Analyzing the Potential of Linguistic Features for Sign Spotting: A Look at Approximative Features

Natalie Hollain
Institute for Computing
and Information Sciences
Radboud University
natalie.hollain@ru.nl

Martha Larson
Center for Language Studies
and Institute for Computing
and Information Sciences
Radboud University
martha.larson@ru.nl

Floris Roelofsen
Institute for Logic,
Language and Computation
University of Amsterdam
f.roelofsen@uva.nl

Abstract

Sign language processing is the field of research that aims to recognize, retrieve, and spot signs in videos. Various approaches have been developed, varying in whether they use linguistic features and whether they use landmark detection tools or not. Incorporating linguistics holds promise for improving sign language processing in terms of performance, generalizability, and explainability. This paper focuses on the task of sign spotting and aims to expand on the approximative linguistic features that have been used in previous work, and to understand when linguistic features deliver an improvement over landmark features. We detect landmarks with Mediapipe and extract linguistically relevant features from them, including handshape, orientation, location, and movement. We compare a sign spotting model using linguistic features with a model operating on landmarks directly, finding that the approximate linguistic features tested in this paper capture some aspects of signs better than the landmark features, while they are worse for others.

1 Introduction

Sign Language Processing (SLP) (Bragg et al., 2019; Moryossef and Goldberg, 2021) is the field of research that studies how signs and signed phrases can be recognized, retrieved and spotted in videos. Key approaches differ with respect to whether they attempt to leverage linguistics/phonology, and the way in which they do it, as shown in Figure 1. Some recent work used pixel information as input (Approach **A**) without explicitly considering linguistic features that are relevant for sign language (e.g. handshape and orientation of the hand) (Momeni et al., 2020; Jiang et al., 2021). On the other hand, earlier work in sign language recognition proposed methods to extract phonological properties of signs from pixel information (Bowden et al., 2004; Von Agris et al., 2008; Han et al., 2009; Zaki and Shaheen, 2011) (Approach **B**). Other approaches have applied a landmark detection tool, such as OpenPose, to obtain the location of landmarks in the body from the pixel input and used them to train a model (Ko et al., 2018; Ko et al., 2019) (Approach **C**). Angle and distance features which approximate the phonological properties of a sign have also been extracted from these landmarks (Shin et al., 2021; Hussain et al., 2022; Farhan and Madi, 2022) (Approach **D**). SLP research seeks to use approximative features where possible to avoid the computational overhead of calculating features that reflect linguistic properties exactly. An approximation is considered sufficiently good if it contributes to the performance of an SLP system.

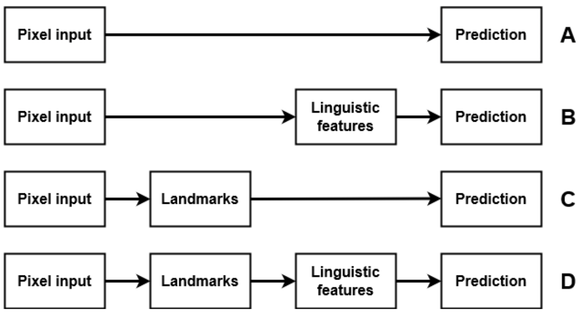


Figure 1: Four methods for sign language processing

Incorporating linguistic features holds great promise for improving SLP in terms of generalizability and explainability. The drawback of incorporating linguistic features based on pixel information, as in Approach B, is that this method is sensitive to particular properties of the training data, such as the lighting conditions, the skin colour of the signer, the color and shape of the signer's clothes, and the recording background. Approach D, which we pursue here, improves on this by implementing a modular approach which is potentially more robust because linguistic features are extracted from landmarks rather than pixel input.

The purpose of this paper is to move research adopting Approach D beyond the current state of the art. We make two contributions. First, we expand the inventory of approximative linguistic features that are used for SLP. Second, we seek to understand when linguistic features deliver an improvement over the landmark features from which they are produced. In contrast, previous work solely focused on the ability of linguistic features to improve performance of SLP systems and did not examine what makes these features important.

The reason why linguistic features extracted from landmarks can be anticipated to be more robust is that the modularity of this approach makes it possible to use existing tools for landmark detection, such as Mediapipe (Zhang et al., 2020), OpenPose (Cao et al., 2017), or MMPose (Sengupta et al., 2020). Mediapipe, for instance, has been trained on a large-scale, in-the-wild dataset (as well as curated and synthetic data) with high variability in background, lighting conditions, the skin colour of subjects, and other visual artifacts. In addition, the modular nature of the approach makes it straightforward to incorporate future improvements of landmark detection technologies as they become available.

This paper focuses on the task of sign *spotting*, which has as its goal to determine when a given target sign occurs in a video of continuous signing. Sign spotting is distinct from sign recognition, because we need to establish *when* a sign occurs in a given video. Recognition only uses video segments in which one isolated sign is performed.

Building on the aforementioned work in sign language recognition (SLR) (Shin et al., 2021; Hussain et al., 2022; Farhan and Madi, 2022), we first detect landmarks with Mediapipe and then extract linguistically relevant features from these landmarks. In the extraction phase, we expand on previous work in that we do not only extract features that serve as an approximate representation of the handshape of the signer, but also features that correspond to other relevant properties, such as the orientation of the hand, its location relative to the body, and its movement through space.

