Technische Universität Berlin
Electrical Engineering and Computer Science
Institute of Software Engineering and Theoretical Computer Science
Algorithmics and Computational Complexity (AKT)

Using Temporal Graph Comparison for Sign Language Recognition

Paula Natascha Ruß

Thesis submitted in fulfillment of the requirements for the degree
"Bachelor of Science" (B. Sc.) in the field of Computer Science

November 2021

Supervisor and first reviewer: Prof. Dr. Rolf Niedermeier
Second reviewer: Prof. Dr. Markus Brill
Co-Supervisors: Dr. Vincent Froese and Malte Renken

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und eigenhändig sowie ohne unerlaubte fremde Hilfe und ausschließlich unter Verwendung der aufgeführten Quellen und Hilfsmittel angefertigt habe.

Berlin, den _____ _____

Datum Unterschrift

Zusammenfassung

Temporale Graphen sind nützlich, um Daten aus der realen Welt zu modellieren, da sie Änderungen über die Zeit nachbilden können. In dieser Arbeit nutzen wir temporale Graphen und Methoden zur Bestimmung der Ähnlichkeit zwischen temporalen Graphen für automatisierte Gebärdenspracherkennung auf Wortebene. Das Ziel der automatisierten Interpretation von Gebärdensprache ist die Kommunikation mit hörenden Menschen zu vereinfachen und die Erkennung von Gebärden auf Wortebene ist dabei ein wichtiger Schritt. Wir stellen ein temporales Graphmodell für Gebärden der amerikanischen Gebärdensprache (ASL) vor, bei dem wir Körperbezugspunkte als Knoten verwenden und die Kanten zwischen den Knoten die räumliche Nähe der Bezugspunkte anzeigen. Die x- und y-Koordinaten der Bezugspunkte wurden aus bestehenden Videoaufzeichnungen einer Gebärde extrahiert. Wir vergleichen dann die resultierenden temporalen Graphen mit zwei verschiedenen Methoden, mit der *dynamic temporal graph warping distance* [1] und mit einem *temporalen Random Walk Kernel* [2]. Basierend auf den Ähnlichkeitswerten verwenden wir den K-Nearest Neighbor Algorithmus und eine Support-Vector Maschine für die Klassifizierung von Gebärden. Mit unserem Graphmodell und mit dem temporalen 2-Schritte Random Walk Kernel können wir bei 100 verschiedenen Gebärden bis zu 62 % der Gebärden des Testsatzes korrekt übersetzen und bis zu 47 % bei 300 verschiedenen Gebärden. Damit sind unsere Ergebnisse vergleichbar mit denen eines 2D-Convolutional Networks und mit weiteren Verbesserungen könnte es ein alternativer Ansatz für die Gebärdenspracherkennung werden.

Abstract

Temporal graphs are useful to model real-world data since they can represent temporal changes. In this thesis, we use temporal graphs and temporal graph proximity measure methods for automated word-level sign language recognition. The goal of automated sign language interpretation is to simplify the communication between hearing and hearing-impaired people and word-level sign language recognition is one important step. We propose a temporal graph model for American Sign Language (ASL) signs, where we use body keypoints as vertices and the edges indicate the proximity of the keypoints. The x- and y-coordinates of the keypoints are extracted from existing video recordings of a sign. We measure the similarity of the resulting temporal graphs with two different methods, with the *dynamic temporal graph warping distance* [1] and with a *temporal random walk kernel* [2]. Based on the similarity values, we use k-nearest neighbor or support-vector machines for the classification of signs. With our graph model and the temporal 2-step random walk kernel, we can predict up to 62 % correctly the meaning of the test signs on a dataset of 100 different signs and up to 47 % on 300 different signs. Our results are comparable to those of a 2D-convolutional network and with further improvements, it could be an alternative approach for sign language recognition.

Contents

Chapter 1

Introduction

Temporal graphs can be used to model real-world data that change over time, such as communication or transportation networks. The advantage compared to static graphs is that time information can be covered, which also makes analyzing temporal graphs challenging. In the last few years, the interest in temporal graphs increased and they have become a major subject in algorithmic graph theory (e.g., [3, 4, 1, 2]).

A temporal graph (also called *temporal network* [5, 6], *time-varying graph* [7], *link stream* [8, 9]) consists of a constant set of vertices, but the edges between the vertices vary over time. Intuitively, a temporal graph is a sequence of static graphs with the same vertices but with different edge-sets. Temporal graphs can also be represented by labeling the edges with the layers in which they are present. An example of both representations are given in Figure 1.1.

A sign usually consists of a movement, which changes the positions of the body keypoints, especially the hands. Since the movement follows a certain pattern, the positions are time-dependent and therefore we can convert signs to temporal graphs. In this thesis, we use temporal graphs to model signs of the American Sign Language, where the vertices represent the keypoints on the human body and the temporal edges indicate whenever two points get close to each other.

Sign language is one of the most important communication tools for hearing impaired people. However, communicating with listeners is often difficult and needs to be done in written form. Therefore, research is being done on automated sign language interpretation to simplify communication. There are two main tasks, word-level, and sentence-level sign language recognition, in this thesis we address automated word-level recognition. Sign language recognition is challenging because a sign is expressed by body and hand motions, facial expressions, and sometimes also the mouth movement [10]. All sign languages have their characteristics and like in spoken languages, there are dialects in sign languages. We use the American Sign Language (*ASL*), which is mostly used by people in the United States of America and Anglophone Canada [11]. With the recent developments in image recognition, movements can be automatically extracted from existing video recordings and larger datasets were presented [12, 13].

The latest approaches in sign language recognition are from the field of machine learning, such as convolutional networks [14, 15, 13, 12]. This thesis, however, presents a different, more simple approach, where we represent signs as temporal graphs, and

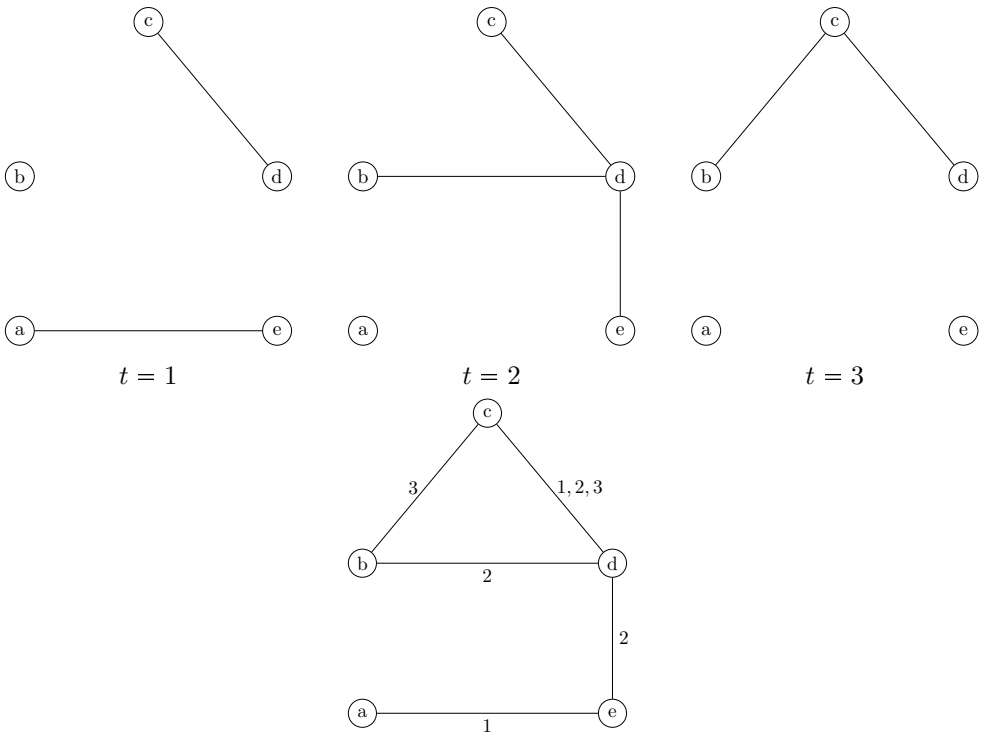


Figure 1.1: Example of two ways to represent a temporal graph with a lifetime of 3. The top row shows the different layers of the temporal graph while in the bottom row the edges are labeled with the layers in which they exist.

we use methods to measure the similarity of temporal graphs, the *dtgw-distance* [1] and a *temporal random walk kernel* [2]. For classification, we use algorithms like k-nearest neighbor [16] and support-vector machines [17].

Related work. There are two main approaches to recognize sign language, sensor-based and vision-based.

For the sensor-based approaches, a signer needs special hardware to capture signs, such as special gloves/markes for the hands and fingers [18, 19, 20, 21, 22, 23, 24] or other sensors, like leap motion controllers or accelerometers [25, 26, 27, 28]. Further, multiple cameras [29] or depth cameras like the Kinect camera [30, 31] have been used to get 3D information. Using such sensors leads to a better tracking of movements, however, special sensors pose an obstacle as not every signer has access to these.

Vision-based approaches try to track the movements relying only on 2D information received from one RGB camera. This simplifies the acquisition of larger datasets, as videos from the web can be collected. To extract features from videos, general approaches for feature detection in images [32, 33] are used, or approaches, which are specifically designed to recognize relevant features for a sign language [34, 35, 36]. Cao *et al.* [37] use so-called part affinity fields to extract body, hand [38], and face keypoints and called the method *OpenPose*. Two of the largest datasets, ML-ASL [13] with 1,000 signs and WLASL [12] with over 3,000 signs, have been created using OpenPose [37, 38].

To model temporal dependencies, dynamic time warping (DTW) [39, 40, 41, 42], Hidden Markov Models (HMM) [39, 40] or extended HMM [43, 44, 45, 46] are used. Recurrent neural networks model the temporal dependencies [12, 13] for 2D-convolutional networks. 3D convolutional networks are used to capture spatio-temporal features simultaneously [14, 15, 13, 12]. Elakkiya [47] gives a detailed overview of the approaches.

