Data Science Master Thesis

Radboud University

Towards Explainable Sign Spotting Systems: an Exploration of Approximative Linguistic Features and Evaluation Methods

Author:
Natalie Hollain
s1018472
natalie.hollain@ru.nl

First supervisor:
Prof. M.A. Larson
m.larson@cs.ru.nl
Radboud University

Second supervisor:
Dr. L.F.M. ten Bosch
louis.tenbosch@ru.nl
Radboud University

External supervisor:
Dr. F. Roelofsen
f.roelofsen@uva.nl
University of Amsterdam

September 12, 2023

Abstract/Executive Summary

This research project carried out an initial exploration into how sign spotting, the task of detecting when a target sign occurs in a given video, can be performed in a more explainable manner. Explainability demands that a system is correct, robust and interpretable to humans [1], [2]. Inspired by domain knowledge being used to increase the explainability and interpretability of systems in other domains [3], [4], we investigate the possibility of using a knowledge-based approach for sign spotting.

One manner in which knowledge about sign language can be incorporated into sign language systems is through linguistic insights. Current sign spotting systems typically do not make use of such knowledge [5], thus limiting their interpretability. Similarly, evaluation methods for sign spotting do not draw on linguistic knowledge, resulting in a lack of explainability since they fail to robustly estimate model performance given an incomplete ground truth. Updates to the known ground truth, in particular the addition of challenging sign annotations, can significantly alter the estimated performance. Moreover, current evaluations do not reflect user expectations for sign spotting systems because spottings are allowed to occur after a relevant segment has already started. Users thus have to put in effort, such as rewinding the video, to watch the full relevant segment, which was found to not reflect what users expect [6].

The goal of this thesis is to address these limitations using a knowledge-based approach. We incorporate linguistic knowledge about sign language into a sign spotting system and evaluation method. We aim to enhance explainability by enabling a sign spotting analysis based on linguistic insights. Furthermore, we develop linguistic features to ensure our model uses knowledge-based inputs as the basis for its decision-making. In this way, we hope to increase the explainability of current methods.

To address the need for explainable sign spotting systems, we implemented features for a sign spotting model that approximate the basic phonological properties of signs, including handshape, orientation, location and movement [7], [8]. Our features are extracted from landmarks, which are keypoints in the body, such as the fingertips and shoulders, that we detected using a landmark detection tool. As far as we are aware, we are the first to implement a sign spotting model which extracts such features from landmarks. By taking into account the basic four phonological properties, we aim to create explainable sign representations for our model to encode. As a result, it is possible to perform a failure analysis for our model that is facilitated by the linguistic features.

To address the need for explainable evaluation methods for sign spotting, we developed an evaluation that is rooted in the concept of *tolerance to irrelevance* (TTI) [9]. TTI builds on the assumption that users, given an entry point in a video or audio stream, keep watching or listening until their tolerance to irrelevant content is reached. Through this means, our evaluation method reflects the effort it takes for users to use a sign spotting system.

However, TTI, like existing sign spotting evaluations, relies on a full ground truth to reliably determine a model's performance, which may not be available for a sign spotting dataset. We address this limitation by using a novel approach that uses only the most challenging known cases to assess our model performance. These hardest cases are called distractors, which we define as those signs that are most similar to the target sign based on a distance measure. In our work, we develop a novel linguistic distance measure to determine the similarity between signs. Through the usage of these distrac-

tors, we estimate the performance for the full ground truth based solely on the hardest cases from the known ground truth, and assume that this makes our performance estimation robust to the addition of new annotations. We validated this assumption by investigating the effects of updates to the annotations on the performance estimates by our distractor-based evaluation compared to a baseline evaluation that uses random, as opposed to hard, cases. Our results show that the distractor-based evaluations provides a more conservative estimate of the performance of a model and is comparably robust to changes in the annotations compared to the baseline.

We validated our linguistic features using an empirical analysis, where we compared the effectiveness of a non-linguistic baseline that used landmarks directly, to a model using our more explainable, linguistically motivated features that are extracted from landmarks. Moreover, we investigated whether a combination of linguistic and baseline landmark features would result in better performance. The conditions were compared through the use of our distractor-based evaluation. We determined that the combination of the features provided the best performance at the cost of linguistic representativeness.

Contents

1 Introduction **1**
1.1 Research Questions . . . 3

2 Related Work **4**
2.1 Sign language linguistics . . . 4
2.2 Sign Language Processing . . . 5
2.2.1 Sign spotting . . . 5
2.3 Evaluation methods . . . 6

3 Approach **7**
3.1 Feature engineering . . . 7
3.1.1 Feature set comparison . . . 9
3.2 Evaluation . . . 9
3.2.1 Linguistic distance . . . 11
3.2.2 Validation . . . 11
3.2.3 Evaluation of linguistic features . . . 12

4 Method **14**
4.1 Comparison to Watch, Read and Lookup framework . . . 14
4.2 Feature engineering . . . 15
4.2.1 Detecting landmarks . . . 15
4.2.2 Feature extraction . . . 16
4.2.3 Final features . . . 19
4.3 Datasets . . . 19
4.3.1 Corpus Nederlandse Gebarentaal . . . 19
4.3.2 Signbank . . . 19
4.4 Data Preprocessing . . . 19
4.4.1 Corpus Nederlandse Gebarentaal . . . 19
4.5 Preparing dataset . . . 21
4.5.1 Fixed-length inputs . . . 21
4.5.2 Data augmentation . . . 21
4.5.3 Creating batches and shuffling . . . 22
4.6 Sign spotting model . . . 22
4.6.1 Loss function . . . 23
4.7 Feature selection . . . 24
4.8 Test phase . . . 25
4.9 Evaluation . . . 26
4.9.1 Validation . . . 26
4.9.2 Tolerance window size . . . 26
4.9.3 Handling overlap and double predictions . . . 27
4.9.4 Linguistic distance . . . 27
4.9.5 Balancing distractors and target sign annotations . . . 31
4.9.6 Confusable signs . . . 32

5 Experimental setup **34**
5.1 Validation of distractor-based evaluation 34
5.2 Feature set comparison 34

6 Results **36**
6.1 Validation of distractor-based evaluation 36
6.2 Feature set comparison 39
6.2.1 Distractor-based evaluation 39
6.2.2 Comparison using confusable signs 41

7 Discussion **44**
7.1 Approximative phonological features 44
7.2 Evaluation method 45
7.3 Deaf inclusion 46

A Appendix **V**
A.1 Pseudocode linguistic distance computation V
A.2 Proof: mirroring after normalisation V
A.3 Spotting threshold tests VI
A.4 Masking experiments VII
A.5 Full results of the validation of the distractor-based evaluation IX
A.5.1 Dropout ratio = 0.1 IX
A.5.2 Dropout ratio = 0.25 X
A.5.3 Dropout ratio = 0.5 XI
A.5.4 Dropout ratio = 0.75 XII

Acknowledgements

Before getting into the nitty-gritty details of my work, I would like to first acknowledge those people who assisted me throughout the writing of this thesis. After all, it was through their help that this work came to be.
I would like to thank Martha Larson and Floris Roelofsen for giving me the opportunity to work on this thesis and for guiding me through the entire process. The fact that we managed to turn this thesis into not one, but two published papers, is something I would have never dreamed of when starting this project. Without them, none of this would have been possible.
I want to thank Onno Crasborn for providing me with the data of the Corpus Nederlandse Gebarentaal and for enriching my knowledge about sign languages. The talks we had were my introduction to this topic that I knew so little about up until that point. This is what initially inspired me to use linguistics in this thesis.
For their valuable feedback during the process of writing this thesis, as well as allowing me to rant whenever things became too much, I would like to thank Lyke Esselink and Javier Martínez Rodríguez.
Last, but certainly not least, I want to thank my partner Christian Bloks, who advised, comforted and understood me even in the most difficult of times.

Chapter 1

Introduction

Sign spotting, the task of determining when a target sign occurs in a given video, is a task within the field of sign language processing (SLP) [10]. A variety of applications have been proposed for sign spotting systems, which includes search and automatic annotation systems [11]. These applications have the potential to facilitate deaf and hard-of-hearing individuals, as well as researchers who are annotating sign language datasets.
In this project, we carried out an initial exploration into how sign spotting can be performed in a more explainable manner. Explainability is an important aspect to consider when developing an AI system because it facilitates the comprehensibility and transparency of a system's decision-making process. In SLP, explainable systems have already been used for sign language learning [12] and sign language recognition (SLR) [13], [14]. On the other hand, current methods for sign spotting have been found to be lacking in terms of their explainability [11].
To deem a system explainable, we require that it is correct, robust and interpretable to humans [1], [2]. One way in which explainability has previously been facilitated outside of the domain of SLP is through the incorporation of domain knowledge [3], [4]. Inspired by this work, we investigate the possibility of using a knowledge-based approach for sign spotting.
One manner in which knowledge about sign language can be incorporated into sign language systems is through linguistic insights. Current work on SLP, including sign spotting, typically does not leverage such knowledge [5], which limits the interpretability of developed systems. Similarly, evaluation methods for sign spotting do not draw on linguistic knowledge, resulting in a lack of explainability since they fail to robustly estimate model performance given an incomplete ground truth. Updates to the known ground truth, in particular the addition of challenging sign annotations, can significantly alter the estimated performance. Moreover, current evaluations do not reflect user expectations for sign spotting systems because spottings are allowed to occur after a relevant segment has already started. Users thus have to put in effort, such as rewinding the video, to watch the full relevant segment. This has been found to not reflect what users expect [6]. As such, we want to push sign spotting evaluations to be more robust against changes to the known ground truth, and simultaneously more reflective of user expectations.
The goal of this thesis is to address the aforementioned limitations of sign spotting models and evaluation methods, using a knowledge-based approach. We incorporate linguistic insights about sign language into the design of a sign spotting system as well as an evaluation method. Our aim is to enhance the explainability of sign spotting systems by enabling an analysis based on linguistic domain knowledge. Moreover, we develop linguistic features to ensure our model uses knowledge-based inputs as the basis for its decision-making. Through these means, we hope to increase the explainability of current methods.
To address the need for explainable sign spotting systems, we implemented features for a sign spotting model that approximate the basic phonological properties of signs. Our features are based on the work by Stokoe [7] and Battison [8] that specifies the manual

phonology of a sign in terms of the handshape, orientation, location and movement. By taking into account the basic four phonological properties, we aim to create explainable sign representations for our sign spotting model to encode. Therefore, it is possible to perform a failure analysis for our model that is facilitated by the linguistic features.
We extracted the features from landmarks, which are keypoints in the body, such as the fingertips and shoulders, using the landmark detection tool Mediapipe. As far as we are aware, our sign spotting model is the first that uses linguistically motivated features that were extracted from landmarks. We anticipate that linguistic features that are extracted from landmarks are more robust than those which are directly extracted from the visual input, because existing landmark detection tools have 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 [15]. Additionally, the modular nature of this approach makes it simple to incorporate future improvements of landmark detection technologies or linguistic feature extraction methods as they become available.
To address the need for explainable evaluation methods for sign spotting, we developed an evaluation that is rooted in the concept of tolerance to irrelevance (TTI) [9]. TTI bases itself on the assumption that users, when provided with an entry point in a stream of video or audio, will keep watching or listening until their tolerance to irrelevant content is reached. By taking into account the tolerance of users, our evaluation method reflects the amount of effort it takes to use a sign spotting system. Thus, we hope that our evaluation enables a more user-reflective assessment of sign spotting systems.
Like existing sign spotting evaluations, TTI relies on a full ground truth to reliably determine the performance of a model. Since sign spotting datasets have to be manually annotated by human annotators, datasets are typically not fully annotated [16] and thus, the full ground truth is not known. We address this limitation by proposing a novel approach that uses only the most challenging cases in the known ground truth to assess the performance of a model. We term these hardest cases *distractors* and define them as those signs which are most similar to the target sign that we want to spot based on a chosen distance measure. In this research project, we develop a novel linguistic distance measure to assess the similarity between signs. Through the distractors, we estimate the performance for the unknown, full ground truth based solely on the hardest cases from the ground truth. We assume that our evaluation method is therefore robust to the addition of new annotations, and validate this assumption by performing experiments with altered versions of our dataset and testing the change in our evaluation method's output compared to a baseline evaluation that uses random, as opposed to hard, cases. Our results show that the distractor-based evaluations provides a more conservative estimate of the performance of a model and is comparably robust to changes in the annotations compared to the baseline.
To validate our linguistic features, we performed an empirical analysis in which we compared the effectiveness of a non-linguistic baseline which uses landmarks directly, to a model using our more explainable, linguistically motivated features that we extracted from the landmarks. Furthermore, we investigated whether a combination of both the linguistic and baseline landmark features would result in better performance. The conditions were compared through the usage of our distractor-based evaluation. We found that the combination of features provided the best performance at the cost of linguistic representativeness.

1.1 Research Questions

The goal of this research is twofold. First, we determine the impact of incorporating linguistically motivated features into a sign spotting model. In particular, we extract approximative phonological features from landmarks of the hands, and investigate when these features deliver an improvement over using the landmarks directly. Second, we investigate how a sign spotting evaluation can be developed that reflects user effort in using a sign spotting model and is capable of handling an incomplete ground truth. We propose the following main research questions with sub-questions:

RQ1 In which ways can manual phonological properties of sign language be incorporated into a sign spotting model?

- RQ1.1 What do representations based on phonetics reveal about the importance of phonetics in sign spotting?
- RQ1.2 What is the effect of incorporating phonological properties of sign language on model performance?
- RQ1.3 How does incorporating phonological knowledge affect the robustness and applicability of a sign language model?

RQ2 How can we evaluate sign spotting systems in a way that is robust to updates to the known ground truth and is reflective of users?

- RQ2.1 How can an evaluation method be adapted to deal with an incomplete ground truth?

Chapter 2

Related Work

2.1 Sign language linguistics

In sign language linguistics, the phonology of a sign is described in terms of manual and non-manual aspects. While the manual phonology only concerns itself with the hands, non-manual information encapsulates a variety of aspects, such as the body posture and the facial expressions of the signer [17]. In this thesis, we focus only on the manual component of the phonology, leaving the study of the non-manual properties to future work. The manual articulation of a sign is typically described using four attributes: handshape, location, orientation, and movement [7], [8], [18], [19], of which we provide a brief overview.

The *handshape* describes the way in which the fingers of the signing hand(s) are configured. It can be analyzed in terms of which fingers are extended and which are folded into the palm, whether they are curved or straight and how they are positioned in terms of their distance to the other fingers and thumb. If both hands are used in the sign, they can either have the same or a different handshape.

The *location* of the hand specifies where the utterance of the sign takes place. Signs can be performed in neutral space, which is the space in front of the signer's torso. They can also be performed by the head, the neck or the arm. The location may also be specified in a relative manner, for instance when one hand is located next to the other hand in a two-handed sign.

The *orientation* of the hand is specified in an absolute or relative manner. Absolute orientation defines the orientation of the hand without making reference to how it is positioned in relation to the body. By contrast, relative orientation takes the body as the frame of reference when specifying the orientation. Absolute orientation is the dominant perspective in most phonological models [18].

The *movement* of the hand(s) can be specified in terms of its shape, size, direction, tensity and whether there is repetition. In symmetrical, two-handed signs, both hands make the same movement in a synchronized manner. On the other hand, asymmetrical two-handed signs are performed such that the two hands are either not performing the same motion, or the motion is identical but for its synchrony.

While *handedness* is not specified as a phonological aspect in the literature we consulted, it is also linguistically relevant whether a sign is performed with one or two hands [18]. We refer to handedness as a linguistic component of a sign from now on.

The phonological makeup of a sign only partially determines how it is articulated in reality. The specific characteristics of the signer, such as their gender, age, emotional state and loudness, are one set of factors which impact the way a sign is performed in reality. Furthermore, which signs precede and follow the current sign influences the articulation because there is a transition from one sign to the next. The effect of such factors is studied in the field of sign language phonetics [20], [21]. Our longer-term goal is to develop more accurate and explainable sign spotting systems through the incorporation of insights from sign language phonology and phonetics. In this thesis, we focus on extracting the basic phonological parameters – namely the handshape, orientation,

location, and movement – for a sign spotting model, and leave the incorporation of phonetic factors and non-manuals to future research.

2.2 Sign Language Processing

Researchers in the field of sign language processing (SLP) study how signs can be retrieved, recognized and spotted within video footage and images. Key approaches in SLP differ with respect to whether they attempt to leverage linguistic information about sign language and in which way they do it, as shown in Figure 2.1.

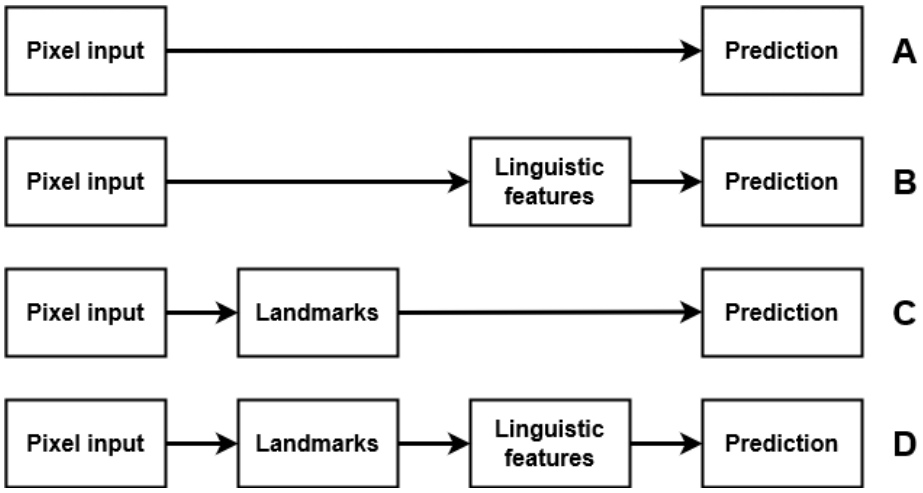

Figure 2.1: Four methods for sign language processing

Some recent work follows Approach **A**, where pixel information is used as the sole input, and linguistic features that are relevant for sign language, such as the handshape or orientation of the hand, are not explicitly considered [22], [23]. In contrast, earlier work proposed methods to extract phonological properties of signs from pixel information [24]–[27], which aligns with Approach **B**. Other approaches have applied landmark detection tools, such as MMPose or Openpose, to obtain the location of body landmarks from the pixel input and used them as input to a model [28], [29] (Approach **C**). Distance and angle features which may approximate the phonological properties of a sign, in particular the handshape, have also been extracted from landmarks [30]–[32] (Approach **D**).
Incorporating linguistic knowledge holds great promise for improving the explainability of SLP systems. A potential drawback of incorporating linguistic features based directly on pixel information, as is done in Approach B, is that it is sensitive to particular visual properties of the training data, such as the lighting conditions, the shape and color of the signer's clothes, the skin colour of the signer, and the recording background. We hypothesize that Approach D improves on this by implementing a modular pipeline that is potentially more robust because linguistic features are extracted from landmarks rather than pixel input.