We compare a sign spotting model which makes use of these approximative linguistic features with one that operates on landmarks directly (approach **D** and **C**, respectively). We find that the approximate linguistic features tested in this paper capture some aspects of signs better than the landmark features, while they are worse for others. Our code is made available on Github[1].

2 Background

2.1 Sign language phonology and phonetics

Sign language phonology studies the articulation of signs within and across different sign languages. Typically, the phonological properties of a sign are split up into manual and non-manual properties. Non-manual properties pertain to the face, in particular the mouth, and the signer's body posture (Pendzich, 2020). Manual phonological properties pertain to the shape, orientation, location and movement of the signer's hands (Stokoe, 1960; Battison, 1978; Van der Kooij, 2002; Sandler, 2012; Brentari et al., 2018; Brentari, 2019). We focus here on manual phonological properties.

The phonology of a sign is not the only factor that influences how the sign is articulated in reality. The specific characteristics of the signer, such as their emotional state, language background, age and gender, can change how signs are performed in practice. Moreover, the linguistic context in which the sign is uttered, in particular the previous and subsequent sign, is an important factor in a sign's articulation. The concrete realisation of signs, as influenced by these factors and more, is studied in the field of sign language phonetics (Crasborn, 2012; Tyrone, 2020). In this work, we focus on the basic phonological parameters that we introduced above, leaving the study of phonetics to future work.

2.2 Sign spotting

We give a brief overview of notable work on sign spotting before describing what distinguishes our

[1]https://github.com/nataliehh/Linguistic-Features-for-Sign-Spotting

work from what has already been done. A variety of methods have been applied in previous work, including dynamic time warping (Viitaniemi et al., 2014), conditional random fields (Cho et al., 2009; Yang and Lee, 2010), hierarchical sequential patterns (Ong et al., 2014) and hidden Markov models (Elmezain et al., 2008). Typically, these approaches were applied to datasets that only contained a small set of signs and signers. More recently, the focus has been on the application of deep learning methods, such as 3D convolution (Jiang et al., 2021; Wong et al., 2023; Enrıquez et al., 2022).

We highlight in particular the work of Momeni et al. (2020), which proposed a framework for continuous sign spotting called 'watch, read and lookup'. A model was trained to create sign spotting embeddings using sparsely annotated videos and examples from a video dictionary of signs. The authors use BSL-1k, a dataset that contains videos of BBC broadcasts that have been interpreted in sign language. Interpreted signing is distinct from 'natural signing', the latter being faster and less distinctly signed (Bragg et al., 2019).

Our work contrasts with current approaches to sign spotting, which either used ad-hoc datasets or operated directly on pixel input. We use a dataset which matches most of the criteria described by Bragg et al. (2019). Furthermore, although linguistic features have been used for other SLP tasks, we are the first to our knowledge to investigate their potential for sign spotting.

3 Method

3.1 Data

We use the Corpus Nederlandse Gebarentaal (CNGT) (Crasborn and Zwitserlood, 2008; Crasborn et al., 2008) to train our sign spotting model. It contains 72 hours of video footage of 104 signers conversing in Dutch Sign Language (NGT), recorded at 25 fps. Circa 15% of the corpus is annotated, which is equivalent to 162k annotations of 3.2k unique signs. CNGT is annotated using NGT Signbank (Crasborn et al., 2014), which contains information about the phonological properties of signs discussed in Section 2.1.

The corpus consists of videos of 'natural' signing, where signers are in conversation and are not signing in a more proper manner than usual (Crasborn and Zwitserlood, 2008). The dataset contains footage that is compatible with real-world applications (Bragg et al., 2019), and contains a large amount of different signs and signers. Thus, CNGT forms a good basis for SLP applications.

We prepare the data of CNGT for our model as follows. First, we split the annotations into a train, validation and test set. We ensure that the training set does not contain the same signers as the validation or test set to make our system signer-independent. We filter out signs which are not seen during training, as well as signs for which no linguistic information is available in the NGT Lexicon in Signbank, since we require such information for our performance analysis. After this preprocessing step, 118k annotations of 2.7k unique signs remain. We use a data split of approximately 80/10/10, with 90k train, 10.5k validation and 9.5k test annotations.

To create more variety in the training set, we augment it by mirroring the footage. This is done to ensure that one-handed signs occur signed with both the right and the left hand. Similarly, two-handed signs where one hand is dominant now also occur with each hand being dominant. After the augmentation, we have 180k train instances. We found that our model converges more consistently with this augmentation than without it.

Due to the fact that signs have variability in how long they are signed, the annotations in our dataset are of variable length. Thus, to make the input compatible with a neural network architecture, we ensure that our inputs are transformed to a fixed length. We select a target length of 10 frames, which is equal to the mean duration of the annotations in the corpus. Annotations that are shorter than 10 frames are simply padded with zeros to the target length. For annotations that are too long, we undersample to 10 frames.