To the best of our knowledge, we are the first to study sign language recognition with temporal graph comparison methods. An overview of temporal graphs can be found in [48, 49, 4]. There are two approaches for the proximity measure on temporal graphs, the dynamic temporal graph warping (dtgw) distance [1] and temporal graph kernels [2]. The dtgw-distance [1] combines dynamic time warping and vertex mapping, based on assigned vertex signatures, to measure the similarity between two temporal graphs. The temporal graph kernels [2] use static kernels like the random walk kernel [50, 51, 52] or the Weisfeiler-Lehman graph kernel [53] on temporal graphs by mapping them to static graphs which encode the temporal dependencies.

Our contributions. The goal of this thesis is to analyze how ASL signs can be modelled as temporal graphs, and how well the similarity measure methods dtgw-distance [1] and a temporal graph kernel [2] perform regarding classification of signs. We derive the temporal graphs from the extracted body and hand keypoints from video recordings of the signs, provided by the WLASL dataset [12]. The dataset is one of the largest datasets of ASL signs on a word-level. We use the similarity measures between two temporal graphs, to predict the English counterpart of a sign on the word-level.

We show that our temporal graph model with a temporal 2-step random walk kernel achieves an accuracy of 62.79% when classifying between 100 different signs and 47.01% on 300 different signs, which is an improvement compared to the results of the pose based convolution network [12]. The dtgw-distance performs not as good as the temporal graph kernel did, reaching only 26.74% on 100 different signs.

The results show that our approach achieves comparable results on the dataset and with further improvements, it may be an alternative to convolutional network approaches for sign language recognition.

Chapter 2

Preliminaries

In this chapter we will give basic notation and concepts used in this work For $T \in \mathbb{N}$, we write $[T]$ to refer to the set $\{1, 2, ..., T\}$.

Static Graphs. We use basic notation according to Diestel [54].

Let $G = (V, E)$ denotes a *static (undirected) graph*, V denotes the set of vertices and $E \subseteq \{\{v, w\} \mid v, w \in V, v \neq w, \}$ denotes a set of edges. A *vertex-labeled static graph* $G = (V, E, \ell)$ consists additionally of a vertex labeling function $\ell : V \to \Sigma$ that assigns a label to each vertex, where Σ denotes a finite alphabet. The number $|E(v)|$ of edges at a vertex v is called degree $d_G(v) = d(v)$ of v.

Undirected Temporal Graphs. We adapt the notation from Froese *et al.* [1]. An *undirected temporal graph* is a $(T + 1)$-tuple $\mathcal{G} = (\mathcal{V}, \mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3, ..., \mathcal{E}_T)$, where $\mathcal{V}$ denotes a non-empty set of vertices, $T \geqslant 1$ is called the lifetime of $\mathcal{G}$ and $\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3, ..., \mathcal{E}_T$ is a sequence of edge-sets $\mathcal{E}_i \subseteq \{\{v, w\} \mid v, w \in \mathcal{V}, v \neq w, \}$, $i \in [T]$. A *layer* of $\mathcal{G}$ is the static undirected graph $\mathcal{G}_i = (\mathcal{V}, \mathcal{E}_i)$ at timestep $i \in [T]$.

A *vertex-labeled undirected temporal graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E}_1, \mathcal{E}_2, ..., \mathcal{E}_T, \ell)$ consists additionally of a labeling function $\ell : \mathcal{V} \times T \to \Sigma$, which assigns a label to each vertex at each timestep $t \in [T]$.

Vertex Mapping. A vertex mapping between two vertex sets V and W assigns to a vertex in V a vertex in W and vice versa [1]. If the vertex sets do not have the same size, not all vertices of the larger vertex set are mapped. Formally, a vertex mapping $M \subseteq V \times W$ is a set of tuples in which each vertex $x \in \mathcal{V} \cup \mathcal{W}$ is in at most one tuple and M contains $\min(|\mathcal{V}|, |\mathcal{W}|)$ tuples. $\mathcal{M}(\mathcal{V}, \mathcal{W})$ indicates the set of all vertex mappings between $\mathcal{V}$ and $\mathcal{W}$. $V_M \subseteq V$ are the vertices that are mapped to vertices in W and $W_M \subseteq W$ are the vertices that are mapped to vertices in V.

Vertex Signature Graph Distance on Static Graphs. We define a graph distance based on vertex signatures according to Jouili and Tabbone [55] and Froese *et al.* [1].

Given a vertex mapping between the vertices of two graphs, the *vertex signature graph distance* is the sum of the distances between mapped vertices. The distance between

two mapped vertices is defined as the distance between their vertex signature, which is assigned to the vertices by a vertex signature function.

Let $G = (V, E)$ and $H = (W, F)$ be two static graphs. A vertex signature function $f_G : V \to \mathbb{Q}^k$ assigns to a vertex $v \in V$ a value, which encodes information about it, and similarly for the vertices in H with $f_H : W \to \mathbb{Q}^k$. Let $n : \mathbb{Q}^k \times \mathbb{Q}^k \to \mathbb{Q}$ be a metric. The cost of a vertex mapping M between V and W is defined as

$$C(G, H, M) = \sum_{(u,v)\in M} n(f_G(u), f_H(v)) + \sum_{v\in V\setminus V_M} \Delta_G(v) + \sum_{w\in W\setminus W_M} \Delta_H(w).$$

The (predefined) cost for a vertex $v \in V$ that is not mapped to a vertex in W is $\Delta_G(v) \in \mathbb{Q}$, similar the cost of a not mapped vertex $w \in W$ to a vertex in V is $\Delta_H(w)$. One of the sums $\sum_{v\in V\setminus V_M} \Delta_G(v)$, $\sum_{w\in W\setminus W_M} \Delta_H(w)$ will always be zero.

The vertex signature graph distance between G and H is defined as

$$D(G, H) = \min_{M\in\mathcal{M}(V,W)} C(G, H, M).$$

Dynamic Time Warping (DTW). Dynamic time warping is an algorithm that aligns temporal sequences of different speeds and also different lengths to one another. Each element of the sequences must be matched to at least one element in the other sequence, this matching is called *warping* and the sequence of the warped elements is called *warping path* [56]. The *dynamic time warping distance* [56, 1] between two temporal graphs is based on the warping path between the layers of the temporal graphs $\mathcal{G} = (\mathcal{V}, \mathcal{E}_1, \mathcal{E}_2, ..., \mathcal{E}_T)$ and $\mathcal{H} = (\mathcal{W}, \mathcal{F}_1, \mathcal{F}_2, ..., \mathcal{F}_U)$.

A *warping path* of order $T \times U$ is a set $p = \{p_1, ..., p_L\}$ of $L \geqslant 1$ pairs $p_l = (i_l, j_l)$, where

- $p_1 = (1, 1)$ and $p_L = (T, U)$ and
- $p_{l+1} \in \{(i_l + 1, j_l + 1), (i_l, j_l + 1), (i_l + 1, j_l)\} \forall 1 \leqslant l < L$.

All warping paths of order $T \times U$ are denoted by $\mathcal{P}_{T,U}$.

Dynamic Temporal Graph Warping (dtgw) Distance. We adapt the notation from Froese *et al.* [1].

The *dynamic temporal graph warping (dtgw)* distance [1] is a similarity measure between two temporal graphs based on vertex signature mapping and dynamic time warping. The vertex signature is specified by a function, which assigns to every vertex one or more numbers. Then, the vertices of both graphs are mapped in order that the cost of the mapping is minimized. As the graphs are temporal, the layers of the graphs are warped by dynamic time warping and the cost of the mapping in each layer is computed. The vertex mapping between the temporal graphs is constant over the layers of the temporal graphs. The dtgw-distance is the sum of the costs of the optimal mapping on the layers of the temporal graphs which are warped in the optimal warping path.

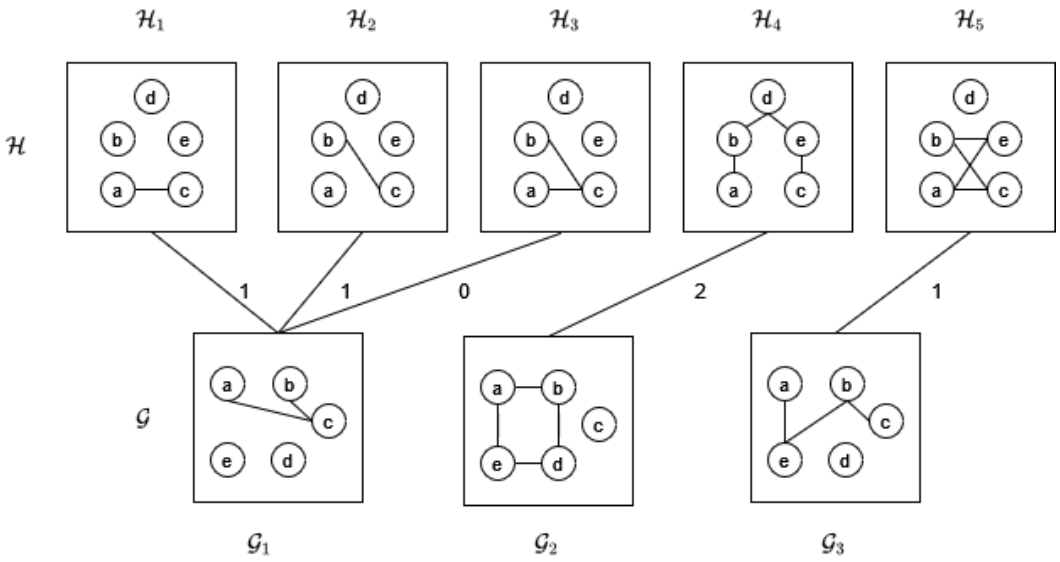


Figure 2.1: Example of the dtgw-distance between two temporal graphs $\mathcal{G}$ with lifetime 3 and $\mathcal{H}$ with lifetime 5. The vertex mapping M is represented by the letters in the nodes and the dynamic warping path is $p = \{(1,1),(2,1),(3,1),(4,2),(5,3)\}$. The vertex signature functions for $\mathcal{G}$ and $\mathcal{H}$ use the degree of the vertex as vertex signature. The metric is the absolute value of the difference. The dtgw-distance is $\text{dtgw}(\mathcal{H},\mathcal{G}) = 1 + 1 + 0 + 2 + 1 = 5$.

