**Automatic Thai Finger Spelling Transcription.**

**Running title**

Automatic Thai Finger Spelling Transcription.

**Abstract**

This article explores a transcription of a video recording Thai Finger Spelling (TFS)— a specific signing mode used in Thai sign language— to a corresponding Thai word. TFS copes with 42 Thai alphabets and 20 vowels using multiple and complex schemes. This leads to its unique challenges and necessity to take an issue of processing video data uncommon in spelling schemes of other sign languages. Our proposed system, Automatic Thai Finger Spelling Transcription (ATFS), processes a signing video in 3 stages: ALS marking video frames in order to easily remove any non-signing frame as well as conveniently group frames associating to the same alphabet, SR classifying a signing image frame to a sign label (or its equivalence), and SSR transcribing a series of signs into an alphabet. ALS utilizes TFS practice of signing different alphabets at different locations. SR and SSC employ well-adopted spatial and sequential models. Our ATFS has been found to achieve Alphabet Error Rate 0.256 (c.f. 0.63 of the baseline method). Our findings also reveal a benefit of smoothing mechanism in ALS, which could lead to over 15.88% improvement.

**Keywords:** *Sign Language Transcription, Thai Finger Spelling Transcription, Sign Sequence Classification Sign Video Transcription.*

# Introduction

Sign language is a main communication channel of a deaf community. However, sign languages are not universal nor mutually intelligible with each other. Each country creates its own sign language with its own grammar and lexicon. There are more than one-hundred different sign languages around the world and also many different gesture alphabets to represent their alphabets. A finger spelling is used in a sign language to represent alphabets for spelling names of people and any meaning which has not been defined as a semantic sign. Various schemes have been employed for finger spelling: American, French, and Russian Sign Languages employ a one-hand scheme; British and Australian Sign Languages employed two-hand scheme. In order to cope with 42 alphabets and 20 vowels, Thai Finger Spelling (TFS) employ a one-hand scheme with an extension using multi-postures for the alphabets and a two-hand scheme for the vowels[1]. Noted that, finger spellings of many large-alphabet-repertoire languages, e.g., Chinese and Japanese, resort to a phonetic system for a manageable set of spelling signs.

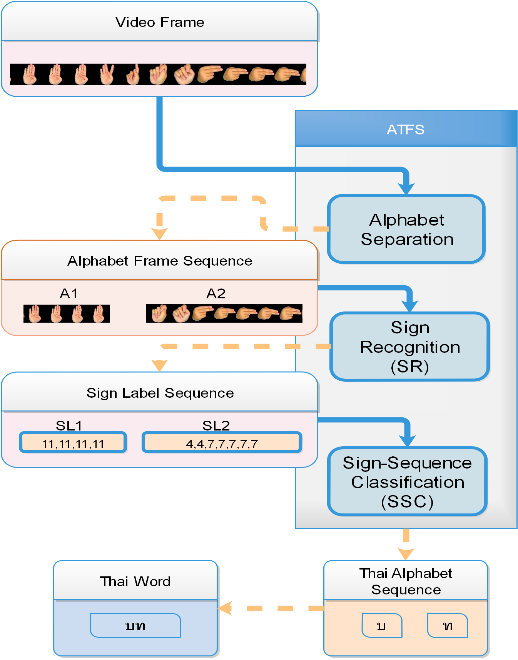
In addition, the mobile phone technology allows deaf people to use videophone to communicate with other deaf people, but it is diffecult to comnunicate between deaf people and normal people. The translatter is need to tanslate from sign languate to text or voice for this situation. An automatic sign language translation is an improtance technology for communicate betweem them. A deaf people and normal people would take benefit from this technology to improve the communication that would make a better society..

## Literature Review.

A Finger Spelling Recognition (FSR) system takes an image of a signing hand and automatically gives a corresponding sign label. P. Nakjai and T. Katanyukul [1] has addressed FSR for TFS using their customized CNN and reported an average accuracy of on a single-image setting. P. Nakjai, P. Maneerat, and T. Katanyukul [2] has employed a Yolo-based Darknet-19[3] to both locating a hand and reading a sign. has reported mAP also on a single-image setting, but with a complex background. [4] has employed VGG-16[5] and reported mAP on a single-image setting, as well as discovered Latent Cognizance—an innovative mechanism to allow an out-of-context awareness. Although addressing TFS video transcription is in need for a practical FSR as [1] discussed, an investigation into this issue has so far been lacking. Therefore, this article proposes Automatic Thai Finger Spelling Transcription (ATFS) to address transcription of a video clip signing TFS. Unlike other sign languages, TFS employs multiple and complex schemes, as shown in Table 1. This uniqueness renders challenges and necessity of addressing video setting, which is uncommon in other sign languages.

**Table 1** Example of Sign languages and their finger spelling scheme.

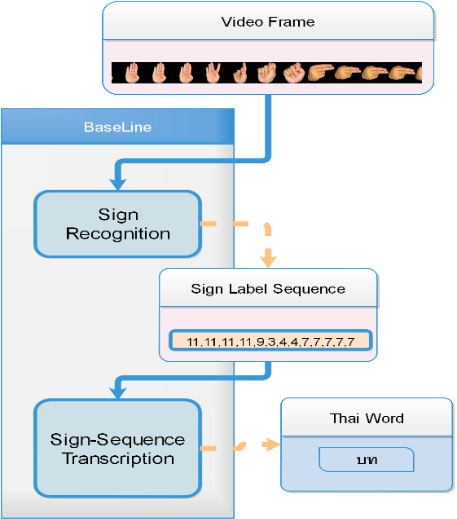
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sign Language | Number of Signs | Number of Alphabets/Characters | Signing  Scheme | Automatic Transcription  Approach | Remark |
| British[6] | 26 signs | 26 | Two-handed posture | Classification |  |
| American [7], [8] | 24 signs and  2 movements | 26 | Single-posture |  |
| Russian [9] | 24 signs and  8 movements | 32 |  |
| Arabic [10], [11] | 28 signs and  2 movements | 30 |  |
| French [12] | 23 signs and  3 movements | 26 |  |
| Chinese [13] | 30 signs | 3000+ | Phonetic System  (30 sounds) |
| Japanese [14], [15] | 41 signs and  5 movements | 46 (Hiragana) | Phonetic System  (46 Hiragana) |
| Thai [1], [16] | 25 signs ( consonants),  6 signs(non-consonants) and  17 palm locations | 42 consonants,  4 intonations and  20 vowel visuals | Single-posture,  Multi-posture,  and Hand- mapping | Our Approach targeted  25 signs covering 42 alphabets |  |



