Springer Data Mining and Knowledge Discovery manuscript No.
(will be inserted by the editor)

On Sign Language Recognition from Skeletal Trajectory Data with Interpretable Classifiers

Joachim Gudmundsson · Martin P. Seybold · John Pfeifer

Received: date / Accepted: date

Abstract Recent advances in tracking sensors and pose estimation software enable smart systems to use trajectories of skeleton joint locations for supervised learning. We study the problem of accurately recognizing sign language words, which is key to narrowing the communication gap between hard and non-hard of hearing people.

Our method explores a geometric feature space that we call 'sub-skeleton' aspects of movement. We assess similarity of feature space trajectories using natural trajectory distance measures, which enables clear and insightful nearest neighbor classification. The simplicity and scalability of our basic method allows for immediate application in different data domains with little to no parameter tuning.

We demonstrate the effectiveness of our basic method, and a boosted variation, with experiments on data from different application domains and tracking technologies. Surprisingly, our methods improve on or are on par with more complex recent state-of-the-art algorithms for all tested data sets.

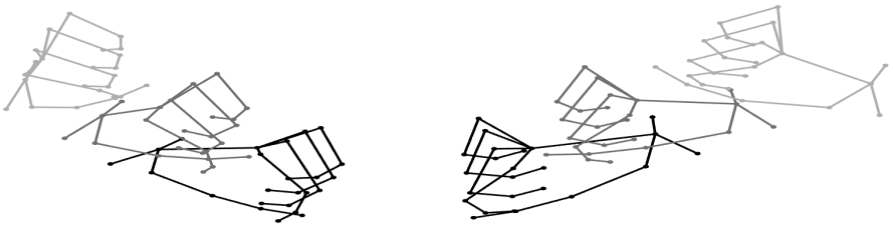
Keywords classification · data mining · spatio-temporal · sign language · machine learning · feature construction

1 Introduction

The problem of automatically and accurately identifying the meaning of human body movement has gained research interest due to advances in motion capture systems, artificial intelligence algorithms, and powerful hardware. Wearable tracking technology, such as micro electromechanical systems, have been shrinking in size and cost while improving in accuracy and availability. Also, less invasive motion capture systems such as Microsoft's Kinect or the Leap Motion controller have been used to capture motion of human actors for recently released benchmark data sets (e.g. KinTrans (linedanceAI, 2020), LM (Hernandez et al., 2020),

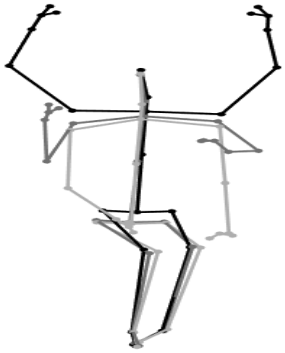
School of Computer Science, University of Sydney, Sydney, Australia

J. Pfeifer E-mail: johnapfeifer@yahoo.com



(a) LM right/left hand joints for sign *coat*.

(b) KinTrans upper body joints for sign *credit card*.



(c) NTU RGB+D full body joints for action *cheer up*.

Fig. 1: Human skeletal joint examples showing movement from earlier (lighter) to later (darker) time frames. See Section 5.1 for data set details.

NTU RGB+D (Shahroudy et al., 2016), see Figure 1). The systems typically capture several input sources, e.g. RGB video and depth map, to track a bounding box of the actor and the skeletal joint positions therein, for every sensor frame.

This work studies the problem of recognising patterns in human sign language. Human sign languages are systems of communication that use manual movement patterns of arms and hands as well as non-manual elements, such as head, cheek and mouth posture, to convey meaning. We focus on data sets that consist of extracted, single word labeled inputs along with the sequences of skeletal joint locations of the actors. Our goals are to attain high classification accuracy with acceptable learning and query latency on diverse data sources, requiring little to no parameter tuning. We are particularly interested in simple, interpretable methods that, in turn, provide insight for curation of publicly available catalogs of sign language.

Though there are recent works (Adaloglou et al., 2020; Rastgoo et al., 2020) on sign language recognition from video data, our problem is typically considered as a special case of human action (Presti and La Cascia, 2016) or gesture (Mantecón et al., 2019) classification from *skeleton-based* data, both receiving tremendous attention from the research community. Improved solutions for this special case might provide the deaf community with computational tools that improve communication between signers and sign language illiterates (Young et al., 2019; Kushalnagar et al., 2018).

The Core of the Problem "One of the biggest challenges of using pose-based features is that semantically similar motions may not necessarily be numerically similar" (Yao et al., 2011). This statement, that pervades action recognition articles in various forms, is a key motivation for our method. Clearly, actors perform movement patterns with varying temporal windows and varying speed/pause modulations therein. Hence (overly) simplistic similarity measures can easily be fooled to return meaningless values.

We solve this with *speed-invariant* similarity measures that simultaneously capture the shape and spatial location of trajectories, without the use of numeric frame numbers.

1.1 Related Work

The survey by Presti and La Cascia (2016, Table 4) provides an overview of previous human action recognition methods and their benchmark performance, and includes popular skeleton-based benchmark data sets such as Berkley's Multimodal Human Action Database (MHAD) and a contribution by the University of Central Florida (UCF) (Ofli et al., 2013; Ellis et al., 2013). It categorizes the foundation of known approaches in automatically mined joint features (M) as well as user-specified features on dynamics (D) and joints, where the latter is sub-categorized in Spatial Relations (S), Geometric Relations (G), and Key-Pose Dictionaries (K).

We briefly discuss the benchmark leading methods. 'LDS-mSVM' (D) considers trajectories as result of Linear Dynamic Systems, on location and velocity variables that are sub-sampled from skeleton limbs over different time scales, on which Multiple Kernel Learning is used (Chaudhry et al., 2013a).

'Riemann-kNN' (G) fixes a skeleton-localized coordinate systems, for transformation invariance. Using differential geometry, a spatial trajectory is considered as curve in the associated velocity space that, after speed vector scaling, all have unit L_2 function norm. Then velocity curves are assessed by the 'elastic shape' distance in the kNN classification (Devanne et al., 2014a, Section III).

Category (S) also includes recent methods using Graph Convolutional Networks which were successfully adapted to architectures with one vertex per joint and frame. 'ST-GCN' contains spatial edges, between the joint neighbors, and temporal edges, between a joint's previous and next frame neighbor (Yan et al., 2018). The spatial and temporal convolutions are performed interleaved, and long-range spatial edges were introduced independently by Li et al. (2018). Improvements of this method were achieved based on key poses, frame-wise and sequence-wise attention edges, adaptive self-attention, motifs, bone features, 3-frame temporal look-ahead, Neural Searching, and cross space-time edges spanning up to a user-specified hop distance (Li et al., 2019a,b; Shi et al., 2019a; Wen et al., 2019; Shi et al., 2019b; Gao et al., 2019; Peng et al., 2020; Liu et al., 2020). Recently Hernandez et al. (2020) provided 'Kinematics-LSTM' (S) that, using convolutional LSTM networks on kinematic features, achieves highly accurate sign language recognition from Leap Motion skeleton data.

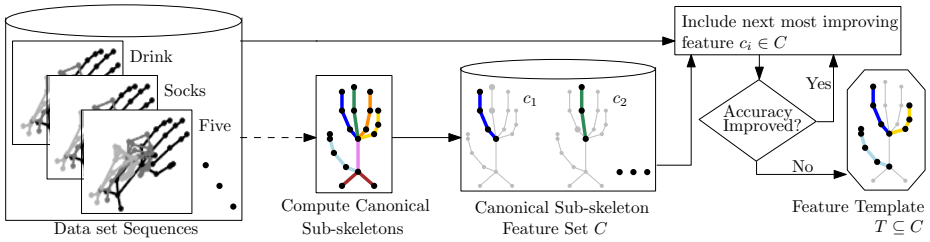


Fig. 2: Automated mining from sub-skeleton features produces a highly discriminative Feature Template.

