Recognition of sign language is a difficult task which often requires tedious annotations by sign language experts. End-to-end learning attempts that bypass frame level annotations have achieved some success in limited datasets, but it has been shown that high quality annotations improve performance drastically. Recent unsupervised learning methods using deep neural networks have achieved successes in learning feature extraction. Yet a technique for high quality frame level classification using unsupervised techniques does not exist. In this paper, we assign labels of an isolated Sign Language frame level classification using unsupervised techniques does not exist. SL datasets are not large and balanced enough for meaningful end-to-end learning. As Farag and Brock stated in their study, publicly available SL datasets are not large and balanced enough for meaningful end-to-end learning. Hence, segmenting SL videos is crucial and needed for studying the underlying structures of SLs using deep neural networks. Our study is based on RGB data whereas [15] proposes a solution depending on skeleton data.

Similar to the task defined in this paper, [16] also aims at recognition of a set of 30 hand shapes as bricks of Chinese sign language from RGB images. Aly et al. approached the task using depth images for both detection of hand shapes and signer independent clustering of American sign language videos [17]. Another relevant work by Kadir et al. approaches the task of sign language recognition in unconstrained conditions by detecting body centred descriptions of activity by coining linguistic description of these activities, and by-passes temporal modeling [18].

Recent approaches to sign language recognition focus on continuous sign language recognition [19, 20]. Huang et. al [21] used multimodal inputs and have shown success in extracting discriminative spatio-temporal features from videos. Pigou et. al constructed a successful network that can recognize 20 Italian gestures with high accuracy [22]. For extracting more information and reaching higher accuracies on SL we need bigger annotated data-sets as in [23]. Current datasets use a combination of finger spelled signs, isolated one-handed signs, or two handed signs. Larger datasets recorded in realistic settings are needed to extract linguistic features of a sign language from datasets.

In this paper, we followed the approach of using sparse autoencoders (SAE) to discover the set of underlying KHSs that is assumed to compose the SL’s “language model”. This idea has recently been proposed for acoustic unit discovery Zerospeech challenge, in which a set of acoustic units are retrieved from untranscribed speech
recordings [24]. This task is analogous to the challenge overtaken in this paper since the underlying KHSs could be considered as tokens of meaningful units to utter a SL sentence, just like acoustic units are used in concatenation to speak.

For this, we trained a SAE in an unsupervised fashion, which simultaneously clusters input frames on an intermediate layer after encoding, and reconstructs them at the output of the decoder layer. Therefore in sparse autoencoders, a sparsity cost is also utilized in addition to the reconstruction loss. In the next section, we describe the methodology adopted in this work and we present our findings over the experiments conducted on the Bosphorus Sign Turkish Sign Language Database (BS-TSLD).

2. METHODOLOGY
The main objective of this study is to discover the underlying hand-shapes that are used in Turkish Sign Language (TSL). We aim to seek the recurring hand-shapes in the TSL corpus and find the minimum number of such figures, we refer to as KHSs, so that the information is conveyed using them as units of meaningful tokens. As the first step, we use skeleton features of Kinect and retrieve the sub-frames that contain the active hand within each frame of the SL video. We adopt two sets of feature representations from these hand frames, meanwhile examining the representation power of hand crafted features like Histogram of Oriented Gradients (HOG) and deep learning based representation such as the bottle neck activations of resnet18. PCA mapping to lower dimensions is also applied to each of these features to reduce variability.

2.1. Sparse Autoencoder (SAE)
Sparse autoencoder (SAE) architecture is coined recently in an acoustic unit discovery task in the Zerospeech 2019 challenge, which is a similar task to that of detecting key hand shapes. In SAE, an autoencoder system with a decoder-encoder structure is trained, except in the embedding layer a softmax is employed such that this layer acts as a posterior probability vector over the set of underlying clusters. Hand frame representations are fed into the input layer of this autoencoder and the sparse representation is obtained through an intermediate feed-forward softmax layer. It is hoped that this embedding will yield a sequence of posterior probability vectors during prediction and enable an automated labelling of SL videos. In other words, it is expected that the activations of this layer act as the ‘hidden’ or the ‘unknown’ states (the KHS units to be discovered). To enforce sparsity of this representation, this layer is penalized with negative L2-norm, so that the representation resembles a set of meaningful units to utter a SL sentence, just like acoustic units are used in concatenation to speak.

\[
P = \text{encoder(softmax}(x))
\]

\[
x = \text{decoder}(p)
\]

(1)

It is expected that similar hand-shapes will activate the same cluster node at the intermediate layer and, the resulting \(p\) will generate a similar enough representation, i.e. a centroid at the output so that the following combined cost is minimized:

\[
J_{\text{SAE}} = \sum_{\forall t} \|x_t - \bar{x}_t\|_2^2 + \lambda \|p_t\|_2^2
\]