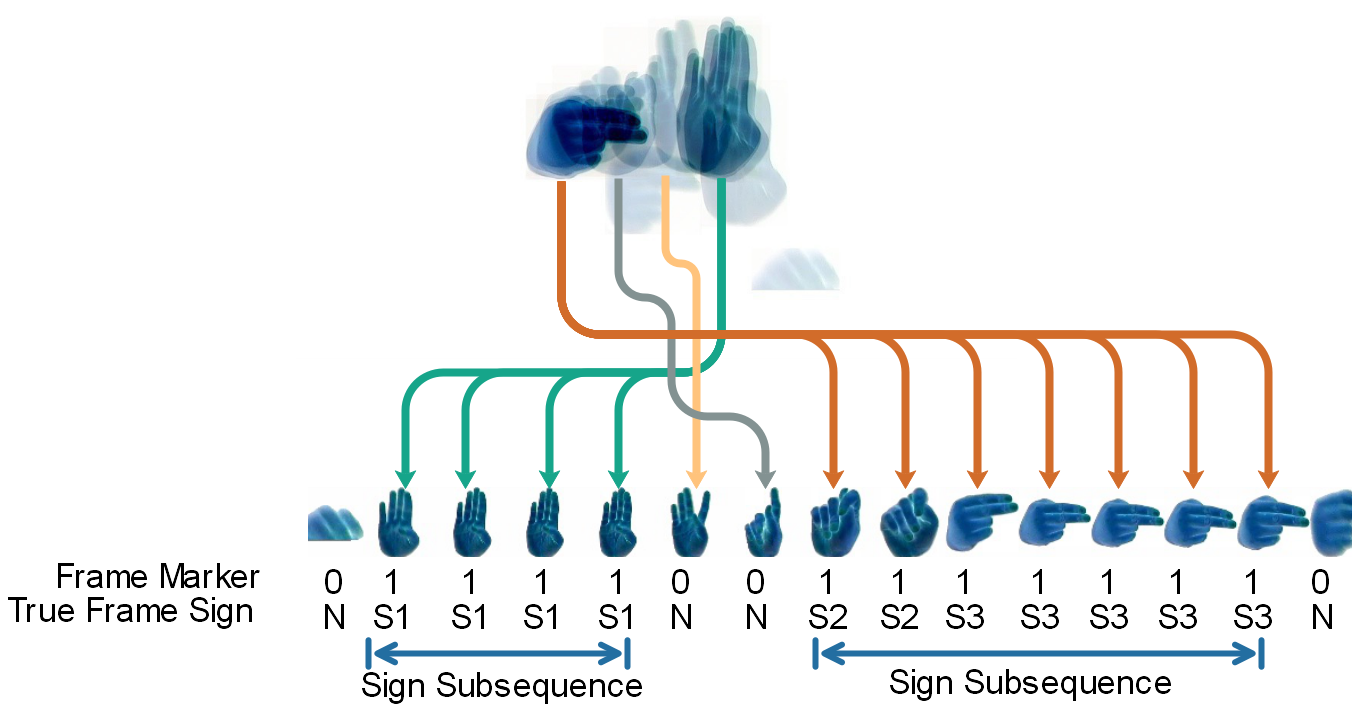
**Figure 1** Automatic Thai Finger Spelling Transcription (ATFS). The ATFS has 3 states that are Alphabet separation stage, Sign recognition state, and Sign-sequence classification state. The alphanet separation state take a sequence image as an input and marked consecutive sign frames to indicate a same an alphabet that are call Alphabet Frame Sequence . The second state take and the alphanet frame sequence and provide a sign label sequnce. The final state transcribe the sign label sequence to a single Thai alphabet.

# The Proposed System.

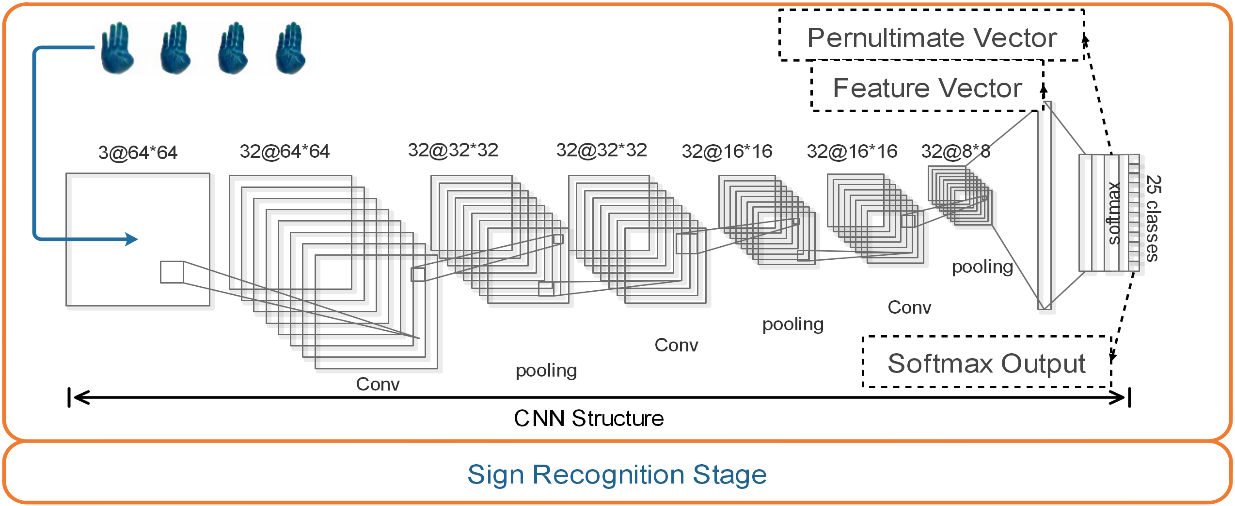
This section describes the process of our proposed Automatic Thai Finger Spelling Transcription (ATFS) system, targeted 25 signs (covering all 42 Thai alphabets, not including 20 vowels). The ATFS system takes a video clip as an input and transcribes to a Thai word. Our ATFS system can transcribe only one word per video clip. It has 3 main stages: the first stage is alphabet-separation; the second stage is sign recognition; and the last stage is the sign-sequence classification. Figure 1 illustrates the ATFS system.



**Figure 2** Baseline Approach is a basis sign transcribbing system. Baseline appoach developed based on gredy algorithm that consider a subsequence of consecutive signs of the same label whose length is longer than a prespecified number. A subsequence of consecutive signs will be marked to be a sign label and transcribed to a Thai alphabet.



**Figure 3** Alphabet-separation (ALS) stage. A sequence of image frames, shown as superimposed image on the top, illustrates signing of a word with two alphabets. A signer generally positions each signing at a slightly different location. Signing postures are broken down and displayed in order from left to right; arrows associates each signing to its position. Below the signing postures, frame markers (output from ALS stage) and true frame signs are shown. The whole signing spells two-alphabet word “บท”, composed of 3 signs: S1 “บ”; (S2,S3) “ท”.



**Figure 4** CNN structure of SR stage follows [1]. The CNN structure has convolution layer that has 32 filters and is followed by ReLUs function as an activattion function. Each filter is of size 3 x 3. Max-pooling layer are applied after the convolution layer. The final layer is a fully-connected layer that has 1024 hidden node and is followed a soft-max function to predict 25 classes.

## The alphabet-separation stage.

The first stage, called alphabet-separation (ALS) stage, takes a video clip and provides frame markers to indicate which frames belong to the same alphabet and which frames are non-signing and should be discarded.