Formally, let $\mathcal{G} = (\mathcal{V}, \mathcal{E}_1, \mathcal{E}_2, ..., \mathcal{E}_T)$ and $\mathcal{H} = (\mathcal{W}, \mathcal{F}_1, \mathcal{F}_2, ..., \mathcal{F}_U)$ be two temporal graphs.

The *dtgw-distance* between the temporal graphs $\mathcal{G}$ and $\mathcal{H}$ for the vertex signature functions $f_{\mathcal{G}_1}, ..., f_{\mathcal{G}_T} : V \to \mathbb{Q}^k$ and $f_{\mathcal{H}_1}, ..., f_{\mathcal{H}_U} : V \to \mathbb{Q}^k$ is

$$\text{dtgw}(\mathcal{G}, \mathcal{H}) = \min_{M \in \mathcal{M}(\mathcal{V},\mathcal{W})} \min_{p \in \mathcal{P}_{T,U}} \sum_{(i,j)\in p} C(\mathcal{G}_i, \mathcal{H}_i, M).$$

Figure 2.1 shows an example of the dtgw-distance between two temporal graphs with 5 vertices and a lifetime of 5 and 3.

Random Walk Kernel. We adapt the notation from Oettershagen *et al.* [2].

A *random walk kernel* is a similarity measure between two static labeled graphs [50, 51, 52, 2]. It counts the common walks of both graphs. Two walks are considered to be common if the labels encountered on the walks are equal. A j-step random walk kernel considers walks up to length j.

Chapter 3

Methods

In this chapter, we introduce our model to represent an ASL sign as a temporal graph in detail. First, we will briefly outline the dataset, which consists of video recordings of signs from the internet. In Section 3.2 we describe how the temporal graphs are derived based on the x- and y-coordinates of body, face, and hand keypoints. Then, we present the similarity measure methods that are used to compare the temporal graphs.

3.1 Dataset

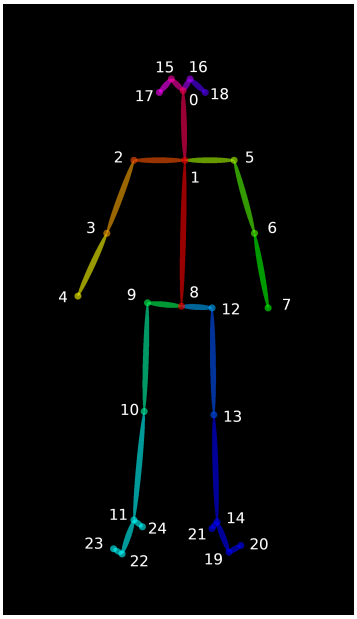
We use the dataset of Word-Level American Sign Language (WLASL) curated by Li *et al.* [12][1]. It contains video recordings of more than 3,000 signs with on average 10 samples per sign and from 119 different signers. They collected the videos from different online sources and annotated them with information about the meaning of the sign, the dialect, and the signer seen in the video. Further, they normalized the frame rate of every video to 25 frames per second.

For every frame, they extracted 55 keypoints, 13 body and 21 keypoints per hand, with OpenPose [37, 38] (see Figure 3.1). As mostly the upper body is recorded in the videos, the OpenPose keypoints for the legs (8-14) and the feet (20-23) were omitted. The resulting data consist of the x- and y-coordinates as well as a detection confidence of the keypoints for each frame [37].

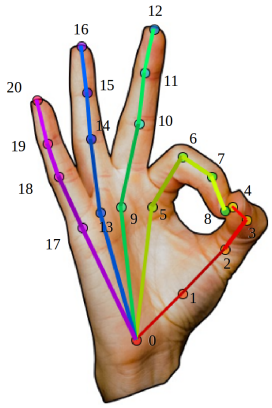
They also ranked the signs by the number of samples and created the subsets *asl100*, *asl300*, *asl1000*, and *asl2000*. *Asl100* contains the 100 signs with the most samples per sign and has an average of 20 samples, whereas *asl2000* contains 2000 signs and has only 10 samples on average (see Table 3.1). Every sign has at least 7 samples.

As there are different dialects in sign languages, not all samples per sign are in the same dialect. The number of samples which represent the same movement may be smaller, for example *computer* has in total 30 samples, but 3 different variations and only 7, 11, and 12 samples per variation.

[1] Available at https://github.com/dxli94/WLASL



(a) Body keypoints that are tracked by OpenPose [57].



(b) Hand keypoints that are tracked by OpenPose [58].

Figure 3.1: Skeleton model of the body and hand keypoints obtained by OpenPose [37, 38]. The numbers are the IDs of the keypoints.

Table 3.1: Overview of the subsets of the dataset with the average number of samples per sign and the minimum number of samples. The sign with the most samples is *book* with 40 samples followed by *drink* with 35 samples.

Subset	Average number of samples (rounded down)	Minimum number of samples
asl100	20	18
asl300	17	14
asl1000	13	10
asl2000	10	7

3.2 Temporal Graphs representing ASL signs

For every sign sample in the dataset, we construct a temporal graph. The vertices are the 55 keypoints detected by OpenPose. The lifetime of the graph is the number of frames and for every timestep, we create an edge set. Our idea is to draw an edge between vertices if we consider these vertices to be close.

There is an edge between two vertices if the Euclidean distance between these vertices, computed between the x- and y-coordinates of the related keypoints, is in a predefined range. These ranges vary, as finger points cannot be as distant to each other as a finger point to a face point. Therefore, we specify different ranges for fingers, hands, and body points.

(a) ASL sign *B* [59, Figure 4] (b) ASL sign *five* [59, Figure 4]

Figure 3.2: ASL signs to measure the distance between the fingers

To determine the ranges, we considered the distances from one finger point to other finger points. The lower bound for all distances is 3 px to sort out mismatched points; for instance, if two keypoints were assigned to the same position. In order to distinguish between splayed fingers and fingers touching each other, we use the ASL signs *five* and *B* (see Figure 3.2) to measure the distance between the fingers. Accordingly, we define the upper bound for the distance between two finger points on one hand as 10 px. Between finger points on different hands the upper boundary is 20 px and between the finger points and the body points the upper boundary is 30 px.

To summarize, given a video recording of a sign and the extracted coordinates by OpenPose, we represent the sign sample as a temporal graph, where the number of frames is the lifetime, the 55 keypoints are the vertices and there is an edge between two vertices if the distance between the keypoints is in a specific range.

In addition, we defined a threshold to only consider the keypoints, which are important in the sign. Especially in signs, where only one hand is used, the keypoints of the other hand can be omitted. In particular, if the hands are not completely inside the boundary box for OpenPose, errors may occur in detecting the keypoints. We define the threshold as the maximum y value for the finger points, as the upper left corner is (0,0). The distances to finger keypoints that are cut off by the threshold are not considered for the temporal graph. In order to determine the y value of this threshold, we use a model called the "eight-head model", which is used by artists [60]. The model suggests that the height of a human is about eight times the height of the head and the torso has about three times the height of the head [61]. The eight-head model is a rough approximation and we use it only as an indicator for the threshold. We choose the threshold to be a bit lower than the elbows and the eight-head model suggests, that it is around 1.75 times the size of the head below the neck [61, Figure 5]. We estimate the head size by using the distance between the keypoints 0 and 1, i.e., between the nose and the neck (see Figure 3.1a). The threshold is set to 1.75 times this nose-neck distance added to the y value of the keypoint 1.

Figure 3.3 shows one frame of the sign *drink*, where the left hand is not used in the sign but would create edges in the temporal graph, as the points are close together. With the threshold, these points are not considered for the graph, as they are below the blue

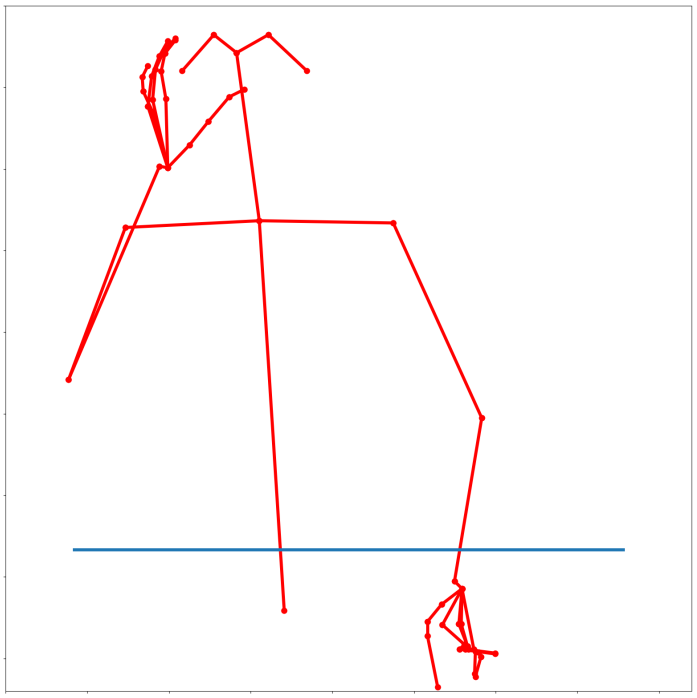


Figure 3.3: Example of the extracted keypoints by OpenPose [37, 38] and the defined threshold on the sign *drink*. The connections between the keypoints indicate the anatomical relation and do not belong to the temporal graph. The blue horizontal line indicates the threshold. The hand points of the left hand would create unnecessary edges in the graph, but with the threshold, they are omitted.

line indicating the threshold. A disadvantage of this threshold could be that important points are cut off if it is set incorrectly. This is particularly likely in the case of children, as the eight-head model is very imprecise for them.

