Proceedings of the International Joint Conference on Neural Networks (IJCNN), pp. 2539-2548, Rio de Janeiro, Brazil, 2018

Sparse Autoencoders for Posture Recognition

Doreen Jirak and Stefan Wermter
Knowledge Technology, Department of Informatics, University of Hamburg, Hamburg, Germany

***Abstract*—Among different gesture types, static gestures or *postures* deliver a broad range of communicative information like commands or emblems. Vision-based processing for posture recognition is the most intuitive yet challenging task in intelligent systems. Achievements in deep learning, specifically convolutional neural networks (CNN), replaced creating hand models or engineering features for automated image feature learning at the expense of large data requirements and long training sessions for optimal parameter tuning. The aim of the present study is to explore the potentials of sparse autoencoders for posture recognition, promoting an alternative method to present convolutional approaches. We conduct experiments with hierarchically designed autoencoders to retain the desired image feature abstractions on two posture datasets with distinct characteristics. The different data properties allow us to demonstrate parameter influences on the network performance. Our evaluation shows that even a shallow network design achieves superior performance compared to a multiple channel CNN, and comparable results on a small dataset with sparse image samples. From our study we conclude that "lightweight" approaches can be viable tools for posture recognition, which are worth more explorations in the future.**

I. Introduction

One aim of the neural network community is to develop neural computational models resembling processing capabilities of the brain, capturing its efficiency and robustness in detection and recognition over different modalities. The progress both in data availability and accelerated computing on GPUs tremendously influenced the neural network and machine learning community in the past years. The term "Deep Learning" signifies deep neural networks (DNN), where especially CNNs address challenging problems primarily in computer vision but also speech recognition and audio processing. Although learning filters instead of handcrafting shows convincingly good performance, the past years' literature demonstrates merely a set of network variations and constraints on training those models for the sake of benchmark performance. Learning becomes an engineering process relying on exhaustive tuning and the provision of correct labels [1] being diametric to the "neurally-inspired" learning principles. Additionally, CNNs are rather robust perceptual tools mirroring early vision processing in the visual cortex, which identifies image feature like edges and shapes. Autoencoders are neural models which provide this feature emergence in the absence of any labels and may offer an unsupervised learning approach in addition or even in substitution to supervised CNNs. Moreover, autoencoders allow a visualization of features obtained after learning and thus open the often criticized 'black box' behavior of neural networks. We want, therefore, to investigate the potential application capabilities of autoencoders for the task of posture recognition, which is an essential research topic in Human-Computer Interaction (HCI) and an important part in natural interactions between humans and robots (HRI). In particular, we are using autoencoder networks with different hierarchical levels to retain feature abstractions similar to CNNs and with neurons implementing a *sparse* firing behavior. In neuroscience it is hypothesized that sparse codes [2] balance neuronal responses to incoming stimuli and the metabolic costs. In the present context, a sparsity constraint enables neurons to react to only specific image features.

In the vision domain, hierarchical learning has been adopted, for instance, in learning features for object recognition with sparse feature detectors [3], deep belief networks (DBN) [4] as well as recurrent convolutional neural networks (CNN) [5]. For action recognition, the introduction of 3D kernel CNNs [6] allowed for spatial learning across images as well as a two-stream CNN for video data [7]. In the unsupervised domain, hierarchical processing employing growing-when-required networks (GWR) was introduced to tackle the problem of classifying new actions without retraining [8]. However, only few approaches consider posture recognition. A study on sign language recognition contrasted CNNs, Deep Belief Network (DBN) and HOG-features combined with Support Vector Machines (SVM) and revealed superior performance of DBN over CNNs, and slightly better results of CNNs versus SVM [9]. Testing the models met realtime conditions for CNNs and DBNs but training the models took hours. Another sign language study compared CNNs and stacked denoising autoencoders (SDAE) [10], the latter showing best performance in training and testing and in runtime comparison. We target a similar approach: particularly, we focus on two papers on vision-based posture recognition introducing posture data with distinct characteristics evaluated on two different approaches, which we describe in the following.

The first paper we are considering addresses elastic graph matching (EGM) [11] inspired by orientation selectivity of neurons in a cat's extrastriate cortex [12]. A graph describes characteristic points of a posture e.g. the fingertips. Local nodes (*jets*) are parametrized 2D Gabor wavelets responding to local image patches regarding spatial size and orientation. The global hand shape is captured by edges labeled with a distance vector. A set of model graphs serve as templates and are compared with new test images regarding the locations and distance similarities. The approach was tested on a set of hand postures with both uniform and complex background (see Figure 1, JTD dataset in the following).