3.2 Landmark detection

For each frame of our dataset, we detect landmarks on the hands and body using Mediapipe. Each hand has 21 landmarks, as shown in Figure 2. While Mediapipe is capable of estimating the x, y, z coordinates of each landmark, the z coordinate is less reliable. As such, we only make use of the 2D coordinates, x and y.

Mediapipe normalises landmarks using the video dimensions (width and height), which means that landmark coordinates are not comparable across videos with different dimensions. Therefore, we reverse the dimension-wise normalisation

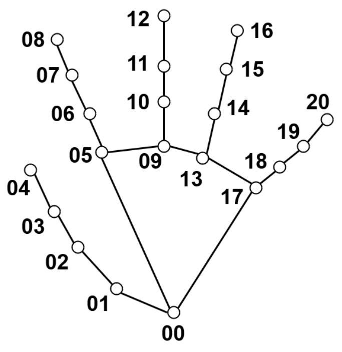


Figure 2: The 21 Mediapipe landmarks for one hand

to convert them back to pixel coordinates. In other words, given a landmark with normalised coordinates $[x, y]$ in a video with dimensions w, h, we perform the operation $[x \cdot w, y \cdot h]$.

After reversing the normalisation, we apply the normalisation described by Celebi et al. (2013). For each frame, we obtain the landmarks of the left and right shoulder, sh_L and sh_R. Every landmark ℓ at a given frame is then normalised as follows:

- We scale ℓ using the absolute distance between the shoulders: $\dfrac{\ell}{\mathrm{abs}(\mathrm{sh}_L - \mathrm{sh}_R)}$
- We center ℓ by subtracting the midpoint of the shoulders: $\ell - \dfrac{\mathrm{abs}(\mathrm{sh}_L + \mathrm{sh}_R)}{2}$.

For our model that uses landmark features, we simply use the normalised landmarks of both hands as our input, or $21 \cdot 2 = 42$ landmarks. Each landmark consists of an x and y coordinate, such that we use $42 \cdot 2 = 84$ features for each frame by using all of the coordinates as features. Thus, each annotation results in a data input of shape $(10, 84)$.

3.3 Linguistic features

To represent the basic phonological parameters, we extracted the following types of features from the normalised landmarks:

- Handshape: the distances and angles between the fingertips, handpalm and wrist.
- Orientation: the angle of the handpalm relative to the torso and the shoulders.
- Location: the x, y coordinates of the wrist(s) and fingertips.
- Movement: the velocity of the wrist.

The handshape angle features are computed using a start, middle and end point triple, (ℓ_s, ℓ_m, ℓ_e), and the arctangent measure:

$$\mathrm{angle}(\ell_s, \ell_m, \ell_e) = \mathrm{atan2}(\ell_{e,y} - \ell_{m,y}, \ell_{e,x} - \ell_{m,x}) - \mathrm{atan2}(\ell_{s,y} - \ell_{m,y}, \ell_{s,x} - \ell_{m,x})$$

where we indicate with the subscript whether the x, y coordinate of the element is used, e.g. $\ell_{1,x}$ indicates the x coordinate of landmark ℓ_1.

For each finger, we compute the angle with the wrist as well as its internal angle. For instance, we get the angle within the thumb using landmarks $[01, 02, 04]$ and get the thumb's angle with the wrist using $[00, 01, 04]$.

For the handshape distance features, we calculate the Euclidean distance between pairs of landmarks ℓ_1, ℓ_2:

$$\mathrm{dist}(\ell_1, \ell_2) = \sqrt{(\ell_{1,x} - \ell_{2,x})^2 + (\ell_{1,y} - \ell_{2,y})^2}$$

To compute the hand orientation, we use Mediapipe's Pose model, which captures the position of landmarks of the entire body. In particular, we use the landmarks of the left shoulder sh_L (pose index 11), right shoulder sh_R (index 12), left hip hip_L (index 23) and right hip hip_R (index 24). We draw two lines using these landmarks: the horizontal line between the shoulders, $(\mathrm{sh}_L, \mathrm{sh}_R)$, and the vertical line in the middle of the torso, $(\dfrac{\mathrm{sh}_L + \mathrm{sh}_R}{2}, \dfrac{\mathrm{hip}_L + \mathrm{hip}_R}{2})$. For the landmarks within the hand, we draw two axes within the hand: one between index 00 and 09, the y-axis, and one between 05 and 17, the x-axis.

Based on the lines that have been drawn for the shoulders, torso and hands, we now compute the slope of each line. For a given line ℓ that consists of a start and end point (ℓ_s, ℓ_e), we compute the slope s_ℓ of the line as:

$$s_\ell = \frac{\ell_{e,y} - \ell_{s,y}}{\ell_{e,x} - \ell_{s,x}}$$

Finally, we can compute the angle between two lines for which we computed the slopes, s_1, s_2:

$$\mathrm{angle}(s_1, s_2) = \arctan(\frac{s_2 - s_1}{1 + (s_2 \cdot s_1)})$$

The hand orientation is then represented by the angles between the x-axis and y-axis of the hand with the shoulders and with the torso. We do not compare the angle of the torso and shoulders, nor the x-axis and y-axis of the hand because these are not relevant for the orientation of the hand relative to the body. As such, we end up with four distinct orientation angles.