1.2 Contribution and Paper Structure

The general idea of our method is to map signed words into a feature space that captures absolute and relative movement of sets of joints. Based on the high-dimensional feature trajectories, we apply standard classifiers that use geometric similarity measures (Section 2) as input. The main technical difficulty is the generation of a descriptive feature space of spatio-temporal trajectories, i.e. determining the most discriminative way to transform spatial joint locations into high-dimensional trajectories. Motivated by human sign language, we consider how to effectively navigate the vast space of absolute and relative movement patterns that subsets of joints describe, which we call 'sub-skeleton' features.

We propose a novel method that *automatically mines* a set of maximal discriminating sub-skeleton features for general skeleton formats (see Figure 2 and Section 3). The main contributions of this are as follows.

- The mining method discovers those movement aspects in the data sets that are highly discriminating. Since all feature trajectories directly relate to the input data, the merits of each classification result can be *interpreted* and visualized naturally in terms of a geometric trajectory similarity measure (e.g. Figure 3).
- Simple Nearest Neighbor and Ensemble Boosted Classification (see Section 4) achieve similar or improved accuracy on popular benchmark sets of diverse tracking technologies. Particularly noteworthy is our high accuracy results on *very small* training sets (see Section 5).
- To the best of our knowledge, we are the first to apply trajectory similarity measures (e.g. the Fréchet distance) as feature predictors for sign language and human action classification problems.

Furthermore, our publicly available[1] implementation achieves very good query latency results on standard, non-GPU, desktop hardware. Surprisingly, classification based on sub-skeleton feature trajectories also improves on benchmarks in Human Action recognition, indicating that our method can be generalized to other recognition problems.

[1] https://github.com/japfeifer/frechet-queries

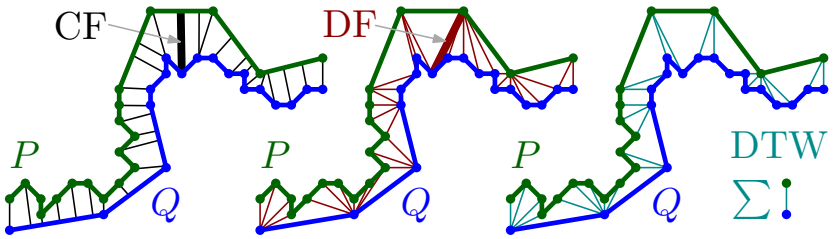


Fig. 3: Measures on two visually similar trajectories P (moves slow then fast) and Q (moves fast then slow) with a total of 19 and 18 frames, respectively.

2 Setup and Problem Definition

A skeleton G is an undirected connected graph where each vertex represents a joint. In our setting G models a part of a human body skeleton where vertices are adjacent if their joints have a rigid connection (e.g. a bone). As input for a word signed by an actor, we are given a sequence S of n frames $\langle G_1, \ldots, G_n \rangle$, and each G_i holds the 3D location of the joints at time step i. Frame frequency is typically constant but depends heavily on the capture technology. Sequences have varying time lengths, for example, between 0.2 and 30 seconds (c.f. Table 1). Every sequence is labeled with one class and the actor's body movement is performed once, or a small number of times, per sign class.

The input data $\mathbb{D}$ is a set of sequences $\{S_1, \ldots, S_\ell\}$ and is partitioned into two sets: the training set $\mathbb{D}_r$ and the testing set $\mathbb{D}_t$. Our aim is to classify each sequence in the test set, using only sequences from the training set, as accurately and efficiently as possible.

2.1 Similarity Measures of Trajectories

Various motion capture devices differ when comparing their frame rates and movement capture accuracy. Furthermore, some actors perform motions faster than others for the same class, even though they describe the same spatial pattern. Our proposed method converts every input sequence of frames into one or more feature trajectories. Therefore, we are interested in trajectory similarity measures that are time-invariant. Continuous Fréchet and continuous Dynamic Time Warp (Alt and Godau, 1995; Munich and Perona, 1999) provide these properties in a strict sense. We do however also discuss the simpler measures discrete Fréchet and Dynamic Time Warp due to their popularity as they are more meaningful in capturing trajectory similarity, especially when compared to other methods such as computing equally sampled Euclidean distances over a sequence. Below we discuss three trajectory distance measures that we tested with our method: continuous Fréchet (CF) distance, discrete Fréchet (DF) distance and Dynamic Time Warp (DTW) (e.g. Figure 3). The first two are metrics, while the latter is not.

A d dimensional trajectory P of size n is a directed polygonal curve through a sequence of n vertices (points) $\langle p_1, \ldots, p_n \rangle$ in $\mathbb{R}^d$ and each contiguous pair of vertices in P is connected by a straight line segment.

Continuous Fréchet (CF) Metric. The continuous Fréchet distance for two trajectories P, Q in $\mathbb{R}^d$ is defined as follows. Imagine one person walks along P and another person walks along Q. Each person must walk along his or her curve from start to finish. Neither person is allowed to travel backwards, but otherwise they are free to vary their speed. The cost of any fixed walk is the maximum distance that is attained between the two people at any time during the walk. Different walks can have different costs, and the continuous Fréchet distance between P and Q, denoted $\mathrm{CF}(P, Q)$, equals the minimum possible cost over all walks. More formally, we have

$$\mathrm{CF}(P, Q) = \inf_{h} \max_{p \in P} \Big\| p, h(p) \Big\|,$$

where $\|\cdot, \cdot\|$ denotes the Euclidean distance and $h : P \to Q$ is a homeomorphism that assigns to every point $p \in P$ a point $h(q) \in Q$.

The continuous Fréchet distance is a (pseudo) metric and can be computed in $\mathcal{O}(mn \log mn)$ time, where the m and n are the sizes of the two trajectories.

Discrete Fréchet (DF) Metric. This distance (Eiter and Mannila, 1994) is very similar to CF, except that the two persons are jumping along their trajectories from vertex to vertex and are never considered to be in the interior of an edge. It can be computed in $\mathcal{O}(mn)$ time.

Dynamic Time Warp (DTW). The DTW similarity measure (Keogh and Ratanamahatana, 2005) only differs slightly from the discrete Fréchet distance in replacing the operation 'maximum' with 'summation' for the distances of the matched vertices. As a result DTW does not fulfill the triangle inequality, and is therefore not a metric. It can be computed in $\mathcal{O}(mn)$ time.

Properties of the trajectory distance measures. The three distances are designed to measure the spatial similarity between trajectories and do not take time into consideration. The different measures have various advantages and drawbacks. The main advantage with the DTW measure is that it is less impacted by outliers and "smooths over" significant differences in comparison to the Fréchet distances which capture the worst-case changes between two trajectories. However, DTW is not a metric and two visually similar trajectories may result in a large measure if the frame rate varies or the same action is performed faster by one subject and slower by another.

CF and DF are metrics and can take advantage of metric indexing (Gudmundsson et al., 2020) which provides very fast query times for k-NearestNeighbor computations. The CF distance is generally regarded to capture visual similarity more precisely than DF, especially if the sampling rate is very sparse, since it takes the continuity of the shapes into account.

3 Mining Sub-Skeleton Features

A feature template is iteratively constructed for a given input $\mathbb{D}_r$ and skeleton G (see Figure 2), and is used to generate feature trajectories. We first provide background motivation and then describe our feature template mining algorithm.