3.3 Similarity Measure Methods for the Sign Temporal Graphs

In order to measure the similarity of two signs, we construct the temporal graphs as described in Section 3.2 and use temporal graph comparison methods. We use the dtgw-distance (Chapter 2) and the random walk kernel on the directed line graph expansion (see Section 3.3.2) to determine the similarity of two temporal graphs. As the computing of dtgw is NP-hard in general [1, Theorem 4.1], we use the alternating minimization (AM) heuristic presented by Froese *et al.* [1, Section 5.3], to estimate the dtgw-distance between

the temporal graphs. First, we outline how the AM heuristic works and define three vertex signature functions for the estimation of the dtgw-distance. Then, we describe the directed line graph expansion for temporal graphs, presented by Oettershagen *et al.* [2, Section 3.2].

3.3.1 Heuristic Computation of dtgw-Distance

Froese *et al.* [1] introduce the alternating minimization heuristic to estimate the dtgw-distance. The AM heuristic is based on the observation [1, Observation 3.1] that for a fixed vertex mapping the warping path can be computed in polynomial time. This also applies to a fixed warping path, where the vertex mapping can be computed in polynomial time. They presented four initializations for the heuristic, two for an initial warping path and two for an initial vertex mapping [1]. We choose to initialize the heuristic with the shortest warping path of length $\max(T, U)$, which is closest to the diagonal, due to the short running time $\mathcal{O}(T + U)$ of this initialization, where T and U denote the lifetimes of the temporal graphs.

Vertex Signature Functions

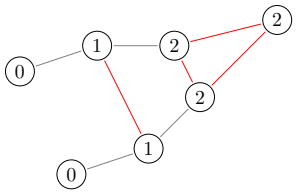
To compute the dtgw-distance, we need the vertex signature functions $f_{\mathcal{G}_1}, ..., f_{\mathcal{G}_T} : \mathcal{V} \to \mathbb{Q}^k$. We use the same vertex signature function for every layer. For our ASL sign temporal graphs, we use the following three functions and compare their performance in Section 4.1.1. The first one uses only the degree of a vertex as a signature. In the second function, we combine the degree with a label, which indicates if the vertex belongs to a body or hand keypoint, so that a hand keypoint is not mapped to a body keypoint. The third function uses even more detailed information about the associated keypoint, as it combines the degree with a label indicating the keypoint ID of the vertex. We use the same keypoint IDs for the left hand as for the right hand, since left-handed signers mirror the movement of a right-handed signer. Therefore, we do not force a vertex mapping based on the keypoint IDs, since we would then have to decide on the right- or left-handed version. Figure 3.4 shows an example graph layer with the vertex signatures assigned by the different vertex signature functions. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}_2, ..., \mathcal{E}_T)$ be a temporal graph and let $\mathrm{d}_t(v) \in \mathbb{N}$ denote the degree of vertex $v \in V$ at timestep t.

Degrees. This vertex signature function assigns the degree to each vertex $v \in \mathcal{V}$ and for each layer $t \in [T]$.

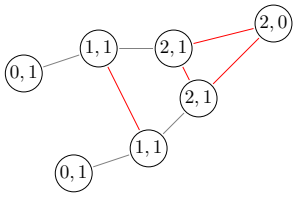
$$f_{\mathrm{d},t}(v) = d_t(v)$$

Degrees combined with body/hands distinction This function assigns for every vertex the degree of the vertex as well as an indication, if it is a body point or a hand point, where 0 denotes body points and 1 denotes hand points. Let $b(v) : V \to [0, 1]$ be a function, which returns 0, if the related keypoint ID belongs to a body point and 1 if it belongs to a hand point.

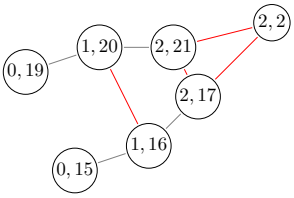
$$f_{b,t}(v) = (\mathrm{d}_t(v), b(v))$$



(a) Example for the degrees function. The vertex label is the degree of the vertex.



(b) Example for the degrees combined with body/hand distinction function. The vertex label is the degree of the vertex followed by a number, which indicates whether it is a body (0) or a hand point (1).



(c) Example for the degrees combined with keypointID labels function. The vertex label is the degree of the vertex followed by the number of the keypoint related to the vertex.

Figure 3.4: Example of the vertex signature functions on one layer of a temporal graph for the thumb and index finger on the right hand and the right shoulder. The gray edges indicate the anatomical connection between the points and the red edges are the edges of the temporal graph.

Degrees combined with keypoint ID labels This vertex signature function uses the degree of a vertex and a label of the vertex, which is the keypoint ID. Let $c(v) : \mathcal{V} \to \mathbb{N}$ be a function, which returns the keypoint ID of a vertex v, except for the left hand, where it returns the identifiers of the right hand to avoid distinguishing between both hands.

$$f_{c,t}(v) = (\mathrm{d}_t(v), c(v))$$

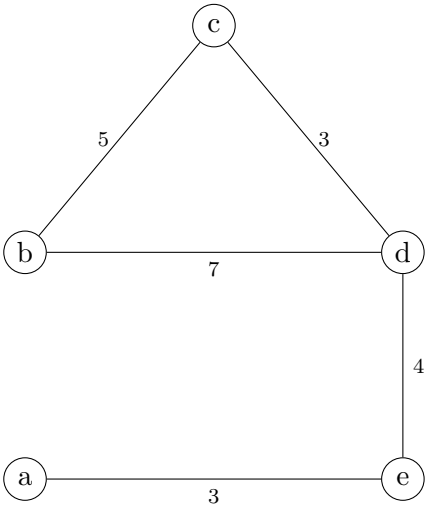
3.3.2 Temporal Graph Kernel

A kernel is another method to measure the similarity of two graphs. Oettershagen *et al.* [2] provide several graph kernels for temporal graphs. They devised different methods to model a temporal graph as a static graph, so that conventional static graph kernels can be applied.

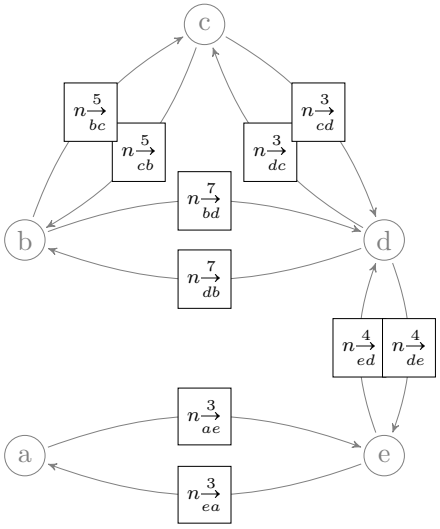
Line Graph Expansion for Kernels We adapt the notation from Oettershagen *et al.* [2].

This representation preserved the temporal information of a temporal graph. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}_1, \mathcal{E}_2, ..., \mathcal{E}_T, \ell)$ be a vertex-labeled temporal graph.

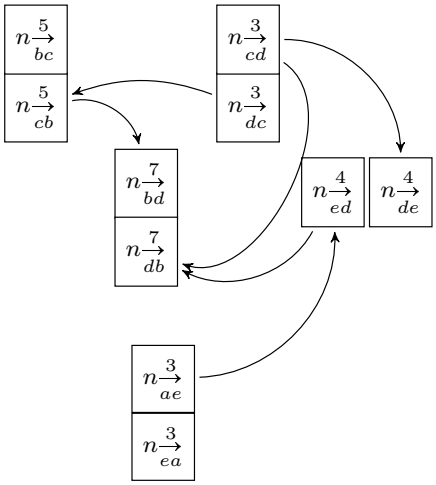
For every edge $e = \{u, v\} \in \mathcal{E}_t, t \in [T]$, two vertices are generated and called $n\overset{t}{\underset{uv}{\rightarrow}}$ and $n\overset{t}{\underset{vu}{\rightarrow}}$. These vertex labels store the temporal information. There is a directed edge between the two vertices $n\overset{t}{\underset{uv}{\rightarrow}}$ and $n\overset{s}{\underset{vw}{\rightarrow}}$ if in the original temporal graph there are the edges $\{u, v\} \in \mathcal{E}_t$ and $\{v, w\} \in \mathcal{E}_s$, with $t < s, t, s \in [T]$ and $u, v, w \in \mathcal{V}$.



(a) A temporal graph $\mathcal{G}$, the number(s) on the edges indicate the timestep(s) of the edges.



(b) Generated vertices for every temporal edge for the line graph expansion representation.



(c) Directed line graph expansion of graph $\mathcal{G}$.

Figure 3.5: Example of a transformation from a temporal graph $\mathcal{G}$ to the direct line graph representation. The initial temporal graph is shown in (a). We generate two vertices for every edge, each of them represents a direct edge and also encodes the timestep of the edge, (b) shows the resulting vertices. Finally, the directed edges between the vertices are drawn and we get the line graph representation (c) of (a).

The labels for the new vertex $n\xrightarrow[uv]{t}$ is $\ell'(n\xrightarrow[uv]{t}) = (\ell(u,t), \ell(v,t+1))$. The directed line graph is defined as $DL(\mathcal{G}) = (\mathcal{V}', E', \ell')$.

Figure 3.5 shows an example of a transformation of a temporal graph to a directed line graph. The size of the directed line graph expansion is in $\mathcal{O}(|\mathcal{E}|^2)$, where $|\mathcal{E}| = \sum_{i=1}^{T} |\mathcal{E}_i|$ and T is the lifetime of the original temporal graph.

The classical static graph kernels can now be used on these graphs, which encode temporal information. We use the *j-step random walk kernel* on the line graph expansion (*DL-RW*) [2] and the vertex labels are set to the keypoint IDs.

3.4 Classification Methods

After we derived the pairwise similarities between the temporal graphs, we use different classification methods to classify signs based on their temporal graph. For the sign classification task, we use the *k-Nearest Neighbor Classification* algorithm [16] and a *C-Support Vector Machine (C-SVM)* [17]. The k-nearest neighbor classifier takes the computed distances as a distance metric for the classification of a test sample. For the computed temporal graph kernel, we also use a C-SVM on the kernel for classification. For the dtgw-distances, we further use a C-SVM in a featured-based dissimilarity space [62], where the dtgw-distances to the other graphs are the features of the sample. With a *Principal Component Analysis (PCA)* [63], we reduce the dimensionality of the dissimilarity space to test how many features are necessary for the classification.

