transformer and targeted JSL. Additionally, as examples use Conformer [19], Kimura et al.'s machine learning model [20] targeted JSL. Signformer [21] also used a module that has been redesigned based on the Conformer architecture.

B. Sensor glove recognition

Several methods have been proposed for recognizing hand shapes based on measurement data acquired by contact-type sensor gloves. These methods can measure data, which includes the flexion of the five fingers, the position and direction of the hand. The measurement data are then sent to a personal computer or microcomputer, and a classification algorithm is used to recognize hand shapes.

As example of a fingerspelling recognition method, Cabrera et al. [22] used the Data Glove 5 Ultra [23] sensor glove with an acceleration sensor to acquire information regarding the degree of flexion of each finger and wrist direction. The study targeted 24 static fingerspelling characters in ASL, excluding "J" and "Z", and achieved a recognition rate of 94.07%. Mummadi et al. [24] prototyped a sensor glove with multiple embedded inertial sensors. The study targeted fingerspellings of French Sign Language, and achieved an average recognition rate of 92% with an F1-score of 91%. Kakoty et al.'s model [25] used kernel-supported vector machine, and targeted a dataset of one-handed Indian sign language alphabets (C, I, J, L, O, U, Y, W), ASL alphabets (A to Z), and signed numbers (0 to 9). The study achieved an average recognition rate of 96.7%. SpellRing [26] used combined active acoustic sensing and Inertial Measurement Unit (IMU) in the ring-shaped devices worn on the thumb. ResNet-18 [27] uses CNN as the backbone and also leverages Connectionist Temporal Classification (CTC) [28].

Furthermore, as examples of a sign language recognition method, Chong et al. [29] placed six IMUs on the back of the palm and on each fingertip to capture their motion and orientations. The study targeted 28 proposed word-based sentences in ASL, and used LSTM. The method achieved an accuracy of up to 99.89%. SmartASL [30] uses IMUs installed in two devices, an earphone and a smartwatch, to include not only manual marker expressions but also Non-manual Marker (NM) expressions. The method used LSTM for data related to hand movements and CNN and LSTM for data related to NM expressions. After that, the Transformer model T5 [31] is used for fine-tuning together with translated English sentences. SignRing [32] uses the IMUs in the ring-shaped devices worn on the index fingers of both hands. It generates data similar to that from the IMU sensors from the sign language videos in the ASLLRP [33] ASL dataset. Then, it uses a model combining CNN and LSTM to train the generated data.

III. First Comparative Analysis

To verify whether the JF recognition rate improved when replacing LSTM with a Transformer Encoder, and to compare the impact of the differences between machine learning model architectures, we trained each model and compared their accuracies. We have the following machine learning models:

- 2LSTM
- branch-2CNN-unit-2LSTM
- Transformer Encoder
- branch-CNN-unit-Transformer Encoder
- branch-2CNN-unit-Transformer Encoder

A. Continuous Japanese fingerspelling dataset

The continuous Japanese fingerspelling dataset used in this study is our previously collected data [2]. The sensor glove used for our previous data collection consists of Arduino Pro Mini and MPU6050, where conductive fiber weaving techniques [34] detect finger movements based on resistance changes, and the MPU6050 detects acceleration and angular velocity. The dataset contains words consisting of three to five fingerspelling characters, with each word comprising 11 dimensions (finger movement: five dimensions, acceleration: three dimensions, angular velocity: three dimensions) $\times$ 960 samples (120 sps $\times$ 8 sec). The dataset includes data from 33 participants, with each person performing 64 words five times each. As preprocessing, we calculated angles using the angular velocity three dimensions and added angles six dimensions ($\sin$ and $\cos$). Using moving average calculations, we also reduced the 960 samples (120 sps $\times$ 8 sec) to 32 samples (4 sps $\times$ 8 sec). Therefore, the input dimensions are 32 samples $\times$ 17 dimensions (length: 32, dim: 17).