2.2.1 Sign spotting

In the following section, we give a brief overview of the sign spotting work that is relevant to our paper. We follow this up by explaining what distinguishes our work from what has been done.
Viitaniemi, Jantunen, Savolainen, *et al.* [33] created a tool to assess the spotting performance of a model on continuous signing. They implemented a baseline model that uses

Dynamic Time Warping (DTW) to demonstrate the effectiveness of it. In their paper, they mention that the used dataset can be regarded as 'artificial' due to its small sample of (five) signers, the studio conditions in which it was filmed, and the videos consisting of example sentences that might not reflect real-life use of the signs. As a result, their work lacks the realistic data that is desired for sign language technologies.
Two approaches that use conditional random fields (CRFs) on continuous signing are implemented by Cho, Yang, and Lee [34] and Yang and Lee [35]. While both papers report high model performance, the datasets which are used are lacking in terms of complexity. More specifically, the footage was recorded in studio conditions, consists of a single signer, and only individual sentences were filmed.
In the paper by Momeni, Varol, Albanie, *et al.* [22], a new framework called 'Watch, Read and Lookup' (WLR) is proposed for continuous sign spotting. Sign spotting embeddings are learned from watching sparsely annotated videos, reading subtitles to find candidate signs and looking up examples in a sign language video dictionary. Their work builds on that done on low-shot action localization and multiple instance learning. The paper uses the BSL-1k dataset that contains videos of BBC broadcasts that have been interpreted in sign language. While BSL-1k matches most of the dataset criteria mentioned in Bragg, Koller, Bellard, *et al.* [5], one aspect to note is that interpreted signing is distinct from 'natural signing', the latter being faster, more spontaneous, and less distinctly signed [5]. Consequently, the applicability of their framework to more spontaneous sign language is unclear.
The discussed approaches do not explicitly incorporate the linguistic knowledge about sign language that was discussed in Section 2.1. Furthermore, we highlighted that the chosen datasets lacked in terms of the number of signers, the type of signing that is presented and the recording conditions. In conclusion, there is a gap in the sign spotting literature that has yet to be filled.

2.3 Evaluation methods

To the best of our knowledge, evaluation methods for sign spotting have so far only been covered in one paper, namely that of Viitaniemi, Jantunen, Savolainen, *et al.* [33]. In this paper, a tool was developed that could be used to assess the spotting performance of a model in the context of continuous signing. This assessment determines a prediction to be correct if its overlap with a ground truth annotation exceeds a threshold.
A field that is related to sign spotting is that of audio segmentation. For this task, the F-measure appears to be one of the more popular metrics [36]–[38]. This metric is also used for tasks like action segmentation [39]. For the task of audio segmentation, a tolerance window is used for the F-measure which makes it so the boundaries, i.e. the start and end point, of a prediction have to be close to the ground truth boundaries. Two tolerance sizes are typically used, namely 0.5 and 3 seconds [36]–[38]. The 3 second tolerance window is supported by the fact that users need about 3 seconds to adjust to viewing a result item [9]. For the 0.5 second tolerance, it is most similar to the strict tolerances applied to the task of precise event spotting [40].
Just like the discussed sign spotting metric, audio segmentation metrics and tolerances are symmetric, since predictions are allowed to start some time before or after the beginning of the annotation. Previous work has considered that this is not ideal for retrieval metrics [41] and as such, it is unclear whether such symmetry reflects the real use case of a sign spotting system. In [6], researchers discovered that users found it annoying when spottings began after the start of the ground truth.

Chapter 3

Approach

In this chapter, we explain our process for developing the two contributions of our work. First, we explain our approach to engineering our linguistically motivated features for a sign spotting model. Second, we formulate an evaluation method for sign spotting which reflects user effort and is capable of handling an incomplete ground truth.

3.1 Feature engineering

Inspired by existing approaches in SLR [30]–[32], we implement features that are extracted from hand landmarks. Previous work only extracted features that approximated the phonological handshape, and thus, we implement new features to capture the movement, location and orientation of the signing hand(s). Below, we give an overview of the types of features that we extract for each of the four basic phonological parameters.

Handshape Similarly to Farhan and Madi [32], we extract angles and distances to represent the handshape of each hand. The angles are computed to assess the curvature of the fingers. We compute two angles for each finger: one which represents the angle within the finger, and one which represents the finger's angle with the wrist. We compute the angle at the midpoint between two points, namely between the fingertip and the base of the finger or the wrist. An example of these measurements is shown in Figure 3.1. For the extraction of the distance features, we measure the distance between pairs of points to represent the spread of the fingers and the relative position of the fingers compared to the wrist. An example can be seen in Figure 3.2.

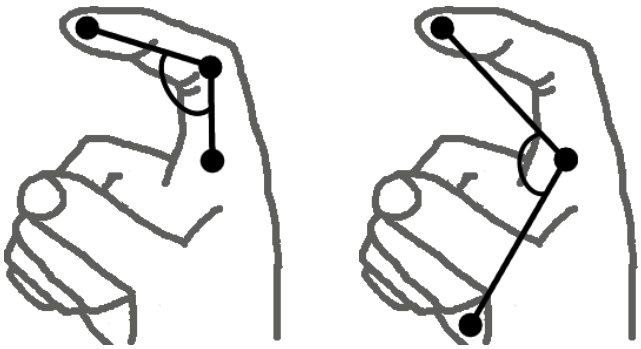

Figure 3.1: The angles measured for a given finger: one shows the angle within the finger (left), the other shows the angle with the wrist (right)

Orientation To ensure that we get the orientation of the hands irrespective of the camera angle, we compute the orientation relative to the body. We achieve this by drawing two axes within the hands and the torso: one which represents a vertical y-axis and one which represents the horizontal x-axis. A visualisation is shown in Figure 3.3.

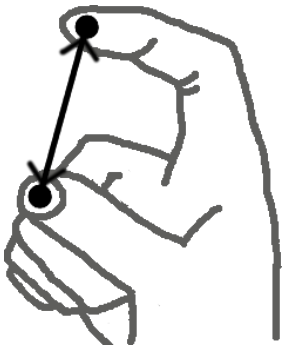

Figure 3.2: The distance measured for a given finger pair (the thumb and pointer finger)

As a result, we can determine the orientation of the hands by computing the angles between pairs of axes. The angles which are of particular interest to us are those which compare an axis within the hand to one within the torso, because the resulting angle is relevant to determining the orientation relative to the body. We therefore leave out angles which compare the axes within the hand or within the torso.

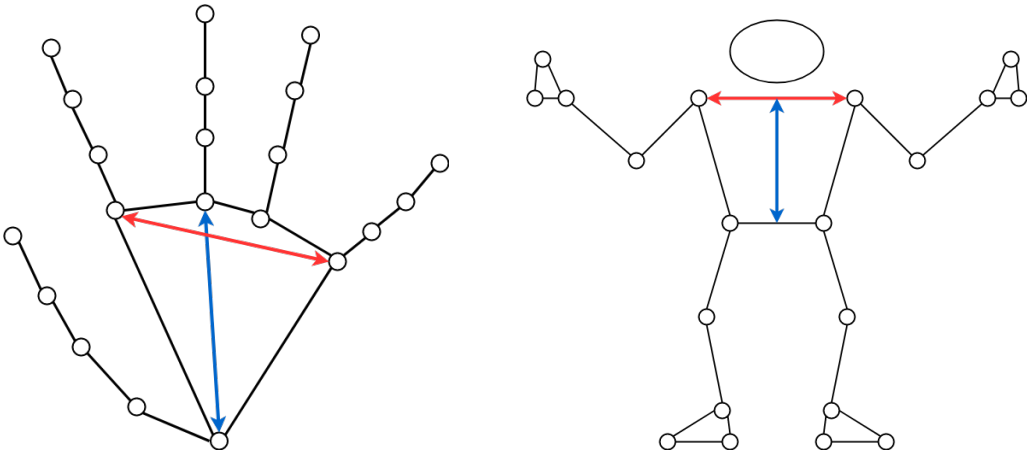

Figure 3.3: The x-axis (red) and y-axis (blue) within the hand (left) and torso (right)

Location For the representation of the location parameter, we directly use the landmarks of the hands. Specifically, we use the landmarks which locate the wrist and the fingertips within the footage. The wrist is chosen as our primary indicator of the location of the hand, because its position is not influenced by the orientation or handshape of the hand. The landmarks of the fingertips are used to give a more fine-grained indication of the location of the hands, for example in signs where one of the fingers makes contact with the face or the arm.
Moreover, the location of a sign is partially characterised by the interaction between the hands. We thus capture the location of the hands relative to each other by calculating the distance between the wrists of both hands.

Movement We represent the movement parameter by measuring the velocity of the hand over time. As before, the wrist is taken as the location of the hand at a given time, such that we can compare its location between consecutive frames to determine the velocity. By determining in which manner the location of the hand changes in terms

of both the vertical and horizontal direction, we can determine what type of movement the hand is making.

3.1.1 Feature set comparison

We compare three sets of features in this thesis. Firstly, we develop a set of linguistic features that follows the description provided in Section 3.1, which we hope provides a more explainable and knowledge-based representation of signs. We input these features into a sign spotting model, and call it the linguistic (feature) model. Secondly, we use the landmarks, on which the linguistic features are based, directly as input for the training of the model, calling it the landmark (feature) model. By comparing the two conditions, we hope to determine when the linguistic, knowledge-based features that build on the landmarks provide an improvement over using an approach that does not use knowledge about sign language. Lastly, we investigate the possibility of combining both sets of features to create a combined (feature) model that could benefit from the interaction between the landmarks and linguistics.

3.2 Evaluation

For our evaluation method, our goal is to develop an explainable evaluation that reflects user effort and can handle an incomplete ground truth. To achieve the first goal, we adopt *tolerance to irrelevance* (TTI) [9] as the basis for our evaluation metric for sign spotting. This metric only considers the starting times of annotations and predictions. It determines a prediction to be correct if its starting time falls within some window of tolerance, *tol*, *before* the ground truth. For instance, if we have target sign S with some annotation s_i and its start point t_{s_i}, and a prediction p_j with its start point t_{p_j}, the prediction would be deemed to be correct if:

$$t_{p_j} \in [t_{s_i} - tol, t_{s_i}]$$

In other words, we only tolerate if the prediction starts a little before or at the same time as the ground truth annotation. This sets our evaluation method apart from previous evaluations developed for sign spotting, since they allowed for a prediction to start after the beginning of the annotation. In other words, a prediction was deemed to be correct if:

$$t_{p_j} \in [t_{s_i} - tol, t_{s_i} + tol]$$

Previous research has shown that this does not reflect what users expect [6], because they may have to rewind the video to see the full annotated segment [41]. In conclusion, we hope to better reflect the effort that is required from users using our sign spotting model by basing our system on TTI.

One common problem with datasets that are used for SLP is that they are often not fully annotated. Such an incomplete ground truth means that segments which are not annotated do not necessarily imply that nothing is being signed in them. Instead, the annotation for such a section may simply be missing. Current sign spotting methods, as well as TTI, require that the full ground truth is known to assess the performance of a model.

To deal with an incomplete ground truth, we choose a subset of the known ground truth that we assume to be representative of the complete ground truth. We select elements for this subset based on a distance measure: for each target sign, we find the most similar other signs according to this measure. We define the annotations of these similar signs

as *distractors* and assume that, if our model is able to ignore these similar distractors when spotting a target sign, more dissimilar signs will be ignored as well. In this way, the performance on the subset of distractors that are hard to ignore should indicate a lower bound of the model's performance on the full ground truth.
The type of distance measure that should be chosen depends on the use case of the distractor-based evaluation. In this paper, our chosen distance measure is *linguistic distance*: we choose distractors based on their phonological properties as described in Section 2.1. In the next section, we will elaborate on how distances are defined using this measure.
Given a target sign S, we define the following notation for our evaluation:

$$\begin{array}{lll} P(S) & = \{p_1, \ldots, p_n\} & \text{predicted occurrences of } S \\ T_{P(S)} & = \{t_{p_1}, \ldots, t_{p_n}\} & \text{starting times of } p_1, \ldots, p_n \\ A(S) & = \{s_1, \ldots, s_m\} & \text{annotated occurrences of } S \\ T_{A(S)} & = \{t_{s_1}, \ldots, t_{s_m}\} & \text{starting times of } s_1, \ldots, s_m \\ D(S) & = \{d_1 \ldots d_l\} & \text{distractors for } S \\ T_{D(S)} & = \{t_{d_1} \ldots t_{d_l}\} & \text{starting times of } d_1 \ldots d_l \end{array}$$

Based on this notation, we can then define true positives (TP) and false negatives (FN) as follows, given the starting time of an annotation, t_{sj}:

$$\begin{aligned} TP: &\quad \exists t_{pi} \in T_{P(S)} : t_{pi} \in [t_{sj} - tol, t_{sj}] \\ FN: &\quad \forall t_{pi} \in T_{P(S)} : t_{pi} \notin [t_{sj} - tol, t_{sj}] \end{aligned}$$

Furthermore, the false positives (FP) and true negatives (TN) given the starting time of a distractor, t_{di}, are defined as:

$$\begin{aligned} FP: &\quad \exists t_{pi} \in T_{P(S)} : t_{pi} \in [t_{di} - tol, t_{di}] \\ TN: &\quad \forall t_{pi} \in T_{P(S)} : t_{pi} \notin [t_{di} - tol, t_{di}] \end{aligned}$$

We show an example of predicted spottings for a target sign and its distractors in Figure 3.4. Assume we are spotting S, with $T_{A(S)} = \{t_{s1}, t_{s2}\}$. Then, the start times of the distractors are defined as $T_{D(S)} = \{t_{d1}, t_{d2}\}$. The white bars show the tolerance windows of the targets, while the black bars indicate the tolerance windows of the distractors. The start of the predictions are shown as t_{p_i}.

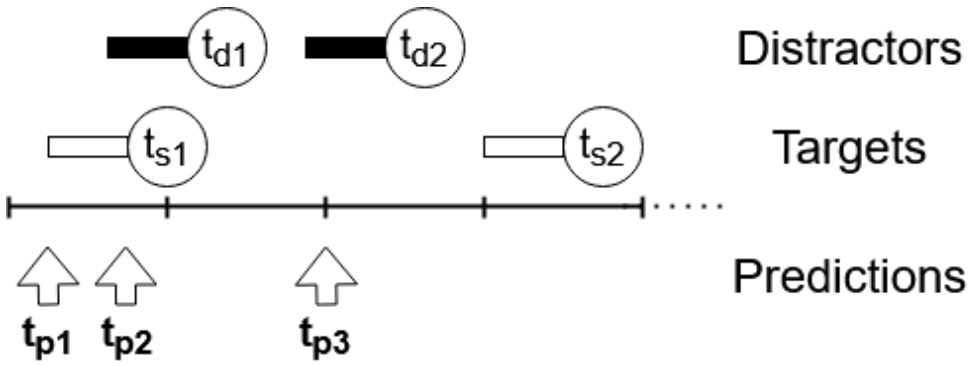

Figure 3.4: Example of predicted spottings for a target sign and its distractors

As t_{p1} falls within the tolerance window of t_{s1}, it is counted as a TP spotting. Because we already spotted t_{s1}, t_{p2} is a redundant prediction of t_{s1}. Moreover, t_{p2} does not only fall within the tolerance window of a target - it also overlaps with the target window t_{d1}. When implementing the evaluation method, both of these problems have to be dealt with in some manner. For example, t_{p2} could be discarded if we assume it would get aggregated with the first prediction. In Chapter 4, we discuss how we handle the problem of overlap between tolerance windows and double predictions.
We register t_{p3} as a FP spotting, since it is within the tolerance window of t_{d2}. If t_{p3} did not exist, we would count a TN evaluation for t_{d2}. There are no predictions at all in the tolerance window of t_{d1}, thus, this is counted as a TN.
Based on this definition of our evaluation method, we can define $accuracy = \frac{TN+TP}{TN+TP+FN+FP}$. We may then specify the *accuracy* at a specific tolerance level by defining $accuracy@tol$, or if we abbreviate it, $acc@tol$. Other metrics that are used in binary classification, such as *precision* ($\frac{TP}{TP+FP}$) and *recall* ($\frac{TP}{TP+FN}$) can also be used. We refer to these metrics, in the context of our tolerance-based evaluation, as $precision@tol$ and $recall@tol$, or $prec@tol$ and $rec@tol$ for short.

3.2.1 Linguistic distance

In this thesis, we propose a distance measure based on linguistic distance. The computation of this distance is done as follows. Assume we have two signs, A and B, for which we know their phonological makeup:

$$phonology(A) = [a_1, a_2, a_3, ..., a_n]$$
$$phonology(B) = [b_1, b_2, b_3, ..., b_n]$$

Where a_m and b_m are the m-th phonological properties that specify A and B, respectively. Together, the properties describe the phonology as discussed in Section 2.1.
We start by assuming that the signs are the same, i.e. $distance(A, B) = 0$. For each property, we can then compare whether the two signs have the same specification, i.e. whether $a_m = b_m$. If the property is the same for both signs, we say that they are similar in this sense and do not increase the distance:

$$if\ a_m = b_m : distance(A, B) \leftarrow distance(A, B)$$

Where $X \leftarrow Y$ indicates that X is updated to have value Y. If the property is different between the signs, we increase the distance by 1:

$$if\ a_m \neq b_m : distance(A, B) \leftarrow distance(A, B) + 1$$

We repeat this for all of the properties to obtain the linguistic distance between the signs A and B. The pseudocode of the linguistic distance computation can be found in Appendix A.1. The linguistic distance between all signs is computed in this manner to ensure that we can compare any two signs in terms of their linguistic similarity. Distractors may then be selected based on which signs have the lowest linguistic distance to each other.

3.2.2 Validation

We conducted a validation of our distractor-based evaluation method to determine whether it performs as intended. In particular, we investigate whether our key assumption holds, namely whether our evaluation is robust to the addition of new annotations. We speculate that this assumption holds for several reasons. First, we hypothesize that

the selection of the hardest cases based on linguistic distance will give us a more conservative estimate of the model performance than if we selected random cases. In this way, we hope that we do not overestimate the performance on the full ground truth. Second, when new annotations are added, we assume that the hardest cases will be affected less than the selected random cases. In particular, we expect that the change in the estimation of a model's performance will be more consistent for the hardest cases.
To validate whether this reasoning is valid, one option is to add more annotations to our dataset and compare the performance between the original and updated annotations. However, a limitation of this approach is that we need to determine for which signs we want to add new annotations. This has the potential to add bias to the new annotations, since either the hardest or random cases may be disproportionately affected. Ideally, we could compare the hardest and random cases in a fair manner, where both scenarios are equally affected by the change in annotations.
Therefore, we opt to remove annotations rather than adding them. One benefit of this approach is that we can control exactly which annotations are affected and focus on only those cases where we know that the hardest or random cases change. We can ensure that the number of annotations that are dropped for both scenarios is the same, which we hope makes the comparison between them more fair. In summary, we can determine whether our distractor-based evaluation is robust compared to a random baseline by investigating how the performance estimation changes when we drop annotations from our dataset. We determine how to implement this approach in Chapter 4.