The alphabet-separation (ALS) stage takes a video clip as an input and provides frame markers to indicate which frames belong to the same alphabet and which frames are non-signing. Each video frame will be marked as either being a sign (labeled 1) or being a non-sign (labeled 0) frame. Consecutive sign frames are called a sign subsequence. Figure 3 illustrates an alphabet separation stage. Our ALS stage marks signing frames based primarily on signing position. A signer generally positions a different sign at a slightly different location. A centroid of a hand area in a frame image is used to represent a signing position. We investigates five ALS approaches, i.e., D1, D2T, D2M, D2S, and HM. The D1, D2T, D2M, and D2S employ thresholding on a Euclidean distance between centroid of 2 consecutive frames. D1 simply uses a (single) threshold to decide a marker, that can be calculated as follows:

if , and otherwise, (1)

where is a marker of the frame; is pre-defined threshold; Euclidean distance, and ; and are x and y coordinates of a centroid of the frame. D2T, D2M, and D2S employ double thresholding as in Equation 2.

(2)

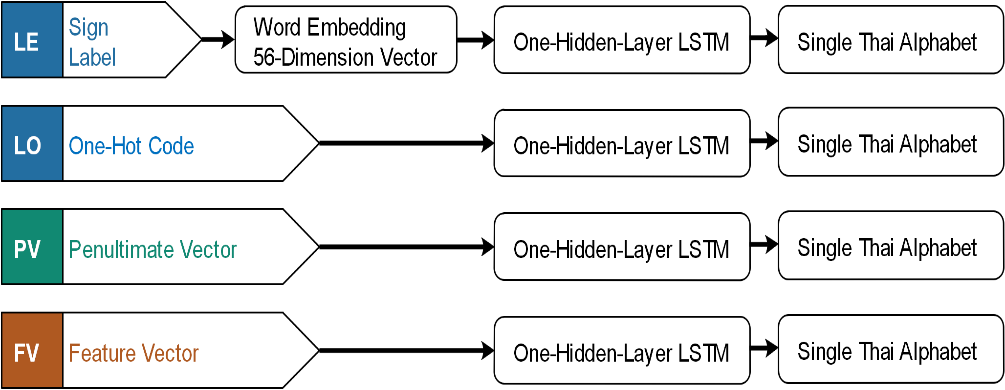
where and are lower and upper thresholds; is a distance at the frame. D2T uses a Euclidean distance . D2M uses moving average distance, , whose value is calculated by , where is a Euclidean distance of the frame; is a user-specific value (moving average window); and for . D2S use a skipping distance, i.e., for and for .

HM employ distance through heatmap mechanism. It also takes centroid frequency into account. It is to utilize frequent proximity of centroids. A more frequent proximity of centroids indicate a more likely that it is a signing frame. Specifically, a heatmap is constructed through Gaussian kernel density method, that can be calculated as follows:

, (3)

where is a heatmap pixel at coordinate; and are x and y coordinates of a centroid at the frame; is a user-specific span parameter. Then, an alphabet area can be identified through thresholding, if and otherwise. Threshold is user-specific. To mark a sign frame, any frame whose centroid lies inside the alphabet area where will be marked as signing frame, if and otherwise.

Moreover, our study employed the Window Frame Smoothing technique(WFS) to improve the output from the alphabet-separation(ALS) stage. Given a sequence of an alphabet-binary output from ALS as , WFS with a hyperparameter corrects the sequence by if for in a step-by-step manner from to . For example, assuming = 5, when the alphabet binary is , WFS will output ; given , WFS will output . According to The sign recognition stage(SR), this is a process to identify a TFS sign from a frame image. Our SR stage takes after [1], using an image classification Convolution Neural Network (CNN).



**Figure 5** Four alternatives in SSC stage.

**The sign-sequence classification stage.**

The sign-sequence classification stage(SSC) predicts a single Thai alphabet from an input sequence. Four alternatives of an SR input sequence are explored. Each alternative represents how SR and SSC are connected. The SSC stage takes sign labels from the SR stage. The SSC stage may represent sign labels simply by one-hot coding (denoted LO) or it may even elaborate the label representation through word embedding[17] (denoted LE). Optionally, the SSC stage can reach further and take a penultimate vector[4]—a value vector before softmax calculation— (denoted PV) or a feature vector (denoted FV) deeper inside the SR stage. Figure 5 illustrates these four alternatives. Given the input sequence , our study predicts an alphabet label using Long Short-Term Memory(LSTM) with one hidden layer. The LSTM has been widely used in sequence classification[18], machine translation[19], speech recognition[20], and video description[21]. Concisely, LSTM predicts label from , where is obtained through , for , when and are weight vector parameters.

## 

# Experiments and Results

Our experiments are to evaluate our proposed system both entirely and separately stagewise. To prepare and evaluate our system, 2 dedicated TFS datasets are acquired.

## TFS Video Dataset.

The Thai Finger spelling(TFS) video dataset is collected from a professional signer. The dataset contains 212 video clips (162 clips for training and 50 clips for testing). The 212 samples are chosen from the top 200 bi-gram Thai words that most frequently appear in Thai names and some ranks may associate to multiple words. Each video shows normal pace and clear signing. Each video clip—lasts 5.37s on average at 29 fps— records a bi-alphabet Thai word—a word with exactly 2 alphabets— corresponding to TFS signs. The first alphabet lasts 0.3-0.8s on average. The second alphabet lasts 0.68-1.1s on average. Annotation has been marked on video frames for TFS signs as well as non-signs. This dataset is intended for alphabet separation, sign-sequence classification, and the evaluation of the entire system.

## TFS Image Dataset.

The image dataset is collected from 12 signers. It contains all 25 hand sign classes. Each sign class is posed 5 times by each signer that makes it to original images, before being augmented to 30,000 images. The 15,000 images are used for training set and the remainings are used for the test set. The image dataset is intended for development and stagewise evaluation of the sign recognition stage.

## Experiment Setting.

The experiments explore various combinations of approaches. Stagewise, the sign-recognition (SR) stage is implemented by CNN with configurations as [1]. The CNN, whose structure is shown in Figure 4, is trained with TFS image dataset. Five approaches—HM, D1, D2T, D2M, and D2S— are investigated for ALS stage. HM uses sigma of 10. D2M uses 3-frame MAV. The HM and D1 use single threshold, i.e., and , respectively. The double thresholds of and are applied to D2T and D2M. The double thresholds of and are applied to D2S. The effects of frame smoothing on alphabet-separation results are also examined on 2 options, i.e., no frame smoothing and frame smoothing with a hyperparameter is 5. The sign-sequence-classification(SSC) stage is implemented by one-hidden-layer LSTM. Five alternatives of LSTM have been assessed, i.e., LSTM with 2, LSTM with 28, LSTM with 56, LSTM with 128, and LSTM with 256 hidden nodes.

In addition, our experiment investigates 4 alternatives of how SR and SSC stages are connected as discussed in 2: LO using one-hot coding is straight forward. Our experiment uses LE with the embedding vector of size 56. PV using a penultimate vector of SR stage is also straight forward. FV uses the last convolution layer of the SR stage as an input of the SSC stage. Evaluation of SSC and the entire system was repeated 5 times.