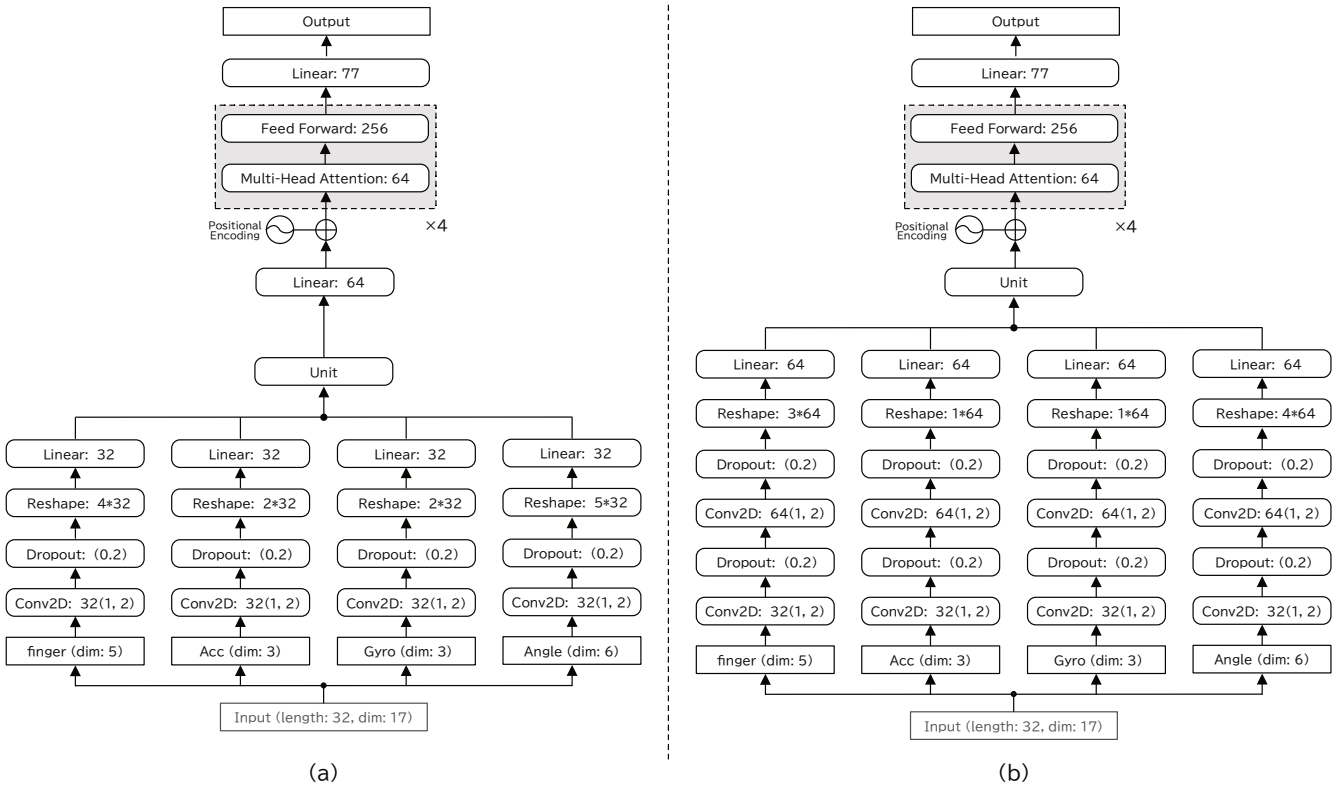

Figure 2. (a) Machine learning model architecture of the "branch-CNN-unit-Transformer Encoder", (b) Machine learning model architecture of the "branch-2CNN-unit-Transformer Encoder".

B. Machine Learning Model Architecture

For the development environment, we used a Docker container image distributed by NVIDIA [35]. The main specifications are as follows. We rebuilt all models, including our previously constructed branch-2CNN-unit-2LSTM, switching from TensorFlow to PyTorch.

- Ubuntu 22.04
- NVIDIA CUDA 12.3.0
- Python 3.10
- PyTorch 2.2.0a0+6a974bec

This motive is verify whether the JF recognition rate of both 2LSTM and Transformer Encoder when combined with CNN, improved compared to when not combined with CNN. Therefore, we included both 2LSTM, which consist of two LSTM layers, and the Transformer Encoder in this comparative analysis. Next, Figure 2 (a) and (b) show the machine learning model architectures that combine one and two layers of CNN with the Transformer Encoder. The reason for this architecture is that, similar to branch-2CNN-unit-2LSTM, the data is split into separate branches for each feature dimension, such as the finger movement, acceleration, angular velocity, and angle, and then input through the CNN layer.

C. Evaluation experiments

We evaluated each machine learning model architecture (Table I). First, we set the epoch to 3,000, and set the patience to 300 for stopping training using EarlyStopping as a measure against overfitting. In addition, we set the batch size to 64 for 2LSTM and branch-2CNN-unit-2LSTM, 16 for Transformer Encoder, and 32 for branch-CNN-unit-Transformer Encoder and branch-2CNN-unit-Transformer Encoder because the batch size was set to the best accuracy in each machine learning model from the result we tested at the batch size 16, 32, and 64, respectively.

1) Comparison of k-Fold Cross-Validation Methods: We evaluated each model using the 5-fold Cross-Validation (CV) and the 10-fold CV. The input data was shuffled and then divided into training and test data, and Table I shows the average and standard deviation of the results of the 5 and 10 runs for each model. Moreover, Table I values are not the values at the epoch when learning was stopped due to early stopping to prevent overfitting, but the values at the epoch when the validation loss was minimized. As shown in Table I, the F-measure micro of each model at the 5-CV and the 10-CV, except for 2LSTM, is over 90%. Comparing Transformer Encoder to 2LSTM, the F-measure micro and macro improved over that of the former. The same improvement was observed in comparing Transformer Encoder combined with CNN to 2LSTM combined with CNN. In particular, the F-measure macro for Transformer Encoder, branch-CNN-unit-Transformer Encoder, and branch-2CNN-unit-Transformer Encoder is improved by nearly 10% when compared to 2CNN-2LSTM, suggesting that the decreased recognition rate due to the large number of instances "ϕ" (the transition movements characters), which was a previous study [2] issue, has been alleviated.