Every subset of joints from skeleton G is considered a sub-skeleton feature f in our feature space F, that we call an *absolute feature*. For example, the left index finger and right leg joints can form one sub-skeleton. Additionally, F can also contain *relative features*, which are subsets of joints but relative to another joint, i.e. center of a moving coordinate system. E.g. the joints of the right arm relative to the neck joint. Note that the union of two absolute sub-skeleton sets is also in F and the same holds for relative sets, if they have the *same* reference joint (this is the 'union' operation of our mining algorithm).

Feature trajectories are generated as follows. Every sub-skeleton feature $f \in F$ maps an input sequence $\langle G_1, \ldots, G_n \rangle$ to a feature trajectory P of size n and dimensionality $3|f|$. Absolute features map each frame G_i to a corresponding trajectory point p_i by projection. In the mapping of relative sub-skeleton features, the 3D location of the reference joint in G_i is subtracted from those joints of f.

Though the space F is independent of the sequences in $\mathbb{D}$, its size is *exponential* in the number of joints $|G|$ of the skeleton. As motivated in Section 1, we ideally want to choose a small number of features from F, that allow us to achieve high accuracy, for a prescribed classifier that is simple and interpretable.

3.1 Feature Template Mining Algorithm

We next describe an algorithm that greedily selects a small number of features $\{f_1, \ldots, f_l\}$, where $l \leq 1 + |G|$. F contains $|G|^2$ many singleton sets (i.e. absolute and relative) and any $f \in F$ can be derived as the union of $|f|$ many singletons, regardless of f being an absolute or relative sub-skeleton. The singletons $C \subseteq F$, which we call *canonical sub-skeletons*, form the basis of our greedy exploration. In the case when $|G|$ is very large, we derive a smaller set C by covering chains of degree 2 vertices with a single set and use only a few branching and leaf vertices as reference joints, i.e. C does not contain singletons but rather small sets of adjacent joints (e.g. index finger or thumb joints, Figure 2).

We use simple, standard classifiers to determine which features to add to the feature template. The simplest is a Nearest Neighbor method that finds the closest label in $\mathbb{D}_r$ based on a trajectory distance measure (see Section 4 for the precise description of our classifiers). To employ underlying classifiers during our greedy exploration, we initially partition the training set $\mathbb{D}_r$ randomly with a 1:2-split into $\mathbb{D}'_r$ and $\mathbb{D}'_t$ and proceed as follows:

1. Start with the empty set $T = \emptyset$.
2. Compute $\forall\ c \in C \setminus T$ the classification accuracy of $T \cup \{c\}$ on $\mathbb{D}'_r$ and $\mathbb{D}'_t$.
3. If one such c improves over the last iteration, add the best c to T and GOTO 2.
4. Return T.

We call the resulting set T the *feature template*, which contains a small number of absolute and relative canonical sub-skeletons. Clearly, the algorithm can be executed for different classifiers and trajectory similarity measures to determine a feature set with (locally) maximal accuracy.

Note that the construction of the feature template from the training set $\mathbb{D}_r$ is only to accommodate fair comparison of the final accuracy against other recognition methods. For the purpose of pure pattern discovery, one may well use the whole data set $\mathbb{D}$ in the selection process.

4 Classifying Feature Trajectories

We focus on classifiers that are solely based on geometric similarity measures between the feature trajectories of the query and training sequences. This not only provides results that can be interpreted and visualized, but also allows us to employ clustering based metric indexing techniques to achieve very small query latency (e.g. Section 5.4). Given the above description of the selection process, the feature template T may contain several absolute features and relative features that have *the same* reference joint. We aggregate those by taking the union of these sets of joints. Let $l(T)$ denote the number of features after the merge. Hence $l(T) \leq 1 + |G|$ and we have an equal number of generated feature trajectories per input sequence. For these, we describe a simple Nearest Neighbor and an ensemble boosted classification technique. Both may achieve slightly improved accuracy by optional pre-processing with natural normalizations, which we describe next.

The spatial coordinates of a sequence S are normalized with a translation, rotation, standardizing the limb lengths, and padding with an at-rest pose. Each is independently applicable. Variations of the first three are common (Presti and La Cascia, 2016), hence we only discuss our at-rest pose briefly. Two sequences for the same class may have similar motion in the middle of their sequence, but different start or end frame joint locations. This can result in a large trajectory distance even though they are the same signed word. An at-rest pose is a standard skeletal pose frame that depicts the subject in a relaxed or neutral state, and can result in a smaller trajectory distance. Each data set has one at-rest pose definition frame, which is inserted at the beginning and end of each sequence.

Our simple Nearest Neighbor (NN-s) classifier concatenates the spatial coordinates of the $l(T)$ feature trajectories frame-wise to derive a *single* trajectory per input sequence. Then, for a given query sequence we return the label of the entity from the training set that has the smallest value in the geometric similarity measure.

Our k-Nearest Neighbor (kNN-s) classifier expands on NN-s by employing a normalized weighted distance scheme to select a predicted label from a set of k-closest results. For a given kNN-s query, the smallest result distance $\bar{d}$ is used to weight the k results inverse proportional in their distance to the query. Then the label with the highest total weight is returned.

Our trajectory Distance Matrix (DM) *single* and *multi* trajectory methods construct training and testing matrices composed of trajectory distances, which are fed into more complex classifiers (e.g. SVM or the random subspace linear discriminant classifier) to boost accuracy results. For DM-s (single trajectory) we use the same concatenation as for NN-s, to build a $|\mathbb{D}_r| \times |\mathbb{D}_r|$ training matrix of geometric similarity values, which is input into the classifier. For a classification query, we compute the row-vector of geometric distance values and then apply the classifier. Our DM-m (multi-trajectory) method considers the similarity values of the individual feature trajectories based on the $|\mathbb{D}_r| \times (|l(T)| \cdot |\mathbb{D}_r|)$ training matrix. Hence, the DM-m matrix contains $|l(T)|$ number of trajectories.

Our classifiers (from NN-s, to kNN-s, to DM-s, to DM-m) contain an increasing amount of information, respectively, with a cost of more distance computations and classifier overhead (e.g. Figure 5). We improve DM scalability for larger training sets by randomly choosing a small number of training sequences per class to reduce the number of $\mathbb{D}_r$ columns (see the LM experiment in Section 5.2).

Benchmark	Type	Jts.	Seq	Class	Sub	Frames/Seq	F./s	Seq/Cl.	Ex.
KinTrans	SL-Body	10	5,166	73	n/a	40 ± 13	30	71	*how*
LM	SL-Hands	54	17,312	60	25	71 ± 21	30	289	*dog*
NTU60	HA-Body	25	44,887	60	40	78 ± 34	30	748	*drop*
UCF	HA-Body	15	1,280	16	16	66 ± 34	30	80	*hop*
MHAD	HA-Body	15	660	11	12	$3{,}602 \pm 2{,}510$	480	60	*clap*

Table 1: Test data sets, showing type (Sign Language or Human Action), skeletal joints, sequences, classes, subjects, mean±SD of frames per sequence, frames per second in Hz, mean sequences per class, and class examples.

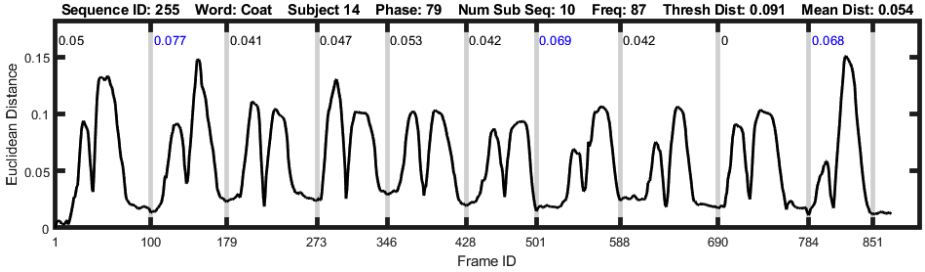


Fig. 4: Example LM raw sequence for the sign *coat* which is performed multiple times. The Fourier transform identifies 10 similar signs for *coat* which are cut into individual segment sequences. A center segment sequence is identified (minimum total distance to all other segment sequences), and distances are shown to the center.

