Proceedings of the 9th Workshop on the Representation and Processing of Sign Languages, pages 189–196
Language Resources and Evaluation Conference (LREC 2020), Marseille, 11–16 May 2020

Unsupervised Term Discovery for Continuous Sign Language

Korhan Polat, Murat Saraçlar
Electrical & Electronics Engineering Department
Bogazici University, Istanbul, Turkey
{korhan.polat, murat.saraclar}@boun.edu.tr

Abstract

Most of the sign language recognition (SLR) systems rely on supervision for training and available annotated sign language resources are scarce due to the difficulties of manual labeling. Unsupervised discovery of lexical units would facilitate the annotation process and thus lead to better SLR systems. Inspired by the unsupervised spoken term discovery in speech processing field, we investigate whether a similar approach can be applied in sign language to discover repeating lexical units. We adapt an algorithm that is designed for spoken term discovery by using hand shape and pose features instead of speech features. The experiments are run on a large scale continuous sign corpus and the performance is evaluated using gloss level annotations. This work introduces a new task for sign language processing that has not been addressed before.

Keywords: unsupervised learning, term discovery, sign language recognition

1. Introduction

Despite the recent advancements in computer vision and deep learning, automatic sign language recognition (ASLR) still remains as a challenging problem and has the potential for improvement. One of the many reasons that hinders development of ASLR systems is the lack of large scale annotated corpora for training supervised deep learning models. Even though there exist plenty of sign language recordings, most of them are not annotated because manual annotation is a labor intensive task which requires linguistic expertise. This brings the need for a language independent, unsupervised learning procedure in order to handle the vast amount resources for sign languages. With this target set, we explore how an unsupervised learning technique in speech processing can be applied in sign language domain to identify lexical structures when there is no labeled data available.

Unsupervised learning has been an active research area in spoken language processing since the majority of the world's languages are low resource in the sense that there are not adequate resources for training models. The extreme case for unsupervised learning, in which there is neither labeled training data nor knowledge of linguistic structure, is referred as the *zero resource* setting (Versteegh et al., 2015; Dunbar et al., 2017). Zero resource speech processing research focuses on two main topics; subword modelling and spoken term discovery. Subword modelling aims to learn speech representations that capture linguistic structures and that are robust for speech recognition. On the other hand, the aim of unsupervised term discovery (UTD) is to find repeating patterns (phonological, lexical or phrasal units) given only the speech features extracted from raw acoustic signals, without any supervision. The output is the hypothesized word types together with token time boundaries for the unknown language. The pioneering work in unsupervised spoken term discovery by Park and Glass (2008) introduces the segmental dynamic time warping (sDTW) algorithm to discover similar segments between two vector time series. Discovered segments are then clustered where each cluster represents the hypothesized word type in that unknown language. Follow up work of Jansen and Durme (2011) proposes an algorithm that reduces the time complexity by applying efficient image processing and randomized bit hashing techniques. Since then, various approaches to this problem have been proposed in Zero Resource Speech Challenges (Versteegh et al., 2015; Dunbar et al., 2017), which are not in the scope of this work.

Here, we define a new task for processing of sign language videos. Unsupervised discovery of sign terms is the task of discovering and segmenting sign glosses automatically, without using any supervisory information (additional modalities, lexical knowledge etc.). This task would provide numerous benefits to sign language and action recognition fields. It can be used as a segmentation tool that proposes gloss time boundaries and it can speed up manual annotation process. Moreover, clustered segments can be treated as weak labels and supervised models can be trained based on these labels. As an initial exploration of this task, we use the method of Jansen and Durme (2011) since it has been used as the baseline method for the ZR Challenges (Versteegh et al., 2015; Dunbar et al., 2017) and its software implementation is publicly available[1]. We adapt this algorithm to run with sign language videos by feeding visual features instead of speech features. Visual features include hand shape and pose features obtained from pre-trained models. We further augment the pose features by training an autoencoder, which is also an unsupervised learning method. The discovery algorithm is run with these features on a large scale continuous sign dataset and results are evaluated using a set of metrics tailored for this task.