TABLE I. MODEL COMPARISON EVALUATIONS IN FIRST COMPARATIVE ANALYSIS. THE PARENTHESES INDICATE THE STANDARD DEVIATION.

machine learning model architecture	k=5, F-measure [%]		k=10, F-measure [%]		k=33 (*random*[a]), F-measure [%]		k=33 (*person*[b]), F-measure [%]	
	micro	macro	micro	macro	micro	macro	micro	macro
2LSTM	89.9 (0.2)	50.5 (1.2)	90.2 (0.3)	51.6 (1.6)	90.4 (0.4)	52.1 (2.1)	88.8 (2.2)	40.5 (9.6)
branch - 2CNN - unit - 2LSTM	91.9 (0.1)	64.6 (0.7)	92.1 (0.2)	65.4 (1.1)	92.2 (0.4)	66.0 (2.5)	90.0 (0.3)	50.7 (13.1)
Transformer Encoder	92.8 (0.3)	72.5 (1.3)	93.3 (0.3)	74.9 (1.8)	93.5 (0.5)	75.8 (2.4)	92.2 (2.8)	69.1 (10.3)
branch - CNN - unit - Transformer Encoder	93.4 (0.1)	75.8 (0.6)	93.6 (0.2)	76.5 (0.8)	93.8 (0.5)	77.7 (2.2)	92.4 (2.9)	70.8 (10.1)
branch - 2CNN - unit - Transformer Encoder	93.3 (0.2)	74.8 (0.9)	93.6 (0.3)	76.6 (1.0)	93.8 (0.4)	77.1 (2.2)	92.4 (2.7)	70.8 (9.1)

[a] *random*: evaluation using randomly selected data for 33-fold CV.
[b] *person*: evaluation where the evaluation set consists of data from a single individual.

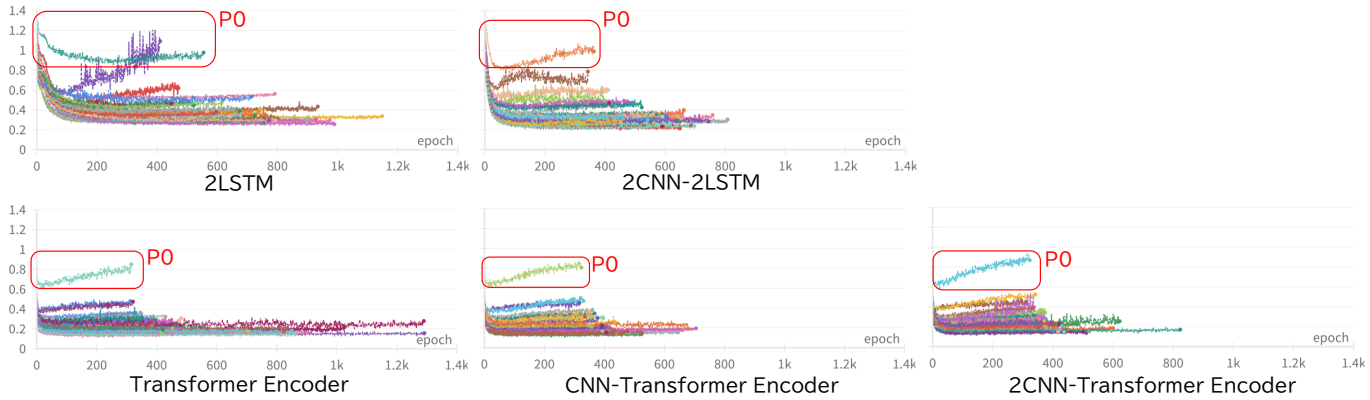

Figure 3. Learning progress graph for each model (horizontal axis: number of epochs, vertical axis: validation loss): The loss for each person in the 33-fold CV method is shown as a line graph, and it was observed that the data for one person (P0) showed a significant deviation from the others in the vicinity of 0.6 and 0.8.

2) The Impact of Individual Differences on 33-Fold Cross-Validation: We evaluated the results using the 33-fold CV, first, with case the input data for one person used as test data and the data for the remaining 32 people used as training data (k=33(person) in Table I), and second, the input data was shuffled and then divided into training and test data (k=33(random) in Table I). Moreover, Table I values are not the values at the epoch when learning was stopped due to early stopping to prevent overfitting, but the values at the epoch when the validation loss was minimized. As described in Section III-C1, the macro average of the F-measurement improved for all three models (Transformer Encoder, branch-CNN-unit-Transformer Encoder, and branch-2CNN-unit-Transformer Encoder) compared to branch-2CNN-unit-2LSTM. However, we found that we needed to consider the significant validation loss for the same person's test data. Figure 3 shows the graph showing the change in validation loss for each model at the 33-fold CV with case the input data for one person used as test data and the data for the remaining 32 people used as training data.

Table II presents a comparative analysis of F1-scores for individual fingerspelling characters across three models: the previous 2CNN-2LSTM model and the two best-performing models (CNN-Transformer Encoder and 2CNN-Transformer Encoder), under three conditions: random data distribution, P0, and P1. A notable improvement was observed in the recognition of challenging characters. While the 2CNN-2LSTM model showed zero F1-scores for 28 characters in the P0 condition, this number significantly decreased to 5 and 4 characters for the CNN-Transformer Encoder and 2CNN-Transformer Encoder models, respectively. Furthermore, the 2CNN-Transformer Encoder under P1 condition demonstrated robust performance, with no characters receiving zero F1-scores, outperforming both the 2CNN-2LSTM and CNN-Transformer Encoder models.

IV. Second Comparative Analysis

Using branch-CNN-Transformer Encoder and branch-2CNN-Transformer Encoder that showed good accuracy in First Comparative Analysis, we examine the impact of the input data for participant P0 identified in the impact of individual differences validation comparison. We also examine the impact of adding BatchNorm-1D and Rectified Linear Unit (ReLU) to each of the two machine learning model architectures. The timing of adding BatchNorm-1D and ReLU is when inputting to the Transformer Encoder module from the CNN module.