As an alternative to hand models, a CNN with three independent input channels implementing additional Sobel

Fig. 1. Examples from the JTD dataset comprising 10 hand postures shown in front of two uniform backgrounds (row 1-2) and a complex background (row 3).

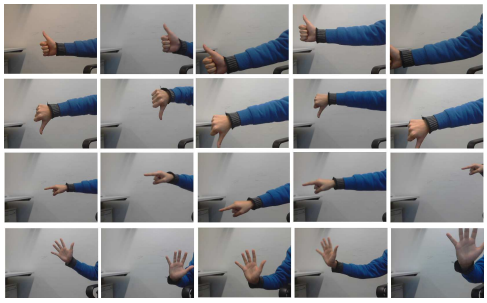

Fig. 2. Examples from the NCD posture set. The first column depicts the default position and posture type. Each posture was also varied in hand orientation and positions in the scene including hand occlusions.

edge dectectors (Multi-Channel Convolutional Neural Network, MCNN) was proposed [13], introducing the NCD dataset obtained from a camera of the NAO humanoid robot. The network was trained using a 2D kernel for image-wise learning and a 3D kernel for image stacks. For comparison, the authors used the JTD database [11] and a set of 4 postures varying in hand orientation and positions (see Figure 2). We use the two datasets in our study and will contrast the different approaches in section VI.

II. Sparse Autoencoders in a Nutshell

An autoencoder is a network that learns an approximate reconstruction of its input x^N, i.e. $x \approx h(Wx + b)$, where W denotes a set of weights including a bias term b, and h is either a linear or nonlinear transfer function. The rationale behind autoencoders is to learn a compact representation by an encoder-decoder scheme. The encoding is a mapping of the N-dimensional input x to a lower dimension M, i.e. $\mathbb{R}^N \mapsto \mathbb{R}^M, M << N$. This mapping is realized by training the connection weights W between the input and the encoder, which capture characteristic features of the images. The decoder reconstructs the input from the resultant representations, which is an input approximation due to the lossy compression. However, autoencoders trained to obtain distinct input features can be beneficial for classification, and the visualization of the encoder weights allows an analysis of resultant representations in the network which contributes to an understanding of what the network actually learns [14].

Sparse Autoencoder Training

The autoencoder learns discriminative features of its input in the absence of any labels. As images usually contain specific patterns and sensor noise, hierarchical or usually called "deep" models provide a way to learn image structures ranging from simple lines to shapes similar to early vision processes in the brain. To achieve this, autoencoders can be stacked where the decoding phase is discarded and instead the feature coding from the first stage is passed as input to a second layer and so on, until a desired level of depth is reached (see Figure 3). A standard autoencoder produces an image manifold from the original input images, and the training performances from such an autoencoder gives a quantitative measure how well this original data was reconstructed. The minimization of the objective function $\mathcal{L}$ is evaluated using the mean-squared-error (MSE). It is common to additionally introduce a penalty coefficient λ on the weight norm $||W||$ to avoid overfitting.

Another constraint on the training in the context of an autoencoder enforces sparse firing of neurons in the hidden layer activated by the presented inputs. This process is based on neuroscientific work on sparse coding schemes in the human brain [2], where neurons respond to specific input stimuli only specializing to e.g. orientation. Sparse firing is also hypothesized to account for the tradeoff between information transmission and a neurons' energy consumption, a principle captured by neuroscientific information maximization models. A sparse firing pattern is achieved when a neuron mostly remains silent, which translates to either an activation[1] of 0 or -1. The average neural activations in the hidden layer are computed as:

$$\hat{\rho}_j = \frac{1}{m} \sum_{i=1}^{m} a_j x^{(i)} \tag{1}$$

where a_j is the activation of the jth hidden neuron, $x^{(i)}$ the ith input and $\hat{\rho}_j$ an approximation of the sparsity parameter ρ in the range $[0; 1]$. As $\hat{\rho}_j = \rho$ is desirable, the training penalizes divergence between these values. A way to quantify this is the Kullback-Leibler divergence $KL(p||q)$ which measures (dis)similarities between probability distributions depending on their parameters p, the theoretical value, and q, its approximation. Note, that $KL(p||q) = 0$ iff $p = q$ and $KL(p||q) \neq KL(q||p)$. The firing pattern is modeled as a Bernoulli distribution with mean ρ because we are interested only in neural "firing" or "not firing". The KL-divergence is computed as:

$$KL(\rho||\hat{\rho}_j) = \rho \log \frac{\rho}{\hat{\rho}_j} + (1 - \rho) \log \frac{1 - \rho}{1 - \hat{\rho}_j} \tag{2}$$

resulting in $\sum_{j=1}^{m} KL(\rho||\hat{\rho}_j)$, where an additional coefficient $\beta > 0$ controls the influence of this sparsity regularization term [15]. With increasing q deviating significantly from p the KL-divergence increases monotonically. An additional constraint to suppress this behavior is supplemented in the overall sparse autoencoder objective function [15], [2]:

[1] depending on the range of the used transfer function

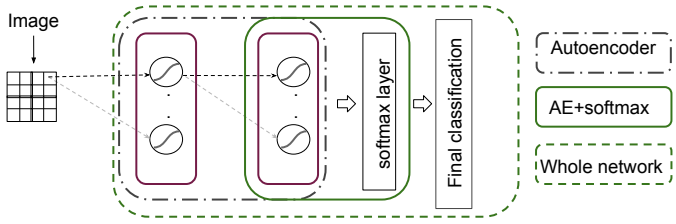

Fig. 3. Neural architecture consisting of an unsupervised learning stage implemented by stacked, sparse autoencoders (grey box) and a subsequent supervised training part for the final classification (green boxes). We show here the $L2$ design, i.e. two stacked autoencoders, for illustration purposes. The number of layers L depend on the complexity of the task.

$$\mathcal{L}(x,y) = \min_{W} \Big[\sum_{i=1}^{m} (h_W(x^{(i)} - y^{(i)})^2 + \lambda(||W||_2^2) + \beta \sum_{j=1}^{m} KL(\rho||\hat{\rho}_j) \Big] \tag{3}$$

where λ is the parameter for the ℓ_2 regularization.

III. Experiments

In the following, we describe the datasets and the preprocessing scheme employed on the data to ensure an understanding of our experiments and to stimulate reproduction on similar tasks.

A. Datasets and Preprocessing

The NCD set is recorded with the NAO camera which has an image resolution of 640×320. Posture performances are varied along the image plane including hand occlusions. To be comparable with the results presented in [13], we follow the same image processing steps and convert first the images from RGB to greyscale. For both datasets, we reduce the images to 28×28 pixel size because we are interested in a minimal setting to investigate the potential of small images with respect to the autoencoder feature representations and their impact on the classification capabilities. Increasing the image sizes would reveal more image detail and consequently the adding of this information usually results in higher performance. As this is rather trivial, we exclude the variable image size.

The JTD set provides 10 gestures performed by 24 subjects in front of 3 different backgrounds[2] (see Figure 1). A posture is always situated frontal to the camera positioned in the image center, despite some few variations due to different hand shapes (e.g. length of fingers). The original image size is 128×128 pixels. We split the JTD dataset into subsets. The letters behind the individual sets determine the background, i.e. W=white, B=black, and A=all images combined, i.e. from the complex and the uniform backgrounds (cf. Figure 1). For the JTD-A subset, the training set comprises 169 posture samples with complex background and 167 each for the uniform background. As a remark, we did not expect reasonable performance for the subset comprising posture with complex backgrounds only due to our constraint on small image sizes. Thus, it is left out in this study.

[2]http://www.idiap.ch/resource/gestures/

TABLE I
Dataset sizes

	Total	Train	Test
JTD-W	240	168	72
JTD-B	239	167	72
JTD-A	718	503	215
NCD	2716	1901	815

TABLE II
Autoencoder parameter

λ_{L_1}	β_{L_1}	ρ_{L_1}	λ_{L2}	β_{L_2}	ρ_{L_2}	N_1	N_2
0.001-0.01	1-4	0.1-0.4	0.01	1-4	0.1-0.4	100	50

B. Training Procedure and Parameters

We further split all datasets into 70% for training and 30% for test. The number of samples for each dataset are listed in table I.

The activation functions for both the encoder and decoder was the logistic sigmoid, i.e. $\frac{1}{1+e^{-a}}$. The result of the encoding was then passed to a softmax layer and subsequently finetuned across the whole network (500 epochs each). We used the 'scaled conjugate gradient' (scg) [16] optimizer for the backpropgation (cf. Figure 3). The autoencoders were stacked using layers $L = \{1, 2, 3\}$. We empirically determined the number of neurons and the value ranges for all other parameters used in this study. They are summarized in table II. We increased the number of neurons to $h_1 = 200$, $h_2 = 100$ and $h_3 = 50$ to ensure proper training for the $L3$ network design. When training networks with three hidden layers we realized that the scg optimization for backpropagation performed worse than the 'resilient backpropagation' (rprop) algorithm [17]. The results from supervised learning were evaluated using crossentropy and were averaged over 20 trials.