3.2.3 Evaluation of linguistic features

We evaluate the performance of a model trained using our approximative linguistic features compared to one trained using landmarks to gain insight into when linguistic features contribute to sign spotting. To achieve this, we introduce the concept of *confusable signs*: signs which only differ from a given target sign by a single phonological property. Confusable signs can be described as a set of the strictest distractors, where we only select those distractors which have a distance to our target sign that is equal to 1. An example of two confusable signs in American Sign Language is shown in Figure 3.5. The sign for 'Dad' (left) is signed identically to the sign for 'Mom' (right), except for the location where it is performed, which is the forehead rather than the chin. As such, 'Dad' is a confusable sign for 'Mom' and vice versa.

Figure 3.5: Two similar signs in American Sign Language: Dad (left) and Mom (right)

We call the phonological property that differs between the confusable sign and the target sign the *Δ property*. For the example in Figure 3.5, the location of the signs would be the Δ property which makes 'Mom' and 'Dad' confusable signs for each other. By determining which confusable signs are mistaken in practice for a given target sign, we are able to pinpoint which Δ properties, and thus which phonological properties, are difficult to distinguish for both of our models. In this manner, we aim to find which

features are more representative of the phonological properties of signs.
Our analysis of the linguistic features uses an adapted version of the distractor-based evaluation. Instead of using the top-f hardest distractors for all signs, we focus only on those signs for which confusable signs are available, that is, there are signs which have a linguistic distance of 1 to the target. All of the annotations that are of a confusable sign are selected, meaning that we do not limit the number of confusable signs. We opted for this choice because we can be certain that all the selected annotations will have a linguistic distance of 1 to the target signs. Contrarily for the distractors, the linguistic distance varies and we therefore have to balance the number that we select to choose the hardest cases. We expand on how the distractors can be balanced in Section 4.9.5.
We use the following notation for the confusable signs, given a target sign S:

$$\begin{aligned} C(S) &= \{c_1 \dots c_l\} && \text{confusable signs for } S \\ T_{C(S)} &= \{t_{c_1} \dots t_{c_l}\} && \text{starting times of } c_1 \dots c_l \end{aligned}$$

True positives (TP), false positives (FP), true negatives (TN) and false negatives (FN) are then defined as:

$$\begin{aligned} TP: &\quad \exists t_{pi} \in T_{P(S)} : t_{pi} \in [t_{sj} - tol, t_{sj}] \\ FN: &\quad \forall t_{pi} \in T_{P(S)} : t_{pi} \notin [t_{sj} - tol, t_{sj}] \\ FP: &\quad \exists t_{pi} \in T_{P(S)} : t_{pi} \in [t_{ci} - tol, t_{ci}] \\ TN: &\quad \forall t_{pi} \in T_{P(S)} : t_{pi} \notin [t_{ci} - tol, t_{ci}] \end{aligned}$$

Where t_{ci} indicates the starting time of a confusable sign annotation. We note that evaluation of the TP and FN instances is defined identically to that for the distractor-based evaluation, such that only the evaluation of FP and TN instances differs. The phonological properties which are difficult to distinguish for each model can then be determined by inspecting the FP and TN instances in terms of their Δ properties.

Chapter 4

Method

4.1 Comparison to Watch, Read and Lookup framework

Of the papers discussed in Chapter 2, one of the closest papers to our work is that of Momeni, Varol, Albanie, *et al.* [22]. Previously, we explained that the paper uses a dataset of interpreted rather than 'natural' signing, and does not incorporate any explicit linguistic representations. In this section, we provide a more in-depth explanation of how our work relates and differs from the Watch, Read and Lookup (WRL) framework.

Our work is similar to that of Momeni, Varol, Albanie, *et al.* [22] in the following manner. In this previous work, the goal was to learn to spot signs by creating embeddings of two types of footage: dictionary footage of isolated signs, and signs performed in continuous footage from a dataset of sign-interpreted news broadcasts. The embeddings are learned by applying a contrastive loss during training, which aims to make embeddings of the same sign similar, while making embeddings of different signs distant. These embeddings are used to spot signs in a video using a sliding window. Our approach adopts the learning of embeddings using a contrastive loss, as well as the usage of a sliding window for the test phase of the model.

Next, we highlight how our work differs from the WRL paper. First, the WRL framework has to use positive *bags* instead of positive *pairs* for its contrastive learning due to the fact that sign variants are not precisely annotated in the dataset that the framework was trained on, BSL-1k. For any sign with different variants (e.g. SIGN-A, SIGN-B), it is simply 'bagged' as one main sign (e.g. SIGN, which could be either SIGN-A or SIGN-B). This has the downside that an embedding has to represent a group of signs, even if these signs are not visually similar to each other. Having to represent multiple signs at once has the potential to create lower-quality embeddings. To avoid the need to bag signs, we select the Corpus Nederlandse Gebarentaal (CNGT) (see Section 4.3.1) for our research. Unlike the annotations in BSL-1k, the annotations in our chosen corpus identify exactly which sign variant is signed. As such, we can adapt their framework to learn positive *pairs* as opposed to positive *bags*.

Second, the choice of the CNGT dataset is made because it is more precisely and more densely annotated than BSL-1k. BSL-1k contains over 1000 hours of footage that is annotated with approximately 280k annotations, whereas CNGT consists of 72 hours of footage for which we found there to be about 121k annotations. While BSL-1k has more than double as many annotations as the latter, it is also noteworthy that BSL-1k has been automatically annotated, with a reported annotation accuracy of approximately 70% for a sample set [16]. On the other hand, CNGT is manually annotated by Deaf assistants and was additionally supervised by linguists for annotation errors [42]. As such, we assume that this process resulted in more precise annotations than those which were acquired for the BSL-1k dataset.

Third, our approach extracts approximative linguistic features from landmarks that were extracted using Mediapipe, while WRL uses pixel information directly as input to its model. Although a feature extraction phase is present in their model architecture, the extraction process is part of the model training and is not made transparent. In

our model, feature extraction is instead done as a preprocessing step after we detect landmarks with Mediapipe. This allows us to extract features which are linguistically motivated, which we hope makes our approach more explainable than WRL.

4.2 Feature engineering

4.2.1 Detecting landmarks

We extract features from our video data by using Mediapipe, a landmark detection tool. For each video frame, this tool detects where specific keypoints of the body, such as the shoulders and wrists, are located. Mediapipe consists of a variety of 'models', with each model focusing on a different area of the body. In Figure 4.1[1], we show the two models used in this thesis: the Hand and Pose models. We predominantly use the Hand model in this thesis, due to the fact that it captures more landmarks within the hands than the Pose model. This allows us to get a fine-grained representation of the hands that is suitable for SLP applications. The Pose model is used whenever we need landmarks other than the hand landmarks, such as the shoulders or hips. Unless specified otherwise, the landmark indices that we use from now on refer to the landmarks of the Hand model. Each landmark consists of x, y, z coordinates, of which we use only the x and y coordinates. The z coordinate, which represents depth, is discarded because Mediapipe infers a 3D perspective from a 2D input, making this coordinate less reliable than the (2D) x, y coordinates [43]. From now on, when we refer to a specific landmark index, both the x and y coordinate are included unless specifically stated otherwise. For instance, the landmark at index 00 refers to the x, y coordinates of the wrist landmark. We denote the x, y coordinates of a landmark ℓ as ℓ_x, ℓ_y respectively.
The landmark coordinates that are returned by our landmark detection tool are in the range $\big[0, 1\big]$. This is because they are normalised automatically by Mediapipe using the height and width of the video. However, this means that the coordinates of landmarks in two videos of different resolutions are not comparable, since they are normalized differently. We remedy this by converting the landmarks to their pixel coordinates: for each landmark $\ell = (\ell_x, \ell_y)$, we perform the operation $(\ell_x \cdot w,\ \ell_y \cdot h)$. The width w and height h are retrieved from the video.
Now, we normalize the pixel coordinates to make the landmarks invariant to the distance and orientation of the person to the camera. This normalisation is based on [44]. We first normalise each landmark ℓ by the (absolute) distance between the left and right shoulder (sh_L, sh_R respectively):

$$\ell_{scaled} = \frac{\ell}{|sh_L - sh_R|} \tag{4.1}$$

We use the absolute distance between the two shoulders to scale the landmark, since we do not care about the sign of the distance. Then, our scaled version of landmark ℓ, which we denote as ℓ_{scaled}, is normalised by subtracting the shoulder midpoint sh_M. Since Mediapipe does not provide a landmark for the shoulder midpoint, we compute sh_M ourselves by taking the middle of the coordinates of the left and right shoulder. Then, we scale sh_M to ensure that we center ℓ_{scaled} within the scaled space. In other

[1]Pose model image taken from https://github.com/google/mediapipe/blob/master/docs/solutions/pose.md

words, we perform the following computations:

$$sh_M = (sh_{M,x}, sh_{M,y}) = (\frac{sh_{L,x} + sh_{R_x}}{2}, \frac{sh_{L,y} + sh_{R_y}}{2}) \tag{4.2}$$

$$sh_{M,scaled} = \frac{sh_M}{|sh_L - sh_R|} \tag{4.3}$$

$$\ell_{centered} = \ell_{scaled} - sh_{M,scaled} \tag{4.4}$$

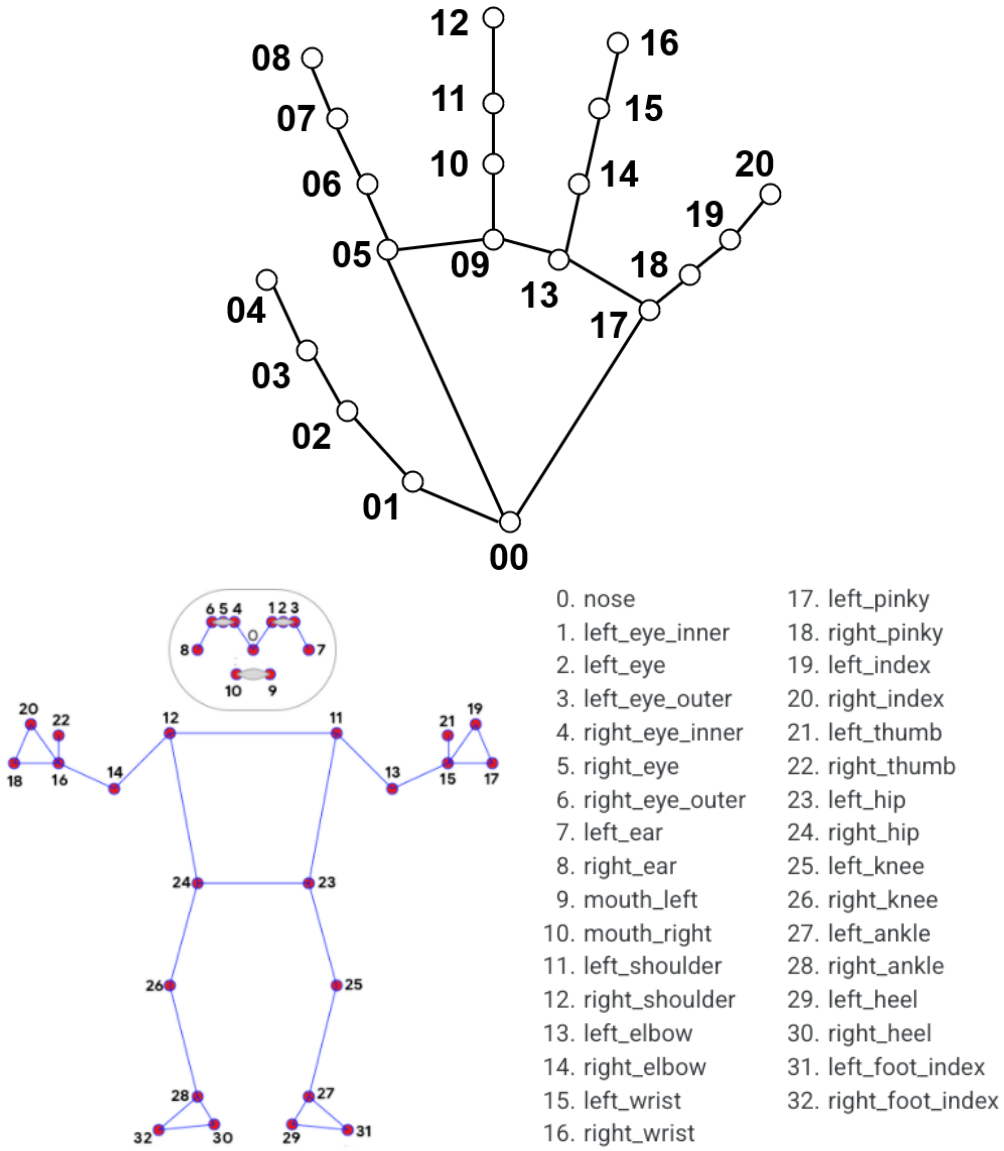

Figure 4.1: Overview of two Mediapipe models: the Hand model (top) and Pose model (bottom)

4.2.2 Feature extraction

To capture the manual phonology of sign language, we extract approximative linguistic features from our landmarks to represent the handshape, orientation, location and movement of the sign in the following manner:

1. **Handshape:** we select the fingertips, handpalm and wrist landmarks and compute the **distances and angles** between them. The distances are computed for landmark pairs ℓ_1, ℓ_2 using Euclidean distance:

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

For the calculation of the angle features, we use landmark triples (ℓ_s, ℓ_m, ℓ_e) which represent the start, middle and end point for which to compute the angle. We compute two types of angles: those within the fingers and those between the fingertips and the wrist. For the first angle type, we take the landmarks of the fingertip, the middle of the finger and the base of the finger at the handpalm. This results in $(finger_{tip}, finger_{mid}, finger_{base})$ as our triple for a given finger. The second type is computed using the landmarks of the fingertip, the base of the finger and the wrist. This triple can be described as $(finger_{tip}, finger_{base}, wrist)$. For both of these computations, the second listed landmark is the one where the angle is measured. We compute the angle for a tuple (ℓ_s, ℓ_m, ℓ_e) with the following formula:

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

Where $atan2$ refers to the 2-argument arctangent. We indicate with the subscript whether the x, y coordinate of the element is used. For instance, $\ell_{1,x}$ indicates the x coordinate of landmark ℓ_1.

2. **Orientation:** the orientation of the hand is captured by specifying an x-axis and y-axis in the hand. We define the x-axis as the line between the base of the index finger and the base of the pinky finger, with Mediapipe landmark indices $[5, 17]$. The y-axis is defined as the line between the base of the middle finger and the wrist, with Mediapipe indices $[0, 9]$. These axes within the hand are then compared to two other lines: drawn in the middle of the torso and one between the shoulders. This is done so we can compute the **orientation** relative to the way the signer is sitting. We use the Mediapipe pose model to extract the location of the hips and shoulders. The middle of the torso is taken as starting at the center of the hips and ending at the center of the shoulders.
The orientation is computed as follows. For each of the lines that have been drawn for the shoulders, torso and hands, we first compute the slope. Given a line ℓ that consists of a start and end point (ℓ_s, ℓ_e), we compute the slope s_ℓ of the line as:

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

From the slope values, s_1, s_2, of two lines, we can compute the angle between said two lines:

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

3. **Location:** for each frame, we locate the **position** of the wrist and take this to be the position of the hand. The wrist's location stays the same regardless of the handshape or orientation of the hand. This results in a simple but effective way to capture the location of the hand.
Additionally, the location of a sign is partially characterised by the interaction between the hands. Thus, we capture the location of the hands relative to each

other by calculating the horizontal and vertical distance between the wrists of both hands. We compute the difference between the x and y coordinates, resulting in two features.

4. **Movement:** we obtain the **velocity** of the hand by calculating the distance of the wrist between two consecutive frames. 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. The Euclidean distance is computed 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 repeat this 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.

For the features which are coordinates, we flatten their x, y coordinates to make them fit into a one-dimensional array.
In Table 4.1, we show the features and their respective indices. The *Ind. Mediapipe Landmarks* column shows which indices from the Mediapipe are used, while the *Type* column indicates what type of feature is computed.

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 4.1: Feature indices

Note that for features describing the hand orientation, we also use the Mediapipe pose model landmarks. In particular, we use pose indices 11 (left shoulder), 12 (right shoulder), 23 (left hip) and 24 (right hip).
We highlight any features which have been adapted from previous work. Underlined features are taken from Farhan and Madi [32]. All distance features have been taken from Shin, Matsuoka, Hasan, *et al.* [30]. The remaining features are all novel contributions by this thesis.

4.2.3 Final features

In summary, we use 120 features per frame for the linguistic condition, for which the extraction was described above. For the landmark features, we use all of the manual landmarks extracted by Mediapipe. One hand is described by 21 landmarks, for each of which we extract the x, y coordinates. Thus, we have $21 \cdot 2 = 42$ coordinates per hand, which results in 84 features per frame since we use the landmarks of both hands in our feature extraction.

4.3 Datasets

4.3.1 Corpus Nederlandse Gebarentaal

We use the *Corpus Nederlandse Gebarentaal* (CNGT) [42], [45] as our continuous video dataset for the sign spotting task. It consists of 72 hours of video footage of 104 signers of Dutch Sign Language (DSL), with about 15% having been annotated by linguists. This is equivalent to 162k annotations of about 3.2k signs. CNGT matches most of the requirements for datasets set out in [5] since it 1) uses native signers, 2) contains a variety of signers who are from different regions and of different ages and genders, which allows for an accurate depiction of DSL, and 3) provides footage of more 'natural', continuous signing, where the signers are in conversation and are not trying to sign in a more proper manner than usual [42].

4.3.2 Signbank

We have selected NGT Signbank [46] as our sign dictionary. This database was created for the specific purpose of annotating Dutch sign language corpora such as CNGT [46], which makes it a perfect companion for our chosen corpus. NGT Signbank contains phonological information about more than 4000 Dutch signs, such as the handshape, orientation, location and movement of the hand(s). The relevant phonological attributes that we use for our analysis are shown in Table 4.2.