A. Evaluation experiments

For each branch-CNN-Transformer Encoder and branch-2CNN-Transformer Encoder, we evaluated the results using the 10-fold CV with three different combinations: add BatchNorm-1D and ReLU, removing the input data for participant P0, applying both modifications. The input data was shuffled and then divided into training and test data, and Table III shows the average and standard deviation of the results of the 10 runs for each model. Moreover, Table III values are not the values at the epoch when learning was stopped due to early stopping to prevent overfitting, but the values at the epoch when the validation loss was minimized. We also included the 10-fold CV values from Section III-C1 to check improvement from the no-applying case. CNN-Transformer Encoder and 2CNN-Transformer Encoder showed improvement in the F-measure macro compared to the no-applying case from case removing

TABLE II. THE F1-SCORES FOR EACH OF THE FINGER CHARACTERS IN THE CHARACTERS IN THE MODEL COMPARISON IN THE FIRST COMPARATIVE ANALYSIS.

2CNN-2LSTM						CNN-Transformer Encoder						2CNN-Transformer Encoder					
k=33		P0		P1		k=33		P0		P1		k=33		P0		P1	
chi	41.3	ho	0.0	me	0.0	du	55.3	nu	0.0	nu	0.0	du	53.9	na	0.0	di	33.3
ho	44.2	bo	0.0	hu	0.0	di	59.7	da	0.0	yo	0.0	di	58.6	so	0.0	nu	50.0
pe	46.6	pa	0.0	du	0.0	chi	63.9	di	0.0	du	40.0	chi	63.8	ke	0.0	re	52.6
du	48.9	hu	0.0	re	21.1	nu	65.2	ze	0.0	yu	50.0	pe	64.3	di	0.0	ma	53.8
te	49.9	zi	0.0	xya	25.0	pe	67.1	ke	0.0	re	53.3	ho	68.6	pu	14.3	ho	54.5
pu	53.4	ha	0.0	ni	27.3	ho	68.0	ta	11.8	di	54.5	bo	70.0	ho	16.7	te	54.5
hu	53.7	no	0.0	ne	28.6	pi	69.9	pi	13.3	ra	54.5	so	70.7	ko	20.0	chi	54.5
he	54.5	ze	0.0	pe	28.6	tsu	71.4	ko	15.4	chi	54.5	he	70.8	a	22.2	xtsu	59.5
di	54.6	ni	0.0	di	37.5	yu	71.7	ha	17.4	ta	57.1	tsu	70.9	ze	23.5	ra	60.0
so	54.6	ro	0.0	yu	40.0	na	71.9	du	17.4	xyo	60.0	nu	71.8	chi	24.4	du	62.5
ni	55.9	pi	0.0	ho	40.0	ta	72.6	te	18.2	me	63.2	pi	71.9	du	26.1	yo	66.7
ko	56.0	te	0.0	he	44.4	se	72.7	e	19.0	ma	64.0	se	72.4	pi	26.7	yu	66.7
pi	56.3	xtsu	0.0	te	44.4	ro	72.8	wo	21.7	ne	66.7	yu	72.6	da	26.7	hu	66.7
ha	56.4	chi	0.0	wo	46.2	yo	72.9	o	25.0	de	66.7	na	73.0	pe	28.6	ni	66.7
ga	57.4	so	0.0	ro	47.6	he	73.1	ho	26.7	hu	66.7	ha	73.1	e	28.6	ta	66.7
ro	57.6	yu	0.0	nu	50.0	so	73.1	xya	26.7	ho	66.7	ta	73.2	ge	28.6	zu	66.7
ka	57.8	bi	0.0	chi	51.6	ha	73.2	ro	28.6	zo	66.7	o	73.9	wo	28.6	ka	66.7
pa	57.9	bu	0.0	xtsu	52.2	me	73.9	bu	28.6	ha	66.7	ro	74.3	ha	28.6	go	69.2
tsu	58.3	da	0.0	bo	53.3	de	73.9	so	28.6	pa	66.7	yo	74.4	tsu	30.5	xyu	69.6
bo	58.8	ko	0.0	ko	53.3	ko	74.0	pe	28.6	bo	70.6	hu	74.5	bu	30.8	ha	70.0
ne	60.3	ke	0.0	de	54.5	bo	74.0	go	30.0	pe	70.6	te	74.6	te	30.8	sa	70.2
xtsu	60.8	ga	0.0	ha	55.6	hu	74.0	tsu	30.3	hi	71.0	wo	74.9	me	31.3	xya	70.6
me	61.3	pu	0.0	pa	57.1	su	74.4	wa	31.6	ro	71.4	hi	75.0	ro	31.6	pe	70.6
zi	62.5	a	0.0	hi	57.1	xya	75.1	bi	31.6	ge	72.7	me	75.0	gu	33.3	zi	70.6
ya	62.5	di	0.0	ze	57.1	te	75.4	yo	31.6	tsu	72.7	re	75.3	o	33.3	ya	70.8
wa	62.5	wo	0.0	ra	57.1	ni	75.4	no	33.3	ka	72.7	xya	75.4	pa	33.3	de	71.4
nu	62.5	pe	0.0	ke	57.1	re	75.5	de	35.3	wa	73.7	ni	75.5	bi	33.3	pa	71.4
hi	62.6	du	0.0	xyu	58.3	da	75.5	ra	35.3	go	74.1	ge	75.5	yo	33.3	za	71.4
de	63.0	su	9.7	pu	58.8	ba	75.6	a	36.4	za	75.0	su	75.6	mo	34.4	ku	71.7
a	63.2	ru	9.7	tsu	58.8	hi	75.6	pu	37.5	wo	75.0	ne	76.2	yu	37.5	a	72.7
da	63.2	ta	10.0	go	61.5	ge	75.6	ge	37.5	xtsu	75.0	de	76.2	zo	37.5	gu	73.7