In the field of unsupervised sign language recognition, a similar work to ours is presented by Nayak et al. (2012). They propose a Bayesian method to find the most oc-

[1] github.com/arenjansen/ZRTools

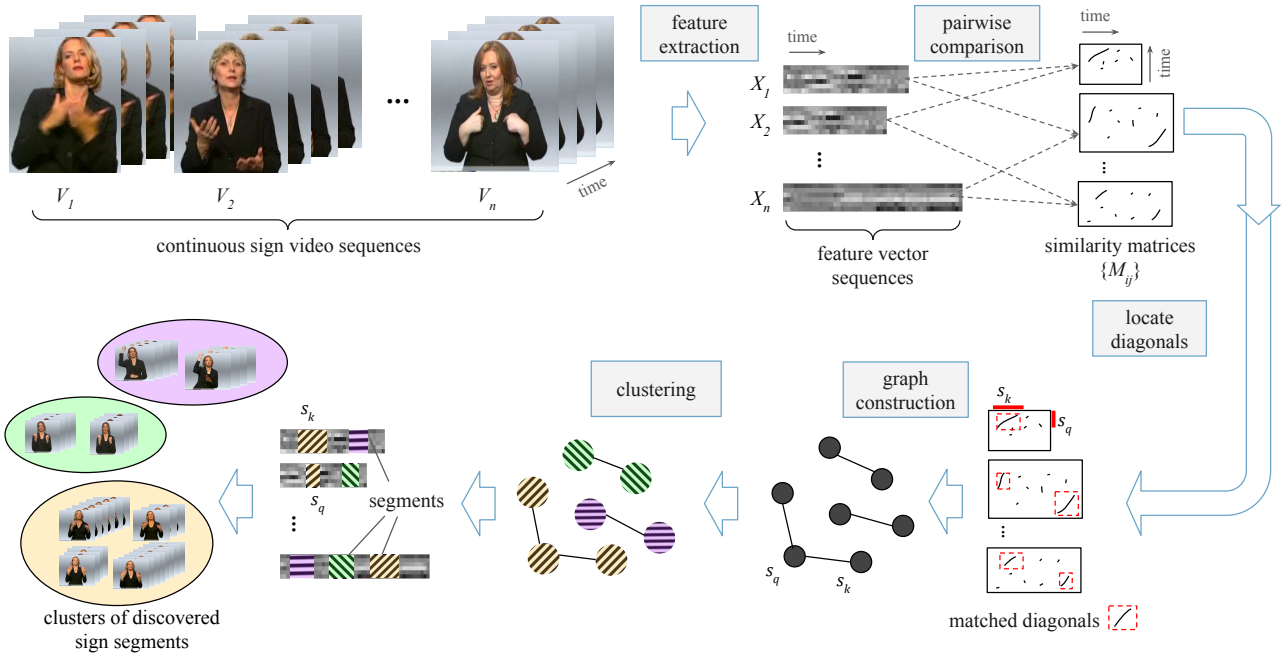

Figure 1: Flow of the unsupervised term discovery algorithm.

curring signs from continuous sign sequences given the information that how many signs are common among these sequences. They report the system performance based on localization of most common signs in 155 sentences from American Sign Language. Even though they do not use labels, their work differs from ours since they use the knowledge of how many sign segments should be discovered from each sequence beforehand. However in UTD, we do not know whether any two sequences share a common sign or not. From the perspective of sub-unit representation, Theodorakis et al. (2014) introduce a sign language phonetic modelling framework in which signs are segmented into dynamic and static sub-units, in an unsupervised fashion. Evaluation of subunit modelling is carried out with regard to ASLR performance on isolated sign datasets. Kelly et al. (2011) and Pfister et al. (2013) use multiple instance learning for extracting isolated signs but utilize text as weak supervision. Our method differs from these in the sense that it does not rely on any supervisory information and analyzes large scale data without knowing whether there are matching segments between pairs of sequences. Our work has the potential for aiding the annotation process for large scale sign dataset when no weak labels (speech, subtitle) are available. It might also be helpful for discovering glosses when little is known about a new sign language.

The rest of the paper is organized as follows. In Section 2 the term discovery algorithm is explained. Experimental setup for sign term discovery is given in Section 3. Implementation details and the results are discussed in Section 4.

2. Term Discovery Algorithm

Our work is based on the algorithm of Jansen and Durme (2011), which is composed of discovery and clustering steps (Figure 1). The discovery step yields pairs of matching segments that are similar to each other. Then an adjacency graph is constructed from the pairs of matching segments and similar segments are clustered together. Details of these steps are explained in the following sections.