Feature Ind.	**Type**	**Ind. Mediapipe Landmarks**
0 – 24	Handshape (Angles)	[01,02,04], [00,01,04], [05,06,08], [00,05,08], [09,10,12], [00,09,12], [13,14,16], [00,13,16], [17,18,20], [00,17,20], **[02,03,04]**, **[05,06,07]**, **[06,07,08]**, **[09,10,11]**, **[10,11,12]**, **[13,14,15]**, **[14,15,16]**, **[17,18,19]**, **[18,19,20]**, **[04,00,08]**, **[08,00,20]**, **[16,17,20]**, **[08,05,12]**, **[04,05,20]**, **[08,13,20]**
25 – 39	Handshape (Distances)	[00,04], [00,08], [00,12], [00,16], [00,20], [04,08], [04,12], [04,16], [04,20], [08,12], [08,16], [08,20], [12,16], [12,20], [16,20]
40 – 43	Hand orientation	[00,09], [05,17] + Pose: [11, 12, 23, 24]
44 – 55	Wrist, fingertip locations	00, 04, 08, 12, 16, 20
56 – 58	Wrist velocity	00
59 – 117	Features other hand	*See features 0 – 58*
118 – 119	Distance between wrists	00

Table 1: Feature indices

The location of the hand is simply represented using the x, y coordinates of the wrist and the fingertips. To capture the movement of the hand, we compute the velocity of the wrist. We do this in three different ways: first, we compute the Euclidean distance between the location of the wrist at the current frame and the last frame. This is done in the same manner as for the handshape distance features. Second, we separately store the difference between the x coordinate of the wrist between these two frames. We do the same for the y coordinate to obtain the third feature. This way, we capture both an average velocity that combines the x, y coordinates, as well as the horizontal and vertical velocity.

Finally, we capture the horizontal and vertical distance between the wrists of the hands. These features are chosen because the location of a sign is partially characterised by the interaction between the hands. We compute the difference between the x and y coordinates, resulting in two features.

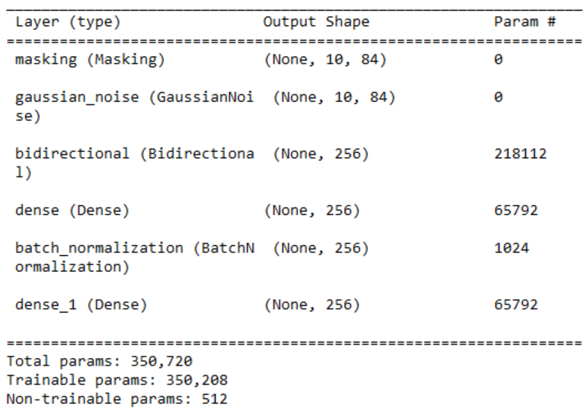
In Table 1, the extracted features are displayed. The *Ind. Mediapipe Landmarks* column shows which indices from the Mediapipe hand model are used, while the *Feature Ind.* column indicates the indices of our created features. Note that the shown indices are only for the left hand. The right hand's indices are equivalent modulo 59, e.g. the first feature of the right hand that computes the angle for landmark indices $[01, 02, 04]$, is at index 59. In total, we use 120 features to represent the phonological properties of both hands.

Some of the extracted features are adapted from previous work. **Bold** values indicate features taken from Farhan et al. (2022). All distance features are adopted from Shin et al. (2021). The remaining features are novel.

3.4 Model architecture

Based on Momeni et al. (2020), we develop a model which learns to create embeddings from our input features, such that inputs of the same sign result in similar embeddings while inputs of different signs result in dissimilar embeddings. The model that we chose for our experiments is a LSTM network. LSTMs can extract temporal information from data sequences and have been a popular tool for natural language processing (Chai and Li, 2019). While more sophisticated architectures are available these days, our goal is not to select the best model but rather to engineer meaningful features. We tested multiple configurations of our network and selected one which performs well for both the landmark and linguistic features. Our chosen configuration is shown in Figure 3.

We start with a masking layer to deal with our zero padding, followed by a Gaussian noise layer which creates variability in our data to make the model generalize better. We empirically found that a standard deviation of $\sigma = 0.001$ for the noise is suitable. It is followed by a biLSTM layer with $2 \cdot 128 = 256$ nodes, and two dense layers of size 256. We use batch normalization between the dense layers for training stability. A batch size of



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 masking (Masking)           (None, 10, 84)            0

 gaussian_noise (GaussianNoi  (None, 10, 84)           0
 se)

 bidirectional (Bidirectiona  (None, 256)              218112
 l)

 dense (Dense)               (None, 256)               65792

 batch_normalization (BatchN  (None, 256)              1024
 ormalization)