We use the k-nearest neighbor and PCA implementation from Scikit-learn [64] and the C-SVM implementation from Scikit-learn [64] based on libSVM [65].

Chapter 4

Experiments

In this chapter, we analyze our graph representation using the dtgw-distance and the temporal graph kernel DL-RW. For the dtgw-distance estimation, we also compare the three vertex signature functions we defined. At the end, we compare our accuracies to the results Li *et al.* [12] achieved on the WLASL dataset[2]. We implemented the construction of our temporal graph and adapted the implementation of dtgw[3] [1] in Python. For the temporal graph kernel, we used the implementation from Oettershagen *et al.* [2][4]. As some signs are quite similar, we consider in addition to the class label with the highest probability (*top-1*), the five (*top-5*), and ten most probable (*top-10*) class labels, as the correct meaning of the sign may be deduced from the context.

We ran the experiments on a 3.60 GHz Intel Xeon E5-1620 processor system single-threaded, the operating system was Ubuntu 18.04.5 LTS.

4.1 Temporal Graph Representation of Signs

In this section, we test our graph model and investigate the impact of the threshold under which the hand points are ignored. In Section 4.1.1 we evaluate the three vertex signature functions defined in Section 3.3.1 for the dtgw-distance estimation and their impact on the classification accuracy. Next, we test the temporal graph kernel DL-RW on the graphs in Section 4.1.2. Further, we compare the results of the dtgw-distance with the best vertex signature function to results of the temporal graph kernel DL-RW.

4.1.1 Results of the Vertex Signature Functions for the dtgw-Distance

In two separate experiments, we compared the three vertex signature functions on the temporal graphs, which were constructed with and without threshold. In the first experiment, we used k-nearest neighbor with the dtgw-distances as distance metric. For the second experiment, we used a C-Support Vector Machine (C-SVM) with a linear kernel on the dissimilarity space, where the dtgw-distances to the training samples are the features of a sample.

[2]Available at https://github.com/dxli94/WLASL
[3]Available at https://fpt.akt.tu-berlin.de/software/dtgw/
[4]Available at https://gitlab.com/tgpublic/tgkernel

Table 4.1: Results of the correctly classified signs in percent on the *asl100* subset computed by the AM heuristic for the graph with and without the threshold and for the three vertex signature function. For all experiments, the same train and test split are used with a train and test split ratio of 70:30.

Vertex signature function	Graphs without threshold	Graphs with threshold
degrees	13.47 % ($k = 4$)	17.68 % ($k = 1, 2, 4$)
degrees with 0/1 labels	14.14 % ($k = 1, 2, 3$)	19.02 % ($k = 1, 2$)
degrees with keypoint ID labels	22.05 % ($k = 1, 2$)	30.81 % ($k = 1, 2, 4$)

(a) Classification with k-nearest neighbor with $k \in [1, 50]$. k was set individually per vertex signature function and per graph model to reach the best possible results.

Vertex signature function	Graphs without threshold	Graphs with threshold
degrees	15.82 % (dim=20)	18.35 % (dim=30)
degrees with 0/1 labels	14.14 % (dim=90)	19.53 % (dim=40)
degrees with keypoint ID labels	16.82 % (dim=90)	21.04 % (dim=50)

(b) Classification with C-SVM with a linear kernel in the dissimilarity space with dimension reduction by PCA. For every vertex signature function and both graph models, the dimension was reduced from 100% down to 10% in steps of 10 to detect the dimension that leads to the highest accuracy. The C-parameter was selected from $\{10^{-3}, 10^{-2}, ..., 10^{2}, 10^{3}\}$, however, the C-parameter had hardly any influence on the results.

The first function takes only the degree as vertex signature, the second one combines the degree with a label, which indicates if the vertex is associated with a body or a hand keypoint and the third one combines the degree with a label, which represents the keypoint ID. The metric for all three functions is the Euclidean distance. The average running time of the AM heuristic with the different vertex signature functions was around 11 hours for the asl100 subset, which could be reduced by parallelism, as the AM heuristic takes on average only 4 iterations to converge.

As can be seen in Table 4.1, the graph with threshold and the vertex signature function *degrees with keypoint ID labels* achieved the highest accuracy with k-nearest neighbor. The threshold leads to an improvement of the classification on all vertex signature functions. Using k-nearest neighbor for classification, the body/hand distinction labels only led to a minor improvement compared to just the degree as vertex signature. However, the keypoint ID labels have significantly improved the accuracy, using k-nearest neighbor for classification. For the classification with the C-SVM on the dissimilarity space, the different vertex signature function had less of an impact on the

accuracy.

Overall, the highest accuracy of the dtgw-distance on the asl100 subset was reached by k-nearest neighbor on the temporal graphs, which were constructed with the threshold that omits hand points below it. Nevertheless, the highest achieved accuracy is still low, only around 30 % on the asl100 subset.

An example, where the classification was incorrect on the asl100 subset is *study*. None of the five *study* test samples were correctly classified by k-nearest neighbor. Figure 4.1 shows three warped frames with the keypoints and the associated layers of the temporal graphs of a training sample from *study* and *clothes* compared to a test sample of *study*. Figure 4.1b is the test sample of *study*, which is classified as *clothes*. The temporal graphs of both of these signs are very similar, as we only have 2D information and do not use the orientation of the palm. The test sample has a lifetime of 41, where the layers of the temporal graph did not change by much, as the person already has the hands in position at the beginning of the video and only moves the right hand a bit closer to the left hand for the sign *study*. Figure 4.1a is a training sample of *study* with a lifetime of 77, where the person starts with the hands down and also ends the video that way. The warping path needs to warp every layer of one graph to a layer of the other and the warping of the "start" and "end" phase of the training sign to the test sign increased the distance between the graphs. In the sample of the sign *clothes*, the hands with finger splayed remain close to each other during the movement, even in the "start" and "end" phases. Hence, the distance between the layers of *clothes* warped to the test sign *study* tend to be smaller and it is classified as *clothes*.

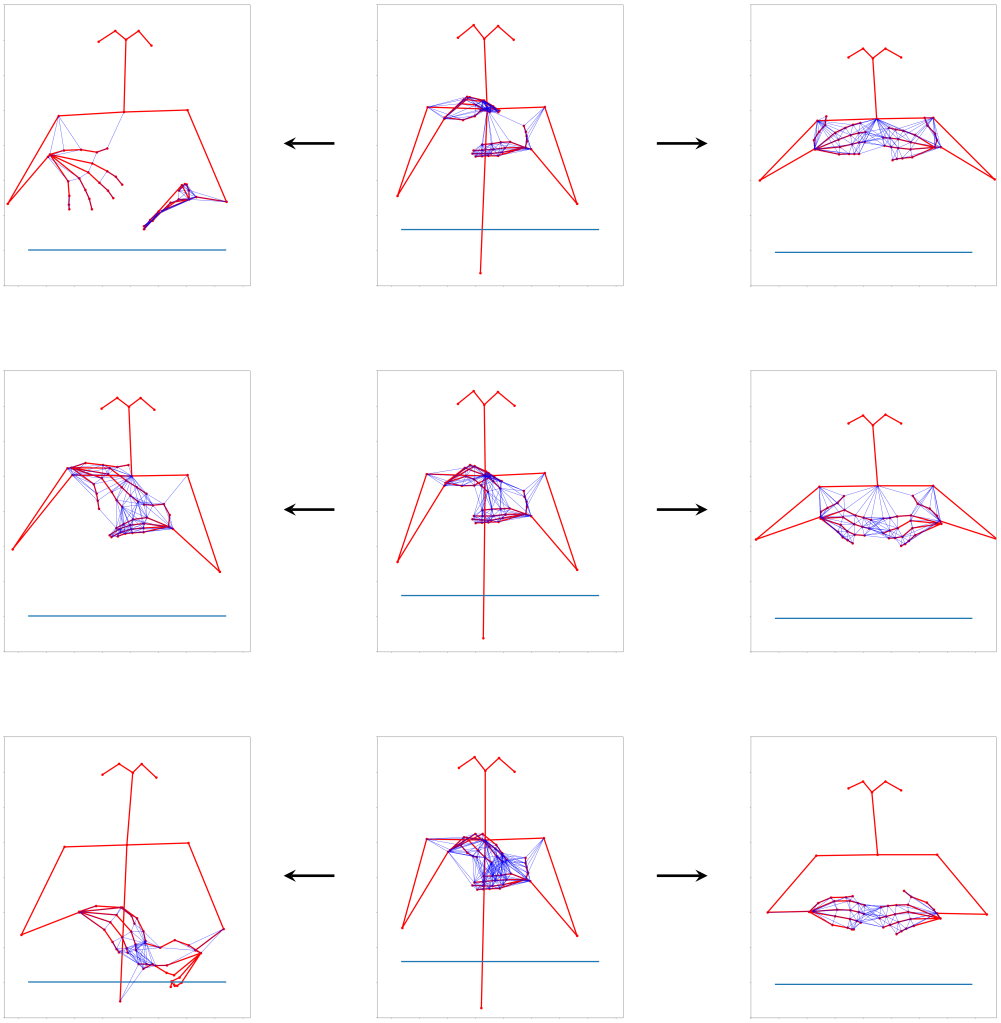
Further, the time warping cannot handle situations where the movement of a sign is repeated multiple times in a video sample, as the warping path needs to be monotonically increasing. For example, the movement of the sign *book* is often executed twice, but only once is possible and samples of both variants are in the dataset.

The average standard deviation of the videos within a sign class is 0.85 seconds [12, Section 3.3], thus around 21 frames. The dynamic time warping can compensate for that if the sign is signed at a different pace, as it can warp one layer of one graph to multiple consecutive layers of the other graph.