5 Experimental Setup and Results

We implemented our methods and ran experiments on five skeleton-based data sets of actors that perform American Sign Language and Human Actions, each captured with different tracking technologies. The experiments investigate if (i) sub-skeleton feature mining discovers highly discriminative movement patterns, (ii) classification accuracy improves on state-of-the art methods for the benchmarks and on small training sets, and (iii) classification queries are answered quickly.

Experiments ran on a standard desktop computer (3.6GHz Core i7-7700 CPU, non-GPU) with MATLAB R2020a software on Windows 10, except for those using competitor ST-GCN (Yan et al., 2018) or the NTU60 data set, which ran on a high-performance computing environment with an NVIDIA V100 GPU and up to 32 CPUs (we attempted to run the ST-GCN author's code on the desktop computer but the learning did not converge within three days).

5.1 Data Sets

We use the skeleton joint-based Human Sign Language and Human Action data sets introduced in Section 1, see Table 1 for an overview.

The American sign language data set KinTrans is exclusively obtained from linedanceAI (2020), hence we are the first to publish results on it. Skeletal movement is captured with a Microsoft Azure Kinect DK by sign-literate individuals in a controlled setting. Some of the sequences contain noise and subject IDs are not

available due to privacy. Only 10 joints are captured: right/left hand/wrist/elbow/shoulder, neck and hip center. Recognizing signed words can be difficult since many signs use detailed finger movement, which is not captured with this data set. Requests for the KinTrans data set can be made directly with linedanceAI (2020).

The American sign language data set LM (Hernandez et al., 2020) captures detailed right/left finger, wrist, and elbow joint movement using the Leap Motion controller. Raw data sequences each contain one subject repeating a particular sign a variable number of times, and are 'cut' into segments that perform the single sign just once. Their work uses a manually created segmentation, which is not publicly available. Therefore, we use the following automated segmentation. Transform a raw sequence into a 1D signal by converting each frame into a $3|G|$ dimensional point which is used to compute the Euclidean distance to the first point of the sequence. Then use the mode in the Fourier power spectrum to obtain the number of segments that are aligned greedily to capture local minima of the 1D signal. See Figure 4 for an example raw sequence cut. Our automated cuts ended up with slightly more sequences (17,312) than the LM author's manual cut (16,890).

The NTU RGB+D human action data set (Shahroudy et al., 2016) captures subject movement using Microsoft Kinect v2 sensors. An action sequence is captured simultaneously by three camera views: one directly facing the subject, and the other two at 45 degree angles. There are 17 different camera setups (placed varying distances relative to the subject), single and mutual subject action classes, and subject ages range from 10 to 35 years old. In our setting we study single-subject sequences, therefore NTU RGB+D sequences with two or more detected subjects are removed, and we name the resulting data set NTU60.

We also test on popular human action data sets from the survey paper by Presti and La Cascia (2016). The University of Central Florida (UCF) data set (Ellis et al., 2013) captures movement with a Kinect sensor at 30Hz and estimates skeletal joints with OpenNI, and Berkeley's Multimodal Human Action Database (MHAD) captures motion at 480Hz with five different systems (Ofli et al., 2013).

To speed up distance computations, we use trajectory simplifications for LM, UCF, and MHAD. Simplification reduces the size of trajectories at the cost of a small distance error. We use the algorithm of Agarwal et al. (2005) as it has a guaranteed bounded error under the continuous Fréchet distance.

5.2 Feature Template Mining

Template mining experiment results for a subset of validation protocols and classifier methods are shown in Table 2. The feature mining training sets $\mathbb{D}_r$ are: randomly selected 20% trainers per class for KinTrans, randomly selected 15 trainers per class for LM and UCF, camera id's 2 & 3 for NTU60, and the first seven subjects for MHAD. Template mining for the NTU60 data set is slightly modified to enhance the generation of meaningful templates: after each mining iteration the three best canonical sub-skeletons c are appended to T, and iterations continue until $|T| \geq 15$. To speed up our implementation, we reduced the canonical sub-skeleton set C to contain fewer basic joint types, which are: all absolute joints, highly active joints relative to less active joints, highly active right joints relative to left counterparts, and leaf node joints relative to degree 2 vertices and up until

Mining Input				Feature Template Result						
Benchm.	$\mathbb{D}_r$	Method	$\|C\|$	Dist.	T	R	L	P	AJ	RJ	DM Class.
KinTrans	20%	NN-s	24	DTW	N	N	N	N	5	0	-
LM	15/class	NN-s	73	CF	**Y**	N	N	N	2	5	-
NTU60	cam. 2&3	DM-m	90	DTW	**Y**	**Y**	N	N	11	7	SVM
UCF	15/class	DM-s	34	DTW	N	N	N	N	4	1	Sub. Disc.
MHAD	7 subjects	NN-s	34	CF	**Y**	N	N	**Y**	0	3	-

Table 2: A subset of feature template mining results for various benchmarks. Mining input columns show: data set benchmark, training set protocol, our classifier method, and the number of canonical sub-skeletons. Feature template result columns show selected: distance similarity measure, normalizations (Translate, Rotate, Limb length, at-rest Pose), total numbers of Absolute and Relative Joints, and Distance Matrix classifier.

Trainers per class	ST-GCN	Ours[2]		
		NN-s	DM-s	DM-m
2	57.6	79.1	77.9	**80.6**
3	76.7	83.9	87.4	**89.6**
5%	81.2	90.1	**92.7**	92.2
10%	96.7	98.0	97.9	**98.4**
20%	99.2	99.3	99.5	**99.8**

Table 3: KinTrans American sign language data set classification accuracy results comparing our methods vs. ST-GCN on training sets of increasing sizes.

a branch. Nevertheless, the reduced set $|C|$ still generates highly discriminative templates, which will be shown in the classification accuracy experiments.

The template mining chose the DTW similarity measure for KinTrans, NTU60 and UCF data sets whereas the CF distance measure is chosen for LM and MHAD data sets. KinTrans is noisier than LM which may account for the different choice of similarity measures between these two data sets. For our DM classification methods, the SVM and subspace discriminant classifier was chosen for the NTU60 and UCF data set, respectively, which shows that our mining process can select different DM classifiers depending on the underlying data. Interestingly, normalizations are completely discarded on two data sets and none use all normalizations. Hence, our feature templates can negate the need for data normalizations.

The number of absolute and relative joints in selected feature templates are small compared to $|C|$, which results in lower dimensional trajectory features and faster distance computations. The most discriminative KinTrans joint is the right hand and for LM it is the right index finger, but interestingly, KinTrans neck and hip center joints were also included (which only have subtle movement), and the UCF and MHAD feature templates contain mostly different joints compared to each other.

The KinTrans template mining took 47 minutes for a specific normalization combination, which we consider reasonable on a standard desktop computer. As a guide, the respective ST-GCN training in Table 3 took more than 11 hours on the high-performance GPU computing environment.

Benchmark	Method	Paper	Accuracy
LM - 5-fold XSub	Kine.-LSTM	Hernandez et al. (2020)	91.1
	NN-s	Ours	83.2
	kNN-s ($k = 20$)	Ours	83.4
	DM-m	**Ours**	**93.8**

Table 4: LM American sign language data set classification accuracy results comparing our methods against Hernandez et al. (2020).