 dense_1 (Dense)             (None, 256)               65792

=================================================================
Total params: 350,720
Trainable params: 350,208
Non-trainable params: 512
```

Figure 3: Model architecture

128 and learning rate of 0.001 are used, inspired by Momeni et al. (2020). We train the model using the Adam optimizer for 10 epochs, which is when it typically starts to converge on the validation set. Due to the strength of contrastive loss reported in the literature (e.g. (Momeni et al., 2020)), we apply supervised contrastive loss to train our model (Khosla et al., 2020).

3.5 Experimental setup

Our experiments compare our expanded inventory of approximative linguistic features against a pipeline using only Mediapipe landmarks. Our main goal is to investigate when linguistic features contribute to sign spotting performance. We train two sign spotting models, one using the linguistic features extracted from the landmarks and the other using the landmark features directly. In order to test our models, we move a sliding window with the same size as our train inputs, 10 frames, over our test set videos. For each window and for each target sign, we compute their cosine distance d. The inventory of target signs consists of all 1038 signs present in the test set. If d is lower or equal to our spotting threshold τ, i.e. $d \leq \tau$, we say the target sign has been spotted. We report our results at $\tau = 0.2$ for both models, which we empirically found to be a good spotting threshold on the validation set.

Each target sign has been seen multiple times during training. As such, using each train embedding individually to find the spottings in a test video requires many comparisons. To reduce the number of comparisons, we create *reference embeddings* for each sign. For a sign S, we first compute embeddings of its training set occurrences. These embeddings are then compared to each other in terms of their cosine distance to each other. We investigate which the embeddings are, on average, closest to all other embeddings of S, and define them to be most representative of S. The top 10% most representative embeddings are averaged to make one reference embedding for S. The predicted spottings of a sign S can then be found using the reference embeddings.

3.6 Evaluation

In this section, we describe how we evaluate the two models. Recall that we aim to achieve insight into when linguistic features are contributing to sign spotting. To this end, our evaluation approach makes use of *confusable signs*: signs which only differ from a given target by a single phonological property. For instance, a pair of signs may only differ in where they are signed, in which case they form confusable signs for each other based on location. We call the single property that differs between the confusable signs the *Δ property*. By investigating which confusable signs are actually mistaken for a given target sign, we are able to discover which phonological properties are difficult to distinguish using each set of features.

We evaluate our sign spotting models by computing the true positive (TP), false negative (FN), true negative (TN) and false positive (FP) evaluations for each model. The FP and TN evaluations are computed by obtaining the confusable signs for each target sign that are present in our test set videos. The confusable signs are selected based on the linguistic properties provided by NGT Signbank (Crasborn et al., 2020).

We begin by analyzing the confusable signs for each target sign and determining their Δ properties. In Table 2, the frequency of the Δ properties in our test set is shown. Notably, a few Δ properties are much more common than others. This may be related to how many confusable signs exist with a particular Δ property. For example, there may be few signs for which only the handshape of the weak hand differs from another sign. Additionally, if confusable signs with a given Δ property are not common signs in our corpus, the Δ property will also not occur frequently.

The TP, FN, TN and FP instances are calculated using *tolerance to irrelevance* (TTI) (De Vries et al., 2004). This metric is based on the assumption that users, when given an entry point in an audio or video stream, keep listening or watching until their tolerance to irrelevant content has been

Δ **property**	**Test set frequency**
Alternating Movement	182
Contact Type	231
Handedness	4263
Handshape Change	299
Location	18078
Movement Direction	16566
Movement Shape	749
Orientation Change	568
Relation between Articulators	42
Relative Orientation: Location	1711
Relative Orientation: Movement	2839
Repeated Movement	1047
Strong Hand	35043
Weak Hand	85

Table 2: Frequency of Δ properties in our test set

reached. TTI is thus relevant to our evaluation, as sign spotting systems should reflect real-life applications (Bragg et al., 2019).

To capture a user's tolerance, TTI makes use of a *tolerance window* which allows for entry points to be located a bit before or at the start of the relevant content, but not after it has begun. The reasoning behind this decision is that it has been found to be annoying to users when entry points are given after the start of the relevant section (He et al., 1999).