We have also employed a rule-based approach as a baseline (Figure 2) to have others compared against. The rule-based approach employs CNN—the same structure and weights as one using by other approaches—to provide a sequence of input signs from a video clip. Then, the sequence of input signs is translated to a Thai word—a sequence of Thai alphabets using prespecified rules. The rules are that (1) a subsequence of consecutive signs of the same label whose length is longer than a prespecified number ( in our experiment) will be transcribed as an output sign of that label; otherwise the subsequence is ignored; (2) to translate from the output signs to the corresponding alphabet, a sequence of output signs is greedy-wise matched to the signs-to-alphabet mapping table. Noted that, there are some alphabets corresponding to single sign and some alphabets corresponding to series of multiple signs.

The base-line greedy scheme is to map the output-sign sequence to alphabets such that the mapping proceeds from the beginning of the sequence and tries to match the longest subsequence possible; then proceeds along the sequence to the position after the matched subsequence until the sequence is exhausted. When no match is possible even for the shortest subsequence (length 1) the subsequence is discarded and the translation proceeds to the position after.

# Evaluation.

Alphabet Error Rate (AER) is used as a metrics for the entire system assessment. Inspired by Word Error Rate (WER)[22], AER measures the minimum number operation, i.e., substitution, deletion, and insertion to make a word under evaluation match the reference. The AER is calculated as follows:, where is a number of substituted alphabets; is a number of deleted alphabets; is a number of inserted alphabets and is a total number of alphabets in the reference word. Stagewise, the sign-recognition stage is tested on the image dataset for its accuracy. The ALS and SSC stages are tested on the video dataset for their F-score and accuracy, respectively.

## Experimental Results.

Table 2 show test results of the ALS stage in F-score. The last column shows percentage of improvement using WFS over not using WFS. All options have been shown to perform well with over F-score 0.8. D2M delivered the best performing results either with or without WFS.

WFS has been shown to contribute from to improvement on any distance-based approach, but shown to contribute very little on HM approach. This may be explained by that HM has its own intrinsic smoothing effect (from gaussian basis) and therefore extra smoothing effect may be just redundant.

In addition, D2M and D2S employ moving-average and skip-frame schemes, respectively. These schemes also provide some degree of smoothing effect and their superior performance to their counterpart D2T may be attributed to their smoothing effect as well.

**Table 2** Alphabet-sepration results.

|  |  |  |  |
| --- | --- | --- | --- |
|  | F-score | | Improvement |
|  | Without WFS | With WFS | % |
| D1 | 0.825 | 0.851 | 2.6 |
| D2T | 0.826 | 0.857 | 3.1 |
| D2M | 0.861 | 0.886 | 2.5 |
| D2S | 0.849 | 0.865 | 1.6 |
| HM | 0.838 | 0.842 | 0.4 |

The SR stage is found to have a test accuracy of . Table 3 shows evaluation results of the SSC stage. The table presents each approach with its best performing hyperparameters—hidden size— along with its accurracy: mean, median, and 1st and 3rd quartiles. Interestingly, despite both deriving their input from sign labels, LO—which straightforwardly uses one-hot coding to represent a sign label— is clearly outperformed by LE—which employs word embedding to represent a sign label. We hypothesize that word embedding may provide an easier (numerically) digestible representation. More practical point is that how SR and SSC stages connect hugely effects the transcription performance. LO and LE, which simply take sign labels from SR and pass them to SSC, were more than outperformed by PV and FV—both reach further into SR stage and take either penultimate or feature vector for SSC input. This indicates a crucial fact that simply taking conventional output from an earlier stage and pass it as an input to the following stage may significantly sacrifice an overall performance. In addition, a sign label, pernultimate vector, and feature vector are in order processed information backwardly along SR stage. Our results may indirectly provide a peek into how information is lost along the deep neural processing path. Since FV and PV show similar performance, this may imply good information preservation from the last convolution layer—providing a feature vector— to the penultimate layer— providing a penultimate vector. As performance drops significantly as we use a label instead of a penultimate vector, this may imply a substantial loss of information along the way from a penultimate layer to a softmax layer to the final label decision, which may have been done through an function. Table 4 shows the overall performance of difference approaches. Table 4 presents each approach with its best performing hyperparameters—hidden size— along with its average AER over 5 repeats. The Benefit columns present the improvement of using WFS technique over not using it.

Our results reveal that a combination of D2M and smoothing for alphabet separation and FV for sign-sequence classification leads to the best performing transcription (AER 0.256). Table 5 provide an example of how prediction outputs look like. It is clearly seemed that WFS helps reduce alphabet duplication significantly.

Affirming the stagewise results, using word-embedding seems to provide an extra beneficial effect on the transcription as shown by LE outperforming LO in both stage-level accuracy and comparable pipeline AER (most noticeable at improvement on D1, with WFS). Noted that we speculate a marginal benefit of using word embedding with feature or penultimate vectors. However, the decisive conclusion may require a dedicated study.

**Table 3** Sign Sequence Classification Results.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Transcription Approach | Hidden Size | Transcription Accuracy | | | |
| Mean | Median | Q1 | Q3 |
| LO | 256 | 0.744 | 0.75 | 0.74 | 0.75 |
| LE | 128 | 0.79 | 0.79 | 0.78 | 0.8 |
| FV | 128 | 0.894 | 0.91 | 0.89 | 0.92 |
| PV | 256 | 0.924 | 0.92 | 0.91 | 0.97 |