IV. Results and Evaluation

In the following, we evaluate the different autoencoder schemes considering their design with layers L_i and separate the classification performance between the unsupervised autoencoder stage with subsequent softmax classification (in the following 'AE+softmax') and the finetuning over the network (in the following 'whole network'). All results apply to the evaluation of the test set of the corresponding dataset and parameters given above.

A. Performances of Architectures - NCD dataset

For an autoencoder with only one layer $L = 1$ the lowest accuracy over all parameter configurations was observed for regularization parameters $\lambda = 0.001$, $\beta = 1$, and sparsity $\rho = 0.1$, i.e. the mean accuracy from AE+softmax was 44.5% and 78.0% when trained over the whole architecture (median: 42.85% and 86.2%). To find an explanation for

the performance difference, we investigated the individual trials and observed that in 3 trials only 1 posture class was learned. As a consequence, both learning stages achieved only 28.8% accuracy, which influenced the global performance. Our assumption that the classes were too imbalanced both in the training and test set could not be supported. Hence, there might be other factors like insufficient feature representations, which we will demonstrate in section V.

Table III shows the influence of the sparse firing parameter ρ and parameters $\lambda = 0.001$ and $\beta = 1$ on the classification performance both expressed as the average calculation (mean performance) and the median over all trials. The results show a peak performance for $\rho = 0.3$ with a median of 93.62%. A high discrepancy between the average and the median calculations hints at large skewness of results across trials. To avoid the influence of outliers, we will only report the median but both measures are shown in the corresponding tables. Increasing the sparsity regularization β dampens this effect, as depicted in the corresponding graphics. Figure 4 and Figure 5 show the boxplots for β parameter values, while we fixed $\rho = 0.3$ and $\lambda = 0.001$ both for AE+softmax and for the final classification. We observe a large span of the accuracy results for $\beta = 1$ ranging from a minimum of 28.8%, which is close to guessing the correct posture, to an outlier with a maximum of 75%. When β is increased, we observe a significant reduction in the result variations concurrently to a rise of the accuracy, best demonstrated in Figure 5 for $\beta = 4$. This parameter configuration yielded the lowest intra-trial variances, which is relevant when discussing the reliability of neural architectures regarding their classification ability. A network which has high intra-trial variability and thus only produces by chance good performance is clearly not desirable in any application of a posture recognition system. Overall, best performance with a median value of 100% was achieved for $\rho = 0.1$, which supports the rule of setting this parameter to a low value. However, we are also interested in the resultant feature representations in the unsupervised learning stage and its impact on the performance. For the mentioned parameters, the AE+softmax stage yields 58.83%, which shows that finetuning is still necessary to obtain good performance.The best result for AE+softmax only was obtained for $\rho = 0.2$ (other parameters constant) yielding an accuracy of 64.48%. From our results it becomes apparent that different values of the parameter ρ produce only slight differences in the overall performance so other factors like experimental variances or analysis of the parameter impact on the feature representations become important. We suspect that variations among the posture images influence the resolution needed in the neuronal responses.

From the evaluation of the single autoencoder network, we infer that parameter configurations degrading or increasing the performance naturally influence the network behavior of such stacked autoencoders. Our results confirm worse performance for both, AE+softmax and the whole network when the sparsity parameters ρ and β are low. Table IV depicts accuracy for increasing ρ_1, i.e. the sparsity firing in the first layer with

TABLE III
VARYING SPARSITY FIRING FOR NCD (L1)

Accuracy (%)	AE+softmax		Whole network	
ρ	mean	median	mean	median
0.1	44.5	42.85	78.02	86.20
0.2	50.27	49.32	87.67	81.28
0.3	56.63	55.21	80.27	**93.62**
0.4	52.02	50.55	73.38	87.79

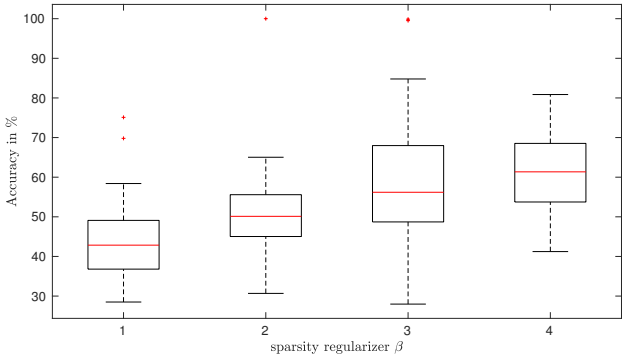

Fig. 4. Boxplot demonstrating the influence of β on the results of the NCD set with $\lambda = 0.001$ and $\rho = 0.3$ of the softmax classifier ($L1$). The red crosses depict outliers.