We formalize TTI for our analysis as follows. For a given ground truth annotation s_j, a TP occurs when a prediction p_i, with onset time t_{p_i}, falls into its tolerance window:

$$t_{p_i} \in [t_{s_j} - tol, t_{s_j}]$$

where t_{s_j} is the onset time of s_j. In contrast, a FN occurs when no prediction falls into this tolerance window. The tolerance *tol* can be chosen depending on the exact context in which TTI is used. There are currently no established tolerance levels for SLP, thus, we consult the related field of audio segmentation for our tolerance. We found $tol = 0.5$ seconds to be a frequently used tolerance for audio (Aljanaki et al., 2015; Smith and Chew, 2013; Smith et al., 2011).

To compute the FP and TN instances, we obtain the confusable signs, $C(S)$, for each target sign S:

$$C(S) = \{A, B, ..., Z\}$$

We then determine when the confusable signs are annotated in CNGT:

$$ANN_{C(S)} = \{A_1, ..., A_m, ..., Z_1, ..., Z_n\}$$

Next, we select the onset time of each confusable sign annotation:

$$T_{ANN_{C(S)}} = \{t_{A_1}, ..., t_{A_m}, ..., t_{Z_1}, ..., t_{Z_n}\}$$

Based on the notation above, we can then define FP and TN evaluations. Given the onset time of an annotation for a confusable sign, $t_{c_j} \in T_{ANN_{C(S)}}$, and a set of predictions $P(S)$ with onset times $T_{P(S)}$, we define the FP and TN evaluations as:

$$FP(t_{c_j}) \text{ iff } \exists t_{p_i} \in T_{P(S)} : t_{pi} \in [t_{c_j} - tol, t_{c_j}]$$
$$TN(t_{c_j}) \text{ iff } \forall t_{pi} \in T_{P(S)} : t_{pi} \notin [t_{c_j} - tol, t_{c_j}]$$

In other words, a predicted spotting of a given target is a FP if it falls within the tolerance window of an annotated occurrence of another sign that is a confusable sign for the target. On the other hand, a TN occurs if we do *not* predict a spotting within the tolerance window of this annotated occurrence of the confusable sign.

Finally, we can analyze the FP and TN instances in terms of their Δ properties to determine which phonological properties are difficult to distinguish for our model. Our general approach to evaluating sign spotting models is further elaborated elsewhere (Hollain et al., 2023).

4 Results

The results of our evaluation are shown in Table 3. The model that was trained with the linguistic features produces more TP spottings than the model trained using landmarks, as well as fewer FP instances for the confusable signs.

Model	**TP**	**FN**	**FP**	**TN**
Linguistic	5442	4274	11395	68263
Landmarks	5380	4336	12292	67366

Table 3: Performance using linguistic and landmark features

We now investigate the capabilities of our linguistic features to capture the linguistic properties of signs, compared to the landmark features. In Figure 4, we display the percentage of FPs per Δ property. The percentage is computed by counting how often the confusable signs with each Δ property, as shown in Table 2, are falsely spotted. For instance, a value of 50% in the Alternating Movement column would indicate that $182 \cdot 0.5 = 91$ of the confusable signs that differ only in this Δ

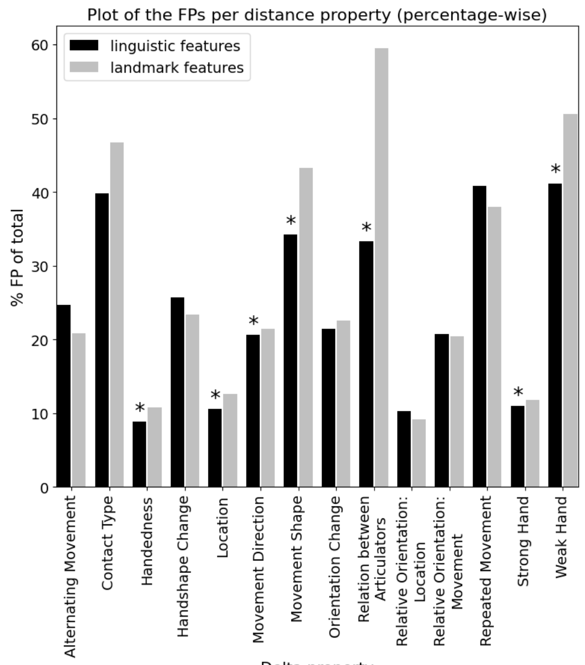


Figure 4: Percentage of confusable signs, per Δ property, that are falsely spotted ($*$=statistically sign. improvement)

property, are falsely spotted. We performed McNemar's test to analyze for which Δ properties there was a significant difference in performance between the models. An asterisk ($*$) is displayed where the difference is significant ($p < 0.05$).

For most Δ properties, the model trained using linguistic features outperforms the one trained with landmarks as it has a lower percentage of FP spottings. While there are some properties for which the model with landmark features produces fewer FP spottings, the difference is never found to be significant. For all Δ properties where we find a significant difference in performance, the linguistic feature model outperforms the landmark model. That said, it is evident that the linguistic feature model needs further improvement, since it still produces a substantial number of FP and FN predictions, and it does not significantly outperform the landmark model for some Δ properties.

5 Conclusion

In this paper, we investigated how linguistic features, extracted from landmarks of the hands and body of a signer, can be used in the context of sign spotting. We built on recent work in sign language recognition which derived an approximate representation of the handshape of a sign from Mediapipe landmarks, and developed our own features to capture the orientation, location and movement of the hands. We compared a sign spotting model

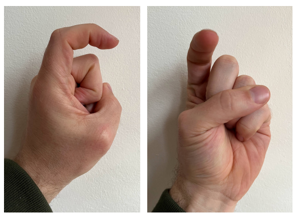


Figure 5: Hooked finger from two viewpoints

that uses these approximate linguistic features with a model that incorporates landmarks directly as training input. Our results show that the model using approximate linguistic features captures some aspects of signs better than the landmark model.

In future work, our approach to extracting linguistic features could be further improved. For example, the trajectory and repetition of movements may be better captured by including additional features besides wrist velocity. Furthermore, it could be interesting to train a model using a combination of landmark coordinates and linguistic features.