4.1.2 Results of the Temporal Graph Kernel

In the following, we present the results achieved by the 2-step random walk kernel on the directed line graph expansion (*DL-RW*) of the temporal graphs with and without threshold and with keypoint IDs as the vertex labels. The average running time was around 10 minutes. We also tried the Weisfeiler-Lehman kernel on the directed line graph expansion on smaller subsets with only 50 signs, but it needed significantly more RAM than the random walk kernel by achieving similar accuracies, which is why we decided to use the random walk kernel. For classification, we used k-nearest neighbor and a C-Support Vector Machine (C-SVM) on the computed kernel.

Table 4.2 reports the accuracies achieved on the same train and test split as in Section 4.1.1. The threshold did improve the classification accuracies by more than 10 % on both classifiers. The C-SVM was slightly better than the k-nearest neighbor classification, it could predict the meaning of over 60 % of the test samples correctly.



(a) Three frames (24, 34, 74) of a training sign of *study*.

(b) Three frames (6, 8, 38) of the test sign *study*.

(c) Three frames (20, 22, 40) of the training sign *clothes*

Figure 4.1: Example of the dynamic time warping distance between the test sign *study* in the middle, a training sample of *study* on the left, and a training sample of *clothes*. The arrows indicate the warped frames. The dark blue lines illustrate the extracted graph layer, and the light blue line is the threshold. The red lines do not belong to the temporal graph, they only represent the anatomical connection between the keypoints. In this example, the dtgw-distance, computed with the vertex signature function *degrees combined with the keypoint ID labels*, is smaller between (b) and (c) than between (b) and (a). Hence, the test sample in the middle of *study* is classified as *clothes* by k-nearest neighbor and not as *study*. The temporal graphs between *study* and *clothes* are relatively similar, as both hands are flat, close to each other, and in front of the chest.

Table 4.2: Results of the correctly classified signs in percent on the *asl100* subset computed by the temporal graph kernel DL-RW (2-step random walk kernel on the directed line graph expansion) for the graphs with and without threshold. For all experiments, the same train and test split as for dtgw are used with a train and test split ratio of 70:30.

Graphs without threshold	Graphs with threshold
45.73 % ($k = 4$)	57.75 % ($k = 1, 2$)

(a) Classification with k-nearest neighbor with $k \in [1, 50]$. k was set individually per graph model to reach the best possible results.

Graphs without threshold	Graphs with threshold
53.10 % ($C = 10^1$)	63.64 % ($C = 10^3$)

(b) Classification with C-SVM on the DL-RW kernel. The C-parameter was selected individually for the graph models from $\{10^{-3}, 10^{-2}, ..., 10^2, 10^3\}$.

Compared to the dtgw-distance, k-nearest neighbor on the similarity measures of the DL-RW kernel classified almost twice as many signs from the test split correctly.

The distances between the temporal graphs are illustrated in Figure 4.2, it shows, that the distances between different signs computed by the DL-RW kernel are often greater than within one sign, as the dtgw-distance between different signs appear to be more similar.

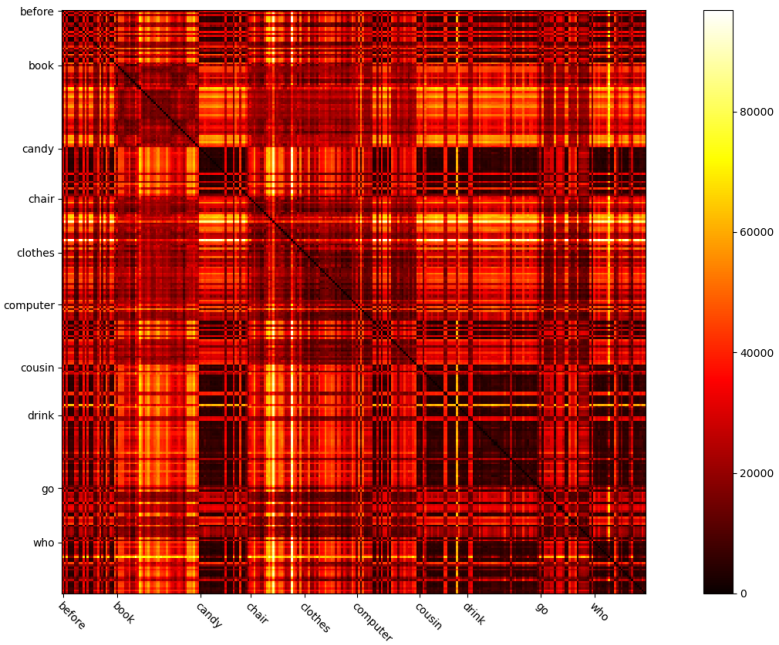
Regarding the example in Figure 4.1, the DL-RW kernel with k-nearest neighbor could correctly classify the test sample Figure 4.1b as *study*. It seems that the DL-RW kernel could handle the missing parts in the test sample better than the dtgw-distance, as the most similar training sample was the training sign of *study* Figure 4.1a.

4.2 Performance Evaluation on the Sign Data Subsets

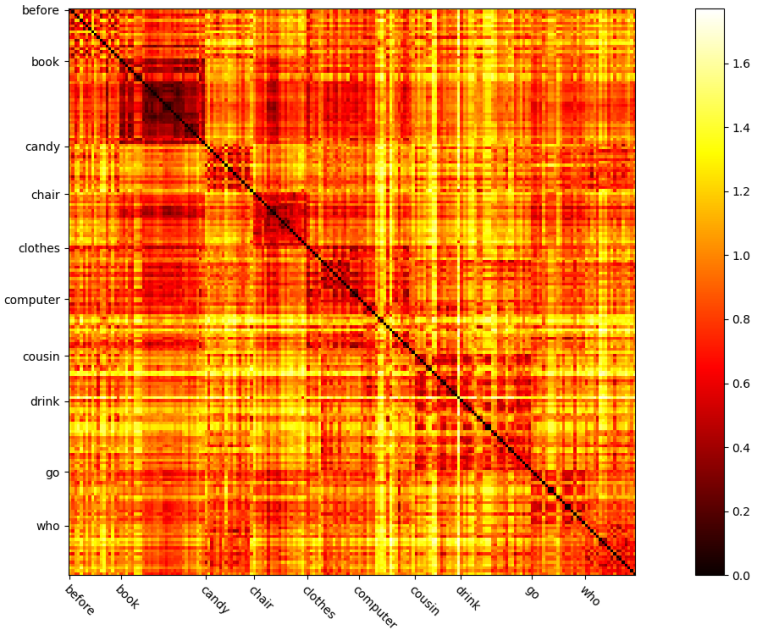
We compare the highest accuracies we achieved on the WLASL dataset [12] subsets *asl100* and *asl300* by using our temporal graph model with the threshold to the accuracies Li *et al.* [12] achieved with Pose-TCGN and I3D on the same training and test split. The pose-based *Temporal Graph Convolution Network (TGCN)* [12, Chapter 4.2.2] uses a deep graph convolution network on the 2D keypoint coordinates extracted by OpenPose, representing the body as a fully-connected graph with weighted undirected edges for every timestep. The I3D is a 3D convolutional network, trained on different datasets [66] and fine-tuned on the WLASL videos [12, Chapter 4.1.2].

For this experiment, the vertex signature function for the dtgw-distance [1] estimation is *degrees with the keypoint ID labels*. For the kernel, we used the implementation of the directed line graph expansion with j step random walk kernel *DL-RW* [2] with $j = 2$ and we used the keypoint IDs as labels. The left hand was labeled with the right-hand IDs, as signs are not linked to a specific hand. The classification method for the dtgw-distance was k-nearest neighbor and for the DL-RW kernel, we used a C-SVM on it.

As Table 4.3 shows, we can reach a top-1 accuracy of 62.79 % with the temporal graph kernel DL-RW on the asl100 subset which is close to the accuracy that I3D achieved.



(a) Heatmap of dtgw-distances computed by the AM heuristic with degrees and keypoint ID labels as the vertex signature.



(b) Heatmap of the distances computed by the Random Walk kernel on the line graph representation (DL-RW)

Figure 4.2: Heatmaps of the pairwise distances for the ten signs with the most samples alphabetically sorted. The darker the area, the smaller the distance between the graphs. It would be best if the distances within one sign are smaller than to other signs, so there should be only darker squares on the diagonal. (a) shows, unfortunately, that this is not for all signs the case with the dtgw-distance, for example, the distances within the sign *candy* and between samples of *candy* to samples of *cousin*, *drink*, and *who* seem to be similar. In heatmap (b), the distances computed by the DL-RW kernel seems to be greater between different signs.

Table 4.3: Results of the top-1, top-5, and top-10 accuracy in percent on the *asl100* subset achieved by the dtgw-distance and the random walk kernel on the directed line graph with random walk length 2 on our temporal graph model compared the accuracies achieved by Li *et al.* [12, Table 3] on the Pose-TCGN and the I3D model on the same training and test split. We could not run the dtgw-distance on the asl300 subset, due to the long running time.

Method	asl100			asl300		
	top-1	top-5	top-10	top-1	top-5	top-10
dtgw-distance	26.74	51.55	66.66			
DL-RW Kernel	62.79	86.05	90.70	47.01	73.35	81.74
Pose-TCGN	55.43	78.68	87.60	38.32	67.51	79.64
I3D	65.89	84.11	89.62	56.41	79.94	86.98

The dtgw-distance with the *degree with the keypoint ID label* function could only achieve a top-1 accuracy of 26.74 %. Due to the long running time, we could not test the dtgw-distance on the asl300 subset.

Nevertheless, the I3D model still shows the highest top-1 accuracy with over 56 % on the asl300 dataset. I3D is advantageous here, as it has already been pre-trained on other ASL sign data and uses the video recordings directly.

Compared to Pose-TCGN, which like ours uses the OpenPose data, we could achieve better classification accuracies with our temporal graphs and the DL-RW kernel on the asl100 and asl300 subsets.