$\beta_1 = \beta_2 = 1$ and $\lambda_1 = 0.001$. Other parameters were fixed as follows: $\rho_2 = 0.1$ and $\lambda_2 = 0.01$. The best performance was again obtained for $\rho_1 = 0.3$ but also with higher variances across trials. Increasing the regularization β now for both layers to $\beta_1 = \beta_2 = 4$ with $\rho_1 = 0.3$ and $\rho_2 = 0.2$ yielded the best performance for the fully trained network with an average accuracy of 100%. This result with the impact of β is shown in Figure 6. We obtained very good results on classification accuracy with a shallow network design. Going "deeper" with more layers would give no further performance improvements but may lead to overfitting. Thus, we did not consider the $L3$ network for further analysis.

B. Performances of Architectures - JTD dataset

Involving autoencoders for the training of posture with uniform background reveals good performance as we will

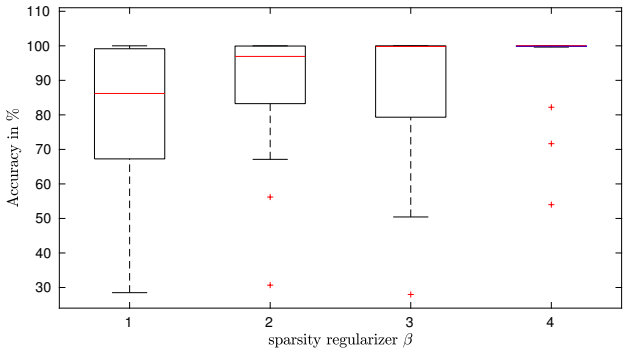

Fig. 5. Boxplot demonstrating the influence of β on the results of the NCD set with $\lambda = 0.001$ and $\rho = 0.3$ after finetuning ($L1$). The red crosses depict outliers.

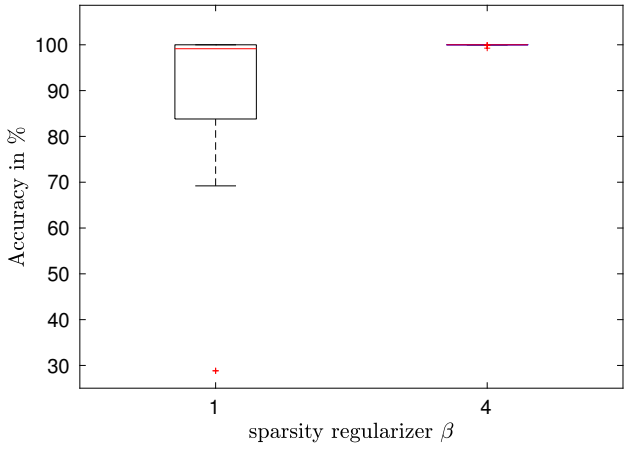

Fig. 6. Regularization effects for the L2 network on the intra-trial variability. Red crosses depict outliers.

TABLE IV
VARYING SPARSITY FIRING FOR NCD (L2).

Accuracy (%)	AE+softmax		Whole network	
ρ	mean	median	mean	median
0.1	40.01	38.59	66.41	73.25
0.2	43.28	42.64	75.54	82.27
0.3	52.82	52.27	87.39	**99.14**
0.4	46.61	49.20	71.48	78.90

demonstrate in the following. We start with the results on the JTD-B subset for both $L1$ and $L2$ networks.

In table V, the results for the effect of sparse firing for $\beta = 1$ and $\lambda = 0.001$ are shown. Here, the best performance was achieved with $\rho = 0.2$ for both AE+softmax and finetuning with a median of 70.83% and 75.70%, respectively. Notably, the median for the remaining sparsity parameter in the last column is similar, which we explain by the small dataset.

In combination with stricter regularization, the performance increases as shown in table VI. Considering only the results from the AE+softmax learning, the best performance of 79.17% was achieved for $\beta = 4$, $\lambda = 0.001$ and $\rho = 0.3$.