o	63.9	wa	13.8	sa	63.2	ke	75.8	yu	37.5	e	75.0	ba	76.3	ne	38.1	me	73.7
su	65.7	ne	14.3	po	64.7	o	76.5	pa	38.1	xya	75.0	bi	76.6	bo	40.0	wa	73.7
yu	66.0	de	14.3	be	66.7	wo	76.6	ne	38.1	ke	75.0	ma	76.9	nu	40.0	ne	73.7
ba	66.0	ri	14.6	da	66.7	ne	77.1	me	38.7	xyu	75.0	pa	76.9	ba	40.0	tsu	75.0
na	66.1	ra	18.2	yo	66.7	gu	77.4	zi	40.0	ru	75.3	xyu	77.1	ki	40.4	e	75.0
ma	66.2	xya	20.0	ma	66.7	ra	77.7	be	40.0	do	75.5	gu	77.7	zu	42.1	bo	76.2
shi	66.3	hi	21.4	ka	66.7	e	77.8	se	40.0	sa	75.5	ko	78.0	re	42.4	gi	76.7
bi	66.3	me	22.2	ta	66.7	ma	77.9	gu	41.0	su	75.9	da	78.0	xtsu	42.6	su	77.2
yo	66.7	ma	22.7	se	66.7	wa	77.9	su	41.6	ku	76.4	xtsu	78.4	su	43.3	no	77.2
po	66.9	ya	24.0	mo	69.1	bi	78.1	re	42.9	ya	76.6	to	78.6	ku	43.8	da	77.8
wo	67.2	go	25.5	za	69.2	ru	78.1	chi	43.5	mo	76.9	e	78.7	xya	44.4	ko	77.8
se	67.2	ba	26.1	wa	70.0	xtsu	78.2	ri	45.9	ni	78.6	ya	78.7	se	46.2	he	78.3
ta	67.3	tsu	26.8	zu	70.6	mo	78.2	gi	46.7	u	79.7	ru	78.8	hu	46.7	hi	78.8
sa	68.1	o	27.6	ya	71.1	ki	78.6	shi	47.1	te	80.0	ra	78.8	go	46.8	-	79.0
e	68.3	ge	28.6	zo	71.4	ya	78.7	ku	47.2	po	80.0	po	78.9	ra	47.1	wo	80.0
re	68.4	zo	29.6	bi	71.4	xyu	79.1	bo	47.6	so	80.0	shi	78.9	no	48.0	xyo	80.0
ru	68.5	re	30.0	ga	71.4	po	79.2	sa	47.9	a	80.0	u	79.0	ga	48.5	pu	80.0
xya	69.0	za	30.0	su	71.7	shi	79.3	hu	48.0	i	80.9	ka	79.1	ta	50.0	mo	80.0
be	69.1	na	30.8	ri	73.5	to	79.7	xtsu	49.1	to	81.0	mo	79.1	po	50.0	ke	80.0
ra	69.4	be	30.8	i	75.2	pa	79.9	ma	50.0	ri	81.3	bu	79.3	ni	50.0	zo	80.0
go	70.4	se	31.3	ru	75.6	a	79.9	ba	50.0	-	81.4	go	79.5	he	50.0	do	80.6
ze	70.8	gi	32.7	u	76.0	u	80.0	na	50.0	gu	82.4	a	79.6	hi	50.0	u	81.0
ge	70.9	he	33.3	to	76.1	ka	80.0	zo	50.0	o	82.4	ke	79.7	ri	50.3	se	81.3
ke	70.9	ku	35.0	e	76.2	i	80.4	mo	50.7	no	83.3	ki	80.2	za	51.9	po	81.5
gu	71.1	yo	35.3	mu	76.5	sa	80.6	ni	51.1	gi	83.6	wa	80.4	de	52.6	ru	82.5
zu	71.9	po	35.3	so	76.9	ga	80.6	ki	51.7	ko	84.2	sa	80.7	shi	52.6	to	83.2
bu	73.4	nu	35.3	ge	76.9	zu	80.6	ru	53.6	ze	85.7	zu	81.0	-	53.6	bu	83.3
i	73.7	do	37.5	zi	76.9	no	81.0	za	53.8	zi	85.7	i	81.0	u	54.2	bi	83.3
zo	73.8	shi	40.5	o	78.3	xyo	81.1	he	53.8	ga	85.7	do	81.5	zi	55.6	ro	83.3
mo	74.0	sa	40.6	do	79.3	go	81.2	-	54.1	ba	85.7	mu	81.5	i	56.0	shi	84.0
u	74.6	ka	40.7	ku	80.0	mu	81.2	zu	54.5	se	86.7	ga	81.6	nn	56.3	so	84.2
ki	75.3	-	41.4	-	81.0	bu	81.5	hi	55.0	he	87.0	no	81.6	ma	56.5	ri	85.4
xyu	75.8	u	43.9	gu	82.4	do	81.6	u	55.4	da	87.5	zo	81.7	gi	57.1	ba	85.7
no	76.1	e	44.4	shi	82.4	za	81.8	i	58.4	zu	87.5	gi	81.8	wa	57.1	i	85.7
za	77.1	i	45.5	gi	82.6	gi	82.0	nn	58.4	shi	88.0	ri	82.6	sa	57.5	nn	85.9
to	77.1	ki	47.9	no	83.0	pu	82.1	mi	58.9	nn	88.3	pu	82.6	ka	57.8	ze	87.5
gi	77.3	mo	48.5	xyo	83.3	ku	82.2	ka	61.2	pi	90.9	ze	82.9	to	58.8	ga	87.5
do	77.4	zu	50.0	nn	84.3	ri	82.8	xyu	61.5	ki	92.0	za	82.9	mi	60.4	mi	88.3
xyo	77.8	to	58.9	a	85.7	-	82.9	po	63.6	mu	92.3	xyo	83.0	ru	63.8	o	90.9
-	78.6	nn	59.2	ba	85.7	mi	83.2	xyo	64.0	be	94.1	zi	83.2	mu	65.2	ki	91.3
ri	78.7	xyu	59.5	ki	88.4	zo	83.3	to	64.8	pu	94.1	ku	83.4	be	66.7	mu	92.3
ku	78.9	gu	63.6	bu	88.9	nn	83.5	mu	65.4	mi	94.6	-	83.5	xyo	66.7	ge	93.3
nn	79.7	mu	63.8	mi	90.9	ze	84.4	do	65.9	ϕ	97.1	nn	83.7	ya	66.7	be	94.1
mu	79.9	mi	64.7	pi	90.9	be	84.8	ga	66.7	bi	100.0	mi	83.8	do	67.4	ϕ	97.0
mi	81.4	xyo	66.7	ϕ	96.5	zi	85.4	ya	69.1	na	100.0	be	86.0	xyu	68.1	na	100.0
ϕ	96.1	ϕ	90.5	na	100.0	ϕ	96.5	ϕ	90.6	bu	100.0	ϕ	96.5	ϕ	90.3	pi	100.0
	66.0		22.7		63.1		77.0		40.3		75.4		77.1		41.7		75.6