2.1. Discovery Step

Starting with a set of feature vectors $\mathcal{X}$, where each vector sequence $X_i \in \mathbb{R}^{d \times T_i}$ is extracted from a continuous signing sequence V_i, the aim is to find pairs of similar sub-sequences by comparing pairs of sequences. For a given pair of feature sequences (X_i, X_j), a pairwise similarity matrix $M_{ij} \in \mathbb{R}^{T_i \times T_j}$ is computed using cosine similarity. If the same word occurs in both sequences, the cosine similarities of the feature vectors corresponding to that time intervals would be high and appear as diagonal lines on the similarity matrix (Figure 1). These regions can be detected by DTW, which is an algorithm that aligns two time series by minimizing alignment costs. By running these steps for all input pairs, we end up with pairs of matching sub-sequences that have low alignment costs.

Both similarity matrix computation and DTW search steps have the time complexity of $O(n^2)$, making it difficult to scale up to large datasets. To combat this limitation, the algorithm of Jansen and Durme (2011) introduces two stages of approximation to perform these steps in $O(n \log n)$ time complexity. These two methods are summarized below.

2.1.1. Approximation of Cosine Similarity Matrix

The first method speeds up the similarity matrix computation by using locality sensitive hashing technique to approximate the pairwise cosine distance computation. At the beginning, feature vectors are normalized such that each dimension has 0 mean and 1 variance across

time. Then applying a random transformation matrix, each vector is projected onto a new 64 dimensional feature space. Projected feature vectors are thresholded at zero to form bit signatures of size 64 (eg. '011...01'). This operation preserves the distance in the original space and enables the approximation of cosine distance by computing the Hamming distance between two bit signatures. Thus if a group of bit signatures are sorted, the signatures that fall nearby would have low Hamming distance between each other. Then the sorted list is linearly swept and for each signature, Hamming distances are computed only between the nearby signatures. This way M is populated sparsely and efficiently without spending time on comparisons that would result in low similarity.

2.1.2. Locating the Diagonal Segments

In the second stage the approximate similarity matrix M is treated as an image. If the same word occurs in both sequences, it would appear as a diagonal line segment on M (Figure 1). These diagonal lines can be located by efficient image processing techniques. First, a diagonal μ-percentile filter of length L is applied which allows the diagonal segments to pass. Then diagonal lines are located with Hough transform which is an algorithm to find lines in an image. Next, segmental DTW search is performed only in the vicinity of the located diagonals instead of exhaustive search over M. The segmental DTW search is terminated when the alignment cost exceeds a threshold C. Using these alignment costs, similarity scores are assigned to matching pairs.

2.2. Clustering Step

As a first step, the matching pairs that have a similarity score less than S_{dtw} are discarded. Using the remaining segments, a graph is formed such that each node corresponds to a discovered segment and the vertices are assigned between the pairs of matching segments. The graph is *de-duplicated* by eliminating overlapping segments. Finally, connected components are found as individual clusters. These clusters are the hypothesized word types and segments are the word tokens.

3. Experimental Setup for Sign Language

The UTD algorithm described above (Jansen and Durme, 2011) is applied to RWTH-PHOENIX-Weather 2014 (Koller et al., 2015) continuous sign dataset by feeding visual features instead of speech features. We implemented some of the metrics that are used in the ZR Challenges (Versteegh et al., 2015; Dunbar et al., 2017) to measure the performance of the UTD algorithm.

3.1. Corpus

A continuous sign dataset which includes gloss annotations with corresponding time boundaries is needed to evaluate this algorithm. In order to satisfy these requirements, we opted to use the RWTH-PHOENIX-Weather 2014 corpus (Koller et al., 2015) which consists of German Sign Language (DGS) interpretation of daily weather forecast on public television, signed by 9 different signers in total. Each video clip is a sign sentence and the glosses for each sentence are annotated manually. However, these manual annotations do not specify the time boundaries of glosses. Follow up work of Koller et al. (2017) uses a Hidden Markov Model (HMM) based forced alignment procedure to find the gloss time boundaries automatically. We used the training set of Multi-Signer setup, since it is the only subset that contains these time boundaries for gloss annotations. We take these automatic annotations as the ground truth labels and use them only for evaluating the performance of the algorithm; the labels are not part of the UTD algorithm.