Chapter 5

Conclusion

We proposed a model to represent American Sign Language signs as temporal graphs by using 2D information of human body keypoints extracted from video recordings by OpenPose [37, 38]. With the 2-step random walk kernel on a directed line graph expansion of the temporal graphs [2], we could predict the English meaning of a sign correctly for 62 % of the test samples on the subset containing 100 different signs of the WLASL [12] dataset.

Using the dtgw-distance to measure the similarity between the temporal graphs of signs could only achieve an accuracy of 30 %, which may be due to the vertex signature functions we used. We also tested other vertex signatures on smaller subsets, as the size of the connected component or the closeness, but these did not perform better than degrees either.

Our model is much simpler than the 3D convolutional network [12] but can achieve similar results on 100 different signs. On the larger asl300 dataset, the top-1 accuracy is less than 50 %. A disadvantage is that we do not use the videos directly like the 3D convolutional network, but first need to extract keypoints, which can be another source of error. Therefore, our approach is not sufficient for larger datasets, but with some improvements, it could be an alternative approach for sign language recognition.

Future work It is a question of future research to investigate the performance of the temporal graph model on more signs, such as the asl1000 or even asl2000 subset of WLASL, which we could not test due to the large size of the directed line graph expansion for the random walk kernel and the long running time of the AM heuristic. Besides, more samples per sign could improve the classification, as some sign variations have only 7 samples. Further, our temporal graph model could be improved, as at the moment the boundaries for the ranges are fixed and with variable boundaries the size of a person would be considered. The temporal graph could also include information of the relative position of limbs, as the hands, so that the position of the hands to each other is represented. Additionally, in some sign language families, for example, the German Sign Language, the mouth movement is important to distinguish between signs, so the temporal graph model needs to be extended for these sign languages. Also, other vertex signature functions for the dtgw-distance computation could lead to an improvement of the classification. For example, a function that uses the keypoint ID labels of not only

the vertex but also from the neighborhood combined with a centrality measure.

In addition, unbounded dynamic time warping [67] could overcome the problem of repetitions of a sign in a sample and other movements that do not belong to the sign, as it does not have the start-end restriction like the classical dynamic time warping.

Literature

[1] V. Froese, B. J. Jain, R. Niedermeier, and M. Renken, *Comparing temporal graphs using dynamic time warping*, *Soc. Netw. Anal. Min.*, vol. 10, no. 1, 50:1–50:16, 2020 (cit. on pp. 5, 9–11, 13, 14, 20, 21, 25, 29).

[2] L. Oettershagen, N. M. Kriege, C. Morris, and P. Mutzel, *Temporal graph kernels for classifying dissemination processes*, in *Proceedings of the 2020 SIAM International Conference on Data Mining, SDM 2020, Cincinnati, Ohio, USA, May 7-9, 2020*, C. Demeniconi and N. V. Chawla, Eds., SIAM, 2020, pp. 496–504 (cit. on pp. 5, 9–11, 15, 21, 22, 24, 25, 29, 33).

[3] A. Casteigts, K. Meeks, G. B. Mertzios, and R. Niedermeier, *Temporal graphs: Structure, algorithms, applications (dagstuhl seminar 21171)*, *Dagstuhl Reports*, vol. 11, no. 3, pp. 16–46, 2021 (cit. on p. 9).

[4] H. Molter, *Classic graph problems made temporal - a parameterized complexity analysis*, Ph.D. dissertation, Technical University of Berlin, Germany, 2020 (cit. on pp. 9, 11).

[5] P. Holme and J. Saramäki, *Temporal networks*, *Physics reports*, vol. 519, no. 3, pp. 97–125, 2012 (cit. on p. 9).

[6] E. Valdano, L. Ferreri, C. Poletto, and V. Colizza, *Analytical computation of the epidemic threshold on temporal networks*, *Phys. Rev. X*, vol. 5, 2 2015 (cit. on p. 9).

[7] N. Santoro, W. Quattrociocchi, P. Flocchini, A. Casteigts, and F. Amblard, *Time-varying graphs and social network analysis: Temporal indicators and metrics*, *CoRR*, vol. abs/1102.0629, 2011. arXiv: 1102.0629 (cit. on p. 9).

[8] N. Gaumont, C. Magnien, and M. Latapy, *Finding remarkably dense sequences of contacts in link streams*, *Soc. Netw. Anal. Min.*, vol. 6, no. 1, 87:1–87:14, 2016 (cit. on p. 9).

[9] T. Viard, M. Latapy, and C. Magnien, *Computing maximal cliques in link streams*, *Theor. Comput. Sci.*, vol. 609, pp. 245–252, 2016 (cit. on p. 9).

[10] W. Sandler, *Symbiotic symbolization by hand and mouth in sign language*, vol. 2009, no. 174, pp. 241–275, 2009 (cit. on p. 9).

[11] G. Mathur and D. J. Napoli, *Deaf around the World: The Impact of Language*. Oxford University Press, Nov. 2010 (cit. on p. 9).

[12] D. Li, C. Rodriguez, X. Yu, and H. Li, *Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison*, in *The IEEE winter conference on applications of computer vision*, 2020, pp. 1459–1469 (cit. on pp. 9–11, 17, 25, 27, 29, 31, 33).

[13] H. R. V. Joze and O. Koller, *MS-ASL: A large-scale data set and benchmark for understanding american sign language*, in *30th British Machine Vision Conference 2019, BMVC 2019, Cardiff, UK, September 9-12, 2019*, BMVA Press, 2019, p. 100 (cit. on pp. 9–11).

[14] J. Huang, W. Zhou, H. Li, and W. Li, *Sign language recognition using 3d convolutional neural networks*, pp. 1–6, 2015 (cit. on pp. 9, 11).

[15] Y. Ye, Y. Tian, M. Huenerfauth, and J. Liu, *Recognizing american sign language gestures from within continuous videos*, in *2018 IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2018, Salt Lake City, UT, USA, June 18-22, 2018*, Computer Vision Foundation / IEEE Computer Society, 2018, pp. 2064–2073 (cit. on pp. 9, 11).

[16] N. S. Altman, *An introduction to kernel and nearest-neighbor nonparametric regression*, *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992 (cit. on pp. 10, 24).

[17] C. Cortes and V. Vapnik, *Support-vector networks*, *Machine Learning*, vol. 20, no. 3, pp. 273–297, Sep. 1995 (cit. on pp. 10, 24).

[18] M. Mohandes, S. A-Buraiky, T. Halawani, and S. Al-Baiyat, *Automation of the arabic sign language recognition*, May 2004, pp. 479 –480 (cit. on p. 10).

[19] K. Hoshino, *Dexterous robot hand control with data glove by human imitation*, *IEICE Trans. Inf. Syst.*, vol. 89-D, no. 6, pp. 1820–1825, 2006 (cit. on p. 10).

[20] R. Y. Wang and J. Popovic, *Real-time hand-tracking with a color glove*, *ACM Trans. Graph.*, vol. 28, no. 3, p. 63, 2009 (cit. on p. 10).

[21] N. El-Bendary, H. M. Zawbaa, M. S. Daoud, A. E. Hassanien, and K. Nakamatsu, *Arslat: Arabic sign language alphabets translator*, in *2010 International Conference on Computer Information Systems and Industrial Management Applications, CISIM, Krakow, Poland, October 8-10, 2010*, IEEE, 2010, pp. 590–595 (cit. on p. 10).

[22] K. Assaleh, T. Shanableh, and M. Zourob, *Low complexity classification system for glove-based arabic sign language recognition*, in *Neural Information Processing*, T. Huang, Z. Zeng, C. Li, and C. S. Leung, Eds., Berlin, Heidelberg: Springer, 2012, pp. 262–268 (cit. on p. 10).

[23] A. Z. Shukor, M. F. Miskon, M. H. Jamaluddin, F. bin Ali Ibrahim, M. F. Asyraf, and M. B. bin Bahar, *A new data glove approach for malaysian sign language detection*, *Procedia Computer Science*, vol. 76, pp. 60–67, 2015, 2015 IEEE International Symposium on Robotics and Intelligent Sensors (IEEE IRIS2015) (cit. on p. 10).

[24] M. I. Sadek, M. N. Mikhael, and H. A. Mansour, *A new approach for designing a smart glove for arabic sign language recognition system based on the statistical analysis of the sign language*, in *2017 34th National Radio Science Conference (NRSC)*, 2017, pp. 380–388 (cit. on p. 10).

[25] X. Zhang, X. Chen, Y. Li, V. Lantz, K. Wang, and J. Yang, *A framework for hand gesture recognition based on accelerometer and EMG sensors*, *IEEE Trans. Syst. Man Cybern. Part A*, vol. 41, no. 6, pp. 1064–1076, 2011 (cit. on p. 10).

[26] C. Chuan, E. Regina, and C. Guardino, *American sign language recognition using leap motion sensor*, in *13th International Conference on Machine Learning and Applications, ICMLA 2014, Detroit, MI, USA, December 3-6, 2014*, IEEE, 2014, pp. 541–544 (cit. on p. 10).

[27] N. Rossol, I. Cheng, and A. Basu, *A multisensor technique for gesture recognition through intelligent skeletal pose analysis*, *IEEE Trans. Hum. Mach. Syst.*, vol. 46, no. 3, pp. 350–359, 2016 (cit. on p. 10).

[28] L. Quesada, G. López, and L. A. Guerrero, *Automatic recognition of the american sign language fingerspelling alphabet to assist people living with speech or hearing impairments*, *J. Ambient Intell. Humaniz. Comput.*, vol. 8, no. 4, pp. 625–635, 2017 (cit. on p. 10).

[29] H. Hongo, M. Yasumoto, Y. Niwa, M. Ohya, and K. Yamamoto, *Focus of attention for face and hand gesture recognition using multiple cameras*, in *4th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2000), 26-30 March 2000, Grenoble, France*, IEEE Computer Society, 2000, pp. 156–162 (cit. on p. 10).