4.4 Data Preprocessing

4.4.1 Corpus Nederlandse Gebarentaal

Sign spotting is a supervised task, thus, we need labeled data. For the purpose of our research, we are therefore only interested in the annotated data of the CNGT. When selecting only the annotated data, we are left with approximately 20 hours of video footage, with each video containing two signers. Next, we section the footage into two parts to separate the signers into their own segments. As a result, we have 40 hours of footage of individual signers.
The total number of annotations in CNGT is approximately 162k annotations of 3.2k unique signs. The indicated number of annotations counts the left and right hand separately and does not take into account that some signs are performed with both hands. If we fuse instances where the left and right hand sign the same sign at the same time into one annotation of a two-handed sign, we end up with about 122k annotations.
For each annotation in CNGT, we are provided with information about when the annotation starts and ends, as well as which sign is performed during this timespan. Some videos have been annotated using the Dutch names of the signs while other times the English names are used. To use a consistent naming convention for all signs, we ensure

Attribute name	Phonological category
Handedness	Handedness
Strong Hand	Handshape, Handedness
Weak Hand	Handshape, Handedness
Handshape Change	Handshape
Relation between Articulators	Location
Location	Location
Relative Orientation: Movement	Orientation
Relative Orientation: Location	Orientation
Orientation Change	Orientation, Movement
Contact Type	Movement
Movement Shape	Movement
Movement Direction	Movement
Repeated Movement	Movement
Alternating Movement	Movement

Table 4.2: NGT Signbank attributes used for the computation of linguistic distance

that all annotations use the Dutch names as follows. Given an annotation in the Corpus, we determine whether it matches with any of the English or Dutch names of the signs in NGT Signbank. If the annotation matches exactly with the Dutch name of a sign, we keep the annotation as-is. If it instead matches with the English name of a sign, we replace the name value of the annotation with the Dutch equivalent. For example, if consider the sign that means 'hello' ('HALLO'/'HELLO') and assume it is annotated in a given video as 'HELLO' (i.e. in English), we convert the annotations of the sign to the Dutch equivalent 'HALLO'.

The conversion to the Dutch naming convention is done in a strict manner, that is, we only use signs which already match exactly with the Dutch name of a sign, or those for which we know a Dutch equivalent. By extension, we remove any annotations for which it is uncertain what sign was performed or the performed sign is obscured from view, as such annotations use an additional marker (? and !, respectively). These annotations will thus not match exactly with the name of any sign and are discarded. About 700 annotations use an uncertainty or obscure marker, which is less than 1% of the total number of annotations.

Our next criterion for the annotation concerns the linguistics of the signs in the Corpus. In particular, we only make use of the annotations of a given sign if NGT Signbank contains linguistic information about the sign. This is done to ensure that we can compare the linguistic makeup of the signs, which is relevant for the evaluation method that we discuss in Section 3.2. After removing signs and annotations which do not have any linguistic annotation in NGT Signbank, we are left with 2.7k unique signs that make up 118k annotations. Our split is approximately 80/10/10, with 90k training, 10.5k validation and 9.5k test instances.

4.5 Preparing dataset

In the previous section, we extracted linguistic and landmark features from CNGT with Mediapipe. Next, we split the videos up by where the annotations occur, such that we have the features for each annotation.

4.5.1 Fixed-length inputs

Our model requires fixed-length inputs, meaning the number of frames should be consistent for each annotation. However, the annotations in our corpus are of variable lengths because signers typically perform signs at varying speeds. We computed the average number of frames the annotations last, which is about 10 frames. Thus, 10 frames is used as our target input length to make sure we capture this average. Not all annotations are already of the desired length and therefore, we do the following:

1. We crop to the annotation timestamp to make a video clip corresponding to the annotation.

2. If the input length is now exactly the desired length (10 frames), we add the clip to our dataset as-is.

3. If the input length is too short, we simply pad our input with zeros until we reach the desired length.

4. If the input length is too long, we undersample the given frames. First, we determine how many times the input length is larger than our target length. We define this value as i, and sample every i-th frame from the input. If i is not a round value, we first round it down to ensure that we retain as much of the input as possible. If we do not obtain exactly 10 frames after this step, we randomly remove values until the desired length is obtained.
   To illustrate how the undersampling works for such input lengths, let us assume that we have 37 frames of input, where we define its indices as:

   $$[0, 1, ..., 35, 36]$$

   Then, we find that $i = 3$, since 10 frames fit into 37 frames thrice ($10 \cdot 3 = 30$) without going over. This means we take every third frame:

   $$[0, 3, 6, ..., 33, 36]$$

   This leaves us with 13 frames, so we need to remove 3 more frames randomly to get 10 frames. If we remove frame $0, 3, 33$ randomly, the remaining frames are:

   $$[6, 9, 12..., 27, 30, 36]$$

4.5.2 Data augmentation

We want to give our model varied examples of the signs within the corpus. Ideally, one-handed signs are sometimes signed left-handed and other times right-handed. Similarly, for asymmetrical two-handed signs which have a dominant hand, it would be ideal if in the signed instances of the sign, the dominant hand is sometimes the right hand and other times the left hand. For the more frequent signs in CNGT, we can expect variety to occur naturally, but for some signs we have very few instances. As such, it is likely

that there is an imbalance in the handedness of these signs.
This is why we also add data augmentation to the train set while preparing the dataset. We achieve a variety in handedness by mirroring the x-coordinates of the landmarks. Given a landmark ℓ which we want to mirror horizontally, and the resolution of the video for which we extracted the landmark, we can mirror ℓ using the video width w:

$$mirror(\ell_x) = w - \ell_x - 1$$

However, this requires us to mirror our data before the feature normalisation, which has the disadvantage that it requires us to store the mirrored landmarks and apply the feature extraction and normalisation separately for them. To avoid this additional computational overhead, we opt to compute the mirroring after the normalisation. In Appendix A.2, we show that the following holds:

$$mirror(\ell_{centered,x}) = -1 \cdot \ell_{centered,x}$$

As such, we can horizontally mirror the normalised landmarks simply by multiplying their x coordinates with -1. We found that the mirror augmentation makes our model train in a more stable manner on the train and validation sets.

4.5.3 Creating batches and shuffling

Contrastive learning uses positive pairs and negative pairs to learn how to create embeddings. Positive pairs are groupings of the same sign, whereas negative pairs are groupings of different signs within a batch. We create stratified training batches to make sure that instances of the same sign, the 'positive' instances, occur together in a batch and are not separated as they would be when randomly assigning them to batches. To create these stratified batches, we use a batch generator. Such a generator does not permit shuffling the batches after being created. To allow for shuffling, we thus recreate the generator after each epoch and make sure that we shuffle the training data before using the new generator.

4.6 Sign spotting model

In line with the approach taken by Momeni, Varol, Albanie, *et al.* [22], we develop a model which learns to create embeddings from our input features. Embeddings are learned in such a manner that instances of the same sign result in similar embeddings while instances of different signs result in dissimilar embeddings. This approach is called *contrastive learning*.
The model that we choose for our experiments is a LSTM network. LSTMs are capable of extracting temporal information from data sequences and have been a popular tool in the field of natural language processing [47]. While more sophisticated architectures have become available in recent times, our goal is not to select the best model but rather to engineer meaningful features. We tested multiple configurations of our LSTM network and selected one which performs well for both the landmark and linguistic features.
In the end, our model uses the architecture shown in Figure 4.2. The first layer adds Gaussian noise to our input, to prevent overfitting by introducing variability in the data that is seen during training. This step is followed by a biLSTM layer with $2 \cdot 256 = 512$ nodes in total. We follow this layer up with two Dense layers, both of size 256, and use batch normalization between the Dense layers for training stability. The model uses a batch size of 128 and learning rate of 0.001 based on [22]. It is trained using the Adam

optimizer for 10 epochs, which is when the validation loss typically starts to converge. Our GPU is the NVIDIA GeForce GTX 1050 Ti and our CPU is the Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz. Our model is implemented using Tensorflow and we use a GPU compatible version of this software. Training for 10 epochs takes approximately 10 minutes using this setup.

```
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 bidirectional (Bidirectiona  (None, 512)              677888
 l)

 dense (Dense)                (None, 256)              131328

 batch_normalization (BatchN  (None, 256)              1024
 ormalization)

 dense_1 (Dense)              (None, 256)              65792

 batch_normalization_1 (Batc  (None, 256)              1024
 hNormalization)

=================================================================
Total params: 877,056
Trainable params: 876,032
Non-trainable params: 1,024
_________________________________________________________________
```

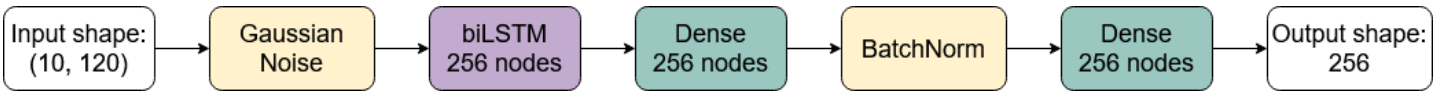

Figure 4.2: Model architecture

4.6.1 Loss function

Many different loss functions have been proposed for contrastive learning, most of which focus on self-supervised contrastive learning. Contrastive learning has as its goal that positive pairs, which are data points with the same label, should be kept close together, while negative pairs, or data points with a different label, should be kept far apart in embedding space. Since we have labeled data, we are able to make use of the supervised contrastive learning loss introduced in the paper by Khosla, Teterwak, Wang, *et al.* [48], SupCon. This method improves on self-supervised approaches by allowing for multiple positive examples based on labeled data instead of creating positive examples from unlabeled data. These changes result in a state-of-the-art loss function for supervised contrastive learning.
SupCon uses a hyperparameter, temperature, to determine whether it is more important to bring the positive examples together or to keep the negatives far apart. We use a temperature of 0.07 based on the work of [22], [49], [50].

4.7 Feature selection

Before we finalize our model, we first determine which of our linguistic features to use. This is done to ensure that we only use those features which are beneficial for training our model. The selection process is performed as follows. We first run our model using all possible features to get a baseline of the validation performance using our distractor-based evaluation. Next, we determine the importance of each feature by comparing to this baseline. We tried multiple methods for feature selection, namely ANOVA, permutation, perturbation and masking, and found that selecting features using masking consistently returned the same features on the validation set. With the other methods, the selected features were inconsistent over different model runs.

The masking of the features was done as follows. We mask one feature within the validation set by replacing it with random noise drawn from a Gaussian distribution ($\mu = 0, \sigma = 0.1$). While masking one feature, all other features are used as-is. This process is repeated for each feature. Then, we apply our evaluation and compare the validation set accuracy of our model to the accuracy of the baseline model. If the results improve or stay consistent, we conclude that the feature can be safely removed. We repeated the masking experiments five times, retraining the model each time to account for the fact that training is non-deterministic and feature importance may therefore vary. We only removed features if they were never found to contribute to the accuracy in any of these runs. All default model parameters are used except for the batch size, which we increase to 1024 to speed up the model predictions which have to be made for each masking experiment.

In Figure 4.3, we display one of the five runs of our masking experiment. The full results of all five runs can be found in Appendix A.4. We observe that the masking of the feature indices $45, 47, 49, 51, 53$ and 55 has the largest impact on the accuracy of the model. Interestingly, these features represent the y coordinates of the wrist and fingertips. On the other hand, the x coordinates of the same landmarks, with indices $46, 48, 50, 52, 54$, are found to not contribute to the performance of the model. One possible explanation for this could be that the horizontal location where a sign gets articulated is typically close to the horizontal bounds of the torso, whereas the vertical location differs more significantly between signs that are performed near the face, the shoulders or the stomach.

Only the vertical velocity of the wrist (index 58) is found to contribute to the model performance, whereas the other velocities (indices $56, 57$) do not. In a similar manner, only the vertical distance between the wrists (index 119) is found to be helpful in increasing the model accuracy, whereas the horizontal distance (index 118) does not have such an impact. Again, the horizontal dimension seems to not be as important for identifying a sign as the vertical dimension. Moreover, feature indices 8 and 25, which represent a handshape angle and distance respectively, are never found to increase the accuracy in any of our model runs. It is possible that these features are closely related to other extracted features, which allows us to remove them during masking without a loss of performance.

In conclusion, we remove the x coordinates of the wrist and fingertip landmarks, the horizontal and Euclidean velocity of the wrist, the horizontal distance between the wrists and two handshape features, because their masking does not decrease our model's performance. The removed features have the indices $[8, 25, 46, 48, 50, 52, 54, 56, 57, 118]$.

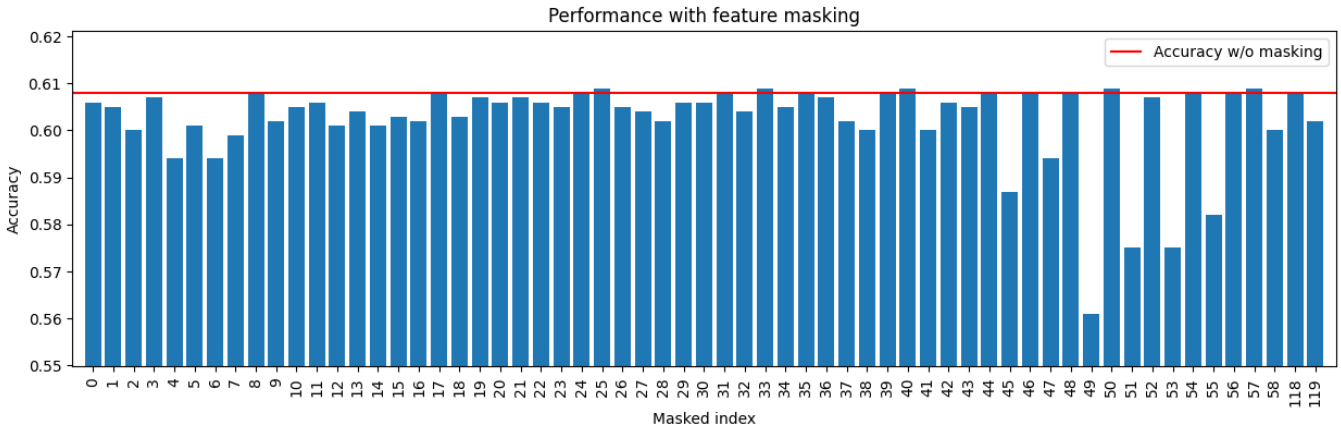

Figure 4.3: Example run of the masking experiment.

4.8 Test phase

After training our sign spotting model, we evaluate our model on a test set of videos which contain different signers from the train set. We use a sliding window with the same size as the model input, which is 10 frames, to slide over the videos. Then, we input each window into our model to obtain an embedding. This embedding of the test video's window can then be compared to the embedding of a sign that we saw during training to determine which sign the window may contain. However, if we were to compare all sign instances which were seen during training, that is, all 90k train set annotations, this would create a large overhead during this phase. As such, we first create *reference embeddings* from the train data as follows:

- Collect the non-mirrored train data points and create an embedding for each of them.
- For all train-set embeddings of a specific sign, we compute their distance to all other train-set embeddings of that sign. For instance, if we have the sign *SIGN-A*, we gather all train embeddings of *SIGN-A* and compute the distances between all of them. The embeddings which are on average most similar to all other embeddings, are then chosen to be the most representative embeddings for the sign (e.g. *SIGN-A*).
- We select the top 10% most representative embeddings of each sign. For each of the embeddings that are chosen to be representative, we average them to create one reference embedding for each sign.

Now, a reference sign can be compared to a given embedding that was created using the sliding window over a test set video. If the cosine distance of a reference sign and a sliding window chunk is below a certain threshold t, the sign is being spotted in that moment by our model. We determined a fitting threshold based on empirical tests on the validation set, which can be found in Appendix A.3, and found that $t = 0.2$ is suitable.

4.9 Evaluation

4.9.1 Validation

We developed an evaluation method to reflect user needs and to deal with an incomplete ground truth. We assumed that the performance on the distractors will provide us with a robust estimate of the performance on the unknown, full ground truth. Our reasoning to support this assumption is that the hardest cases should give us a conservative estimate of the performance and be less affected by changes in annotations when compared to selecting random cases.
To validate our reasoning, we demonstrate our evaluation for two types of distractors: '**standard**' distractors selected using our distance measure, and a set of **random** distractors. We select the random distractors by shuffling the ordering of the distractor candidates, and pick the top-f distractors from this randomized order. The shuffling is done in a seeded manner to ensure that we can reproduce our results.
For the standard group of distractors, we assume that the hardest known cases are included, while for the random group, both easy and hard cases may be present. Furthermore, we expect that the estimated performance on the random distractors will be less predictable when the annotations are updated. We thus aim to demonstrate that the randomly chosen distractors give a higher estimate of the performance than the distractors which are selected using the distance measure, and that the estimate of the performance of our model will fluctuate more with the random distractors than the standard ones.
To demonstrate the validity of our reasoning, we make use of the model trained on the landmark features. We want to avoid validating our evaluation using a set of features that is also linguistically motivated and could therefore bias the results in our favor. By selecting the landmark features to test the validity, we aim to ensure that the assessment is performed in an unbiased manner.
After training our model, we apply our validation experiment as follows. First, we select the distractors for both the standard and random condition, as described above. We apply our evaluation with the chosen distractors, which gives us a baseline performance estimate of our model for both of the conditions. Then, we investigate the effects of updating the annotations by randomly removing a fixed fraction of distractors from both sets. By doing this, we are effectively creating a hypothesized previous version of our dataset in which some annotations were not available yet. In other words, we look at a more incomplete version of our ground truth and specifically look at those versions for which the two distractor sets are equally affected, to allow for a fair comparison between them. After removing a fixed fraction of annotations from both sets, we rerun the distractor selection process to replace the removed distractors, and apply our evaluation method again to get the updated performance for both sets. This estimate of the performance indicates how robust our distractor sets are to a more incomplete ground truth than the one we have access to, and we assume that this extends to the scenario where our known ground truth was updated with more annotations.

4.9.2 Tolerance window size

The size of the tolerance window, *tol*, can take on different values depending on the exact domain we want to use a tolerance-based evaluation for. No specific tolerance window sizes have been determined for sign language processing and as such, we consult the related field of audio segmentation for the size of the tolerance window. As discussed in Chapter 2, two sizes that are typically used in this field are 0.5 and 3 seconds [36],

[37]. We determined that the average sign duration in our dataset is 0.4 seconds, which is close to the stricter audio tolerance window size. Moreover, choosing a small window size decreases the chance that the tolerance windows of different signs overlap. During trial runs with larger tolerance window sizes, we found that the number of distractors that could be selected was noticeably lower. Therefore, we opt for a tolerance window of 0.4 seconds. Considering that the footage in our corpus runs at 25 frames per second, this is equivalent to 10 frames.

4.9.3 Handling overlap and double predictions

We note that the current definition of our evaluation does not restrict that a prediction may overlap with the tolerance window of more than one target annotation and/or distractor. Such a situation makes it difficult to determine which annotation was spotted, particularly if a distractor and a target are spotted by the same prediction and it is unclear whether we have a FP or a TP prediction. To remedy this problem, distractors for a given sign are selected only if their tolerance windows do not overlap with each other or with the tolerance windows of the target sign.
Another restriction that we place on our evaluation is that we only count the first FP or TP spotting within a tolerance window. This is done to ensure we do not evaluate one tolerance window multiple times. The assumption we make is that the other predicted spottings can be ignored if the first one is evaluated, since they could be aggregated into one prediction if they are close enough to each other to fall within the same tolerance window. We leave the process of aggregating the predictions to future work.