Our general approach will also benefit from further improved landmark detection technologies. Current technologies only reliably deliver 2D landmark coordinates. If an accurate estimation of the z coordinate were available, we could work with 3D representations of the hands and bodies of signers. Based on such 3D representations, linguistic features could be extracted in a more robust way. For instance, the left and right image in Figure 5 depict the same handshape viewed from two different angles. Based on 2D landmark coordinates, it would be possible to derive the curvature of the index finger under the perspective on the left (side view), but not under the perspective on the right (front view). From 3D landmark coordinates, the curvature could be derived precisely and reliably.

Another limitation of currently available landmark detection technologies, such as Mediapipe, is that they are not explicitly trained on sign language data. Certain handshapes are frequent in sign languages but may not be as frequent in general-purpose datasets. As a result, the current performance of Mediapipe and similar tools may be limited for such handshapes. That said, an important advantage of the modular approach we adopted is that it allows for the direct incorporation of future improvements of landmark detection technologies.

Finally, while not the focus of this work, we note that the model chosen to demonstrate the performance of the two types of features can be improved. A more sophisticated model architecture may result in better sign spotting performance.

References

Aljanaki, Anna, Frans Wiering, and Remco C Veltkamp. 2015. Emotion based segmentation of musical audio. In *Proceedings of the 16th conference of the international society for music information retrieval*, pages 770–776.

Battison, Robbin. 1978. *Lexical borrowing in American sign language*. Silver Spring, MD: Linstok Press.

Bowden, Richard, David Windridge, Timor Kadir, Andrew Zisserman, and Michael Brady. 2004. A linguistic feature vector for the visual interpretation of sign language. In *European conference on computer vision 2004: 8th european conference on computer vision*, pages 390–401. Springer.

Bragg, Danielle, Oscar Koller, Mary Bellard, Larwan Berke, Patrick Boudreault, Annelies Braffort, Naomi Caselli, Matt Huenerfauth, Hernisa Kacorri, Tessa Verhoef, Christian Vogler, and Meredith Ringel Morris. 2019. Sign language recognition, generation, and translation: An interdisciplinary perspective. In *The 21st association for computing machinery international special interest group on accessible computing conference on computers and accessibility*, pages 16–31.

Brentari, Diane, Jordan Fenlon, and Kearsy Cormier. 2018. Sign language phonology. In *Oxford research encyclopedia of linguistics*. Oxford University Press.

Brentari, Diane. 2019. *Sign Language Phonology*. Cambridge University Press.

Cao, Zhe, Tomas Simon, Shih-En Wei, and Yaser Sheikh. 2017. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7291–7299.

Celebi, Sait, Ali Selman Aydin, Talha Tarik Temiz, and Tarik Arici. 2013. Gesture recognition using skeleton data with weighted dynamic time warping. *Proceedings of the international conference on computer vision theory and applications*, 1:620–625.

Chai, Junyi and Anming Li. 2019. Deep learning in natural language processing: A state-of-the-art survey. In *2019 International conference on machine learning and cybernetics*, pages 1–6. IEEE.

Cho, Seong-Sik, Hee-Deok Yang, and Seong-Whan Lee. 2009. Sign language spotting based on semi-markov conditional random field. In *2009 workshop on applications of computer vision*, pages 1–6. IEEE.

Crasborn, Onno and Inge Zwitserlood. 2008. The corpus NGT: an online corpus for professionals and laymen. *Construction and exploitation of sign language Corpora. 3rd workshop on the representation and processing of sign languages*, pages 44–49.

Crasborn, Onno, Inge Zwitserlood, and Johan Ros. 2008. The corpus NGT. an open access digital corpus of movies with annotations of sign language of the netherlands. *Centre for language Studies, Radboud University Nijmegen*.

Crasborn, Onno, Richard Bank, Inge Zwitserlood, Els van der Kooij, Anique SchÃ¼ller, Ellen Ormel, Ellen Nauta, Merel van Zuilen, Frouke van Winsum, and Johan Ros. 2014. NGT Signbank. *Centre for language Studies, Radboud University Nijmegen*.

Crasborn, Onno, Inge Zwitserlood, Els van der Kooij, and Anique Schüller. 2020. Global signbank manual. Technical report, version 2. Accessed on 5 April 2023.

Crasborn, Onno. 2012. Phonetics. In *Sign language: An international handbook*. De Gruyter.

De Vries, Arjen P, Gabriella Kazai, and Mounia Lalmas. 2004. Tolerance to irrelevance: A user-effort oriented evaluation of retrieval systems without predefined retrieval unit. In *RIAO conference proceedings*, pages 463–473.

Elmezain, Mahmoud, Ayoub Al-Hamadi, Jorg Appenrodt, and Bernd Michaelis. 2008. A hidden markov model-based continuous gesture recognition system for hand motion trajectory. In *2008 19th international conference on pattern recognition*, pages 1–4. IEEE.

Enrıquez, Manuel Vázquez, JL Alba-Castro, L Docio-Fernandez, JCSJ Junior, and S Escalera. 2022. Eccv 2022 sign spotting challenge: dataset, design and results. In *European conference on computer vision workshops*.