[30] K. Lai, J. Konrad, and P. Ishwar, *A gesture-driven computer interface using kinect*, in *IEEE Southwest Symposium on Image Analysis and Interpretation, SSIAI 2012, Santa Fe, New Mexico, USA, April 22-24, 2012*, IEEE Computer Society, 2012, pp. 185–188 (cit. on p. 10).

[31] S. G. M. Almeida, F. G. Guimarães, and J. A. Ramírez, *Feature extraction in brazilian sign language recognition based on phonological structure and using RGB-D sensors*, *Expert Syst. Appl.*, vol. 41, no. 16, pp. 7259–7271, 2014 (cit. on p. 10).

[32] F. Yasir, P. W. C. Prasad, A. Alsadoon, and A. Elchouemi, *SIFT based approach on bangla sign language recognition*, in *IEEE 8th International Workshop on Computational Intelligence and Applications, IWCIA 2015, Hiroshima, Japan, November 6-7, 2015*, IEEE, 2015, pp. 35–39 (cit. on p. 10).

[33] A. Tharwat, T. Gaber, A. E. Hassanien, M. K. Shahin, and B. Refaat, *Sift-based arabic sign language recognition system*, in *Afro-European Conference for Industrial Advancement - Proceedings of the First International Afro-European Conference for Industrial Advancement, AECIA 2014, Addis Ababa, Ethiopia, 17-19 November 2014*, A. Abraham, P. Krömer, and V. Snásel, Eds., ser. Advances in Intelligent Systems and Computing, vol. 334, Springer, 2014, pp. 359–370 (cit. on p. 10).

[34] L. Ding and A. M. Martínez, *Modelling and recognition of the linguistic components in american sign language*, *Image Vis. Comput.*, vol. 27, no. 12, pp. 1826–1844, 2009 (cit. on p. 10).

[35] H. Cooper, E. Ong, N. Pugeault, and R. Bowden, *Sign language recognition using sub-units*, *J. Mach. Learn. Res.*, vol. 13, pp. 2205–2231, 2012 (cit. on p. 10).

[36] C. F. Benitez-Quiroz, K. Gökgöz, R. B. Wilbur, and A. M. Martinez, *Discriminant features and temporal structure of nonmanuals in american sign language*, 2, vol. 9, Public Library of Science, Feb. 2014, pp. 1–17 (cit. on p. 10).

[37] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh, *Openpose: Realtime multi-person 2d pose estimation using part affinity fields*, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019 (cit. on pp. 10, 17, 18, 20, 33).

[38] T. Simon, H. Joo, I. Matthews, and Y. Sheikh, *Hand keypoint detection in single images using multiview bootstrapping*, in *CVPR*, 2017 (cit. on pp. 10, 17, 18, 20, 33).

[39] T. Starner and M. Group, *Visual recognition of american sign language using hidden markov models*, May 1995 (cit. on p. 11).

[40] T. Starner, J. Weaver, and A. Pentland, *Real-time american sign language recognition using desk and wearable computer based video*, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 12, pp. 1371–1375, 1998 (cit. on p. 11).

[41] H. Wang, A. Stefan, S. Moradi, V. Athitsos, C. Neidle, and F. Kamangar, *A system for large vocabulary sign search*, in *Trends and Topics in Computer Vision - ECCV 2010 Workshops, Heraklion, Crete, Greece, September 10-11, 2010, Revised Selected Papers, Part I*, K. N. Kutulakos, Ed., ser. Lecture Notes in Computer Science, vol. 6553, Springer, 2010, pp. 342–353 (cit. on p. 11).

[42] P. Jangyodsuk, C. Conly, and V. Athitsos, *Sign language recognition using dynamic time warping and hand shape distance based on histogram of oriented gradient features*, in *Proceedings of the 7th International Conference on PErvasive Technologies Related to Assistive Environments, PETRA 2014, Island of Rhodes, Greece, May 27 - 30, 2014*, F. Makedon, M. Clements, C. Pelachaud, V. Kalogeraki, and I. Maglogiannis, Eds., ACM, 2014, 50:1–50:6 (cit. on p. 11).

[43] Y. Bengio and P. Frasconi, *Input-output hmms for sequence processing*, *IEEE Trans. Neural Networks*, vol. 7, no. 5, pp. 1231–1249, 1996 (cit. on p. 11).

[44] A. Just, O. Bernier, and S. Marcel, *HMM and IOHMM for the recognition of mono- and bi-manual 3d hand gestures*, in *British Machine Vision Conference, BMVC 2004, Kingston, UK, September 7-9, 2004. Proceedings*, A. Hoppe, S. Barman, and T. Ellis, Eds., BMVA Press, 2004, pp. 1–10 (cit. on p. 11).

[45] A. D. Wilson and A. F. Bobick, *Parametric hidden markov models for gesture recognition*, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 9, pp. 884–900, 1999 (cit. on p. 11).

[46] C. Vogler and D. N. Metaxas, *Parallel hidden markov models for american sign language recognition*, in *Proceedings of the International Conference on Computer Vision, Kerkyra, Corfu, Greece, September 20-25, 1999*, IEEE Computer Society, 1999, pp. 116–122 (cit. on p. 11).

[47] R. Elakkiya, *Machine learning based sign language recognition: A review and its research frontier*, *J. Ambient Intell. Humaniz. Comput.*, vol. 12, no. 7, pp. 7205–7224, 2021 (cit. on p. 11).

[48] P. Holme, *Modern temporal network theory: A colloquium*, *CoRR*, vol. abs/1508.01303, 2015. arXiv: 1508.01303 (cit. on p. 11).

[49] O. Michail, *An introduction to temporal graphs: An algorithmic perspective*, *Internet Math.*, vol. 12, no. 4, pp. 239–280, 2016 (cit. on p. 11).

[50] T. Gärtner, P. A. Flach, and S. Wrobel, *On graph kernels: Hardness results and efficient alternatives*, in *Computational Learning Theory and Kernel Machines, 16th Annual Conference on Computational Learning Theory and 7th Kernel Workshop, COLT/Kernel 2003, Washington, DC, USA, August 24-27, 2003, Proceedings*, B. Schölkopf and M. K. Warmuth, Eds., ser. Lecture Notes in Computer Science, vol. 2777, Springer, 2003, pp. 129–143 (cit. on pp. 11, 15).

[51] M. Sugiyama and K. M. Borgwardt, *Halting in random walk kernels*, in *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds., 2015, pp. 1639–1647 (cit. on pp. 11, 15).

[52] S. V. N. Vishwanathan, N. N. Schraudolph, R. Kondor, and K. M. Borgwardt, *Graph kernels*, *J. Mach. Learn. Res.*, vol. 11, pp. 1201–1242, 2010 (cit. on pp. 11, 15).

[53] N. Shervashidze, P. Schweitzer, E. J. Van Leeuwen, K. Mehlhorn, and K. M. Borgwardt, *Weisfeiler-lehman graph kernels. Journal of Machine Learning Research*, vol. 12, no. 9, 2011 (cit. on p. 11).

[54] R. Diestel, *Graph Theory*, 5th. Springer Publishing Company, Incorporated, 2017 (cit. on p. 13).

[55] S. Jouili and S. Tabbone, *Graph matching based on node signatures*, in *Graph-Based Representations in Pattern Recognition, 7th IAPR-TC-15 International Workshop, GbRPR 2009, Venice, Italy, May 26-28, 2009. Proceedings*, A. Torsello, F. Escolano, and L. Brun, Eds., ser. Lecture Notes in Computer Science, vol. 5534, Springer, 2009, pp. 154–163 (cit. on p. 13).

[56] H. Sakoe and S. Chiba, *Dynamic programming algorithm optimization for spoken word recognition*, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978 (cit. on p. 14).

[57] *Pose Output Format (BODY_25)*, https://cmu-perceptual-computing-lab.github.io/openpose/web/html/.github/media/keypoints_pose_25.png, [Online; accessed 2021-10-06] (cit. on p. 18).

[58] *Hand output format*, https://cmu-perceptual-computing-lab.github.io/openpose/web/html/.github/media/keypoints_hand.png, [Online; accessed 2021-10-06] (cit. on p. 18).

[59] C. Valli, *Linguistics of American Sign Language : an introduction*. Washington, D.C: Gallaudet University Press, 2011 (cit. on p. 19).

[60] S. Spencer, *Zbrush digital sculpting human Anatomy*. John Wiley & Sons, 2010 (cit. on p. 19).

[61] L. De Silva, *Audiovisual sensing of human movements for home-care and security in a smart environment*, *International Journal on Smart Sensing and Intelligent Systems*, vol. 1, Jan. 2008 (cit. on p. 19).

[62] R. P. W. Duin, M. Loog, E. Pekalska, and D. M. J. Tax, *Feature-based dissimilarity space classification*, in *Recognizing Patterns in Signals, Speech, Images and Videos - ICPR 2010 Contests, Istanbul, Turkey, August 23-26, 2010, Contest Reports*, D. Ünay, Z. Çataltepe, and S. Aksoy, Eds., ser. Lecture Notes in Computer Science, vol. 6388, Springer, 2010, pp. 46–55 (cit. on p. 24).

[63] I. Jolliffe, *Principal component analysis*, in *Encyclopedia of Statistics in Behavioral Science*. 2005. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/0470013192.bsa501 (cit. on p. 24).

[64] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, *Scikit-learn: Machine learning in Python*, *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011 (cit. on p. 24).

[65] C.-C. Chang and C.-J. Lin, *Libsvm: A library for support vector machines*, *ACM transactions on intelligent systems and technology (TIST)*, vol. 2, no. 3, pp. 1–27, 2011 (cit. on p. 24).

[66] J. Carreira and A. Zisserman, *Quo vadis, action recognition? A new model and the kinetics dataset*, in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, IEEE Computer Society, 2017, pp. 4724–4733 (cit. on p. 29).

[67] X. Anguera, R. Macrae, and N. Oliver, *Partial sequence matching using an unbounded dynamic time warping algorithm*, in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2010, 14-19 March 2010, Sheraton Dallas Hotel, Dallas, Texas, USA*, IEEE, 2010, pp. 3582–3585 (cit. on p. 34).