Working on this corpus leads to several advantages for our task. One advantage is that there are no significant illumination, angle or scale variations. All videos are recorded in the studio with 25fps and signers face directly to a stationary camera. Another advantage is the sign vocabulary being limited to weather related terms only. This results in limited search space for the algorithm and also less variation in signing of a word type. However one drawback is the low resolution (210x260 pixels) of the recordings, which introduces noise to feature extraction process.

Signer ID	Duration (min)	# Sentences	# Discoverable	
			Types	Tokens
1	130	1475	462	15928
5	125	1296	445	13795
4	82	836	345	7642
8	64	704	327	7242
7	60	646	390	7493
3	45	470	260	5227
9	17	165	203	1763
2	6	49	111	576
6	3	30	69	307
Total	533	5671	803	60927

Table 1: Corpus statistics for training set of RWTH-PHOENIX-Weather Multi Signer dataset. A sign type is discoverable if it occurs two or more times.

Corpus statistics for different signer subsets are given in Table 1. We partitioned the Multi-Signer training set further into three subsets, rather arbitrarily. First subset (signer IDs 3,7) constitutes 20% of the training set and it is used as a development set for tuning the parameters of the UTD algorithm. Another subset (signer IDs 2,4,5,6,9) that covers 45% of the training set is used for training the auto-encoder model. The rest (signer IDs 1, 8) are used as the unseen test set for performance evaluation. Note that we used the labels of the development set (IDs 3,7) only for parameter tuning and model validation.

3.2. Features

Convolutional neural nets (CNN) have shown great success in the recent years. Last layers of CNN's capture high

level visual information and their activations can be used as image features. We considered two different feature extraction methods; activations of a pre-trained hand shape classifier CNN and a pose estimator. Moreover, an autoencoder is trained with pose features and the embeddings from the encoder part are used as the third set of features. Our aim is to explore how different types of features can be used for the UTD task, rather than to make a comparison of their performance.

3.2.1. 2D Pose Estimates

OpenPose pose estimator (Cao et al., 2017) is used for obtaining body and hand keypoint coordinates. We used the 8 upper body joint locations out of 25 body joints in addition to 21 keypoints for each hand. Each location is identified with (x, y) coordinates thus we end up with 100 dimensional feature vectors for each video frame. Normalization is done by taking the neck and wrist locations as origins for body and hands respectively and dividing the features by shoulder lengths.

3.2.2. Pre-Trained Hand Shape Classifier

We use the DeepHand convolutional network (Koller et al., 2016) which is a publicly available pre-trained model. Given right hand patches as input, it is able to classify 60 hand shapes based on SignWriting (Sutton, 2000) notation. The final layer is a softmax layer which normalizes the activations to class-conditional posterior probabilities. For our purposes, using the pre-softmax activations as feature vectors is more applicable since distance the UTD algorithm approximates cosine distance between feature vectors. We further applied PCA for dimensionality reduction. We used the wrist coordinates estimated by OpenPose (Cao et al., 2017) to crop right hand patches.

3.2.3. Autoencoder

Autoencoders are encoder-decoder type neural networks which are trained to predict its input. The challenge comes from the existence of a bottleneck layer in the middle, whose dimension is lower than the input dimension. Therefore the network is forced to learn a more compact representation of its inputs, such that from this representation it should be able to reconstruct the original input. They can be formed in varying depths by stacking layers. Here, we make use of this architecture for the purpose of learning better feature representations as well as the additional benefit of achieving non-linear dimensionality reduction. Specifically, we used this network to augment the pose features, aiming for a feature representation that is more appropriate for computing cosine similarity. An autoencoder is trained using the 100 dimensional OpenPose features from the training set. Then using this trained network, the bottleneck features (encoder outputs) are extracted for the development and test sets.