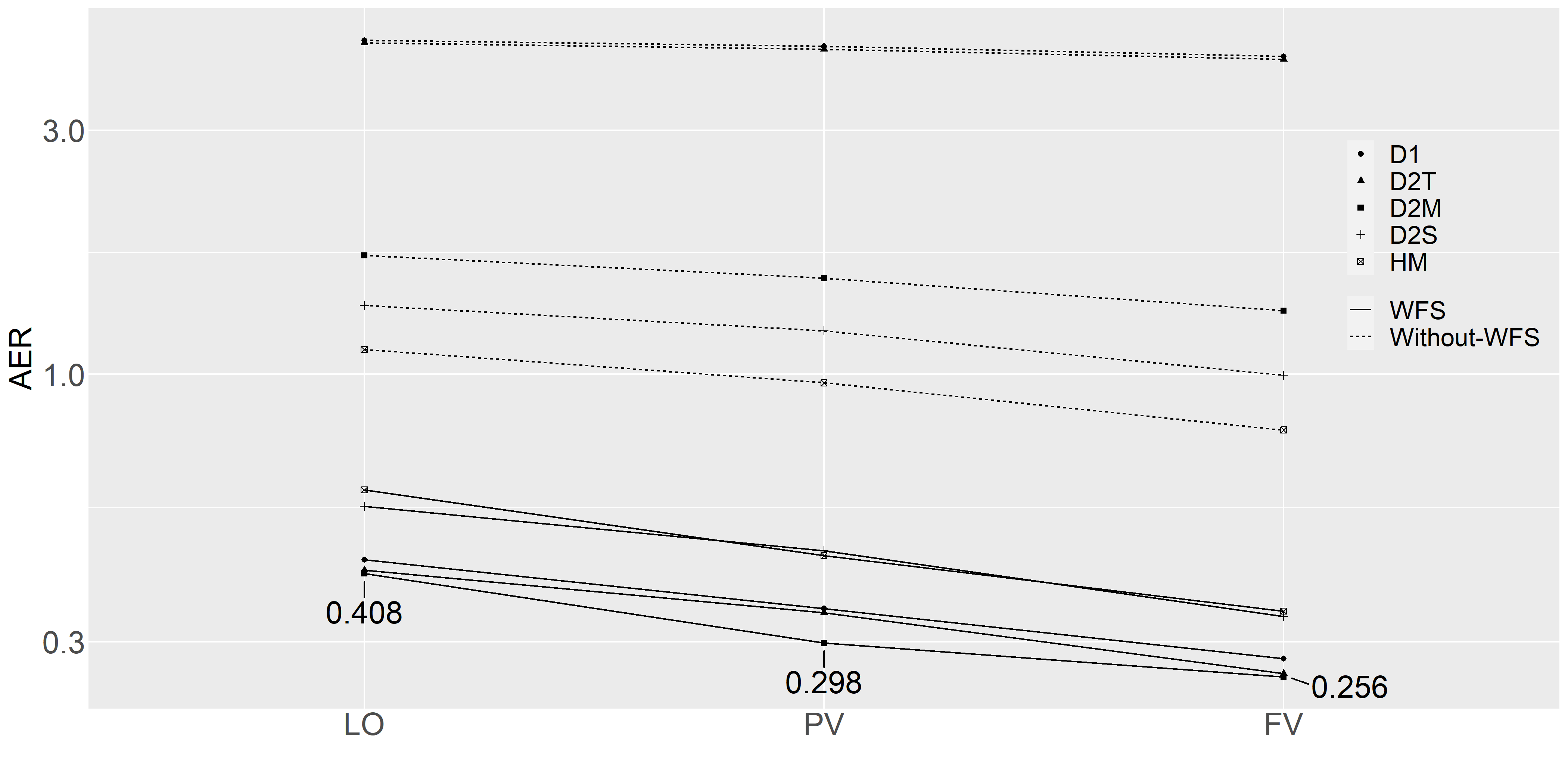
**Table 4** The Overall Performance of the Entire Pipeline.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Without WFS | | | | With WFS | | | | | Benefit of WFS (over not using WFS) | | |
|  | LE | LO | FV | PV | LE | LO | FV | PV | LE | LO | FV | PV |
| D1 | 4.49 | 4.5 | 4.186 | 4.374 | 0.342 | 0.434 | 0.278 | 0.348 | 13.129 | 10.369 | 15.058 | 12.569 |
| D2T | 4.432 | 4.446 | 4.13 | 4.318 | 0.34 | 0.414 | 0.26 | 0.342 | 13.035 | 10.739 | 15.885 | 12.626 |
| D2M | 1.658 | 1.708 | 1.332 | 1.542 | 0.344 | 0.408 | 0.256 | 0.298 | 4.820 | 4.186 | 5.203 | 5.174 |
| D2S | 1.328 | 1.364 | 0.996 | 1.216 | 0.514 | 0.552 | 0.336 | 0.452 | 2.584 | 2.471 | 2.964 | 2.690 |
| HM | 1.052 | 1.118 | 0.778 | 0.962 | 0.528 | 0.594 | 0.344 | 0.442 | 1.992 | 1.882 | 2.262 | 2.176 |
| Baseline | | 0.63 | | | | | | | | | | |

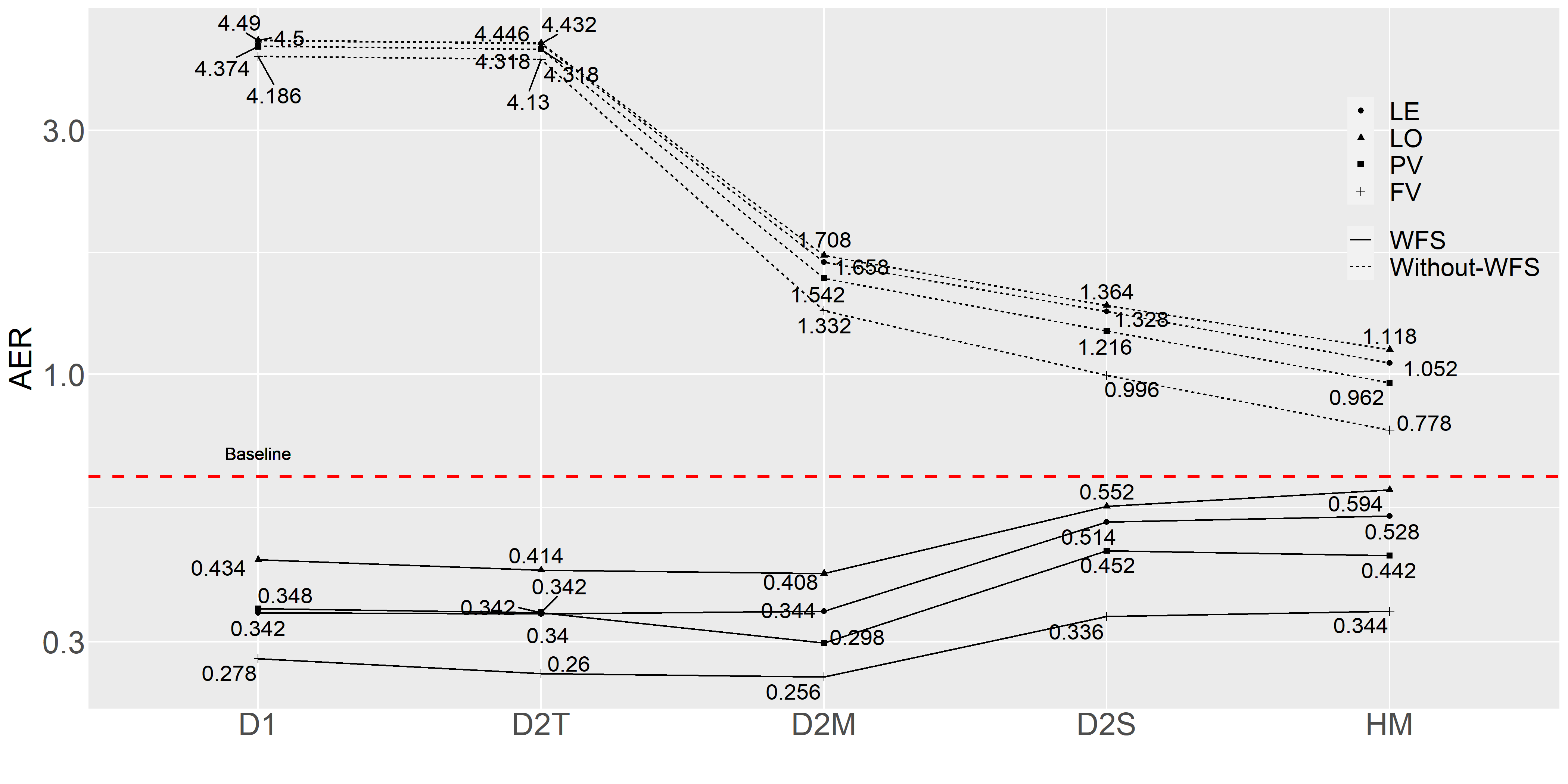
Table 5 Example of prediction results from D2M when using a feature vector.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| FV with D2M | | Without WFS | | With WFS | | Baseline | |
| Sample | GT | Prediction | AER | Prediction | AER | Prediction | AER |
| Data\_63 | ธน | ตหชน | 1.5 | ชน | 0.5 | ตต | 1 |
| Data\_84 | ชล | ชหชกลล | 2 | ชล | 0 | ล | 0.5 |
| Data\_196 | ชพ | คชหชพพ | 2 | ชพ | 0 | หพ | 0.5 |
| Data\_182 | ธง | ตหกนง | 2 | ชง | 0.5 | - | 1 |
| Data\_26 | ธร | ตหธร | 1 | ธร | 0 | ร | 0.5 |
| Data\_111 | ชอ | ภงหชอ | 1.5 | ชอ | 0 | อ | 0.5 |
| Data\_117 | ชว | ชงหกว | 1.5 | ชว | 0 | ว | 0.5 |