TABLE V
VARYING SPARSITY FIRING FOR JTD-B (L1)

Accuracy (%)	AE+softmax		Whole network	
ρ	mean	median	mean	median
0.1	49.93	51.39	64.31	69.44
0.2	56.11	70.83	61.11	**75.70**
0.3	48.33	62.29	52.08	69.44
0.4	47.15	63.40	45.13	69.44

TABLE VI
EFFECT OF β FOR $\rho = 0.2$

Accuracy (%)	AE+softmax		Whole network	
β	mean	median	mean	median
1	56.11	70.83	61.40	75.70
2	61.94	61.80	72.99	76.38
3	67.78	71.52	74.10	77.08
4	69.30	73.61	74.44	**77.78**

TABLE VII
SPARSITY REGULARIZATION ON JTD-B (L2)

Accuracy (%)	AE+softmax		Whole network	
β	mean	median	mean	median
1	50.28	54.86	61.39	73.61
2	57.57	54.17	70.21	71.53
3	50.76	50.00	65.70	70.83
4	55.00	56.94	68.54	72.91

Table VII displays the accuracy on the JTD-B set for a 2-layered, stacked autoencoder with the same fixed parameter set as reported for the evaluation of the NCD set.

The best overall performance achieved for training only AE+softmax was 76.39% for $\beta_{L1} = \beta_{L2} = 4$, $\lambda_{L1} = \lambda_{L2} = 0.001$ with $\rho_{L1} = 0.2$ and $\rho_{L2} = 0.1$. This performance is comparable to the final finetuning step which yielded an average accuracy 77.78%. We observed a significant performance push when finetuning the network for the NCD set. However, the best result for the JTD-B subset here was 81.25% for $\beta_{L1} = \beta_{L2} = 4$, $\rho_{L1} = 0.2 = \rho_{L2} = 0.2$ and $\lambda_{L1} = 0.001, \lambda_{L2} = 0.01$, showing only minor performance differences between the two learning stages for this dataset.

The JTD-W subset contains postures performed in front of a white background. We assumed, that the lack of contrast negatively impacts the performance of the autoencoder in addition to its spare samples. Our evaluation confirms this hypothesis. The overall best performance achievable for $L1$ was 45.83% for $\lambda = 0.007$, $\beta = 4$ and $\rho = 0.4$. This parameter configuration also yielded the best result when considering only training the autoencoder with a final softmax classifier, a median of 24.3%.

Using the $L2$ design showed few improvements only. The parameterization with $\beta_{L1} = \beta_{L2} = 4$, $\lambda_{L1} = \lambda_{L2} = 0.001$ with $\rho_{L1} = 0.2$ and $\rho_{L2} = 0.1$ yielded also the best result for the light background when trained over the whole network but with a considerable performance drop: only an average accuracy of 45.14% was obtained. The decrease in accuracy is even more notable for the AE+softmax combination, yielding 22.22% accuracy with parameters $\lambda_{L1} = \lambda_{L2} = 0.001$, $\beta_{L1} = 3$, $\beta_{L1} = 1$, $\rho_{L1} = 0.2$, and $\rho_{L2} = 0.1$. For both network designs we observed a high intra-trial variability resulting in ranges of performances from, for instance, $9.72\% - 87.5\%$ derived for the trials with the best reported network performance. A closer look into the trial performances reveal that worse performance results from the AE+softmax training phase delivering insufficiently discriminative features to the successive layer and consequently backpropagating through the network fails in learning from this input. As this subset is rather small, finetuning cannot compensate for insufficient representations as shown for the NCD set.

From the slight improvements of the stacked autoencoders ($L2$) we did not expect any more performance increase when adding layers. Initial experiments choosing different parameters supported our assumption and thus, no more essential information can be derived using a $L3$ scheme. However, we expect that for the JTD-A the use of stacked autoencoders

TABLE VIII
SPARSITY REGULARIZATION ON JTD-A

Accuracy (%)	AE+softmax		Whole network	
β	mean	median	mean	median
1	30.69	40.81	33.37	43.88
2	32.58	32.79	44.23	44.07
3	35.42	35.93	49.20	**51.51**
4	28.43	28.60	37.22	41.51

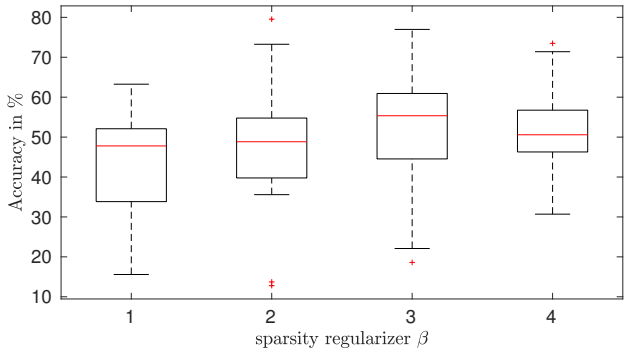

Fig. 7. Boxplot demonstrating the influence of the sparse regularization parameter β on the performance of the final classification results with $\lambda = 0.001$ and $\rho = 0.4$. The red crosses depict outliers.