TABLE III. MODEL COMPARISON EVALUATIONS IN SECOND COMPARATIVE ANALYSIS. THE PARENTHESES INDICATE THE STANDARD DEVIATION.

machine learning model architecture	k=10, F-measure [%]	
	micro	macro
branch - CNN - unit - Transformer Encoder	93.6 (0.2)	76.5 (0.8)
(Remove P0 data)	93.9 (0.2)	77.6 (0.7)
(Add BatchNorm-1D and ReLU)	93.8 (0.2)	77.4 (1.0)
(Remove P0 data & Add BatchNorm-1D and ReLU)	94.1 (0.3)	78.4 (1.0)
branch - 2CNN - unit - Transformer Encoder	93.6 (0.3)	76.6 (1.0)
(Remove P0 data)	93.9 (0.2)	77.3 (1.0)
(Add BatchNorm-1D and ReLU)	93.5 (0.3)	76.1 (1.4)
(Remove P0 data & Add BatchNorm-1D and ReLU)	93.9 (0.2)	77.7 (0.6)

the input data for participant P0. Furthermore, in the case of adding BatchNorm-1D and ReLU, CNN-Transformer Encoder showed improvement, but no improvement was observed for 2CNN-Transformer Encoder.

Finally, we conducted a word-level accuracy evaluation of the best-performing model configuration: the branch-CNN-unit-Transformer Encoder incorporating BatchNorm-1D and ReLU. The evaluation utilized the Letter Error Rate (LER) metric, formulated in (1).

$$\mathrm{LER} = \frac{\text{Substitutions} + \text{Deletions} + \text{Insertions}}{\text{The number of characters in the reference}} \quad (1)$$

Similar to CTC, when evaluating sequences processed by removing "ϕ" tokens from the Encoder's output and merging consecutive identical fingerspelling characters, the system demonstrated strong performance with an average LER of 0.122 (0.008), indicating a low misrecognition rate.

V. DISCUSSION

A. Replacing LSTM with a Transformer Encoder

Our results demonstrated improvement in generalization performance when replacing LSTM with Transformer architectures. However, since the removal of P0 data led to improved macro F-measure scores, we cannot make strong claims about the model's ability to handle individual differences. The distinctive recognition results for P0 raise two potential explanations:

1) Whether this represents an "extreme individual difference"
2) Whether the data contains "noise" that transcends typical individual variations

During data collection, we only gathered limited participant attributes (age, gender, hearing ability, and sign language experience), which prevented us from fully analyzing the characteristics of the removed participant's data. Consequently, we cannot definitively determine whether the observed variations represent individual differences or more significant data anomalies. Thus, we have not necessarily sufficiently evaluated the machine learning model's robustness when the dataset contains outliers. Future data collection efforts should include more comprehensive participant attribute information, such as experience with JSL or Signed Japanese, to enable more thorough analysis. On the other hand, we can conclude that both CNN combination and BatchNorm-1D+ReLU application contributed to performance improvements. However, we have not yet explored the full parameter space for CNN combinations or the relationship with preprocessing parameters used in moving average calculations during dataset construction. Further comparative analysis is needed. Specifically, we plan to investigate varying the moving average calculation from 960 samples (120 sps $\times$ 8 s) to N$\times$8 samples (N sps $\times$ 8 s), along with corresponding adjustments to CNN parameters - modifying kernel size from (1, 2) to (1, N) and stride from (1, 1) to (1, N).