Benchmark	Method	Paper	Accuracy
NTU60 - Xsub	ST-GCN	Yan et al. (2018)	**78.7**[3]
	DM-m	Ours	71.1
NTU60 - Xview	ST-GCN	Yan et al. (2018)	**86.8**
	DM-m	Ours	83.4
UCF - 4-fold	Log. Reg.	Ellis et al. (2013)	95.9
	SVM	Ohn-Bar and Trivedi (2013)	97.1
	dHMM	Presti et al. (2015)	97.7
	SVM	Slama et al. (2015)	97.9
	kNN + Vote	Zanfir et al. (2013)	98.5
	SVM	Kerola et al. (2014)	98.8
	DP+kNN	Devanne et al. (2014b)	99.2
	ST-GCN	Yan et al. (2018)	**99.7**
	NN-s	Ours	97.7
	DM-m	Ours	99.5
MHAD - XSub	k-SVM	Ofli et al. (2013)	80.0
	ST-GCN	Yan et al. (2018)	89.8
	SVM	Ofli et al. (2014)	95.4
	CNN	Ijjina and Mohan (2014)	98.4
	MKL-SCM	Chaudhry et al. (2013b)	**100**
	NN-s	Ours	99.3
	DM-m	Ours	**100**

Table 5: Human action data set classification accuracy results, showing that our classifiers generalize well compared to other human action-based methods.

5.3 Classification Accuracy

We compare the accuracy of our simple classifiers against various state-of-the-art skeleton-based recognition approaches discussed in Section 1.1, using five diverse data sets from Table 1. Experiments with the new KinTrans data set gradually increase the amount of training data for each subsequent validation protocol, i.e. randomly choose training sets that contain either $2, 3, 5\%, 10\%$, or 20% sequences per class and use the remaining sequences as the respective testing sets. Experiments with previously released data sets use the same validation protocols as in the comparison papers: 5-fold cross-subject for LM, cross-subject and cross-camera view for NTU60, 4-fold cross for UCF, and first seven subjects for training for MHAD. In all DM-m classifier experiments, we use the full number of distance matrix columns, except for LM where we reduced it by randomly choosing two sequences per class/subject.

[2] kNN-s results are omitted as they did not outperform NN-s on the KinTrans data set.

[3] For the complete NTU RGB+D data set (single *and* multi-subject sequences), ST-GCN achieves accuracy's of 81.5% and 88.3% for Xsub and Xview, respectively, and the current best is Liu et al. (2020) who achieve accuracy's of 91.5% and 96.2%, respectively.

To investigate the effectiveness of our method on sign language data sets, we compare our accuracy results against: (i) publicly available code of the recent ST-GCN method using our KinTrans data set (see Table 3), and (ii) LM data set accuracy results of Hernandez et al. (2020) (see Table 4). The KinTrans experiment investigates the effectiveness of our method on varying training set sizes compared to ST-GCN. Training and testing data for each validation protocol experiment are fed into the ST-GCN application environment in NTU skeletal format, using all KinTrans joints without information loss. To achieve the best possible ST-GCN results, we tuned the learning parameters, replicated each sequence in the training sets exactly 40 times, and report the best accuracy result out of three independent training runs per experiment. In the lowest training information experiment (2 trainers per class) we improve over ST-GCN by more than 20%. As more training information is introduced, we still outperform ST-GCN, even with our simple NN-s classifier. For the LM sign language data set, our DM-m method improves on the accuracy results of Hernandez et al. (2020) which uses a more complex LSTM neural network.

Table 5 compares our accuracy results against three human action data sets. We achieve 99.5% and 100% accuracy on the UCF and MHAD data sets, respectively, which compares favorably to competitor results. For the NTU60 data set we achieve accuracy results that are similar to the performance of ST-GCN which uses a more complicated graph convolutional network. We are surprised that we achieve similar accuracy's on the NTU60 data set compared to ST-GCN, since our approach assumes that a subject's body movement is performed once, yet the NTU60 data set contains sequences with variable repetitious movement.

The experiments show that the accuracy performance of our simple methods generalizes well over different data domains (sign language / human action), training set sizes (small / large), and skeleton capture formats (coarse body / detailed hands).

5.4 Query Time

To investigate the query latency of our distance based classifiers, we run experiments on training sets of different sizes from the KinTrans data set with each of our methods. The testing sets always contain the remainder of the whole data set and the DM methods compute the full number of distance columns. For the metric distance measure CF, we use the approximate and exact Nearest Neighbor search structure from Gudmundsson et al. (2020). The results in Figure 5 show average wall-clock time per query, average number of trajectory distance computations per query, and overall classification accuracy for our NN-s (blue) and DM methods (black).

All NN-s and DM-s classifiers are under 200ms per query and 'NN-s CF-apx' is the fastest since it only uses distance approximations.

5.5 Classifier Interpretability

The use of trajectory similarity measures as feature predictor inputs make it easy to interpret how well a classifier will perform. Closer distances signify greater sim-

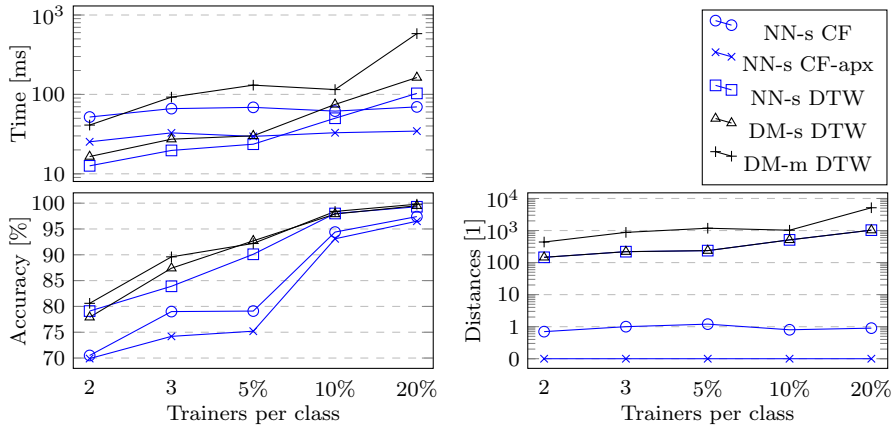


Fig. 5: Query latency of our classifiers for various training sets sizes using the KinTrans data set. Shown are average time per query (top), overall classification accuracy (bottom left), and average number of trajectory distance computations per query (bottom right).

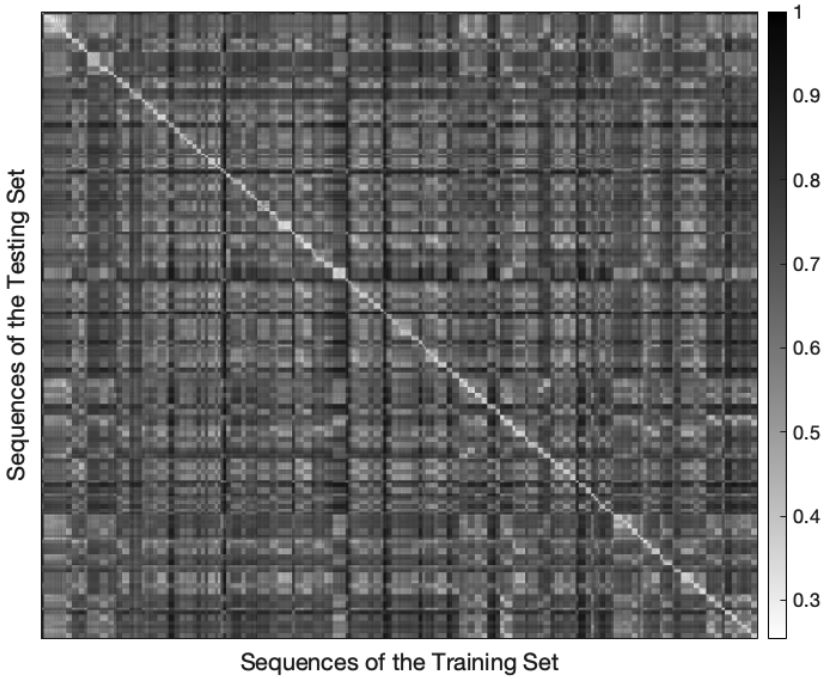


Fig. 6: The KinTrans 'NN-s DTW' 10% trainers per class heat map, which contains the distances (normalized to $[0, 1]$) of derived feature trajectories. There are $4,655$ testing data rows and 511 training data columns, which are both sorted by 73 class labels.