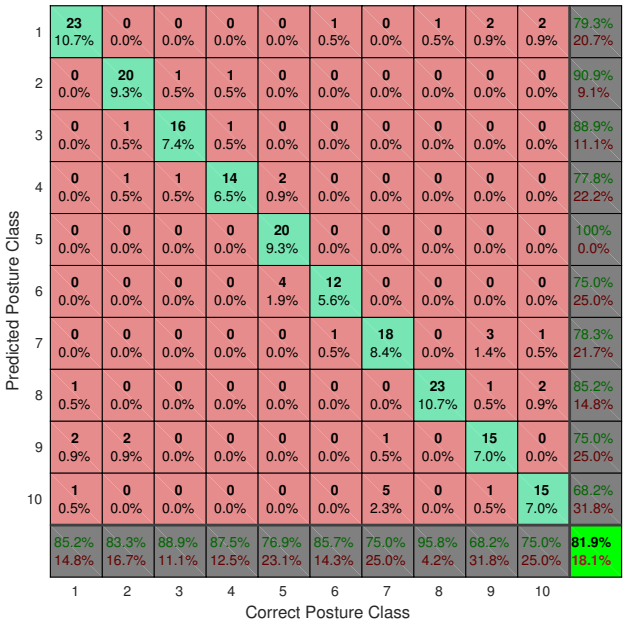

Fig. 8. Confusion matrix for JTD-A using a deep sparse autoencoder ($L3$).

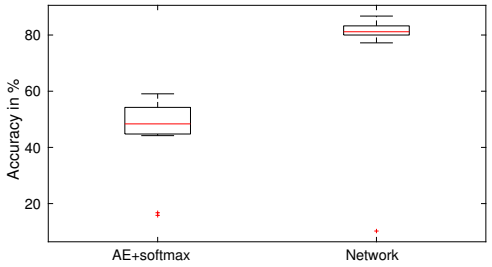

Fig. 9. Performances and variances for the two learning stages in a three-layered autoencoder using the JTD-A posture set. The red crosses depict outliers, showing that the network is error-prone to local minima even when the parameter configurations yield globally good results.

capturing complexities in the images might be beneficial. We will show that our evaluations confirm this assumption.

Our evaluation of the JTD-A set confirms low performances when incorporating AE+softmax only. Similar to the findings for the NCD set, increasing the neuronal firing behavior is beneficial for the classification but with noticeable less improvement. The best performance was achieved for $\rho = 0.4$, yielding a median value of 47.80%. Again, our results show no crucial role of λ on the performance. Increasing the sparsity constraint β led to better performance achieved for $\beta = 3$ with $\rho = 0.4$ (cf. table VIII) and $\lambda = 0.001$, concretely a median of 51.1%.

Among all subsets, the JTD-A dataset is the most challenging set in this study due to merging the three different image background conditions into the training and test sets, and our results show that despite supervised learning the single autoencoder scheme did not yield sufficient feature representions. Additionally, compared to the NCD set, all JTD sets comprise fewer samples while containing 2.5 times more posture classes. However, using a deeper autoencoder network turned out to be beneficial for the JTD-A set. Using the $L3$ design demands careful choice of a suitable backpropagation algorithm. Propagating gradients along a network hierarchy might lead to the well-known problem of vanishing or exploding gradients and thus the optimization procedure on backpropagation learning plays a crucial role. Here, we chose the rprop algorithm [17]. Based on our study observations of the influences of $\{\lambda, \beta, \rho\}$ we only varied the latter. Concretely, we let $\rho_1 = 0.2$ and varied ρ_2 and ρ_3, which showed only performance increase for the AE+softmax stage but not in a significant way comparable to the whole network (cf. Figure 9). A performance of 82.32% was achieved for $\rho_2 = 0.1$ and $\rho_3 = 0.05$, similar to 82.80% for $\rho_2 = 0.15$ and $\rho_3 = 0.1$. To show further support, a confusion matrix sampled from an individual trial on a test set demonstrates an accuracy of 81.9%. This figure is exemplary but representative as space is limited here. It shows that posture 10 was mostly confused with posture 7, which can be explained by the similar hand position (bent hand) but while posture 7 shows only the little finger, posture 10 includes the thumb. This information might have been missing in the classification. Similarly, most confusion is detectable between posture 5 and posture 6, being the most similar postures among all others in the JTD dataset, that is, pointing to the left with either index finger or index and middle finger.