4.9.4 Linguistic distance

Our chosen measure of distance for the selection of distractors is linguistic distance. To compute this distance, we use NGT Signbank to get the phonological properties of each sign. We previously described which attributes we selected from this dataset (see Section 4.3.2).
Given the phonology of the signs in NGT Signbank, we now define our linguistic distance measure. Given a pair of two signs a, b, we initialize their distance to 0. Then, we can compare the signs for each of the attributes. For instance, we can find whether the handedness of sign a and sign b. If a, b have the same value for some attribute, they are linguistically similar for this phonological aspect and we therefore do not increase the linguistic distance between a, b. If they have a different value for some attribute, for instance that a is one-handed whereas b is two-handed, we increase the distance by 1.

Phonological attribute	Missing values
Handedness	9.95%
Strong Hand	10.03%
Weak Hand	59.44%
Handshape Change	87.26%
Relation between Articulators	90.74%
Location	15.22%
Relative Orientation: Movement	38.42%
Relative Orientation: Location	83.72%
Orientation Change	87.04%
Contact Type	64.78%
Movement Shape	89.11%
Movement Direction	44.58%
Repeated Movement	0.0%
Alternating Movement	0.0%

Table 4.3: Percentage of missing values for each phonological attribute in NGT Signbank

Handling unknown values

Unfortunately, the phonology of a gloss or sign in NGT Signbank is often not completely annotated as can be seen in Table 4.3. Manually annotating this data is beyond the scope of this thesis and we leave this contribution to future work. As such, we have to decide how to deal with two scenarios for a given comparison between an attribute for two signs:

1. The attribute is specified for one of the signs but is not specified for the other sign.
2. The attribute is not specified for either of the two signs.

First, we ascertain why attributes are sometimes not specified for a sign. Some values are left out simply because no annotator has had the opportunity to annotate them, yet other times a value is intentionally left out because it is not seen as relevant[2]. For example, the handshape of the weak hand is not specified for one-handed signs, since the singular signing hand is determined to be the strong hand.
Based on this observation, we determine that scenario 1, where we compare an unknown value with a known one, can be dealt with by adding to the linguistic distance. In other words, we determine that two signs are distant for a given property if one value is unknown while the other is specified. We come to this decision because of two reasons. First, if the attribute is intentionally left out because it is irrelevant, that makes it inherently different from a value that is filled in because it is relevant. Second, if the attribute for one sign is missing but is relevant, it is not statistically likely that it has the same phonology for this property as the other sign.
We can apply similar reasoning to scenario 2. If both values are left out because they are irrelevant, then they can be regarded as being the same for this attribute. Additionally, if both values are unknown but relevant, it is statistically more likely that they match because they are both unknown than if one of the values was known. In conclusion,

[2]According to personal correspondence with an author of the dataset.

we decide for scenario 2 that a comparison between two unknown values can be ignored instead of counting it towards the linguistic distance between the signs. To demonstrate

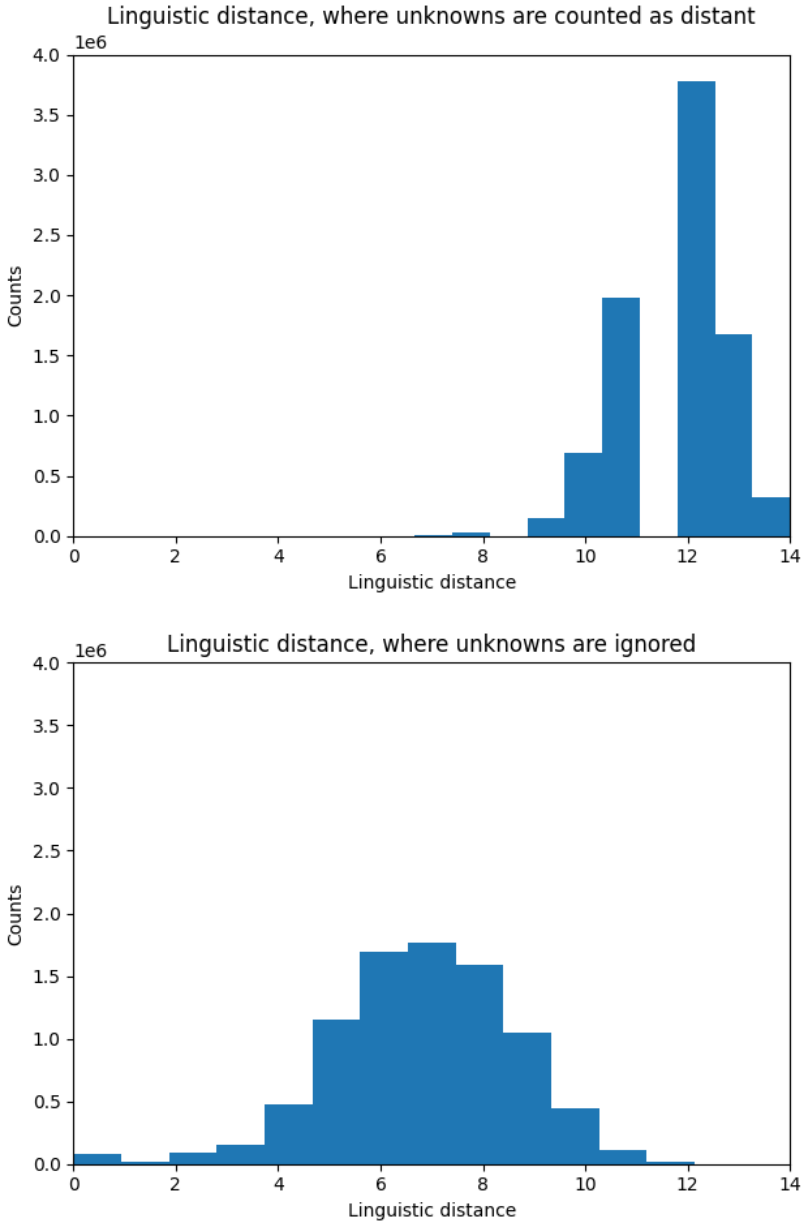

Figure 4.4: Histogram of the distribution of linguistic distances where we either count unknown values as distant (top) or ignore them (bottom)

that our reasoning is sound, we now show the impact of choosing to handle the unknown values by ignoring them, as compared to counting them as distant. Figure 4.4 shows the distribution of linguistic distances for the two conditions. In the histogram of the distances where we ignore the unknowns, we see that the distribution is more or less normally distributed. On the other hand, the histogram of the condition where we count the unknowns as distant, we can see that the distribution is heavily skewed to the right, indicating that signs are on average found to be more distant. Because the normal distribution is often used to represent natural phenomena, we use this as a first argument for why ignoring the unknowns is more sensible.

We also investigated the effects of ignoring the unknowns on two different categories of sign similarity: homonyms and minimal pairs. Homonyms are signs that appear visually similar to a given sign, though they may have different phonological properties. At the same time, homonyms may also be visually and phonetically identical to a sign but have a different meaning. Minimal pairs are signs which are phonologically similar to each other, thus only a minimal number of linguistic attributes should be different. Our assumption is that homonyms and minimal pairs will typically have a small linguistic distance to the sign to which they are visually or phonetically similar. We make use of NGT Signbank to obtain the known homonyms and the minimal pairs for all signs.
Our analysis uses the rankings of the homonyms and the minimal pairs. For each sign S, we determine whether it has any known homonyms and/or minimal pairs. We then rank the most similar signs to S in terms of their linguistic distance, as computed using our linguistic distance measure that builds on NGT Signbank. Typically, the homonyms and minimal pairs of S are expected to have a relatively low distance to the given sign. For a ranking system where rank 0 is chosen as signifying the sign closest to a target, we thus would expect the homonyms and minimal pairs to have a low rank.

Handling unknowns	**Mean rank homonyms**	**Median rank homonyms**	**Mean rank minimal pairs**	**Median rank minimal pairs**
Ignore	27.88 ± 239.8	0	2.75 ± 36.54	1
Count as distant	19.85 ± 203.18	1	4.82 ± 35.35	2

Table 4.4: Ranking statistics of minimal pairs and homonyms given unknown values

In Table 4.4, the ranking statistics of the minimal pairs and homonyms are displayed. For the mean values, we provide the standard deviation in the form $mean \pm std$.
The median rankings for both types of similar signs are lower and thus more representative when we use the strategy of ignoring the unknowns. In particular for the minimal pairs we can see that the mean is lower for the condition where we ignore the unknowns. Note that the rank of the homonyms is higher for this condition, but that the homonyms may be quite different phonetically from the signs for which they are homonyms.
We also investigated the absolute distance of the homonyms and minimal pairs for each target sign. In Table 4.5, we see that the mean distance of both types of similar signs is much lower when ignoring the unknown values. Once again, the choice for ignoring unknowns is supported by this analysis.

Handling unknowns	**Mean dist. homonyms**	**Median dist. homonyms**	**Mean dist. minimal pairs**	**Median dist. minimal pairs**
Ignore	0.40 ± 1.16	0	1.02 ± 0.21	1
Count as distant	6.47 ± 1.51	1	8.21 ± 1.07	8

Table 4.5: Distance statistics of minimal pairs and homonyms given unknown values

Finally, we also determine how representative the distractors are when we make the choice to ignore unknowns as compared to counting them. We select distractors in the same manner as will be discussed in Section 4.9.5. The results are shown in Table 4.6. We see that the linguistic distances have a lower mean and median distance for the condition where we ignore the unknowns as opposed to the condition where we count

them as distant. Since the distractors should typically be selected to be similar to a target sign, their linguistic distance to this target should be low. Again, this supports our choice for ignoring the unknowns instead of counting them as distant.

Handling unknowns	**Mean linguistic distance**	**Median linguistic distance**
Ignore	3.2 ± 1.63	3
Count as distant	9.28 ± 1.25	9

Table 4.6: Linguistic distance statistics of distractors given unknown values

Summarizing, we found that ignoring the comparison of unknown values results in a better representation of the linguistic distance between signs than when we let this comparison contribute to the linguistic distance.

4.9.5 Balancing distractors and target sign annotations

Previously, we introduced distractors, signs which are similar in some manner to a target sign. We implemented a novel linguistic distance measure to determine the similarity of the candidate distractors to a target. In this section, we consider how to select the actual distractors from the candidates.
While it may seem ideal to use all candidate distractors for a given target sign, we note that the annotations of our corpus are not complete. We can thus, at most, use all known candidates, however, both hard and easy cases will be included in this manner, and it is possible that our estimation of a model's performance will be less conservative than we desire. To ensure our estimate of the performance is conservative, we decide to only select those which are most similar to the target signs. Our assumption is that the performance of a model on this subset reflects the performance on the unknown, complete ground truth because the chosen set of distractors are chosen to be hard negative cases for our model. Thus, the performance on these difficult examples should tell us how well the model performs on easier counterexamples as well.
To select only the hardest distractors, we strike a balance between the number of distractors and target signs. For a target sign S in a given video, we determine its frequency f. Then, we find all other signs that are performed in the video. The annotations of these signs are then determined to be distractor candidates. As we explained before, we filter out any candidates which overlap with the target sign or with each other. From the remaining set, we select the top-f most similar distractors in terms of their linguistic distance to S.
The ratio of distractors that is used for a given frequency f can naturally be tweaked depending on the use case of the model. For instance, it may be sensible to increase the number of distractors that are used if the model should be particularly sensitive. We leave the analysis of the number of distractors that should be used to future work.
For our test set that consists of 9613 annotations, we were able to select 8418 distractors which do not overlap with each other or with a corresponding target sign. Evidently, we are not able to find the top-f distractors for each sign due to this restriction regarding annotation overlap. This imbalance can be dealt with in two ways. First, we may use all 9613 annotations and the selection of 8418 distractors as-is. Second, we can only select target signs with frequency f in a given target video, if we are able to find f distractors for them. In other words, we try to balance the number of annotations and distractors by filtering out the annotations which do not have enough distractors. We

compare these options below.
To investigate how our evaluation method is affected by the balancing of the number of distractors and annotations, we run our distractor-based evaluation on the validation set, both with and without balancing. For the balanced condition, we find that there are 7507 annotations in the validation set for which we are able to find exactly 7507 distractors. For the imbalanced condition, we find 10571 annotations with 9406 distractors. The evaluation is applied to the model trained on the linguistic features. All model parameters were kept to their default values, and we use the optimal spotting threshold ($t = 0.2$). For the tolerance window size, we report at the strictest tolerance level, namely 0.3 seconds.
In Table 4.7, the results of our balancing analysis are shown. Accuracy, precision and recall are abbreviated as Acc, Prec and Rec, respectively. Our results show that the balanced selection of distractors yields a higher performance for all of our metrics. This is due to the balanced condition having fewer target annotations and corresponding distractors. As such, the performance of our model appears to be more optimistic when selecting balanced distractors. Since we want our distractor-based evaluation to provide us with difficult cases to reflect the performance on the full ground truth, we conclude that it is more sensible to use the imbalanced condition for our purposes.

	TP	**FN**	**FP**	**TN**	**Acc**	**Prec**	**Rec**
Balanced condition	3767.0	3740.0	949.0	6558.0	0.688	0.799	0.502
Imbalanced condition	4947.0	5624.0	1297.0	8109.0	0.654	0.792	0.468

Table 4.7: Comparison of the imbalanced and balanced distractor selection

4.9.6 Confusable signs

The Δ properties for our dataset can be determined from the phonological data in NGT Signbank. In Figure 4.8, we display the number of occurrences of the Δ properties within the test set. Apparently, certain Δ properties are much more common that others. We think this is due to two factors. The first factor is the number of confusable signs that exist with a specific Δ property. For example, there may be only a few sign pairs for which only the handedness differs. The second factor is the frequency of the confusable signs within our dataset. If confusable signs with a given Δ property are not common in our corpus, the Δ property will naturally also have a low frequency.

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 4.8: Frequency of Δ properties in our test set

Chapter 5

Experimental setup

In this chapter, we elaborate on the experiments we performed to answer our research questions. We show how the validation of the distractor-based evaluation is performed based on a set of hardest and random distractors. Moreover, we explain how we compare the feature sets in terms of their performance for the distractor-based evaluation and the confusable signs.
Unless stated otherwise, our experiments are performed using the standard hyperparameters for each model and use a spotting threshold of $t = 0.2$ that we selected based on experiments on the validation set.

5.1 Validation of distractor-based evaluation

In our validation experiments for the distractor-based evaluation, we drop out random annotations from the random and the standard distractors to determine how robust both selection methods are to changes in the dataset. We select four different ratios of drop out to test this: $[0.1, 0.25, 0.5, 0.75]$, e.g. 0.1 represents that 10% of the selected distractors is dropped randomly and replaced with other candidate distractors. For each ratio, we repeat the experiment 10 times so we can assess whether our findings are consistent across multiple random dropouts of distractors.
Because the distractors represent the negative counterparts to a target sign, the FP and TN evaluations are affected by a change in the selection of the distractors. On the other hand, the TP and FN evaluations, and by extension the recall, are not affected by changing the distractors and stay consistent between the two conditions. As such, the relevant results for our validation are the FP and TN evaluations, as well as the accuracy and precision. We only report on said relevant metrics in Chapter 6, while the full tables including the non-relevant metrics can be found in the Appendix (A.5). Furthermore, we report on the mean linguistic distance of both sets of distractors for each dropout experiment to demonstrate how the difficulty of the distractor sets changes according to our linguistic distance measure.

5.2 Feature set comparison

The next set of experiments focuses on the differences between the sets of features. We compare the performance of three sets of features: the linguistic features, the landmark features and a combination of both. The three feature sets are compared using the confusable signs as well as the distractor-based evaluation method. For the distractor-based evaluation, we compare the three feature sets in pairs and apply the comparison for each of our metrics (e.g. accuracy, TP). We perform five model runs for each of the feature sets to get an average performance for each metric and to allow us to perform a t-test that determines whether the difference in performance between two sets is significant. The results of the experiments are shown in Section 6.2.1.
For the confusable sign experiments, we start by selecting the best trained model for

each set based on the validation set accuracy. We then apply the evaluation method on the confusable signs to obtain the FP and TN evaluations. Because the confusable signs differ in only one aspect from the target signs, we are able to assess how many FPs each model has for a given Δ property. We use McNemar's test, which tests if two conditions are significantly different from each other, to determine whether the difference in the FPs output by the models is significant. In this manner, we assess how well each phonological property is represented by our feature sets. The experiments for the confusable sign analysis are performed in Section 6.2.2.

Chapter 6

Results

6.1 Validation of distractor-based evaluation

To validate the distractor-based evaluation, we compare the usage of the **standard** distractor selection, which uses the linguistic distance measure, to selecting **random** distractors. Recall that we build on the assumption that the former selection method will choose harder cases while we expect the latter method to choose both easy and hard cases. Therefore, we expect that the random distractors will result in a higher performance estimation and will be less robust to changes in the annotations. Here, we investigate whether this assumption holds.

We performed experiments to determine the impact of dropping a fixed ratio of the random and standard distractors, on the output of the performance estimation. The results are reported at the four dropout ratios, $[0.1, 0.25, 0.5, 0.75]$, as mentioned. Table 6.1 and 6.2 contain the results of the dropout experiments for a dropout ratio of 0.1. We also display the results for a dropout ratio of 0.75, as shown in Table 6.3 and Table 6.4. The other dropout experiments, which are not displayed here because they are consistent with the aforementioned results, can be found in Appendix A.5. We note that *Dropout x* indicates the x-th dropout experiment that was performed, whereas *No dropout* indicates the evaluation with the original, unaltered selection of distractors.

Based on these results, we make a few observations. Our first observation is that the number of distractors, which we obtain from adding up the FP and TN evaluations, is not consistently the same. We speculate that this is the case because some distractor candidates have more overlap in their tolerance windows with other candidates and if they are selected, this means the overlapping candidates can no longer be selected. We also observe that the accuracy, precision and f1-score are consistently estimated to be higher using the random distractors as opposed to the standard ones. This is a result of the number of FP evaluations being about half as many for the random condition. A possible explanation for this observation may be the difference in the mean linguistic distance for the distractor sets, since the random distractors have, on average, a higher linguistic distance to the target signs than the standard distractors. Since the random distractors always give a higher estimate of the model performance, we may conclude that the standard distractors give a more conservative estimate. This is in line with our expectations.