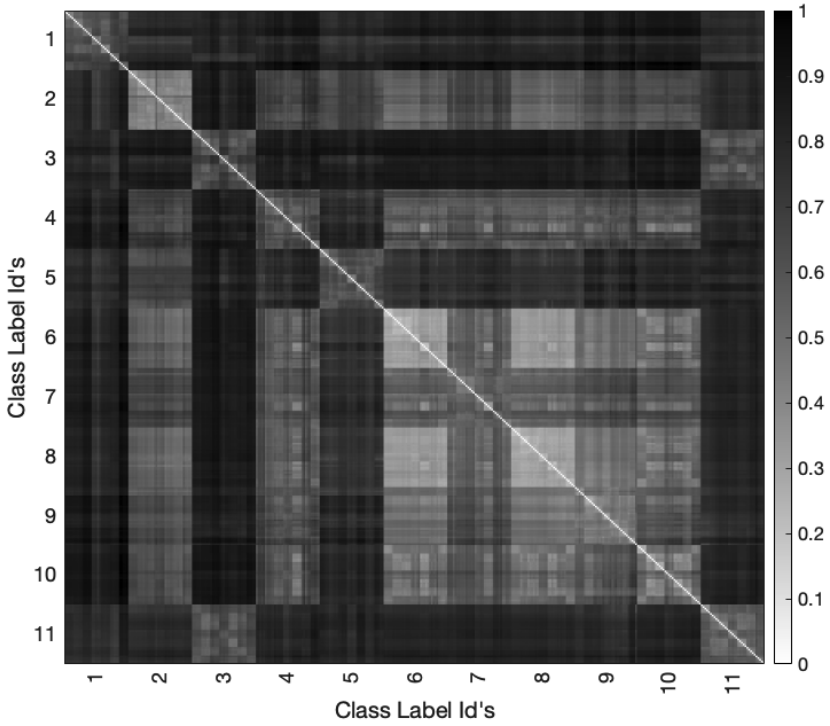


Fig. 7: Training distance matrix heat map for the MHAD DM-m experiment. The DM contains derived feature trajectory distances normalized to $[0, 1]$, has size 384×384, and rows/columns are sorted by class label then subject.

ilarity and further distances less similarity. The greater the separation of distances between classes, the better the classifier will perform. For example, if the Kin-Trans sequences for the class *doctor* all have small distances to each other and large distances to other classes, then the classifier will be able to easily distinguish and correctly classify this sign. Our feature template mining algorithm suggests a feature trajectory that will best separate classes.

We achieve classifier interpretability by visualizing trajectory distances in a heat map, where is it easy to compare distances amongst and between classes, thus providing intuition on the performance of the classifier.

5.5.1 Nearest Neighbor (NN)

The heat map in Figure 6 shows distances between the training and testing trajectories (using the NN-s trajectory feature template derived from the feature mining process). The correct class labels are along the diagonal, and as the figure shows, the diagonal tends to be a lighter color (represents a closer distance) compared to areas outside the diagonal. It is easy to determine which class is predicted for each test sequence simply by locating the smallest distance.

Classifier	$\|C\|$	Trainers per class				
		2	3	5%	10%	20%
NN-s	reduced	79.1	83.9	**90.1**	**98.0**	**99.3**
	full	**80.2**	**84.6**	89.0	96.7	**99.3**
DM-m	reduced	**80.6**	89.6	**92.2**	98.4	**99.8**
	full	**80.6**	**89.7**	87.7	**98.8**	99.5

Table 6: Comparison of KinTrans accuracy results using a template generated with ($|C|$ is reduced) and without ($|C|$ is full) the feature template mining speed up technique.

5.5.2 Distance Matrix (DM)

Figure 7 helps one visualize why the distance matrix classifier achieves good classification accuracy results. The larger squares along the diagonal represent individual class labels that are compared to themselves, and each of these are typically a lighter color (represents a closer distance) since different sequences in the same class tend to be close to one another. Importantly, when one compares a larger square in the diagonal with its corresponding row, the other larger squares in the row tend to be a darker color (represents a further distance) since they are different classes. It is this ability to correctly separate close and far sequences that results in good classification accuracy's. Underpinning this are feature templates and trajectory distance measures, which are used to compute the distance measures in the DM.

5.6 Additional Experiments

We show additional experimental results on template mining speedups, DM scalability, and a mining discovery that identifies a surprisingly information-rich joint.

5.6.1 Feature Template Speedup Effectiveness

Previous experiments speedup the feature template mining process by *reducing* the canonical sub-skeleton set size $|C|$, which in turn reduces the work required in each iteration. This experiment tests if performing feature mining on the *full* $C = |G|^2$ absolute and relative singleton joints materially impacts accuracy results. Table 6 shows KinTrans data set accuracy results using templates generated from both reduced and full sets C during the template mining process. Accuracy's are not materially impacted whether using full or reduced sets C, indicating that the mining speedup technique is both fast and effective.

Analogue *full* set size $|C|$ experiments were also run on the UCF and MHAD data sets, and the accuracy results are 99.5% and 100%, respectively, which are the same compared to those in Table 5 that use the speedup technique.

5.6.2 Scalability of DM Query Time

Table 7 shows the impact of reducing DM-m distance matrix feature columns on the KinTrans 20% trainers per class protocol. For DM-m(c) just one trainer per class is randomly selected for DM feature columns, and compared to DM-m(a) which uses all trainers per class for feature columns, classification accuracy only

Distance Matrix	DM Feature Col. Selection	Num. Feature Columns	Avg. Query Time (ms)	Accuracy	Avg. Distance Computations
DM-m(a)	all $\mathbb{D}_r$/class	1,208	582	99.8	5,140
DM-m(b)	2 $\mathbb{D}_r$/class	146	126	99.8	730
DM-m(c)	1 $\mathbb{D}_r$/class	73	49	99.5	365

Table 7: Impact on runtime and accuracy when reducing distance matrix feature columns with the KinTrans 20% trainers per class protocol. Columns show DM feaure column selection criteria, number of distance matrix feature columns, average query time, overall classification accuracy, and average number of trajectory distance computations per query. DM-m(a) contains all feature columns. DM-m(b) and DM-m(c) randomly select 2 or 1 trainers per class from $\mathbb{D}_r$ for the feature columns, respectively.

drops 0.3% to 99.5%, but query runtimes are 11.8 times faster. Clearly, the DM method can scale to larger data sets by reducing feature columns, which can result in similar accuracy.

5.6.3 Sign Language Hip Movement

Our mining exploration algorithm shows that surprisingly rich information is contained within subtle center-hip joint movement on the KinTrans data set. When T is set to *only the center-hip joint* and a 20% trainers per class protocol is used, the NN-s and DM-m classifiers achieve 91.8% and 95.1 accuracy results, respectively. This is a non-intuitive and surprising result since conventional thinking emphasizes hand movement over other joints.

6 Conclusion

Our work on skeleton-based sign language recognition introduces the sub-skeleton feature space and studies it using speed-invariant similarity measures. Our method automatically discovers absolute and relative movement patterns, which enables highly accurate Nearest Neighbor classification, with acceptable latency, on training data of varying domains, skeleton formats, and sizes. Our distance based classifiers are interpretable and train on basic computing hardware, which make them particularly interesting for data sets that change frequently.