As discussed earlier, taking information deeper into the SR stage seems to allow the SSC stage and the entire pipeline to perform better (as AERs from using FV are lower than ones from using PV and AERs of PV are lower than ones of LO). Figure 6 illustrates this point. Apparently from Table 4, smoothing (WFS) is beneficial in every case, with larger effect on D1 and D2T, both do not have moving average or skip-frame mechanism. With WFS, all approach outperform the baseline as shown in Figure 7.



**Figure 6** Three alternative AER.



**Figure 7** AER Comparing of Sign transcription approach with different feature input.

Sign sequence separation Discussion.

For alphabet separation, we conjectured 2 possible indicators, i.e., (1) that distance between centroids of hands in consecutive frames was small when the 2 consecutive frames contain the same alphabet and the distance was large otherwise; (2) that posing time duration of the alphabet was long comparing to a transition time. These 2 conjectures were examined as the following.

**Assumption 1** is that a distance between centroids of hands in consecutive frames is small when the 2 consecutive frames contain the same alphabet and the distance is large otherwise. That is, two consecutive frames are corresponding to the same alphabet when their centroids of hands appearing in both frames locate close to each other. Figure 3 illustrates the rationale.

Assumption 1 Verification. To evaluate assumption 1,212 signing video clips were hand-marked

for signs and non-signs on all image frames. Given that all clips correspond to 2-gram words, sign and non-sign labels along with frame orders in the sequence can be used to group frames into 3 categories: signing frame of the 1st alphabet (called “First Alphabet”), signing frame of the 2nd alphabet (called “Second Alphabet”), and non-signing frame (called “Transition”). Then, a centroid of a hand in each frame can be located and distance between centroids in two consecutive frames can be computed using Euclidean distance. Figure 8 shows the distances in boxplots.

Figure 8 has the y-axis representing distance in a number of pixels. Due to possibility of variation in a signer’s hand sizes and shapes and visual sizes due to a camera effect, we derive a measure, which is relative to a signer’s hand size. Given a signer’s hand contained in a bounding box of pixels (hand sign #11 in [1]), the diagonal is pixels. Our proposed measure is a percentage of the underlying distance (in pixels) to the diagonal length of the signer’s hand (also in pixels). For example, an average distance of the 1st alphabet is pixels, which is equivalent to (of the diagonal length of the palm posed as sign #11 in [1]).

Figure 8 shows distances of the three groups. Apparently, the distances in the alphabet groups (average 2.3% and 1.6% in the First Alphabets and the Second Alphabets, respectively) are much smaller than the distance in the Transition group (average 14.35%). This finding strongly supports the Assumption 1.

**Assumption 2** is that posing time duration of the alphabet is long comparing to a transition time.

**Assumption 2 Invalidation.** To evaluate Assumption 2, the same dataset as in Assumption 1 evaluation is used. Time duration can be inferred from a number of consecutive frames labels as a particular type. For example, frames No.16 to No.30 all are marked as the first alphabet. The time duration of the first alphabet is 15 frames (@ 29 fps, 15/29 = 0.517 seconds).

Figure 9 shows time duration of each group. While transition time is quite short (17/29 = 0.586 seconds on median) compared to time duration of the Second Alphabets (24/29 = 0.828 seconds on median), it is still difficult to distinguish the transition time from time duration of the First Alphabets. This finding is against the Assumption 2, especially when considering the First Alphabets and Transition. Therefore, as Assumption 2 is invalidated, we discard time duration as a factor from our alphabet-separation approach. We emphasize this point, since we have found that Assumption 2 is a common misconception perceived by many of our peers.