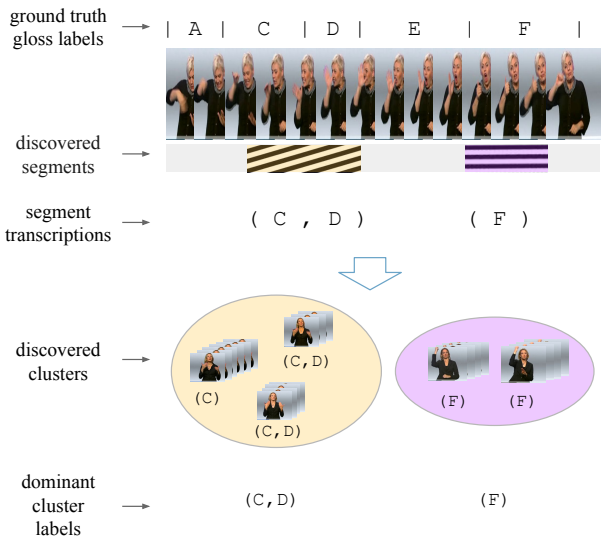

Figure 2: During the evaluation stage, the discovered segments are assigned transcriptions based on the overlapping gloss annotations. Then the metrics are calculated using the transcriptions for each segment. (Gloss labels and images are representative.)

3.3. Evaluation Metrics

Evaluation metrics that we use are inspired by the ZR Challenge (Versteegh et al., 2015; Ludusan et al., 2014). The evaluation for spoken language UTD is based on phoneme level transcriptions however the gloss labels provided by the dataset are not equivalent to phoneme level units. Therefore we modified the evaluation scheme to be compatible with the gloss level annotations.

For the dataset that we use, the labels with time boundaries are aligned using an HMM based model. Koller et al. (2017) used three state HMM's, which model a sign gloss with three sub-units. Therefore the labels they provided indicate the sub-unit indices but we ignore the sub-unit indices and consider only the whole gloss (e.g. $\texttt{WOLKE}_{start}$, $\texttt{WOLKE}_{mid}$, $\texttt{WOLKE}_{end}$ are treated as WOLKE). The annotations also contain *garbage labels* (denoted as 'si' for silence), which might correspond to movement epenthesis and therefore we do not consider it as a target term in evaluation. A discovered segment is mapped to a sequence of ground truth labels if the segment covers at least 50% of that label (see Figure 2). The metrics explained below are calculated using this transcription scheme.

Coverage: The total duration of non-overlapping discovered segments to the duration of all discoverable target segments in the dataset. A target segment contains a gloss that is repeated more than once in the dataset.

Cluster purity: This is a metric that is commonly used to evaluate clustering algorithms. Each cluster is mapped to the most common sequence of ground truth labels. Then purity is the ratio of the segment transcriptions that agree with their dominant cluster label to all discovered

Figure 3: Example clips of three segments that are clustered together.

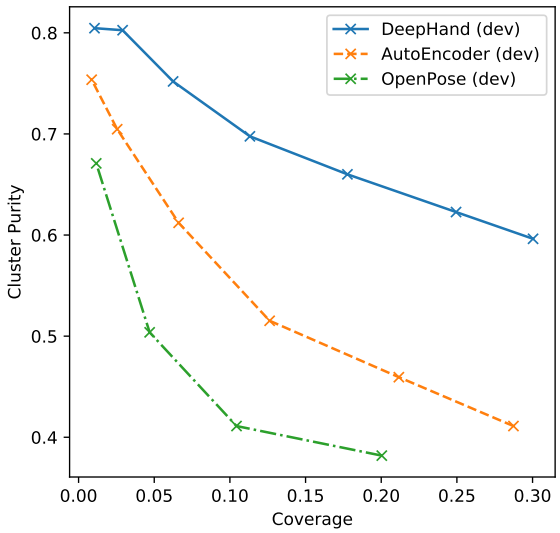

Figure 4: Coverage vs purity curves of sign term discovery using different features in the development set.

segments. For example, if most of the segments in a cluster are mapped to `(C,D)` label and another segment from the same cluster is mapped to `(C)`, it will be penalized due to not having the dominant cluster label. So the purity for the two clusters shown in Figure 2 would be 4/5, since 4 of the segment transcriptions out of 5 agree with the dominant cluster label. More than one cluster may be mapped to the same label, allowing many-to-one mappings.

Even though cluster purity is not included in ZR challenges, here we implement this metric because it is simpler and gives a more intuitive understanding about the quality of clusters. The following metrics are the ones that we implemented for the purpose of enabling comparison with the unsupervised spoken term discovery results. They are computed in terms of precision, recall and their geometric mean (F-scores). Detailed explanations regarding the calculations can be found in (Ludusan et al., 2014).