B. Practical use of the Transformer Encoder

Our study focused solely on the Transformer Encoder component and demonstrated its practical applicability for interface implementation. Specifically, not only did we achieve a LER of 0.122 (0.008), but we also recorded a total inference time of 0.732 s for processing the entire evaluation dataset. Given that the evaluation dataset consisted of 1,042 words, we estimated an average inference time of 0.702 ms per word.

C. Limitation

The current recognition system is not designed for real-time processing; rather, it begins recognition only after receiving a complete word input. Moreover, our continuous fingerspelling dataset consists solely of word-level data, and similar to SpellRing [26], does not include sentence-level data. Thus, we cannot evaluate continuous fingerspelling recognition at sentence-level including also NM expressions and grammatical omission. In addition, as our implementation only utilizes the Transformer Encoder, we have not conducted comparative analyses involving Transformer Decoder or CTC [28]. Moreover, our analysis does not include comparisons with other advanced architectures used in previous studies, such as Conformer [19], Spatial Temporal Graph Convolutional Networks [18], and Signformer [21].

VI. CONCLUSION AND FUTURE WORKS

In this study, we conducted a quantitative evaluation using the CNN-LSTM combined model as a baseline to assess whether the Transformer Encoder could improve Japanese fingerspelling recognition rates. Our results demonstrated that for 76 Japanese fingerspelling characters, the system achieved average micro and macro F-measures of 93.8% (0.2) and 77.4% (1.0), respectively, with a word-level LER of 0.122 (0.008). We confirmed that replacing LSTM with Transformer improved generalization performance. Future work should investigate machine learning models incorporating both Transformer Encoder and Decoder architectures. Additionally, comparative analyses including CTC and HMM approaches are necessary. Further our research will also extend to comparisons with other advanced architectures, such as Conformer [19], Spatial Temporal Graph Convolutional Networks [18], and Signformer [21]. Finally, we plan to examine the spatial characteristics differentiating fingerspelling from sign language through comparative analysis using our collected one-handed sign language dataset [36].

ACKNOWLEDGMENT

This work was supported by JSPS KAKENHI Grant Numbers JP19K11411, JP24K14243.

REFERENCES

[1] T. Tsuchiya, A. Shitara, F. Yoneyama, N. Kato, and Y. Shiraishi, "Sensor glove approach for japanese fingerspelling recognition system using convolutional neural networks," in Proceedings of The Thirteenth International Conference on Advances in Computer-Human Interactions (ACHI 2020), 2020, pp. 152–157.

[2] Y. Shiraishi, A. Shitara, F. Yoneyama, and N. Kato, "Sensor glove approach for continuous recognition of japanese fingerspelling in daily life," International Journal on Advances in Life Sciences, vol. 14, 2022, pp. 53–70.

[3] V. Ashish et al., "Attention is all you need," in Advances in Neural Information Processing Systems, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.

[4] N. Mukai, N. Harada, and Y. Chang, "Japanese fingerspelling recognition based on classification tree and machine learning," in 2017 Nicograph International (NicoInt). New York, NY, USA: IEEE (Institute of Electrical and Electronics Engineers), June 2017, pp. 19–24.

[5] H. Hosoe, S. Sako, and B. Kwolek, "Recognition of jsl finger spelling using convolutional neural networks," in 2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA), May 2017, pp. 85–88.

[6] M. A. Jalal, R. Chen, R. K. Moore, and L. Mihaylova, "American sign language posture understanding with deep neural networks," in 2018 21st International Conference on Information Fusion (FUSION). New York, NY, USA: IEEE (Institute of Electrical and Electronics Engineers), July 2018, pp. 573–579.

[7] M. Kondo, N. Kato, K. Fukui, and A. Okazaki, "Development and evaluation of an interactive training system for both static and dynamic fingerspelling using depth image," IEICE technical report, vol. 114, no. 512, 2015, pp. 23–28, (in Japanese).

[8] P. A. D. Bianco, O. A. Stanchi, F. M. Quiroga, F. Ronchetti, and E. Ferrante, "Signattention: On the interpretability of transformer models for sign language translation," 2024, Available: https://arxiv.org/abs/2410.14506 [retrieved: April, 2025].

[9] L. Tarrés, G. I. Gállego, A. Duarte, J. Torres, and X. Giró-i Nieto, "Sign language translation from instructional videos," in 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2023, pp. 5625–5635.

[10] D. Amanda et al., "How2sign: A large-scale multimodal dataset for continuous american sign language," in 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 2734–2743.

[11] N. C. Camgoz, S. Hadfield, O. Koller, H. Ney, and R. Bowden, "Neural sign language translation," in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 7784–7793.

[12] RWTH-PHOENIX-Weather 2014, "RWTH-PHOENIX-Weather 2014," 2019, URL: https://www-i6.informatik.rwth-aachen.de/~koller/RWTH-PHOENIX/ [retrieved: April, 2025].