(2)

2.2. Correspondence Sparse Auto Encoder (CoSAE)
The term auto in correspondence autoencoder (CAE) is in fact a misnomer. In CAE training, pairs that are assumed to be similar are fed from the input and output layers to reconstruct each other [25]. It has been shown in [26] that CAEs perform better than denoising or deblurring autoencoders in obtaining discriminative representations in an unsupervised manner. Similar to CAEs, correspondence sparse autoencoders (CoSAE) have an intermediate clustering layer to extract the cluster assignments directly. For this, the hand frames that may correspond to the same KHS are obtained from the dataset using dynamic time warping (DTW) alignment path obtained from the pairs sign video sequences. All pairs of videos that correspond to the same sign, \(X \in \mathbb{R}^{D \times N}\) and \(Y \in \mathbb{R}^{D \times M}\) are aligned with DTW and the alignment path \(\Phi\) is used to retrieve the frames of \(Y\) that correspond to the frames of \(X\). A dropout of \(p = 0.5\) was applied after the decoder. The SAE and CoSAE training architectures are demonstrated in Figure 1.

2.3. Baseline Method and the Ground Truth
We implemented the method of Deep Clustering [11] and took it as a baseline. In [11], images are fed into CNNs to obtain a discriminative representation on the penultimate layer. K-means clustering is applied to these bottleneck features prior to each epoch of neural network training and the one-hot vectors depicting the k-means clustering assignments are used to train the network. This procedure is repeated at each epoch and the normalized mutual information (NMI) between the actual class assignments are observed. Similarly, we observed the NMI scores of the SAE and CoSAE learning after each epoch as a reference. The main difference of our work from this approach is that we aim to address the two objectives simultaneously, i.e. the sparsity of the intermediate layer and the reconstruction loss.

One possible useful outcome of this approach is that it enables automatic labelling of the frames as a sequence of KHSs and trans-
tion states, a procedure that is extremely expensive to employ by human interaction. We use the KHS assignments that are obtained and labelled frame-by-frame by human experts as ground truth labels. Since the number of KHS classes are not known prior to execution, in addition to NMI, we monitor the frame accuracy rate, calculated via a learned mapping between the cluster assignments and the actual class labels.

2.3.1. Evaluation Criteria

The information that is shared between cluster labels and ground truth can be measured by two different criteria. The first criterion is the Normalized Mutual Information (NMI) that gives a score of accordance between two clusterings of the same dataset. NMI is expressed as:

\[ NMI(C, K) = \frac{2 \times I(C; K)}{\sqrt{H(C)H(K)}} \]  

where \( C \) and \( K \) are the multinomial distributions of class and cluster labels, respectively. \( H \) denotes entropy and \( I(C; K) \) is the mutual information between \( C \) and \( K \). It should be noted that this measure is independent of the number of classes and the clusters and, gives a value in the range \([0, 1]\).

A directly interpretable measure is the frame error rate, or the frame accuracy rate which is the proportion of correctly classified frames. To make such an evaluation possible, we propose a many-to-one ‘mapping learning’ that converts the cluster assignments to their corresponding class labels using the confusion matrix. This mapping is obtained using the following formula:

\[ \mathbf{M} = \hat{\mathbf{M}} \]  

where \( \hat{\mathbf{M}} \in \mathbb{R}^{C \times K} \) is the non-square confusion matrix having class labels in the rows and cluster labels in the columns. It is multiplied with \( \mathbf{W} \in \mathbb{R}^{K \times C} \), the weight matrix to be learned which computes \( \hat{\mathbf{M}} \in \mathbb{R}^{C \times C} \), showing the class confusion matrix. The aim is to maximize the number of samples in the diagonal of the resulting matrix \( \hat{\mathbf{M}} \). There are also two regularization terms that sparsify row and column elements of the weight matrix, hence the weight matrix becomes a mapping of cluster labels to the class labels. The idea behind this maximization technique is that if the clusters can represent clusters in classes then we can label only the clusters by looking at a couple of highly activated elements and using the learned map \( \mathbf{W} \) for labeling the whole dataset.

3. EXPERIMENTS

3.1. Dataset

We ran our experiments on a subset of BosphorusSign Turkish Sign Language Database (BS-TSLD)\(^{[27]}\). The subset of the dataset we used covers 11 signed sentences that are performed by 6 different signers and hand labelled in a frame-wise manner. The KHSs in the dataset represent context as well as shape. That is to say, they do not only differ in finger configuration, but also in the position of the body. Samples of these ground truth KHSs can be seen in Figure\(^2\).