Next, we investigate whether the standard distractors result in a more robust performance estimate than the random distractors when the annotations are updated. When comparing the results reported for dropout = 0.1 and dropout = 0.75, we find that the random distractors give a more similar estimate to the condition with no dropout than the standard distractors. We note that there is a jump in the performance estimate when we use a dropout of 0.75 with the standard distractors, but that this change in the estimation is consistent over multiple dropout experiments. That is, approximately the same results are reported for all of the evaluations for a given dropout ratio. We hypothesize that this is because dropped distractors have to be replaced with annotations that have a higher linguistic distance to our target signs. Indeed, the mean linguistic

distance is reported to be higher when we perform the dropout experiments.
Furthermore, the standard distractors still consistently result in a performance estimate that is more conservative than the estimate using random distractors. In fact, there are some dropout experiments, such as 'Dropout 8' and 'Dropout 9' with dropout = 0.75, where the random distractors give a more conservative estimate with dropout than without. Similarly, the mean linguistic distance is typically lower when we drop distractors from the random set. Thus, we confirm that the random distractors contain a mix of both hard and easy cases.
In conclusion, our experiments support our hypothesis about the effectiveness of the distractor-based evaluation. The evaluation using the standard distractors consistently resulted in a more conservative estimate of the performance of our sign spotting model compared to the random distractors. Additionally, the standard distractors gave more consistent results in the performance estimation over consecutive dropout experiments. These findings validate our assumptions and demonstrate the importance of using linguistic insights in the development of an evaluation that is robust to an incomplete ground truth.

	FP	**TN**	**Accuracy**	**Precision**	**F1-score**	**Mean ling. dist.**
No dropout	1777	7910	0.685	0.751	0.637	3.007
Dropout 1	1770	7903	0.685	0.752	0.637	3.066
Dropout 2	1748	7932	0.686	0.754	0.638	3.064
Dropout 3	1753	7923	0.686	0.754	0.638	3.064
Dropout 4	1737	7935	0.686	0.755	0.638	3.068
Dropout 5	1761	7922	0.685	0.753	0.638	3.071
Dropout 6	1725	7953	0.687	0.757	0.639	3.062
Dropout 7	1735	7940	0.687	0.756	0.639	3.069
Dropout 8	1734	7938	0.687	0.756	0.639	3.06
Dropout 9	1739	7941	0.687	0.755	0.638	3.069
Dropout 10	1756	7918	0.686	0.753	0.638	3.066

Table 6.1: Dropout = 0.1 for **standard** distractors

	FP	TN	Accuracy	Precision	F1-score	Mean ling. dist.
No dropout	962	8728	0.727	0.848	0.669	5.807
Dropout 1	966	8707	0.726	0.848	0.669	5.792
Dropout 2	965	8712	0.726	0.848	0.669	5.794
Dropout 3	964	8717	0.726	0.848	0.669	5.807
Dropout 4	976	8702	0.726	0.846	0.669	5.81
Dropout 5	944	8726	0.727	0.85	0.67	5.803
Dropout 6	948	8727	0.727	0.85	0.67	5.809
Dropout 7	975	8703	0.726	0.846	0.669	5.799
Dropout 8	992	8680	0.725	0.844	0.668	5.803
Dropout 9	977	8705	0.726	0.846	0.669	5.821
Dropout 10	943	8734	0.728	0.851	0.670	5.802

Table 6.2: Dropout = 0.1 for **random** distractors

	FP	TN	Accuracy	Precision	F1-score	Mean ling. dist.
No dropout	1777	7910	0.685	0.751	0.637	3.007
Dropout 1	1484	8096	0.698	0.783	0.648	3.522
Dropout 2	1474	8100	0.699	0.785	0.649	3.52
Dropout 3	1500	8064	0.697	0.782	0.648	3.52
Dropout 4	1447	8115	0.7	0.787	0.65	3.517
Dropout 5	1465	8105	0.699	0.786	0.649	3.513
Dropout 6	1469	8114	0.699	0.785	0.649	3.522
Dropout 7	1473	8097	0.698	0.784	0.649	3.517
Dropout 8	1486	8072	0.698	0.783	0.648	3.519
Dropout 9	1473	8110	0.699	0.785	0.649	3.527
Dropout 10	1486	8095	0.698	0.783	0.648	3.523

Table 6.3: Dropout = 0.75 for **standard** distractors

	FP	TN	Accuracy	Precision	F1-score	Mean ling. dist.
No dropout	962	8728	0.727	0.848	0.669	5.807
Dropout 1	908	8662	0.728	0.855	0.672	5.748
Dropout 2	915	8651	0.727	0.854	0.671	5.769
Dropout 3	878	8699	0.729	0.859	0.673	5.78
Dropout 4	916	8650	0.727	0.854	0.671	5.774
Dropout 5	907	8671	0.728	0.855	0.672	5.758
Dropout 6	926	8631	0.727	0.853	0.671	5.744
Dropout 7	891	8677	0.729	0.858	0.673	5.766
Dropout 8	944	8626	0.726	0.85	0.67	5.773
Dropout 9	941	8635	0.726	0.851	0.67	5.748
Dropout 10	914	8663	0.728	0.854	0.671	5.772

Table 6.4: Dropout = 0.75 for **random** distractors

6.2 Feature set comparison

6.2.1 Distractor-based evaluation

To determine when each set of features is beneficial to use, we first determine the performance of the sets for the distractor-based evaluation. In Table 6.5, we show the results of running the evaluation in terms of the binary evaluations (TP, FN, TP, TN) as well as the accuracy, precision, recall and F1-score. We compare the feature sets in pairs, where *Set 1* and *Set 2* specify which features are used. The bold values in the *Set 1 avg.* and *Set 2 avg.* columns indicate which model is better on average, whereas those in the *p-value* column highlight significant differences ($p < 0.05$) between the two sets.

Metric	Set 1	Set 2	Set 1 avg.	Set 2 avg.	p-value
TP	landmarks	linguistic	5251.0	**5256.8**	0.841
TP	landmarks	combined	5251.0	**5264.4**	0.705
TP	linguistic	combined	5256.8	**5264.4**	0.855
FN	landmarks	linguistic	4457.0	**4451.2**	0.841
FN	landmarks	combined	4457.0	**4443.6**	0.705
FN	linguistic	combined	4451.2	**4443.6**	0.855
FP	landmarks	linguistic	**1742.0**	1758.6	0.462
FP	landmarks	combined	1742.0	**1692.4**	**0.002**
FP	linguistic	combined	1758.6	**1692.4**	**0.017**
TN	landmarks	linguistic	**7945.0**	7928.4	0.462
TN	landmarks	combined	7945.0	**7994.6**	**0.002**
TN	linguistic	combined	7928.4	**7994.6**	**0.017**
Accuracy	landmarks	linguistic	0.680	0.680	0.545
Accuracy	landmarks	combined	0.680	**0.684**	0.098
Accuracy	linguistic	combined	0.680	**0.684**	0.077
Precision	landmarks	linguistic	**0.751**	0.749	0.339
Precision	landmarks	combined	0.751	**0.757**	**0.001**
Precision	linguistic	combined	0.749	**0.757**	**0.005**
Recall	landmarks	linguistic	0.541	**0.542**	0.846
Recall	landmarks	combined	0.541	**0.542**	0.741
Recall	linguistic	combined	0.542	**0.542**	0.891
F1-score	landmarks	linguistic	0.629	0.629	0.921
F1-score	landmarks	combined	0.629	**0.632**	0.340
F1-score	linguistic	combined	0.629	**0.632**	0.378

Table 6.5: Feature set comparison results for the test set

Our observations are as follows. First, we note that the only significant differences are found to be between the combined feature model and the other models. In particular, the combined features result in a significantly better performance in terms of the FP and TN evaluations, as well as precision. Since precision is calculated as $\frac{TP}{TP+FP}$ and $TN = distractors - FP$, we conclude that the difference in performance is directly caused by the number of FPs that the models produce. Thus, we find that the combined feature model is significantly better than the other models in terms of its ability to distinguish distractors from a given target sign. The models trained on the linguistic features and landmark features are never shown to be significantly different and as such, we conclude that they are comparable to each other.

In summary, our comparison of the feature sets demonstrates that the model trained on the combined features stand out as the most effective in terms of its ability to correctly ignore distractors. It thus surpasses the other models in performance, indicating that a combination of knowledge-based features and the original landmarks has the potential to increase a model's effectiveness. Simultaneously, we maintain the explainability of the chosen features because the linguistic features are included in the combined model.

Still, all feature sets result in a considerable number of FN and FP evaluations, which shows that there is room for improvement regardless of which features we choose.

6.2.2 Comparison using confusable signs

To compare our feature sets in terms of their ability to represent the phonological properties of signs, we conducted experiments with the confusable signs. The results are presented in Figure 6.1. We indicate the results in percentages, which are computed by counting how often the confusable signs with each Δ property, as shown in Table 4.8, 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 property, are falsely spotted.

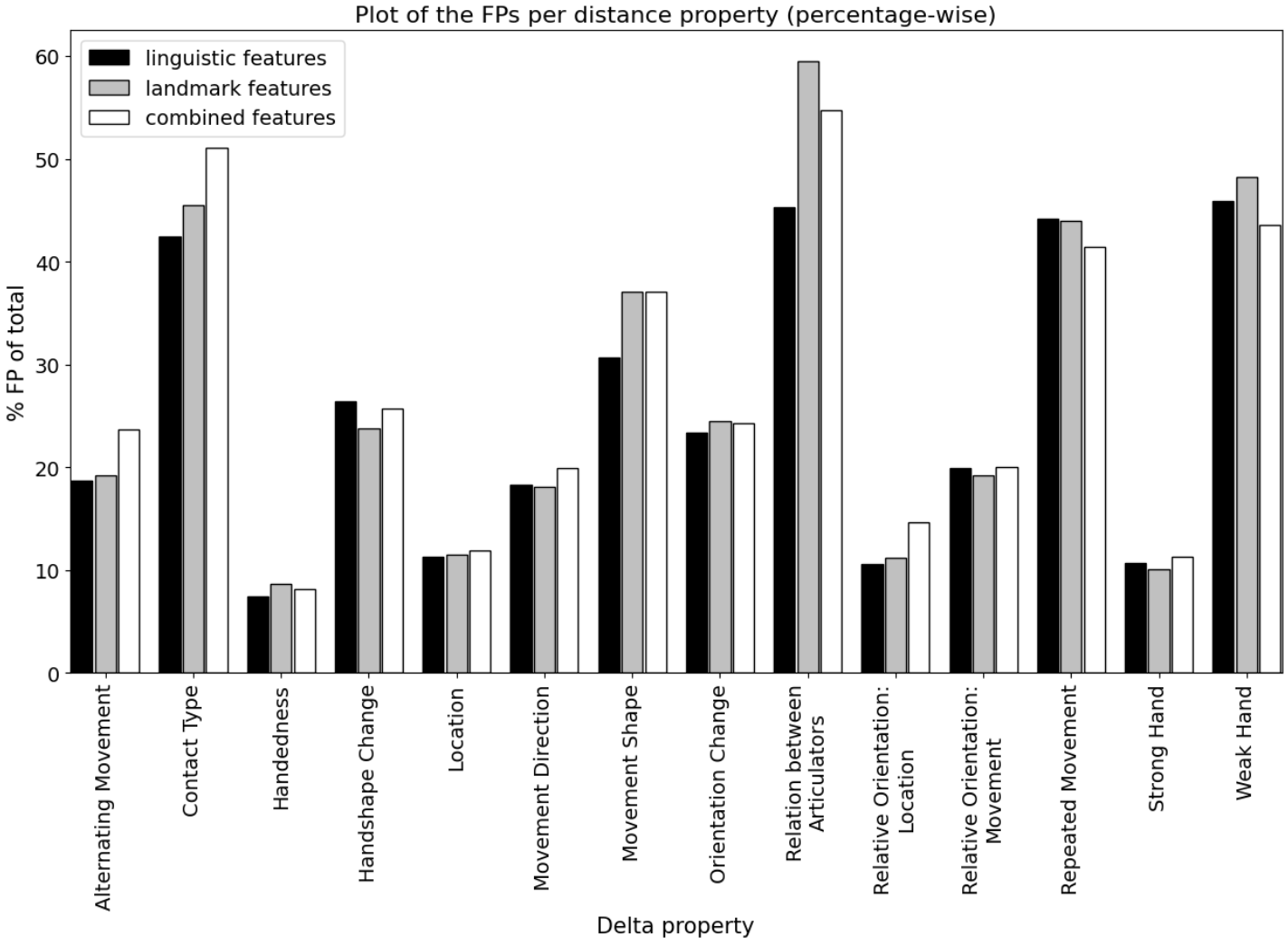

Figure 6.1: Percentage of confusable signs, per Δ property, that are falsely spotted per feature set

Our observations reveal several key findings about the feature sets. Firstly, we find that the linguistic features typically result in fewer or a similar amount of FP spottings of the confusable signs compared to the landmark features. Only the *Handshape change* property results in a clear difference between the percentage of FPs between the two models. It thus appears that the linguistic features are overall more capable of representing the

phonological properties than the landmarks. Secondly, we observe that the combined features often result in a comparable or higher number of FPs compared to the other two models. For example, the number of *Movement Shape* FPs is approximately the same between the landmarks and combined features. Two exceptions are the *Repeated Movement* and *Weak Hand* properties, for which the combined features result in fewer FPs than both of the other feature sets.

We performed McNemar's test to analyze for which Δ properties there was a significant difference in performance between the models, and display the results in Table 6.6. Significant p-values ($p < 0.05$), as well as the feature set with the lowest number of FPs, are indicated in bold. Notably, the linguistic and landmark features are never found to be significantly worse than the combined features in terms of the number of FP evaluations for a given Δ property. In fact, they are often found to have significantly fewer FP evaluations compared to the model that was trained on the combined features. For instance, the combined features result in approximately 300 more FPs for the *Movement Direction* property, which is found to be significant. The linguistic model is furthermore found to produce significantly fewer FPs compared to the landmark model for some Δ properties, and is never found to be significantly worse except for the *Strong Hand* property, for which the landmark model produces significantly fewer FPs than the linguistic features. Overall, both the linguistic and landmark features produce better linguistic representations than the combined features. Furthermore, the linguistic features typically provide the best representation of the phonological properties, but there is still room for improvement given the number of FP evaluations that occur. In contrast, the combined feature model was found to outperform the other models for the distractor-based evaluation. We conclude that the combined features are the better choice if we want to select the model with the best performance, but that the other models are more beneficial to choose when explainability and linguistic representativeness are desired.

Set 1	Set 2	Δ property	FP (Set 1)	FP (Set 2)	p-value
landmarks	linguistic	Alternating Movement	35	**34**	1.0
landmarks	combined	Alternating Movement	**35**	43	0.21592
linguistic	combined	Alternating Movement	**34**	43	0.18845
landmarks	linguistic	Contact Type	105	**98**	0.4701
landmarks	combined	Contact Type	**105**	118	0.12443
linguistic	combined	Contact Type	**98**	118	**0.0126**
landmarks	linguistic	Handedness	371	**319**	**0.01019**
landmarks	combined	Handedness	371	**349**	0.28637
linguistic	combined	Handedness	**319**	349	0.10607
linguistic	combined	Handshape Change	79	**77**	0.89176
landmarks	linguistic	Handshape Change	**71**	79	0.40278
landmarks	combined	Handshape Change	**71**	77	0.55569
linguistic	combined	Location	**2034**	2145	**0.00938**
landmarks	linguistic	Location	2085	**2034**	0.24947
landmarks	combined	Location	**2085**	2145	0.17061
linguistic	combined	Movement Direction	**3032**	3297	**0.00000**
landmarks	combined	Movement Direction	**2992**	3297	**0.00000**
landmarks	linguistic	Movement Direction	**2992**	3032	0.44136
linguistic	combined	Movement Shape	**230**	278	**0.00014**
landmarks	combined	Movement Shape	278	278	0.93116
landmarks	linguistic	Movement Shape	278	**230**	**0.00012**
landmarks	combined	Orientation Change	139	**138**	1.0
landmarks	linguistic	Orientation Change	139	**133**	0.62722
linguistic	combined	Orientation Change	**133**	138	0.66804
landmarks	combined	Relation between Articulators	25	**23**	0.72367
linguistic	combined	Relation between Articulators	**19**	23	0.28884
landmarks	linguistic	Relation between Articulators	25	**19**	0.11385
landmarks	combined	Relative Orientation: Location	**192**	251	**0.00008**
linguistic	combined	Relative Orientation: Location	**181**	251	**0.00001**
landmarks	linguistic	Relative Orientation: Location	192	**181**	0.50687
landmarks	linguistic	Relative Orientation: Movement	**544**	566	0.33272
landmarks	combined	Relative Orientation: Movement	**544**	569	0.26469
linguistic	combined	Relative Orientation: Movement	**566**	569	0.92064
landmarks	combined	Repeated Movement	460	**434**	0.10512
linguistic	combined	Repeated Movement	462	**434**	0.07249
landmarks	linguistic	Repeated Movement	**460**	462	0.94896
linguistic	combined	Strong Hand	**3759**	3969	**0.00054**
landmarks	linguistic	Strong Hand	**3537**	3759	**0.00061**
landmarks	combined	Strong Hand	**3537**	3969	**0.00000**
landmarks	combined	Weak Hand	41	**37**	0.38648
landmarks	linguistic	Weak Hand	41	**39**	0.78927
linguistic	combined	Weak Hand	39	**37**	0.77283

Table 6.6: McNemar's test for the Δ properties

Chapter 7

Discussion

In this thesis, we aimed to increase the explainability of sign spotting systems using a knowledge-based approach. We demonstrated how sign language linguistics could be incorporated into both a sign spotting model and an evaluation method. Below, we discuss our proposed methodology and results in terms of their limitations and potential avenues for future work. The discussion is split up into sections based on the two research questions that we proposed in this work (see Section 1.1).

7.1 Approximative phonological features

The aim of the first research question, RQ1, was to develop approximative phonological features extracted from landmarks and analyze in which situations they deliver an improvement over using landmarks directly. Furthermore, we investigated whether a combination of phonological features and landmarks is beneficial. We found that the combination of the features results in a better overall performance, but that it is less capable of representing the phonological properties of signs than the landmark or phonological features by themselves.

Some limitations have to be noted regarding the approximative features introduced in this work. Firstly, the features that we implemented do not cover all components of the phonology of signs. In Section 6.2.1, we found that all feature sets that were used resulted in a large number of false positive and false negative evaluations. By developing features which better represent a variety of signs, the effectiveness of the features may be improved. For example, the inclusion of the non-manuals, such as mouthings and body posture, could be an interesting avenue for future research to investigate.