In future work we seek to extend our methods to recognition problems that do not assume segmented input data. Popular skeleton-based benchmark data has variable repetitions of the action in each recording, e.g. NTU120 and Kinetics (Kay et al., 2017; Liu et al., 2019), or contains continuous streams of different sign language words per recording (Pu et al., 2018; Wei et al., 2019). To accommodate such recognition problems, we seek to extend our automated, Fourier based, segmentation method or investigate sub-trajectory clustering approaches. Moreover, we are interested in the effectiveness of other k-Nearest Neighbor weighting schemes, potentially taking trajectory clusters into account. Eventually we seek to further improve our already acceptable query latency by extending known speed up techniques for CF and DTW to *higher dimensional* trajectory input.

Acknowledgements The authors acknowledge the technical assistance provided by the Sydney Informatics Hub, a Core Research Facility of the University of Sydney. J.G. is funded by the Australian Government through the Australian Research Council DP180102870.

References

Adaloglou N, Chatzis T, Papastratis I, Stergioulas A, Papadopoulos GT, Zacharopoulou V, Xydopoulos GJ, Atzakas K, Papazachariou D, Daras P (2020) A comprehensive study on sign language recognition methods. arXiv preprint arXiv:200712530

Agarwal PK, Har-Peled S, Mustafa NH, Wang Y (2005) Near-linear time approximation algorithms for curve simplification. Algorithmica 42(3-4):203–219

Alt H, Godau M (1995) Computing the Fréchet distance between two polygonal curves. International Journal of Computational Geometry & Applications 5(1–2):75–91

Chaudhry R, Ofli F, Kurillo G, Bajcsy R, Vidal R (2013a) Bio-inspired dynamic 3D discriminative skeletal features for human action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp 471–478

Chaudhry R, Ofli F, Kurillo G, Bajcsy R, Vidal R (2013b) Bio-inspired dynamic 3d discriminative skeletal features for human action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp 471–478

Devanne M, Wannous H, Berretti S, Pala P, Daoudi M, Del Bimbo A (2014a) 3-d human action recognition by shape analysis of motion trajectories on Riemannian manifold. IEEE Transactions on Cybernetics 45(7):1340–1352

Devanne M, Wannous H, Berretti S, Pala P, Daoudi M, Del Bimbo A (2014b) 3-d human action recognition by shape analysis of motion trajectories on Riemannian manifold. IEEE transactions on cybernetics 45(7):1340–1352

Eiter T, Mannila H (1994) Computing discrete Fréchet distance. Tech. Report CD-TR 94/64, Christian Doppler Laboratory for Expert Systems, TU Vienna, Austria

Ellis C, Masood S, Tappen M, LaViola J, Sukthankar R (2013) Exploring the trade-off between accuracy and observational latency in action recognition. International Journal of Computer Vision 101(3):420–436

Gao X, Hu W, Tang J, Liu J, Guo Z (2019) Optimized skeleton-based action recognition via sparsified graph regression. In: Proceedings of the 27th ACM International Conference on Multimedia, pp 601–610

Gudmundsson J, Horton M, Pfeifer J, Seybold MP (2020) A practical index structure supporting Fréchet proximity queries among trajectories. arXiv preprint arXiv:200513773

Hernandez V, Suzuki T, Venture G (2020) Convolutional and recurrent neural network for human activity recognition: Application on american sign language. PloS one 15(2):e0228869

Ijjina EP, Mohan CK (2014) Human action recognition based on mocap information using convolution neural networks. In: 2014 13th International Conference on Machine Learning and Applications, IEEE, pp 159–164

Kay W, Carreira J, Simonyan K, Zhang B, Hillier C, Vijayanarasimhan S, Viola F, Green T, Back T, Natsev P, et al. (2017) The kinetics human action video dataset. arXiv preprint arXiv:170506950

Keogh E, Ratanamahatana CA (2005) Exact indexing of dynamic time warping. Knowledge and Information Systems 7(3):358–386

Kerola T, Inoue N, Shinoda K (2014) Spectral graph skeletons for 3d action recognition. In: Asian conference on computer vision, Springer, pp 417–432

Kushalnagar P, Moreland CJ, Simons A, Holcomb T (2018) Communication barrier in family linked to increased risks for food insecurity among deaf people who use american sign language. Public Health Nutrition 21(5):912–916

Li B, Li X, Zhang Z, Wu F (2019a) Spatio-temporal graph routing for skeleton-based action recognition. In: Proceedings of the 33rd AAAI Conference on Artificial Intelligence, vol 33, pp 8561–8568

Li C, Zhong Q, Xie D, Pu S (2018) Co-occurrence feature learning from skeleton data for action recognition and detection with hierarchical aggregation. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence, pp 786–792

Li M, Chen S, Chen X, Zhang Y, Wang Y, Tian Q (2019b) Actional-structural graph convolutional networks for skeleton-based action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 3595–3603

linedanceAI (2020) The KinTrans project. https://www.kintrans.com

Liu J, Shahroudy A, Perez ML, Wang G, Duan LY, Chichung AK (2019) NTU RGB+D 120: a large-scale benchmark for 3d human activity understanding. IEEE Transactions on Pattern Analysis and Machine Intelligence

Liu Z, Zhang H, Chen Z, Wang Z, Ouyang W (2020) Disentangling and unifying graph convolutions for skeleton-based action recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 143–152

Mantecón T, del Blanco CR, Jaureguizar F, García N (2019) A real-time gesture recognition system using near-infrared imagery. PloS one 14(10):e0223320

Munich ME, Perona P (1999) Continuous dynamic time warping for translation-invariant curve alignment with applications to signature verification. In: Proceedings of the 7h IEEE International Conference on Computer Vision, vol 1, pp 108–115

Ofli F, Chaudhry R, Kurillo G, Vidal R, Bajcsy R (2013) Berkeley mhad: A comprehensive multimodal human action database. In: 2013 IEEE Workshop on Applications of Computer Vision, IEEE, pp 53–60

Ofli F, Chaudhry R, Kurillo G, Vidal R, Bajcsy R (2014) Sequence of the most informative joints (smij): A new representation for human skeletal action recognition. Journal of Visual Communication and Image Representation 25(1):24–38

Ohn-Bar E, Trivedi M (2013) Joint angles similarities and hog2 for action recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops, pp 465–470

Peng W, Hong X, Chen H, Zhao G (2020) Learning graph convolutional network for skeleton-based human action recognition by neural searching. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol 34, pp 2669–2676

Presti LL, La Cascia M (2016) 3D skeleton-based human action classification: A survey. Pattern Recognition 53:130–147

Presti LL, La Cascia M, Sclaroff S, Camps O (2015) Hankelet-based dynamical systems modeling for 3d action recognition. Image and Vision Computing 44:29–43

Pu J, Zhou W, Li H (2018) Dilated convolutional network with iterative optimization for continuous sign language recognition. In: Proceedings of the 27th