3.2. Experimental set-up

We conducted our experiments using two different set of features: HOG as a hand crafted feature and the pre-trained resnet18 bottleneck layer activations\(^{[28]}\) as a deep learning-based feature. Both features are mapped to 256 dimensions with PCA. A feed-forward encoder is used with a dropout rate of \( p = 0.5 \) followed by a softmax layer to get the probability mass function \( p \) over \( K \) clusters. The decoder layer is also a feed-forward network such that the input is reconstructed as a weighted sum of the rows of this matrix. The \( \lambda \) parameter in \(^{(2)}\) decides the trade-off between the reconstruction loss and the sparsity constraint obtained by the L2-norm of the intermediate layer. It should be noted that the intermediate layer has a maximum L2-norm when it is a one-hot vector since it is output of a softmax activation and sums to one. Several \( K \) and \( \lambda \) values are tested and compared with the baseline. Models are trained using adam optimizer using Keras toolkit with tensorflow backend\(^{[29]}\).

3.3. Results

Our main aim was to obtain the cluster assignments on the intermediate sparse layer (\( p \)) by minimizing the reconstruction loss. The intermediate layer is expected to ‘classify’ the input frame into one of its \( K \) classes, and then in the decoder layer, this frame (or its correspondence) is reconstructed using the probability mass function observed at the \( p \)-layer. We observed in Figure\(^{[3]}\) that the neighboring frames have consistent class-assignments, or probability distributions. The NMI and the accuracy, obtained by treating this layer as the classification layer, is observed as the training progresses similar to the work proposed in \(^{[11]}\).

We have experimented with various cluster numbers (\( K \)) and sparsity weight values (\( \lambda \)). We obtain good results using the parameters provided in Table\(^{[1]}\). Since there is no means of validation and early stopping in our unsupervised setting, and the main objective of a high frame accuracy rate is different from the training...
loss (mse+sparsity), there is no early stopping criterion available and hence we trained all models for the same number of epochs.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K$</td>
<td>256</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>1.0</td>
</tr>
<tr>
<td>batch size</td>
<td>16</td>
</tr>
<tr>
<td>number of epochs</td>
<td>50</td>
</tr>
<tr>
<td>learning rate</td>
<td>0.01</td>
</tr>
</tbody>
</table>

Table 1. Parameters for SAE/CoSAE training

Table 2 gives the frame accuracy rate and the NMI scores of the proposed methods, compared with the baseline. We also included the performance of the ordinary k-means clustering with Euclidean distance and MMSE cost, since it could be considered as a natural baseline for clustering applications. It can be observed that hand crafted HOG features yield better performance than resnet18 bottleneck activations, which were trained on imagenet data. One interesting observation is that the improvement brought by the collaborative training is more significant on resnet18 bottlenecks, although it also improves the results with HOG. Overall, the best accuracy is obtained with CoSAE and HOG features. Using the NMI metric also indicates the same results: CoSAE gives the best KHS labeling results.

<table>
<thead>
<tr>
<th>Methodology</th>
<th>Accuracy</th>
<th>NMI</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>resnet18</td>
<td>HOG</td>
</tr>
<tr>
<td>k-means</td>
<td>0.2995</td>
<td>0.5998</td>
</tr>
<tr>
<td>Deep Cluster</td>
<td>0.4840</td>
<td>-</td>
</tr>
<tr>
<td>SAE</td>
<td>0.4008</td>
<td>0.7026</td>
</tr>
<tr>
<td>CoSAE</td>
<td>0.7433</td>
<td>0.8307</td>
</tr>
</tbody>
</table>

Table 2. Performance of the proposed systems

Figure 4 demonstrates the evolution of accuracy as the training proceeds using the CoSAE method, using two different $\lambda$ values. The order of correspondence is altered in each epoch, which yields an additional increase in the accuracy. It can be observed that keeping the $\lambda$ values small results in a lower performance and higher fluctuation. It should be noted that training without correspondence (i.e. only SAE) fails to exhibit the drastic jumps in accuracy apparent in the Figure 4. Furthermore, going back to the conventional SAE, i.e. generating the input at the output, every other epoch in CoSAE training provides further improvements in the accuracy, which implies that local minimums are avoided with the use of corresponding frames.

4. CONCLUSION AND FURTHER WORK

In this paper we proposed an unsupervised method for detecting KHSs that convey meaningful information in Turkish sign language videos. We used correspondence sparse autoencoders that can be used to automatically label the frames of sign language videos using their sparse intermediate layer activations. Experiments conducted on a subset of BS-TID dataset show that this method can be used as a reliable tool to cluster and label sign language videos. Simple HOG features worked better than ResNet18 features trained on ImageNet. Fine tuning the ResNet on hand data may yield better features. In this work, we performed classifications on a frame basis, without temporal modeling. Our future work will incorporate temporal models such as recurrent neural networks.

5. ACKNOWLEDGMENTS

Authors would like to thank Mehmet Burak Kurutmaz for his invaluable help with the cluster-class mapping learning system that is used in frame error rate calculation.

6. REFERENCES