Secondly, the features that were developed in this thesis made use of 2D landmark coordinates. Current landmark detection tools, such as Mediapipe, do not give reliable estimates of the depth of the landmarks. If the z coordinate could be reliably estimated, we could adopt 3D representations of the hands and bodies of signers and extract linguistic features in a more robust way. For instance, the left and right image in Figure 7.1 depict the same handshape as it appears from two different angles. It would be possible to derive the curvature of the index finger from a side view (left image) based on 2D landmark coordinates, but not from a front-facing view (right image). From 3D landmark coordinates, the curvature could be derived precisely and reliably. In future work, the implementation of 3D feature representations could be compared to the usage of 2D features. Alternatively, a system that is capable of inferring the 3Dh configuration from the hand based on a 2D image, such as the model proposed in [51], could be adopted to mimic the ability of humans to infer pose from a singular perspective.

Another limitation of current landmark detection tools is that they are not explicitly trained on sign language. Particular handshapes that are common in a given sign language may be underrepresented in the data that tools such as Mediapipe are trained on. It appears promising to fine-tune existing detection tools on sign language data to prepare them for the specific task at hand.

While not the focus of this work, we note that the model that was used in this thesis

may not be optimal for the application of sign spotting. Since the apparent effectiveness of our features is in part influenced by the ability of our model to utilize them, it appears promising for future work to make use of more sophisticated model architectures. For example, a model that is capable of handling variable input lengths could allow us to more precisely spot signs in videos if they are of a length that deviates from the average annotation length. A pre-trained model architecture, such as a transformer, could be fine-tuned to allow us to benefit from training on similar datasets outside of sign language corpora. In addition to this, a more explainable model architecture could be chosen to further the explainability of sign spotting systems.

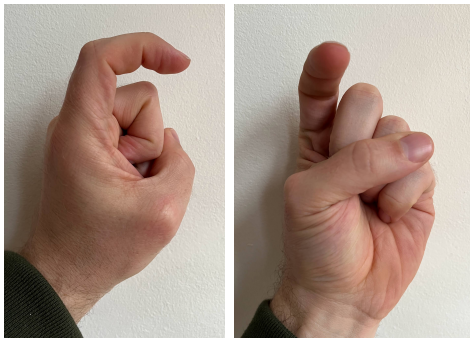

Figure 7.1: Hooked finger from two viewpoints

7.2 Evaluation method

For RQ2, the second research question, our aim was to develop an evaluation method for sign spotting that is reflective of users and that is capable of handling an incomplete ground truth. To achieve the first aim, we based our evaluation on tolerance to irrelevance (TTI) as proposed by De Vries, Kazai, and Lalmas [9], which explicitly considers the effort it takes for users to use a spotting system. To achieve the second aim, our evaluation makes use of distractors, which are annotations of signs that are most similar to a given target sign based on a chosen distance metric. We proposed a novel linguistic distance measure to select these hardest, most similar cases. We demonstrated that our distractor-based evaluation provides a more conservative estimate of the model performance than a baseline evaluation that made use of randomly chosen annotations as distractors, and that it was comparably robust against updates to the annotations. This approach has several limitations, which we will cover below.

As previously discussed, our linguistic distance measure makes use of linguistic information from NGT Signbank. We computed the linguistic distance by comparing the phonological properties of signs, and adding to the distance if the phonological properties did not align. This approach to calculating the distance between signs has a number of limitations. First, the phonological properties that are annotated in NGT Signbank are incomplete. While we managed to work around this issue by ignoring any comparisons between unknown properties, this does not tackle the core issue of linguistic parameters not being available. Future work should ideally make use of complete linguistic information, which requires that the available phonological information is further annotated.

Second, the manual phonology of a sign is not the only factor that plays a role in the execution of a sign. Non-manuals, in particular the mouthings that accompany a sign, may be the only factor that distinguishes two signs. Because our distance measure only takes into account the manual components of signs, we would therefore conclude that the

signs are more similar to each other than they are in reality. Additionally, the realization of a sign is influenced by phonetics, which includes factors such as the emotional state of the signer. The incorporation of the phonetics of sign language could be an interesting research direction for future work.
Third, the linguistic distance introduced in this work used a binary weighting for the distance and did not take the nuanced difference between phonological properties into account. In other words, our calculation either assigned a distance of 0 if two phonological properties were the same, or a distance of 1 if they were different. In reality, certain properties may be inherently more similar than others. A two-handed asymmetrical sign, for instance, may be both phonologically and visually more similar to a two-handed symmetrical sign than a one-handed sign. In a similar manner, a handshape where only one finger is extended could be interpreted to be phonologically closer to a handshape where two fingers are extended, than to a handshape using five extended fingers. Moreover, it may be that certain phonological properties are more important in the distance calculation than others. Future work could investigate how to weight the distances to better capture the nuanced differences in phonology between signs.

7.3 Deaf inclusion

Lastly, we would like to stress the importance of including the Deaf community in sign language research. This thesis was developed in an all-hearing team, and this impacted the manner in which the topic of sign language was approached. Previous work has demonstrated the importance of including Deaf researchers to develop systems that are useful to Deaf users and to secure adoption of new technologies [5]. Additionally, Deaf people should be included in focus groups to determine which needs have to be addressed by SLP systems. We hope that future research will work towards more inclusion of the community.

Bibliography

[1] N. Burkart and M. F. Huber, "A survey on the explainability of supervised machine learning," *Journal of Artificial Intelligence Research*, vol. 70, pp. 245–317, 2021.

[2] V. Belle and I. Papantonis, "Principles and practice of explainable machine learning," *Frontiers in big Data*, p. 39, 2021.

[3] M. Gaur, K. Faldu, and A. Sheth, "Semantics of the black-box: Can knowledge graphs help make deep learning systems more interpretable and explainable?" *IEEE Internet Computing*, vol. 25, no. 1, pp. 51–59, 2021.

[4] A. Samih, A. Adadi, and M. Berrada, "Towards a knowledge based explainable recommender systems," in *Proceedings of the 4th International Conference on Big Data and Internet of Things*, 2019, pp. 1–5.

[5] D. Bragg, O. Koller, M. Bellard, *et al.*, "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*, 2019, pp. 16–31.

[6] L. He, E. Sanocki, A. Gupta, and J. Grudin, "Auto-summarization of audio-video presentations," in *Proceedings of the seventh ACM international conference on Multimedia (Part 1)*, 1999, pp. 489–498.

[7] W. Stokoe, "Sign language structure, an outline of the visual communications systems of american deaf," *Studies in Linguistics Occasional Paper*, vol. 8, 1960.

[8] R. Battison, *Lexical borrowing in American sign language*. Silver Spring, MD: Linstok Press, 1978.

[9] A. P. De Vries, G. Kazai, and M. Lalmas, "Tolerance to irrelevance: A user-effort oriented evaluation of retrieval systems without predefined retrieval unit," in *RIAO Conference Proceedings*, 2004, pp. 463–473.

[10] A. Moryossef and Y. Goldberg, *Sign Language Processing*, https://sign-language-processing.github.io/, Accessed: Jan. 27, 2023, 2021.

[11] M. Vázquez Enríquez, J. L. A. Castro, L. D. Fernandez, J. C. Jacques Junior, and S. Escalera, "Eccv 2022 sign spotting challenge: Dataset, design and results," in *European Conference on Computer Vision*, Springer, 2022, pp. 225–242.

[12] P. Paudyal, J. Lee, A. Kamzin, M. Soudki, A. Banerjee, and S. K. Gupta, "Learn2sign: Explainable ai for sign language learning.," in *IUI Workshops*, 2019.

[13] J. McCleary, L. P. García, C. Ilioudis, and C. Clemente, "Sign language recognition using micro-doppler and explainable deep learning," in *IEEE Radar Conference*, IEEE, 2021, pp. 1–6.

[14] D. R. Kothadiya, C. M. Bhatt, A. Rehman, F. S. Alamri, and T. Saba, "Signexplainer: An explainable ai-enabled framework for sign language recognition with ensemble learning," *IEEE Access*, 2023.

[15] F. Zhang, V. Bazarevsky, A. Vakunov, *et al.*, "Mediapipe hands: On-device real-time hand tracking," *arXiv preprint arXiv:2006.10214*, 2020.

[16] S. Albanie, G. Varol, L. Momeni, *et al.*, "Bsl-1k: Scaling up co-articulated sign language recognition using mouthing cues," in *European Conference on Computer Vision*, Springer, 2020, pp. 35–53.

[17] N.-K. Pendzich, *Lexical nonmanuals in German Sign Language: Empirical studies and theoretical implications*. De Gruyter, 2020.

[18] E. Van der Kooij, *Phonological categories in Sign Language of the Netherlands: The role of phonetic implementation and iconicity*. Netherlands Graduate School of Linguistics, 2002.

[19] U. Klomp, *A descriptive grammar of Sign Language of the Netherlands*. LOT, 2021.

[20] O. Crasborn, "Phonetics," in *Sign language: An international handbook*, De Gruyter, 2012.

[21] M. Tyrone, "Phonetics of sign language," in *Oxford Research Encyclopedia of Linguistics*, Oxford University Press, 2020. DOI: 10.1093/acrefore/9780199384655.013.744.

[22] L. Momeni, G. Varol, S. Albanie, T. Afouras, and A. Zisserman, "Watch, read and lookup: Learning to spot signs from multiple supervisors," in *Asian conference on computer vision*, 2020.

[23] T. Jiang, N. C. Camgöz, and R. Bowden, "Looking for the signs: Identifying isolated sign instances in continuous video footage," in *16th IEEE International Conference on Automatic Face and Gesture Recognition*, IEEE, 2021, pp. 1–8.

[24] R. Bowden, D. Windridge, T. Kadir, A. Zisserman, and M. Brady, "A linguistic feature vector for the visual interpretation of sign language," in *8th European Conference on Computer Vision*, Springer, 2004, pp. 390–401.

[25] U. Von Agris, J. Zieren, U. Canzler, B. Bauer, and K.-F. Kraiss, "Recent developments in visual sign language recognition," *Universal Access in the Information Society*, vol. 6, pp. 323–362, 2008.

[26] J. Han, G. Awad, and A. Sutherland, "Modelling and segmenting subunits for sign language recognition based on hand motion analysis," *Pattern Recognition Letters*, vol. 30, no. 6, pp. 623–633, 2009.

[27] M. M. Zaki and S. I. Shaheen, "Sign language recognition using a combination of new vision based features," *Pattern Recognition Letters*, vol. 32, no. 4, pp. 572–577, 2011.

[28] S.-K. Ko, J. G. Son, and H. Jung, "Sign language recognition with recurrent neural network using human keypoint detection," in *Proceedings of the 2018 Conference on Research in Adaptive and Convergent Systems*, 2018, pp. 326–328.

[29] S.-K. Ko, C. J. Kim, H. Jung, and C. Cho, "Neural sign language translation based on human keypoint estimation," *Applied sciences*, vol. 9, no. 13, p. 2683, 2019.

[30] J. Shin, A. Matsuoka, M. A. M. Hasan, and A. Y. Srizon, "American sign language alphabet recognition by extracting feature from hand pose estimation," *Sensors*, vol. 21, no. 17, p. 5856, 2021.

[31] M. J. Hussain *et al.*, "Intelligent sign language recognition system for e-learning context," *Computers, Materials & Continua*, vol. 72, no. 3, pp. 5327–5343, 2022.

[32] Y. Farhan and A. A. Madi, "Real-time dynamic sign recognition using mediapipe," in *IEEE 3rd International Conference on Electronics, Control, Optimization and Computer Science*, IEEE, 2022, pp. 1–7.

[33] V. Viitaniemi, T. Jantunen, L. Savolainen, M. Karppa, and J. Laaksonen, "S-pot–a benchmark in spotting signs within continuous signing," in *Proceedings of the 9th International Conference on Language Resources and Evaluation*, European Language Resources Association, 2014, pp. 1892–1897.

[34] S.-S. Cho, H.-D. Yang, and S.-W. Lee, "Sign language spotting based on semi-markov conditional random field," in *2009 Workshop on Applications of Computer Vision*, IEEE, 2009, pp. 1–6.

[35] H.-D. Yang and S.-W. Lee, "Simultaneous spotting of signs and fingerspellings based on hierarchical conditional random fields and boostmap embeddings," *Pattern Recognition*, vol. 43, no. 8, pp. 2858–2870, 2010.

[36] A. Aljanaki, F. Wiering, and R. C. Veltkamp, "Emotion based segmentation of musical audio," in *Proceedings of the 16th Conference of the International Society for Music Information Retrieval*, 2015, pp. 770–776.

[37] J. B. Smith and E. Chew, "A meta-analysis of the mirex structure segmentation task," in *Proceedings of the 14th International Society for Music Information Retrieval Conference*, vol. 16, 2013, pp. 45–47.

[38] J. B. L. Smith, J. A. Burgoyne, I. Fujinaga, D. De Roure, and J. S. Downie, "Design and creation of a large-scale database of structural annotations.," in *Proceedings of the 12th International Society for Music Information Retrieval Conference*, 2011, pp. 555–560.

[39] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks for action segmentation and detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 156–165.

[40] J. Hong, H. Zhang, M. Gharbi, M. Fisher, and K. Fatahalian, "Spotting temporally precise, fine-grained events in video," in *Proceedings of the European Conference on Computer Vision 2022*, Springer, 2022, pp. 33–51.

[41] M. Eskevich, W. Magdy, and G. J. Jones, "New metrics for meaningful evaluation of informally structured speech retrieval," in *European Conference on Information Retrieval*, Springer, 2012, pp. 170–181.

[42] O. Crasborn and I. Zwitserlood, "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*, pp. 44–49, 2008.

[43] Y. Lin, X. Jiao, and L. Zhao, "Detection of 3d human posture based on improved mediapipe," *Journal of Computer and Communications*, vol. 11, no. 2, pp. 102–121, 2023.

[44] S. Celebi, A. S. Aydin, T. T. Temiz, and T. Arici, "Gesture recognition using skeleton data with weighted dynamic time warping.," in *Proceedings of the International Conference on Computer Vision Theory and Applications*, 2013, pp. 620–625.

[45] O. Crasborn, I. Zwitserlood, and J. Ros, "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*, 2008.

[46] O. Crasborn, R. Bank, I. Zwitserlood, *et al.*, "Ngt signbank," *Nijmegen: Radboud University, Centre for Language Studies*, 2016.

[47] J. Chai and A. Li, "Deep learning in natural language processing: A state-of-the-art survey," in *International Conference on Machine Learning and Cybernetics*, IEEE, 2019, pp. 1–6.

[48] P. Khosla, P. Teterwak, C. Wang, *et al.*, "Supervised contrastive learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 18 661–18 673, 2020.

[49] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9729–9738.

[50] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance discrimination," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3733–3742.

[51] M. Ivashechkin, O. Mendez, and R. Bowden, "Improving 3d pose estimation for sign language," in *IEEE International Conference on Acoustics, Speech, and Signal Processing Workshops*, IEEE, 2023, pp. 1–5.

Appendix A

Appendix

A.1 Pseudocode linguistic distance computation

Algorithm 1 Linguistic distance computation

```
1: procedure LINGUISTIC_DISTANCE(L_A,L_B)                    ▷
   L_A : List of phonological properties for sign A          ▷
   L_B : List of phonological properties for sign B
2:     dist ← 0
3:     for m in range(len(L_A)) do
4:         a_m ← L_A[m]
5:         b_m ← L_B[m]
6:         if a_m == b_m then
7:             dist ← dist
8:         end if
9:         if a_m ≠ b_m then
10:            dist ← dist + 1
11:        end if
12:    end for
13:    return dist
14: end procedure
```

A.2 Proof: mirroring after normalisation

We first show that the x coordinate of each landmark that is used to normalise the x coordinate of a landmark ℓ, has to be mirrored:

$$\ell_{centered,x} = \ell_{scaled,x} - sh_{M,scaled,x}$$

$$\ell_{centered,x} = \frac{\ell_x}{|sh_{L,x} - sh_{R,x}|} - \frac{sh_{M,x}}{|sh_{L,x} - sh_{R,x}|} = \frac{\ell_x - sh_{M,x}}{|sh_{L,x} - sh_{R,x}|}$$

$$mirror(\ell_{centered,x}) = \frac{mirror(\ell_x) - mirror(sh_{M,x})}{|mirror(sh_{L,x}) - mirror(sh_{R,x})|}$$

The normalisation is done as in Section 4.2.1. Given that a landmark ℓ in a video of width w is mirrored as $mirror(\ell) = w - \ell - 1$, we can mirror each landmark individually:

$$mirror(\ell_x) = w - \ell_x - 1$$
$$mirror(sh_{L,x}) = w - sh_{L,x} - 1$$
$$mirror(sh_{R,x}) = w - sh_{R,x} - 1$$
$$mirror(sh_{M,x}) = w - sh_{M,x} - 1$$

If we use these equations to compute $mirror(\ell_{centered,x})$, we obtain the following:

$$\begin{aligned} mirror(\ell_{centered,x}) &= \frac{(w - \ell_x - 1) - (w - sh_{M,x} - 1)}{|(w - sh_{L,x} - 1) - (w - sh_{R,x} - 1)|} \\ &= \frac{w - \ell_x - 1 - w + sh_{M,x} + 1}{|w - sh_{L,x} - 1 - w + sh_{R,x} + 1|} \\ &= \frac{-\ell_x + sh_{M,x}}{|sh_{R,x} - sh_{L,x}|} \end{aligned}$$

Since $|a - b| = \sqrt{(a-b)^2} = \sqrt{(b-a)^2} = |b - a|$, we thus find that:

$$mirror(\ell_{centered,x}) = \frac{-\ell_x + sh_{M,x}}{|sh_{R,x} - sh_{L,x}|} = \frac{-\ell_x + sh_{M,x}}{|sh_{L,x} - sh_{R,x}|} = -1 \cdot \ell_{centered,x}$$

A.3 Spotting threshold tests

To determine which spotting threshold t is suitable for our experiments, we trained our model using each set of features and then used our distractor-based evaluation using a selection of spotting thresholds, namely $0.15, 0.2, 0.25$ and 0.3. This set was selected based on some initial runs, which revealed that the optimal spotting threshold was always somewhere between 0.15 and 0.3. Values outside of this range never resulted in a higher model performance. We then determined that the threshold for which we find the highest accuracy, is the most suitable spotting threshold.
In the Tables below, we have plotted the performance in terms of accuracy, precision and recall for each set of features. $t = 0.20$ consistently results in the highest accuracy for each of the feature sets. Therefore, we selected this threshold for all of our models.