International Joint Conference on Artificial Intelligence, pp 885–891
Rastgoo R, Kiani K, Escalera S (2020) Sign language recognition: A deep survey. Expert Systems with Applications p 113794
Shahroudy A, Liu J, Ng TT, Wang G (2016) Ntu rgb+ d: A large scale dataset for 3D human activity analysis. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1010–1019
Shi L, Zhang Y, Cheng J, Lu H (2019a) Skeleton-based action recognition with directed graph neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 7912–7921
Shi L, Zhang Y, Cheng J, Lu H (2019b) Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 12026–12035
Slama R, Wannous H, Daoudi M, Srivastava A (2015) Accurate 3d action recognition using learning on the Grassmann manifold. Pattern Recognition 48(2):556–567
Wei C, Zhou W, Pu J, Li H (2019) Deep grammatical multi-classifier for continuous sign language recognition. In: IEEE 5th International Conference on Multimedia Big Data, IEEE, pp 435–442
Wen YH, Gao L, Fu H, Zhang FL, Xia S (2019) Graph CNNs with motif and variable temporal block for skeleton-based action recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol 33, pp 8989–8996
Yan S, Xiong Y, Lin D (2018) Spatial temporal graph convolutional networks for skeleton-based action recognition. In: Proceedings of the 32nd AAAI Conference on Artificial Intelligence, pp 7444–7452
Yao A, Gall J, Fanelli G, Gool LV (2011) Does human action recognition benefit from pose estimation? In: Proceedings of the 22nd British Machine Vision Conference, pp 1–11
Young A, Oram R, Napier J (2019) Hearing people perceiving deaf people through sign language interpreters at work: on the loss of self through interpreted communication. Journal of Applied Communication Research 47(1):90–110
Zanfir M, Leordeanu M, Sminchisescu C (2013) The moving pose: An efficient 3d kinematics descriptor for low-latency action recognition and detection. In: Proceedings of the IEEE international conference on computer vision, pp 2752–2759

A Additional Data Set Statistics

We describe additional KinTrans, LM, and NTU60 data set statistics.

A.1 KinTrans

The data set has varying numbers of sequences per class which can make it more difficult for some classifiers to achieve high accuracy, since some classes are over or underrepresented in the training set. The sequences per class mean is 71, min 18, max 200, and standard deviation 44. Table 8 shows the number of sequences per class label.

A.2 LM Segmentation Results

As discussed in the main paper, we implemented a simple, automated segmentation process to cut LM sequences into smaller sequences that each perform a single sign once. The sequences per class mean is 288, min 242, max 347, and standard deviation 22. Table 9 shows the derived number of sequences per class label.

A.3 NTU60 Single Subject Results

Per the main paper, we extract only single-subject sequences from the NTU RGB+D 60-class human action data set, and insert them into a data set called NTU60. The NTU60 sequences per class mean is 748, min 16, max 919, and standard deviation 338. Table 10 shows the derived number of sequences per class label.

B Normalization Methods

We briefly discuss the other three sequence normalization methods that our automated mining is allowed to choose for a data set.

A subject's joints within a sequence can be *translated* to the origin such that a key joint (e.g. neck) is at the origin for the first frame (or every frame if one chooses).

A subject's joints can be *rotated* about the x, y, or z axis relative to the first frame or every frame via rotation matrices and matrix multiplication. Examples include rotating the subject to face the camera with their hip directly beneath the neck or having the back of the hands face the camera with the wrist directly beneath the knuckles. This can be toggled such that, for example, the subject faces the camera for the first frame or every frame. The rotation and translation normalizations can be useful when various subjects start from disparate coordinates or have different azimuth and elevations.

A subject's limbs can be normalized to *standardized lengths* using linear algebra: for each frame start at a center skeletal joint and work towards the extremities, modifying limb vector magnitudes to the standard length, while preserving the original limb vector direction. Standard limb lengths are pre-computed by analyzing subjects in a data set and obtaining mean (standard) limb lengths. This normalization can help when there are substantial body size differences between subjects.

C ST-GCN Experiment Parameter Details

The following parameters were used to obtain the best accuracy in ST-GCN experiments on the high-performance computing environment: batch_size: 20, epochs: 80, base learning rate: 0.1, weight_decay: 0.0001, optimizer: SGD, nesterov: True. The ST-GCN training runtimes on the KinTrans data set for the 2, 3, 5%, 10%, and 20% trainers per class experiments were 106, 156, 170, 358, and 713 minutes, respectively (times shown are the fastest of three independent runs for each experiment).

Class Label	Num.	Class Label	Num.	Class Label	Num
America	23	*great*	59	*please*	90
I	200	*have*	110	*request*	90
ID	90	*head*	20	*setup*	90
allowed	20	*help*	120	*sick*	20
and	190	*here*	155	*small bill*	90
baggage	20	*hi*	90	*stay*	20
bank	90	*high*	20	*system*	18
bank statement	90	*hotel*	20	*taxi*	20
can	20	*how*	120	*teller*	90
check	61	*how much*	90	*temperature*	20
cheque	90	*hurts*	21	*thank you*	170
claim	20	*insurance card*	20	*they*	20
company account	89	*interest rate*	90	*this*	20
credit card	90	*international*	90	*to*	30
customer service	90	*manager*	90	*today*	90
days	20	*many*	19	*verfication*	90
deposit	90	*medical*	61	*visit*	60
diabetes	20	*money*	90	*want*	90
dizzy	21	*monthly fee*	90	*what*	90
do	30	*my*	119	*where*	110
doctor	20	*need*	90	*with*	20
email	90	*new account*	90	*withdraw*	90
family	60	*no*	90	*yes*	150
feel pain	20	*now*	90		
fund transfer	90	*pay balance*	90		

Table 8: KinTrans data set class labels and number of sequences per class. The classes are common words used in the airport, bank, and doctor's office.

Class Label	Num.	Class Label	Num.	Class Label	Num
0	309	*Coat*	271	*More*	320
1	319	*Cold*	347	*Orange*	299
2	307	*Come*	306	*Pig*	266
3	291	*Cost*	298	*Please*	276
4	303	*Cry*	281	*Red*	283
5	314	*Dad*	285	*Shoes*	315
6	317	*Deaf*	243	*Small*	279
7	298	*Dog*	280	*Socks*	270
8	292	*Drink*	279	*Stop*	278
9	298	*Egg*	309	*Store*	283
10	242	*Finish*	326	*Thanks*	273
Big	277	*Go*	319	*Warm*	302
Blue	242	*Good*	315	*Water*	283
Brush	296	*Green*	267	*What*	296
Bug	257	*Happy*	274	*When*	266
Candy	282	*Hot*	292	*Where*	293
CarDrive	264	*Hungry*	295	*Why*	266
Cat	267	*Hurt*	293	*With*	308
Cereal	264	*Milk*	308	*Work*	293
Clothes	292	*Mom*	300	*Yellow*	244

Table 9: LM data set class labels and number of sequences per class.

Class Label	Num.	Class Label	Num.
brushing hair	916	*reach into pocket*	907
brushing teeth	912	*reading*	907
check time (from watch)	906	*rub two hands together*	908
cheer up	915	*salute*	903
clapping	907	*shake head*	903
cross hands in front (say stop)	891	*sitting down*	909
drink water	908	*sneeze-cough*	902
drop	914	*staggering*	896
eat meal-snack	913	*standing up (from sitting position)*	919
falling	894	*take off a hat-cap*	915
giving something to other person	29	*take off a shoe*	915
hand waving	917	*take off glasses*	913
handshaking	20	*take off jacket*	919
hopping (one foot jumping)	905	*taking a selfie*	906
hugging other person	17	*tear up paper*	904
jump up	906	*throw*	916
kicking other person	35	*touch back (backache)*	895
kicking something	907	*touch chest (stomach-heart pain)*	894
make a phone call-answer phone	906	*touch head (headache)*	894
nausea or vomiting condition	894	*touch neck (neckache)*	895
nod head/bow	903	*touch other persons pocket*	28
pat on back of other person	20	*typing on a keyboard*	912
pickup	915	*use a fan (hand/paper)-feel warm*	895
playing with phone-tablet	906	*walking apart from each other*	37
point finger at the other person	16	*walking towards each other*	183
pointing to something with finger	903	*wear a shoe*	913
punching-slapping other person	42	*wear jacket*	918
pushing other person	28	*wear on glasses*	910
put on a hat-cap	913	*wipe face*	904
put the palms together	903	*writing*	906

Table 10: NTU60 data set class labels and number of sequences per class.