[13] RWTH-PHOENIX-Weather 2014-T, "RWTH-PHOENIX-Weather 2014-T," 2019, URL: https://www-i6.informatik.rwth-aachen.de/~koller/RWTH-PHOENIX-2014-T/ [retrieved: April, 2025].

[14] C. C. Patel and P. Patel, "A comparative analysis of sign language recognition approaches across varied sign languages," in Universal Threats in Expert Applications and Solutions, V. S. Rathore, V. Piuri, R. Babo, and K. S, Eds. Singapore: Springer Nature Singapore, 2024, pp. 355–372.

[15] M. De Coster, D. Shterionov, M. Van Herreweghe, and J. Dambre, "Machine translation from signed to spoken languages: state of the art and challenges," in Universal Access in the Information Society, vol. 23. Singapore: Springer Nature Singapore, 2024, pp. 1305–1331.

[16] X. Shen, Z. Zheng, and Y. Yang, "Stepnet: Spatial-temporal part-aware network for isolated sign language recognition," ACM Trans. Multimedia Comput. Commun. Appl., vol. 20, no. 7, May 2024.

[17] N. Takayama, G. Bemitez-Garcia, and H. Takahashi, "Sign language recognition based on spatial-temporal graph convolution-transformer," Journal of the Japan Society for Precision Engineering, vol. 87, no. 12, 2021, pp. 1028–1035.

[18] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, no. 1, Apr. 2018.

[19] G. Anmol et al., "Conformer: Convolution-augmented transformer for speech recognition," in Interspeech 2020, 2020, pp. 5036–5040.

[20] T. Kimura, T. Miura, and K. Kanda, "AI RECOGNITION OF JAPANESE SIGN LANGUAGE AND ITS APPLICATION," Journal of International Scientific Publications: Language, Individual & Society, vol. 18, 2024, pp. 1–10.

[21] E. Yang, "Signformer is all you need: Towards Edge AI for Sign Language," 2024, Available: https://arxiv.org/abs/2411.12901 [retrieved: April, 2025].

[22] M. E. Cabrera, J. M. Bogado, L. Fermin, R. Acuna, and D. Ralev, "Glove-based gesture recognition system," in Adaptive Mobile Robotics. World Scientific, 2012, pp. 747–753.

[23] 5DT, "5DT Data Glove 5 Ultra," 2019, URL: https://5dt.com/5dt-data-glove-ultra/ [retrieved: April, 2025].

[24] C. K. Mummadi, F. P. P. Leo, K. D. Verma, S. Kasireddy, P. M. Scholl, and K. Van Laerhoven, "Real-time embedded recognition of sign language alphabet fingerspelling in an imu-based glove," in Proceedings of the 4th International Workshop on Sensor-Based Activity Recognition and Interaction, ser. iWOAR '17. New York, NY, USA: Association for Computing Machinery, 2017, pp. 1–6.

[25] N. M. Kakoty and M. D. Sharma, "Recognition of sign language alphabets and numbers based on hand kinematics using a data glove," Procedia Computer Science, vol. 133, 2018, pp. 55–62.

[26] H. Lim et al., "Spellring: Recognizing continuous fingerspelling in american sign language using a ring," 2025, Available: https://arxiv.org/abs/2502.10830 [retrieved: April, 2025].

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015, Available: https://arxiv.org/abs/1512.03385 [retrieved: April, 2025].

[28] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in Proceedings of the 23rd International Conference on Machine Learning, ser. ICML '06. New York, NY, USA: Association for Computing Machinery, 2006, p. 369–376.

[29] T.-W. Chong and B.-J. Kim, "American sign language recognition system using wearable sensors with deep learning approach," The Journal of the Korea Institute of Electronic Communication Sciences, vol. 15, no. 2, 2020, pp. 291–298.

[30] J. Yincheng et al., "Smartasl: "point-of-care" comprehensive asl interpreter using wearables," Proc. ACM Interact. Mob. Wearable Ubiquitous Technol., vol. 7, no. 2, Jun. 2023.

[31] R. Colin et al., "Exploring the limits of transfer learning with a unified text-to-text transformer," J. Mach. Learn. Res., vol. 21, no. 1, Jan. 2020.

[32] L. Jiyang et al., "Signring: Continuous american sign language recognition using imu rings and virtual imu data," Proc. ACM Interact. Mob. Wearable Ubiquitous Technol., vol. 7, no. 3, Sep. 2023.

[33] C. Neidle, A. Opoku, and D. Metaxas, "Asl video corpora & sign bank: Resources available through the american sign language linguistic research project (asllrp)," 2022, Available: https://arxiv.org/abs/2201.07899 [retrieved: April, 2025].

[34] R. Takada, J. Kadomoto, and B. Shizuki, "A sensing technique for data glove using conductive fiber," in Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems, ser. CHI EA '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 1–4.

[35] NVIDIA, "Nvidia docs hub - nvidia optimized frameworks," 2025, URL: https://docs.nvidia.com/deeplearning/frameworks/pytorch-release-notes/rel-23-11.html [retrieved: April, 2025].

[36] A. Shitara, T. Kasama, F. Yoneyama, and Y. Shiraishi, "One-handed signs: Standardization for vehicle interfaces and groundwork for automated sign language recognition," in Proceedings of The Seventeenth International Conference on Advances in Computer-Human Interactions (ACHI 2024), 2024, pp. 174–181.