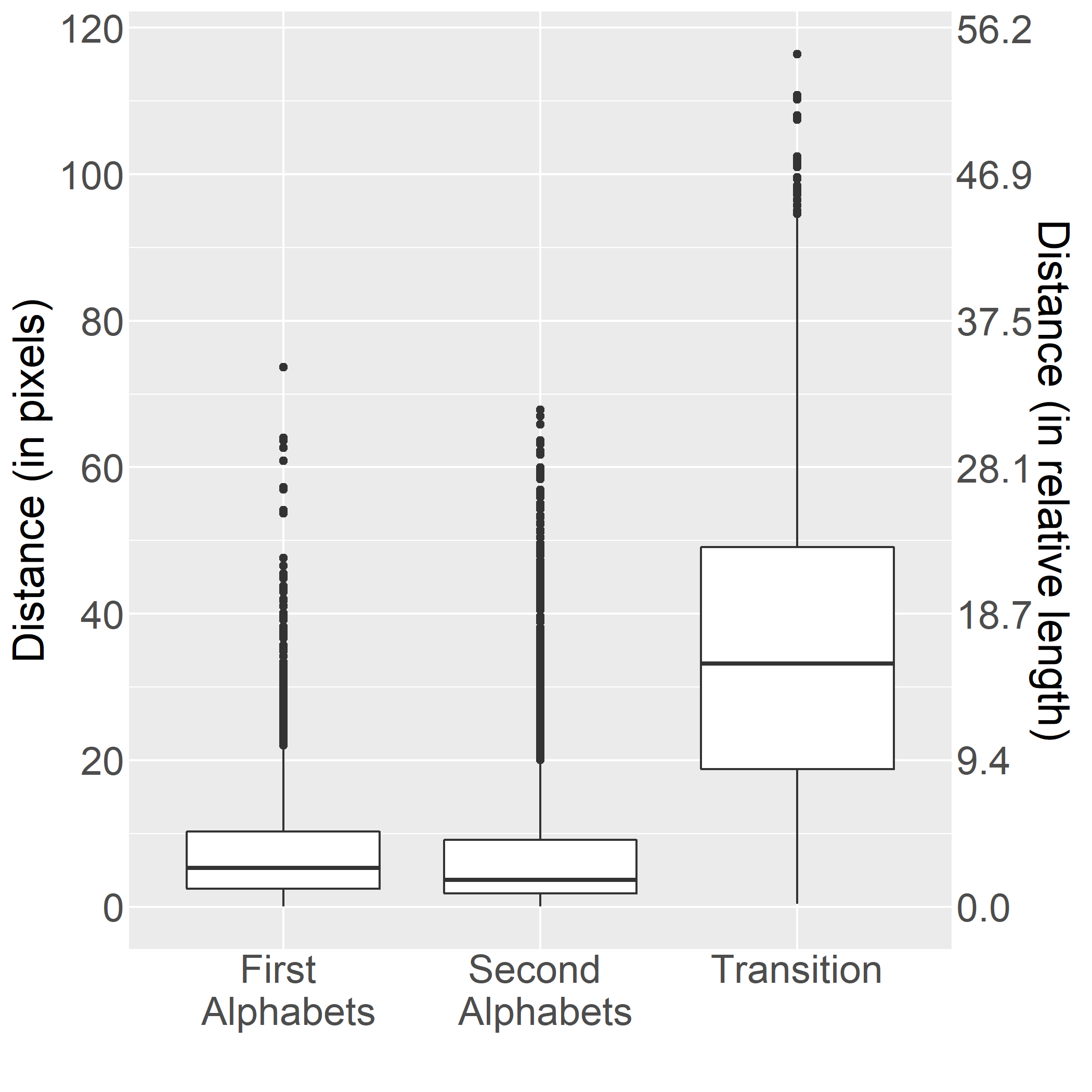


Figure 8 Boxplots of distance between centroids in consecutive frames by groups. Median distances: 5.32 pixels (2.49%), 3.7 pixels (1.74%), 33.18 pixels (15.55%) for First Alphabets, Second Alphabets, and Transition, respectively.

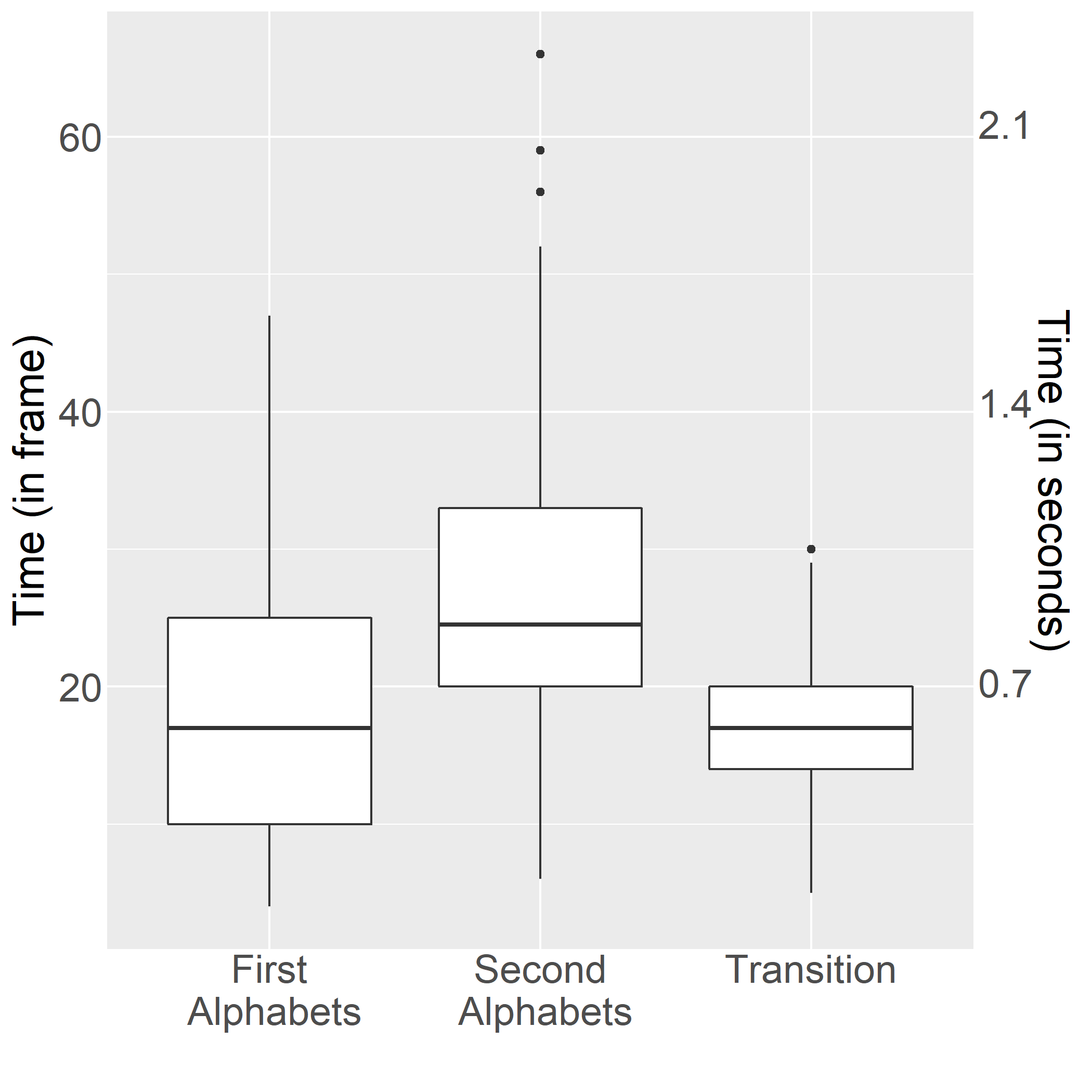


Figure 9 Time duration of signing per category. Median times: 17 (0.59 secs), 24.5 (0.48 secs), 17 (0.59 secs) for First Alphabets, Second Alphabets, and Transition, respectively.

# Conclusion and Discussion

In this article, we proposed an Automatic Thai Finger Spelling Transcription (ATFS), using a three-stage transcription process as well as a Window Frame Smoothing (WFS) mechanism to improve overall performance of ATFS. The first stage, called alphabet-separation (ALS) stage, takes a video clip and provides frame markers to indicate which frames belong to the same alphabet and which frames are non-signing and should be discarded. The D2M alternative, which uses distance between centroids of hands area in two successive frame along with double thresholding and moving average, has been found to be superior to other alternatives with F-score of (without WFS) and (with WFS). The second stage, called Sign-recognition (SR) stage, takes an image frame and outputs a TFS sign (or a vector associating to a sign). This stage was implemented with CNN following [1] and found to be accurate. The final stage, called Sign-sequence classification (SSC) stage, takes a sequence of signs (or a vector associating to a sign) and outputs a Thai word (a sequence of Thai alphabets). Stagewise, SSC stage using penultimate vector(PV) and feature vector(FV) is shown to outperform other option with PV median accuracy of and FV median accuracy of . Among the investigated approaches, D2M with WFS using FV is found to deliver the best entire-pipeline performance, AER of . Our findings reveal a benefit of WFS mechanism in both stagewise and overall performance, with possible improvement upto . To evaluate TFS, which has unique characteristics of employing a multi-posture scheme, we employ Alphabet Error Rate (AER). AER is an alphabet version of Word Error Rate (WER). The result of the overall system shows that all approaches with WFS have been outperforming over other approach. The best performance is D2M with WFS which takes the CNN feature vector as input provided the AER of . In addition, our findings may provide some insight into where information might have lost along the neural processing path. The information seems to be well intact as it passes from the last convolution layer to the penultimate layer. However, the information seems to be substantially lost somewhere along the path from the penultimate layer to softmax layer and then to the final class decision.