	$t = 0.15$	$t = 0.20$	$t = 0.25$	$t = 0.30$
Accuracy	0.623	**0.654**	0.645	0.611
Precision	**0.841**	0.792	0.744	0.689
Recall	0.354	0.468	**0.501**	0.484

Table A.1: Threshold tests for linguistic features

	$t = 0.15$	$t = 0.20$	$t = 0.25$	$t = 0.30$
Accuracy	0.624	**0.658**	0.643	0.605
Precision	**0.844**	0.798	0.740	0.684
Recall	0.356	0.473	**0.502**	0.469

Table A.2: Threshold tests for landmark features

	$t = 0.15$	$t = 0.20$	$t = 0.25$	$t = 0.30$
Accuracy	0.638	**0.664**	0.652	0.616
Precision	**0.850**	0.802	0.748	0.694
Recall	0.384	0.485	**0.517**	0.492

Table A.3: Threshold tests for combined (landmark + linguistic) features

A.4 Masking experiments

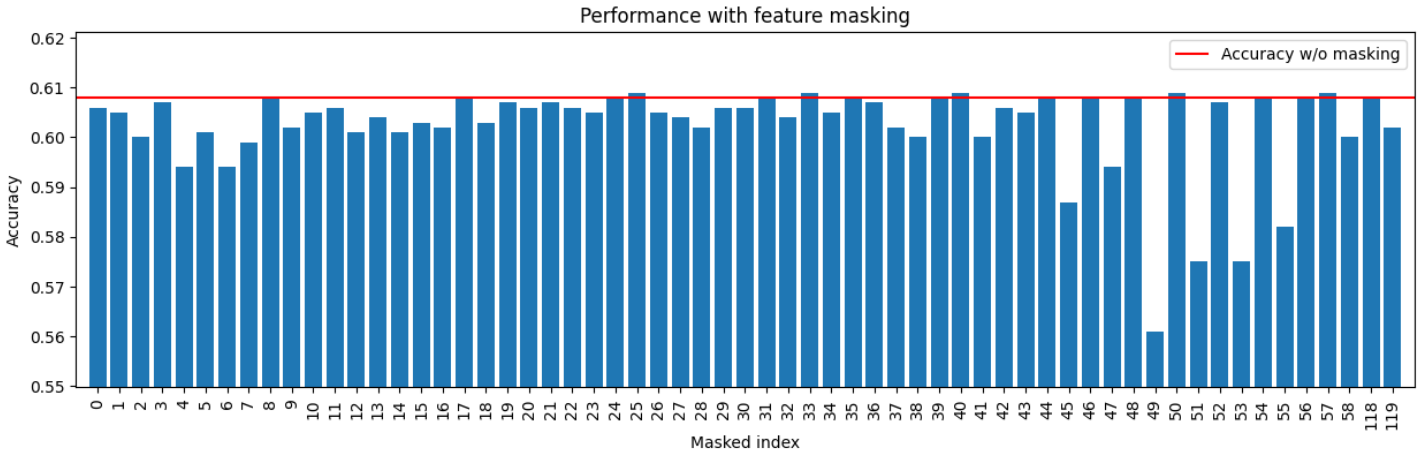

Figure A.1: Run 1 (used as example in Figure 4.3)

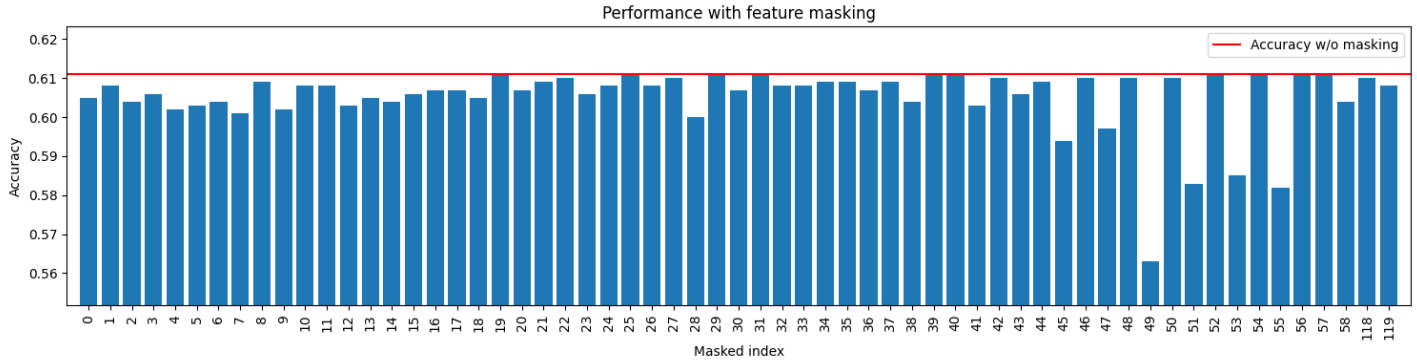

Figure A.2: Run 2

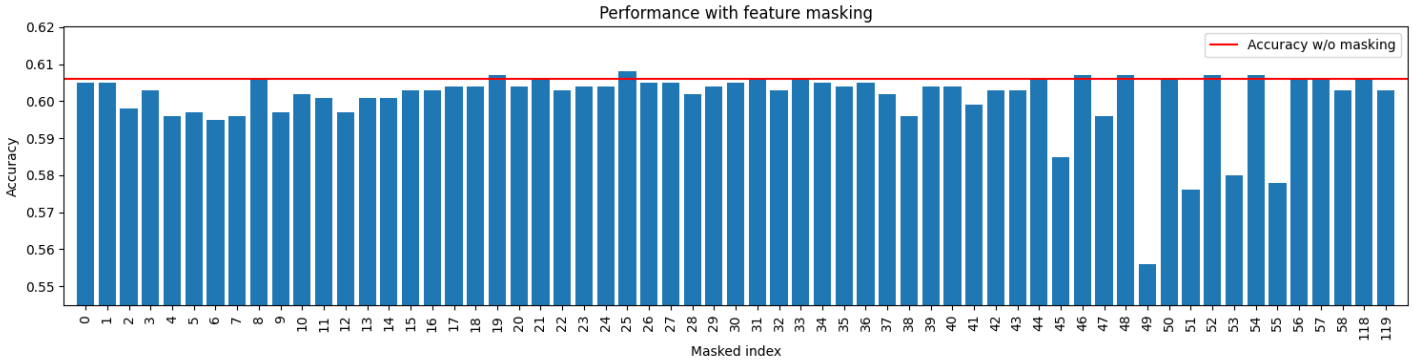

Figure A.3: Run 3

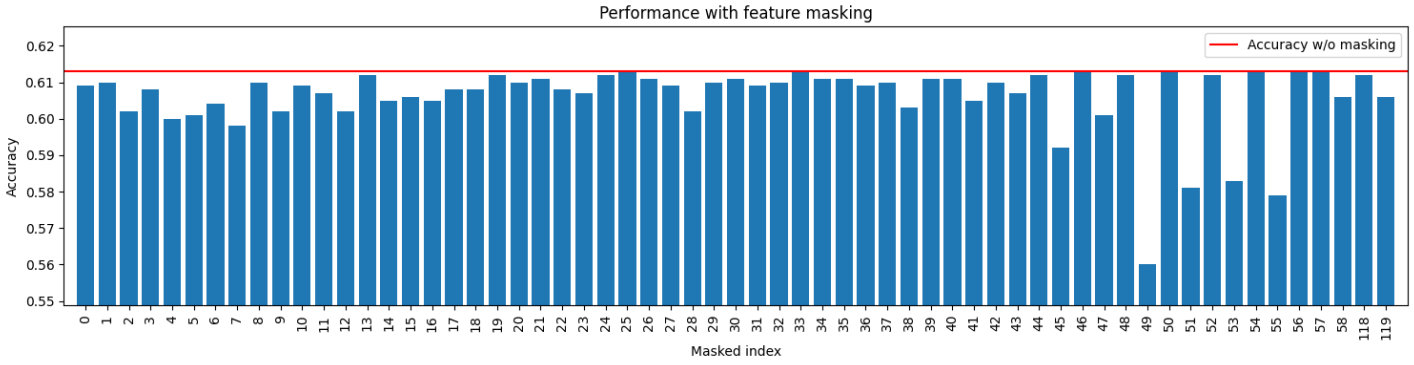

Figure A.4: Run 4

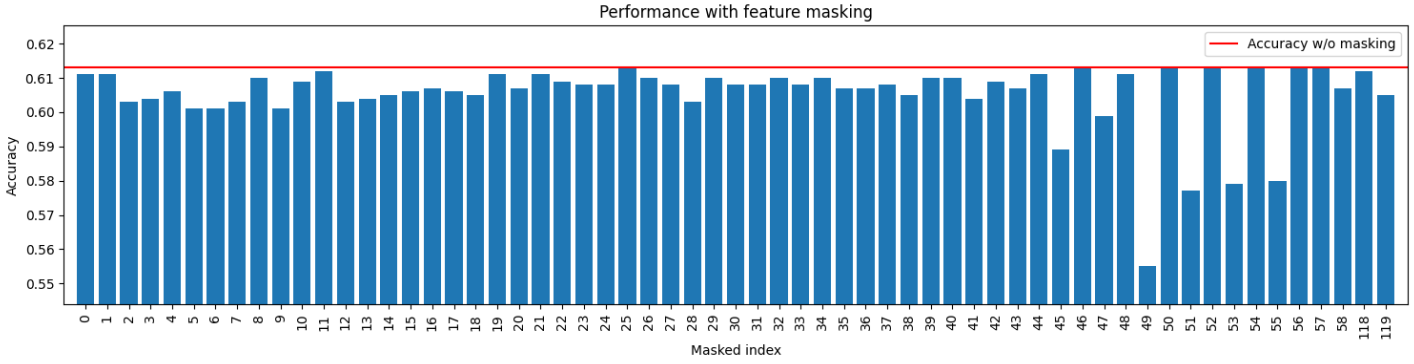

Figure A.5: Run 5

A.5 Full results of the validation of the distractor-based evaluation

A.5.1 Dropout ratio = 0.1

	TP	FN	FP	TN	Acc.	Prec.	Rec.	F1	Mean ling. dist.
No dropout	5369	4339	1777	7910	0.685	0.751	0.553	0.637	3.007
Dropout 1	5369	4339	1770	7903	0.685	0.752	0.553	0.637	3.066
Dropout 2	5369	4339	1748	7932	0.686	0.754	0.553	0.638	3.064
Dropout 3	5369	4339	1753	7923	0.686	0.754	0.553	0.638	3.064
Dropout 4	5367	4339	1737	7935	0.686	0.755	0.553	0.638	3.068
Dropout 5	5369	4339	1761	7922	0.685	0.753	0.553	0.638	3.071
Dropout 6	5369	4339	1725	7953	0.687	0.757	0.553	0.639	3.062
Dropout 7	5369	4339	1735	7940	0.687	0.756	0.553	0.639	3.069
Dropout 8	5369	4339	1734	7938	0.687	0.756	0.553	0.639	3.06
Dropout 9	5369	4339	1739	7941	0.687	0.755	0.553	0.638	3.069
Dropout 10	5367	4339	1756	7918	0.686	0.753	0.553	0.638	3.066

Table A.4: Standard distractors with dropout = 0.1

	TP	FN	FP	TN	Acc.	Prec.	Rec.	F1	Mean ling. dist.
No dropout	5369	4339	962	8728	0.727	0.848	0.553	0.669	5.807
Dropout 1	5369	4339	966	8707	0.726	0.848	0.553	0.669	5.792
Dropout 2	5368	4339	965	8712	0.726	0.848	0.553	0.669	5.794
Dropout 3	5368	4339	964	8717	0.726	0.848	0.553	0.669	5.807
Dropout 4	5369	4339	976	8702	0.726	0.846	0.553	0.669	5.81
Dropout 5	5368	4339	944	8726	0.727	0.85	0.553	0.67	5.803
Dropout 6	5369	4339	948	8727	0.727	0.85	0.553	0.67	5.809
Dropout 7	5369	4339	975	8703	0.726	0.846	0.553	0.669	5.799
Dropout 8	5369	4339	992	8680	0.725	0.844	0.553	0.668	5.803
Dropout 9	5369	4339	977	8705	0.726	0.846	0.553	0.669	5.821
Dropout 10	5369	4339	943	8734	0.728	0.851	0.553	0.67	5.802

Table A.5: Random distractors with dropout = 0.1

A.5.2 Dropout ratio = 0.25

	TP	FN	FP	TN	Acc.	Prec.	Rec.	F1	Mean ling. dist.
No dropout	5369	4339	1777	7910	0.685	0.751	0.553	0.637	3.007
Dropout 1	5369	4339	1702	7958	0.688	0.759	0.553	0.64	3.162
Dropout 2	5369	4339	1646	8009	0.691	0.765	0.553	0.642	3.161
Dropout 3	5369	4339	1660	7993	0.69	0.764	0.553	0.642	3.161
Dropout 4	5367	4337	1682	7976	0.689	0.761	0.553	0.641	3.157
Dropout 5	5368	4339	1642	8005	0.691	0.766	0.553	0.642	3.151
Dropout 6	5369	4339	1672	7978	0.689	0.763	0.553	0.641	3.158
Dropout 7	5368	4339	1704	7953	0.688	0.759	0.553	0.64	3.162
Dropout 8	5368	4339	1688	7959	0.689	0.761	0.553	0.641	3.162
Dropout 9	5367	4339	1667	7990	0.69	0.763	0.553	0.641	3.162
Dropout 10	5367	4339	1679	7977	0.689	0.762	0.553	0.641	3.159

Table A.6: Standard distractors with dropout = 0.25

	TP	FN	FP	TN	Acc.	Prec.	Rec.	F1	Mean ling. dist.
No dropout	5369	4339	962	8728	0.727	0.848	0.553	0.669	5.807
Dropout 1	5369	4339	953	8695	0.727	0.849	0.553	0.67	5.791
Dropout 2	5369	4339	980	8679	0.725	0.846	0.553	0.669	5.787
Dropout 3	5368	4339	946	8705	0.727	0.85	0.553	0.67	5.807
Dropout 4	5368	4339	953	8700	0.727	0.849	0.553	0.67	5.793
Dropout 5	5367	4337	957	8685	0.726	0.849	0.553	0.67	5.793
Dropout 6	5369	4339	946	8704	0.727	0.85	0.553	0.67	5.796
Dropout 7	5369	4339	946	8709	0.727	0.85	0.553	0.67	5.788
Dropout 8	5369	4339	940	8714	0.727	0.851	0.553	0.67	5.793
Dropout 9	5368	4339	950	8706	0.727	0.85	0.553	0.67	5.792
Dropout 10	5368	4339	997	8667	0.725	0.843	0.553	0.668	5.792

Table A.7: Random distractors with dropout = 0.25

A.5.3 Dropout ratio = 0.5

	TP	FN	FP	TN	Acc.	Prec.	Rec.	F1	Mean ling. dist.
No dropout	5369	4339	1777	7910	0.685	0.751	0.553	0.637	3.007
Dropout 1	5367	4339	1609	8012	0.692	0.769	0.553	0.643	3.322
Dropout 2	5367	4337	1581	8025	0.694	0.772	0.553	0.644	3.323
Dropout 3	5367	4339	1593	8022	0.693	0.771	0.553	0.644	3.337
Dropout 4	5369	4339	1599	8021	0.693	0.771	0.553	0.644	3.334
Dropout 5	5368	4339	1606	8011	0.692	0.77	0.553	0.644	3.329
Dropout 6	5368	4339	1600	8018	0.693	0.77	0.553	0.644	3.333
Dropout 7	5367	4339	1557	8054	0.695	0.775	0.553	0.645	3.324
Dropout 8	5367	4339	1594	8020	0.693	0.771	0.553	0.644	3.332
Dropout 9	5368	4339	1574	8035	0.694	0.773	0.553	0.645	3.332
Dropout 10	5366	4339	1596	8025	0.693	0.771	0.553	0.644	3.334

Table A.8: Standard distractors with dropout = 0.5

	TP	FN	FP	TN	Acc.	Prec.	Rec.	F1	Mean ling. dist.
No dropout	5369	4339	962	8728	0.727	0.848	0.553	0.669	5.807
Dropout 1	5367	4339	975	8640	0.725	0.846	0.553	0.669	5.752
Dropout 2	5367	4339	921	8691	0.728	0.854	0.553	0.671	5.768
Dropout 3	5368	4339	958	8659	0.726	0.849	0.553	0.67	5.764
Dropout 4	5366	4337	930	8672	0.727	0.852	0.553	0.671	5.777
Dropout 5	5369	4339	947	8672	0.726	0.85	0.553	0.67	5.774
Dropout 6	5367	4339	938	8681	0.727	0.851	0.553	0.67	5.796
Dropout 7	5368	4339	960	8659	0.726	0.848	0.553	0.669	5.783
Dropout 8	5367	4337	932	8689	0.727	0.852	0.553	0.671	5.784
Dropout 9	5366	4339	919	8689	0.728	0.854	0.553	0.671	5.777
Dropout 10	5366	4339	937	8689	0.727	0.851	0.553	0.67	5.776

Table A.9: Random distractors with dropout = 0.5

A.5.4 Dropout ratio = 0.75

	TP	FN	FP	TN	Acc.	Prec.	Rec.	F1	Mean ling. dist.
No dropout	5369	4339	1777	7910	0.685	0.751	0.553	0.637	3.007
Dropout 1	5367	4337	1484	8096	0.698	0.783	0.553	0.648	3.522
Dropout 2	5368	4339	1474	8100	0.699	0.785	0.553	0.649	3.52
Dropout 3	5366	4339	1500	8064	0.697	0.782	0.553	0.648	3.52
Dropout 4	5356	4335	1447	8115	0.7	0.787	0.553	0.65	3.517
Dropout 5	5366	4339	1465	8105	0.699	0.786	0.553	0.649	3.513
Dropout 6	5365	4337	1469	8114	0.699	0.785	0.553	0.649	3.522
Dropout 7	5357	4337	1473	8097	0.698	0.784	0.553	0.649	3.517
Dropout 8	5366	4339	1486	8072	0.698	0.783	0.553	0.648	3.519
Dropout 9	5365	4337	1473	8110	0.699	0.785	0.553	0.649	3.527
Dropout 10	5366	4337	1486	8095	0.698	0.783	0.553	0.648	3.523

Table A.10: Standard distractors with dropout = 0.75

	TP	FN	FP	TN	Acc.	Prec.	Rec.	F1	Mean ling. dist.
No dropout	5369	4339	962	8728	0.727	0.848	0.553	0.669	5.807
Dropout 1	5366	4337	908	8662	0.728	0.855	0.553	0.672	5.748
Dropout 2	5366	4339	915	8651	0.727	0.854	0.553	0.671	5.769
Dropout 3	5367	4339	878	8699	0.729	0.859	0.553	0.673	5.78
Dropout 4	5356	4335	916	8650	0.727	0.854	0.553	0.671	5.774
Dropout 5	5366	4339	907	8671	0.728	0.855	0.553	0.672	5.758
Dropout 6	5365	4337	926	8631	0.727	0.853	0.553	0.671	5.744
Dropout 7	5365	4337	891	8677	0.729	0.858	0.553	0.673	5.766
Dropout 8	5364	4334	944	8626	0.726	0.85	0.553	0.67	5.773
Dropout 9	5365	4337	941	8635	0.726	0.851	0.553	0.67	5.748
Dropout 10	5367	4339	914	8663	0.728	0.854	0.553	0.671	5.772

Table A.11: Random distractors with dropout = 0.75