Matching quality: A set of metrics that measures how well the pairs of segments within a cluster match in terms of substring completion of their transcriptions. For example if transcription for a pair of segments is `(A,B)` and `(A,B,C)`, the substring matches `(A)`,`(B)`,`(A,B)` will be counted as positive for matching precision. Recall is computed over all possbile substring matches that are discoverable.

Grouping quality: This set of metrics measure the inherent quality of the clustering algorithm. It is a similar metric to cluster purity but here it is computed over pairs of segments that belong to discovered clusters instead of single segments. If the pairs of segments that are in the same cluster have the same transcription the precision is high. If inter-cluster pairs have different transcriptions, then the recall is high.

4. Experiments

Hand shape and pose features are extracted for all subsets. Then using the pose features belonging to the train set, an autoencoder model is trained which is then used to obtain the bottleneck features. For each of these three types of features, optimum parameters are found in the development set by grid search over the parameter values. These optimum values are fixed and the system is evaluated on the unseen test set.

When deciding on the autoencoder architecture, we experimented with various bottleneck dimensions (32, 64, 128), hidden unit sizes (64, 128, 256) and also depths (2, 3, 4). Based on the experiments in the development set, the best model has the bottleneck dimension of 64, 128 hidden units and 3 layer depth for both encoder and decoder parts. We trained this model with Adam optimizer (learning rate=0.02).

4.1. Parameter Tuning

The performance of the UTD algorithm (Jansen and Durme, 2011) depends highly on the parameters and the types of features. The default values used in the spoken UTD does not work for sign language because the sampling frequency, average term lengths and the feature properties are different for these domains. Hence, for each type of features (DeepHand, OpenPose, AE), we run grid search over the parameters only in the development set and pick the best combination of parameter values using the evaluation setup. The term discovery results on the development set are shown in Figure 4. These curves are obtained by sweeping the score threshold S_{dtw}, according to which the pairwise matches are eliminated before the graph clustering step. The curves illustrate the trade of between coverage and purity. The optimum parameters would yield both high coverage and high purity, and

Set	Feature Type	Discovered		Coverage (%)	Matching (%)			Grouping (%)		
		Clusters	Segments		Precision	Recall	F-score	Precision	Recall	F-score
Test	DH	858	2208	12.8	32.5	3.9	6.9	45.0	56.5	50.1
	AE	606	1506	10.4	8.0	1.3	2.3	12.9	23.0	16.5
	OP	181	384	5.4	0.0	0.0	0.0	1.6	8.8	2.7
Dev	DH	179	663	10.5	24.6	4.3	7.4	51.4	69.3	59.0
	AE	210	922	13.7	3.4	3.8	3.6	33.9	53.1	41.4
	OP	94	527	9.8	1.1	3.2	1.6	32.8	61.3	42.8
ZR'15 baseline (Eng)				16.3	39.4	1.6	3.1	21.4	84.6	33.3

Table 2: Sign discovery results obtained using different features (DH: DeepHand, AE: autoencoder, OP: OpenPose) in the development and test set. The baseline results of Zero Resource spoken term discovery challenge are given in the bottom row for comparison.

comparison of different setups can be deduced visually from these curves.

Even though the optimum values of parameters for each feature type vary, they are not much different from each other and here we share a combination that generally yields good results in this setup. The optimum values for the parameters that are described in Section 2.1.2 are as follows: for the percentile filter $L = 11$, $\mu = 0.60$ and for cost threshold $C = 4$.

4.2. Test on Unseen Signers

Using the optimum parameters for each feature type, we run experiments on the test set. We selected the S_{dtw} threshold such that coverage is around 10% and evaluated the discovery results as shown in Table 2. The hand shape features yield the best results for each setup. This might be because the dataset we use is one of the three datasets that DeepHand is trained over. It is trained with hand shape class labels, not the gloss information but nevertheless, having seen these images before might explain the robust performance on this dataset. Using the autoencoder resulted in slight improvements over the pose features both in the development and test sets. It might be the case that the bottleneck features provide a better representation when using cosine similarity for pairwise comparisons. Poor performance of the pose features is not because they are inferior to hand shapes. It can be due to low resolution of the images but more probably, it might not be applicable to compare two set of joint coordinates with cosine similarity. A feature transformation that is more relevant to similarity comparisons should be applied to pose features. We show that even a simple neural net can improve the pose features and more complex models would probably boost the performance. Pose features can be processed by graph convolutional encoders to better capture the connectivity relationships and temporal dependencies between joints. As an exploration of possibilities, here we aim to demonstrate that different types of features can be tailored to achieve better results for this task.