Interestingly, although both representing a sign with one-hot coding and representing a sign through word embedding are base on a sign, word-embedding approach (LE) apparently outperforms one-hot-coding approach (LO) stagewise and overall. We speculate that word-embedding may have provided a computationally easier representation than that of the straightforward one-hot coding. This observation has brought curiosity whether word-embedding could help feature or penultimate vector on the similar matter. This may worth a dedicated study. Noted that, our ATFS relies on an alphabet-separation stage to mark out any non-signing frame. Our previous studies [4], [23] have discovered a promising approach—Latent Cognizance (LC)— to identify an out-of context question, which in this case is a non-sign image. Employing LC in cleaning up non-signing images from a sequence of image frames has not been explored.

**References**

[1] P. Nakjai and T. Katanyukul, ‘Hand sign recognition for thai finger spelling: an application of convolution neural network’, *Journal of Signal Processing Systems*, vol. 91, no. 2, pp. 131–146, Feb. 2019, doi: 10.1007/s11265-018-1375-6.

[2] P. Nakjai, P. Maneerat, and T. Katanyukul, ‘Thai finger spelling localization and classification under complex background using a yolo-based deep learning’, in *ACM international conference proceeding series*, 2019, pp. 230–233.

[3] J. Redmon and A. Farhadi, ‘YOLO9000: Better, faster, stronger’, in *Proceedings - 30th IEEE conference on computer vision and pattern recognition, CVPR 2017*, Jul. 2017, vol. 2017-January, pp. 6517–6525, doi: 10.1109/CVPR.2017.690.

[4] P. Nakjai and T. Katanyukul, ‘Automatic hand sign recognition: Identify unusuality through latent cognizance’, in *Lecture notes in computer science LNAI*, vol. 11081 LNAI, Springer Verlag, 2018, pp. 255–267.

[5] K. Simonyan and A. Zisserman, ‘Very deep convolutional networks for large-scale image recognition’, *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, Sep. 2015, [Online]. Available: http://arxiv.org/abs/1409.1556.

[6] S. Liwicki and M. Everingham, ‘Automatic recognition of fingerspelled words in british sign language’, in *CVPR*, Jun. 2009, pp. 50–57.

[7] C. Oz and M. C. Leu, ‘American Sign Language word recognition with a sensory glove using artificial neural networks’, *Engineering Applications of Artificial Intelligence*, vol. 24, no. 7, pp. 1204–1213, Oct. 2011, doi: 10.1016/j.engappai.2011.06.015.

[8] L. Ding and A. M. Martinez, ‘Modelling and recognition of the linguistic components in American Sign Language’, *Image and Vision Computing*, vol. 27, no. 12, pp. 1826–1844, Nov. 2009, doi: 10.1016/j.imavis.2009.02.005.

[9] M. Hrúz *et al.*, ‘Automatic fingersign-to-speech translation system’, *Journal on Multimodal User Interfaces*, vol. 4, no. 2, pp. 61–79, Jul. 2011, doi: 10.1007/s12193-011-0059-3.

[10] N. El-Bendary, H. M. Zawbaa, M. S. Daoud, A. E. Hassanien, and K. Nakamatsu, ‘ArSLAT: Arabic sign language alphabets translator’, in *2010 international conference on computer information systems and industrial management applications (CISIM)*, Oct. 2010, pp. 590–595, doi: 10.1109/CISIM.2010.5643519.

[11] M. Mohandes, S. Aliyu, and M. Deriche, ‘Prototype Arabic Sign language recognition using multi-sensor data fusion of two leap motion controllers’, in *2015 IEEE 12th international multi-conference on systems, signals & devices (SSD15)*, Mar. 2015, pp. 1–6, doi: 10.1109/SSD.2015.7348113.

[12] A. Ben Jmaa, W. Mahdi, Y. Ben Jemaa, and A. Ben Hamadou, ‘A new approach for hand gestures recognition based on depth map captured by RGB-D camera’, *Computacion y Sistemas*, vol. 20, no. 4, pp. 709–721, 2016, doi: 10.13053/CyS-20-4-2390.

[13] W. Jiangqin and G. Wen, ‘The recognition of finger-spelling for chinese sign language’, in *LNAI*, vol. 2298, Springer Verlag, 2002, pp. 96–100.

[14] N. Mukai, N. Harada, and Y. Chang, ‘Japanese fingerspelling recognition based on classification tree and machine learning’, in *Proceedings - 2017 NICOGRAPH international, NICOInt 2017*, Sep. 2017, pp. 19–24, doi: 10.1109/NICOInt.2017.9.

[15] H. T. C. Machacon and S. Shiga, ‘Recognition of Japanese finger spelling gestures using neural networks’, *Journal of Medical Engineering and Technology*, vol. 34, no. 4, pp. 254–260, May 2010, doi: 10.3109/03091900903580520.

[16] M. Suwanarat and C. Reilly, ‘The thai sign language dictionary. Volume 1’, 1986, [Online]. Available: https://eric.ed.gov/?id=ED275204.

[17] D. Guo, W. Zhou, H. Li, and M. Wang, ‘Hierarchical LSTM for sign language translation’, in *32nd AAAI conference on artificial intelligence, AAAI 2018*, USA, Apr. 2018, pp. 6845–6852, [Online]. Available: www.aaai.org.

[18] C. Zhou, C. Sun, Z. Liu, and F. C. M. Lau, ‘A c-lstm neural network for text classification’, Nov. 2015, [Online]. Available: http://arxiv.org/abs/1511.08630.

[19] I. Sutskever, O. Vinyals, and Q. V. Le, ‘Sequence to sequence learning with neural networks’, *Advances in Neural Information Processing Systems*, vol. 4, no. January, pp. 3104–3112, Sep. 2014.

[20] X. Tian and et al., ‘Deep LSTM for large vocabulary continuous speech recognition’, 2017.

[21] N. Laokulrat *et al.*, ‘Generating video description using sequence-to-sequence model with temporal attention’, 2016.

[22] A. Laurent and et al., ‘Improving recognition of proper nouns in ASR through generating and filtering phonetic transcriptions’, *Computer Speech and Language*, vol. 28, no. 4, pp. 979–996, 2014.

[23] P. Nakjai, J. Ponsawat, and T. Katanyukul, ‘Latent cognizance: What machine really learns’, in *ACM international conference proceeding series*, New York, New York, USA, 2019, pp. 164–170.