Farhan, Youssef and Abdessalam Ait Madi. 2022. Real-time dynamic sign recognition using mediapipe. In *2022 IEEE 3rd international conference on electronics, control, optimization and computer science*, pages 1–7. IEEE.

Han, Junwei, George Awad, and Alistair Sutherland. 2009. Modelling and segmenting subunits for sign language recognition based on hand motion analysis. *Pattern recognition letters*, 30(6):623–633.

He, Liwei, Elizabeth Sanocki, Anoop Gupta, and Jonathan Grudin. 1999. Auto-summarization of audio-video presentations. In *Proceedings of the seventh association for computing machinery international conference on multimedia (part 1)*, pages 489–498.

Hollain, Natalie, Martha Larson, and Floris Roelofsen. 2023. Distractor-based evaluation of sign spotting. *Sign language translation and avatar technology*.

Hussain, Muhammad Jamil, Ahmad Shaoor, Suliman A Alsuhibany, Yazeed Yasin Ghadi, Tamara al Shloul, Ahmad Jalal, and Jeongmin Park. 2022. Intelligent sign language recognition system for e-learning context. *Computers, materials & continua*, 72(3):5327–5343.

Jiang, Tao, Necati Cihan Camgöz, and Richard Bowden. 2021. Looking for the signs: Identifying isolated sign instances in continuous video footage. In *2021 16th IEEE international conference on automatic face and gesture recognition*, pages 1–8. IEEE.

Khosla, Prannay, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Advances in neural information processing systems*, 33:18661–18673.

Ko, Sang-Ki, Jae Gi Son, and Hyedong Jung. 2018. Sign language recognition with recurrent neural network using human keypoint detection. In *Proceedings of the 2018 conference on research in adaptive and convergent systems*, pages 326–328.

Ko, Sang-Ki, Chang Jo Kim, Hyedong Jung, and Choongsang Cho. 2019. Neural sign language translation based on human keypoint estimation. *Applied sciences*, 9(13):2683.

Momeni, Liliane, Gul Varol, Samuel Albanie, Triantafyllos Afouras, and Andrew Zisserman. 2020. Watch, read and lookup: learning to spot signs from multiple supervisors. In *Asian conference on computer vision*.

Moryossef, Amit and Yoav Goldberg. 2021. Sign Language Processing. https://sign-language-processing.github.io/. Accessed: Jan. 27, 2023.

Ong, Eng-Jon, Oscar Koller, Nicolas Pugeault, and Richard Bowden. 2014. Sign spotting using hierarchical sequential patterns with temporal intervals. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1923–1930.

Pendzich, Nina-Kristin. 2020. *Lexical nonmanuals in German Sign Language: Empirical studies and theoretical implications*. De Gruyter.

Sandler, Wendy. 2012. The phonological organization of sign languages. *Language and linguistics compass*, 6(3):162–182.

Sengupta, Arindam, Feng Jin, Renyuan Zhang, and Siyang Cao. 2020. MM-Pose: Real-time human skeletal posture estimation using mmwave radars and cnns. *IEEE sensors journal*, 20(17):10032–10044.

Shin, Jungpil, Akitaka Matsuoka, Md Al Mehedi Hasan, and Azmain Yakin Srizon. 2021. American sign language alphabet recognition by extracting feature from hand pose estimation. *Sensors*, 21(17):5856.

Smith, Jordan BL and Elaine Chew. 2013. A meta-analysis of the mirex structure segmentation task. In *Proceedings of the 14th conference of the international society for music information retrieval*.

Smith, Jordan Bennett Louis, John Ashley Burgoyne, Ichiro Fujinaga, David De Roure, and J Stephen Downie. 2011. Design and creation of a large-scale database of structural annotations. In *Proceedings of the 12th international society for music information retrieval conference*, pages 555–560.

Stokoe, William. 1960. Sign language structure, an outline of the visual communications systems of american deaf. *Studies in linguistics occasional paper*, 8.

Tyrone, Martha. 2020. Phonetics of sign language. In *Oxford research encyclopedia of linguistics*. Oxford University Press.

Van der Kooij, Els. 2002. *Phonological categories in Sign Language of the Netherlands: The role of phonetic implementation and iconicity*. LOT.

Viitaniemi, Ville, Tommi Jantunen, Leena Savolainen, Matti Karppa, and Jorma Laaksonen. 2014. S-pot–a benchmark in spotting signs within continuous signing. In *Proceedings of the 9th international conference on language resources and evaluation*, pages 1892–1897. European Language Resources Association.

Von Agris, Ulrich, Jörg Zieren, Ulrich Canzler, Britta Bauer, and Karl-Friedrich Kraiss. 2008. Recent developments in visual sign language recognition. *Universal access in the information society*, 6:323–362.

Wong, Ryan, Necati Cihan Camgöz, and Richard Bowden. 2023. Hierarchical i3d for sign spotting. In *European conference on computer vision workshops*, pages 243–255. Springer.

Yang, Hee-Deok and Seong-Whan Lee. 2010. Simultaneous spotting of signs and fingerspellings based on hierarchical conditional random fields and boostmap embeddings. *Pattern recognition*, 43(8):2858–2870.

Zaki, Mahmoud M and Samir I Shaheen. 2011. Sign language recognition using a combination of new vision based features. *Pattern recognition letters*, 32(4):572–577.

Zhang, Fan, Valentin Bazarevsky, Andrey Vakunov, Andrei Tkachenka, George Sung, Chuo-Ling Chang, and Matthias Grundmann. 2020. Mediapipe hands: On-device real-time hand tracking. *arXiv preprint arXiv:2006.10214*.