Using DeepHand on the test set, the glosses that are found most accurately are given in Figure 5. The occurrences represent the number of times the gloss type is clustered

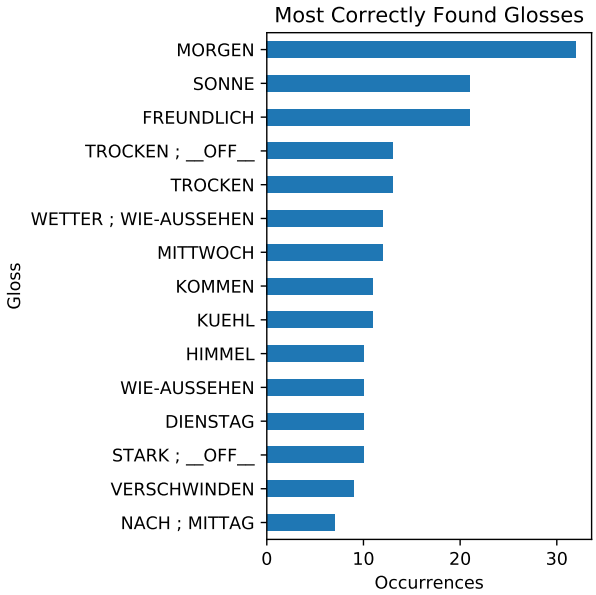

Figure 5: The glosses that the algorithm is able to cluster with %100 purity.

with 100% purity. Here some of the discovered types consist of two or more consecutive glosses and it shows that the algorithm can discover sequences of glosses (n-gram) if they occur frequently. It is observed that it can discover up to 4-gram. In Figure 6, some of the most confused gloss types are shown. The numbers between pairs of glosses are the number of co-occurrences in a cluster that has less than 50% purity. These words that are confused with each other have similar semantics and almost identical signings except for the mouthing. This suggests the importance of incorporating non-manual features to the feature extraction.

4.3. Discussion and Future Work

The proposed approach can aid sign language community in numerous ways. First of all, it can be very useful in cases where there are large amounts of sign videos to be annotated but not enough available resources. The algorithm proposes segments and clusters them so that each cluster corresponds to a hypothesized gloss. An ed-

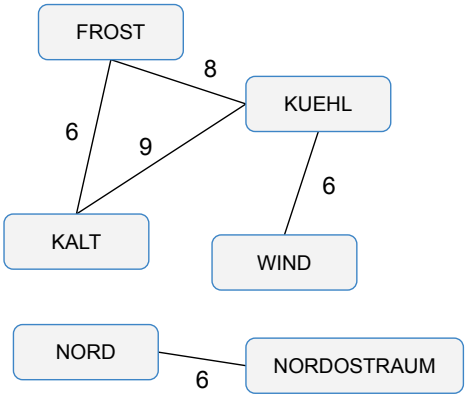

Figure 6: This graph shows some of the most confused gloss pairs. The numbers indicate how many times the pair is grouped together in a cluster that has less than 50% purity.

ucated annotator can easily purify the discovered clusters by eliminating the false segments and then saving the segments from pure clusters as annotations. By doing so, a significant amount of data can be annotated in short time. Given the pre-computed features, 1 hour video is processed in about 15 minutes by a 16-core CPU and an annotator can review the discovered clusters quickly with a proper software. However, the quality of the segmentation boundaries is not assessed in this work. In a future study, a psycho-linguistic experiment can be carried out, where the subjects are shown signs that are segmented by humans versus the UTD algorithm and they are asked to decide on which segmentation seems more natural. This would validate the potential use case of the UTD method as an automatic segmentation tool. Another benefit may arise when we want to train an ASLR system on a sign language that does not have enough resources. The clusters found by the UTD algorithm can provide weak supervision for training models, such as correspondence autoencoders proposed by Kamper et al. (2015). Finally researchers can use this as a tool to build lexicon for a new sign language.

Most of the clusters contain 2 or 3 segments. This is caused by the way the adjacency graph is clustered. As the clustering algorithm, connected components method simply thresholds the edge weights and groups the nodes that remain connected. However, using a more sophisticated algorithm (eg. modularity based), some of the similar clusters can be further joined, hence grouping recall can be increased. Analysis of such algorithms for UTD is done by Lyzinski et al. (2015) and the comparison of these algorithms on unsupervised sign discovery can be a subject of future study.

One of the drawbacks of this work is having used only one corpus for development and testing. Although the testing is done on unseen signers, the language is the same and the recording conditions are almost identical. A future study may include another sign corpus with a different sign language for testing. This would enforce the system to be language independent and would require better feature representations that can generalize well.

5. Conclusion

In this paper, we introduce a new task for sign language processing; unsupervised sign term discovery. The aim is to discover gloss types by clustering segments from a continuous sign dataset using only the video signal. We show that a highly acclaimed spoken term discovery algorithm can be run on continuous sign language videos by using visual features. The results show that, using appropriate features, the algorithm can achieve similar performance compared to its application in the spoken language domain. We believe that the studies targeting this task will lead to better annotation and ASLR systems in the future.

6. Acknowledgements

This work is supported in part by the Scientific and Technological Research Council of Turkey (TUBITAK) under Project 117E059.

7. Bibliographical References

Cao, Z., Simon, T., Wei, S.-E., and Sheikh, Y. (2017). Realtime multi-person 2D pose estimation using part affinity fields. In *Proc CVPR*, pages 1302–1310.

Dunbar, E., Cao, X., Benjumea, J., Karadayi, J., Bernard, M., Besacier, L., Anguera, X., and Dupoux, E. (2017). The zero resource speech challenge 2017. In *Proc. ASRU*, pages 323–330, Dec.

Jansen, A. and Durme, B. V. (2011). Efficient spoken term discovery using randomized algorithms. In *Proc. ASRU*.

Kamper, H., Elsner, M., Jansen, A., and Goldwater, S. (2015). Unsupervised neural network based feature extraction using weak top-down constraints. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5818–5822, April.

Kelly, D., Mc Donald, J., and Markham, C. (2011). Weakly supervised training of a sign language recognition system using multiple instance learning density matrices. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 41(2):526–541, April.

Koller, O., Forster, J., and Ney, H. (2015). Continuous sign language recognition: Towards large vocabulary statistical recognition systems handling multiple signers. *Computer Vision and Image Understanding*, 141:108–125, December.

Koller, O., Ney, H., and Bowden, R. (2016). Deep hand: How to train a CNN on 1 million hand images when your data is continuous and weakly labelled. In *Proc. CVPR*, pages 3793–3802, June.

Koller, O., Zargaran, S., and Ney, H. (2017). Re-Sign: Re-Aligned End-to-End Sequence Modelling with Deep Recurrent CNN-HMMs. In *Proc. CVPR*, pages 3416–3424, July.

Ludusan, B., Versteegh, M., Jansen, A., Gravier, G., Cao, X.-N., Johnson, M., and Dupoux, E. (2014). Bridging the gap between speech technology and natural language processing: an evaluation toolbox for term discovery systems. In *Proc. Language Resources and Evaluation Conference*, May.

Lyzinski, V., Sell, G., and Jansen, A. (2015). An evaluation of graph clustering methods for unsupervised term discovery. In *Proc. Interspeech*.

Nayak, S., Duncan, K., Sarkar, S., and Loeding, B. L. (2012). Finding recurrent patterns from continuous sign language sentences for automated extraction of signs. *Journal of Machine Learning Research*, 13:2589–2615.

Park, A. S. and Glass, J. R. (2008). Unsupervised pattern discovery in speech. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(1):186–197, Jan.

Pfister, T., Charles, J., and Zisserman, A. (2013). Large-scale learning of sign language by watching TV (using co-occurrences). *Proceedings of the British Machine Vision Conference*, pages 1–11.

Sutton, V. (2000). Sign writing. *Deaf Action Committee (DAC) for Sign Writing*.

Theodorakis, S., Pitsikalis, V., and Maragos, P. (2014). Dynamic-static unsupervised sequentiality, statistical subunits and lexicon for sign language recognition. *Image Vision Comput.*, 32:533–549.

Versteegh, M., Thiollière, R., Schatz, T., Cao Kam, X.-N., Anguera, X., Jansen, A., and Dupoux, E. (2015). The zero resource speech challenge 2015. In *Proc. Interspeech*, pages 3169–3173.