V. STATISTICAL DIFFERENCES AND VISUALIZATIONS

We also investigated the performance difference between the AE+softmax learning stage of the architecture and the whole network performing the network finetuning step.

Our motivation is to investigate whether the AE+softmax learning stage is able to produce a sufficient representation of the input with correspondingly good performance competitive to network finetuning. If this is the case, we can think of an integrating of such learning modules into larger scale application where convolutional training would be too time-consuming. To address the question, we have to evaluate the performance difference between the two stages. Therefore, we conducted the McNemar test which is a statistical test used to evaluate performances of two distinct classifiers under the hypothesis (H_0) that both methods have equal capabilities. The assumption is rejected, when performance results 'significantly' differ which means that one classifier outperforms the other. For the test we assume a significance level of $\alpha = 0.01$. We applied the test on the performance obtained from all trials, i.e. we got 20 responses whether the performances were equally good or differ (in a statistical sense).

In an intra-trial analysis on the performances compared between AE+softmax and the whole network, we see crucial differences for the NCD set as visible in our result section. In detail, neither for $L1$ nor for the $L2$ design learning by AE+softmax performed equally good than the training over the whole network. To be more specific, for $L1$ with parameters $\lambda = 0.001$, $\beta = 4$, and $\rho = 0.2$, which resulted in a median result of 99.94% (average: 95.81%), only one trial achieved the same performance of 99.88%. The distribution of individual results is shown in Figure 10. The test can also be used to identify learning cases where the autoencoder failed to encode sufficiently good representation directly affecting also the final classification. This becomes especially apparent for parameter configurations which show already lower average performance results. One such example is when $\beta = 1$ while keeping the other parameter values as just described. The intra-trial analysis revealed 3 cases, where the autoencoder learned only 1 class representation, which consequently also the subsequent finetuning could not resolve. Figure 11 demonstrates the results and shows, for instance for trial 2, a performance of 28.47% for both learning stages. Interestingly, this fact is not mirrored when referring to the averaged values (cf. table III). Similar results are noticeable for the $L2$ design - in almost all cases within and across trials the finetuning performed superior than the AE+softmax training. However, the graphs also show the impact of insufficient neural encodings of the image for both learning stages (within a range of $26.26\% - 32.15\%$). In such cases, Figure 16 shows exemplary the corresponding representations for JTD-W when learning fails (we obtained similar visualizations for NCD).

For the JTD-W we observed high variations within the individual trials. Figure 12 demonstrates exemplary the distinct accuracy ranges for the $L2$ design. It shows, that for five trials the hierarchical structure fails to produce discriminative features from the input images which impacts the successive training over the whole network. At the same time, trial 8 and trial 14 seemed to achieved reasonable training with corresponding good test performance for both learning stages. In detail, in trial 8 an accuracy of 87.5% was obtained, for trial

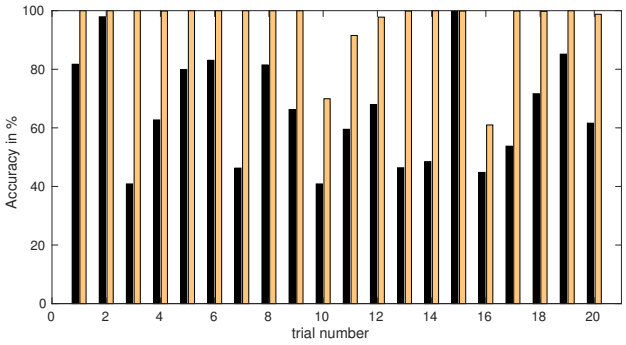

Fig. 10. Comparison of performance results within trials for parameters $\lambda = 0.001$, $\beta = 4$, and $\rho = 0.2$ for the NCD set and $L1$ design. The black bars show the accuracy obtained for training only AE+softmax, the orange bars depict the performance results for the whole network. Only for trial number 15 the classifier achieved similar performance.

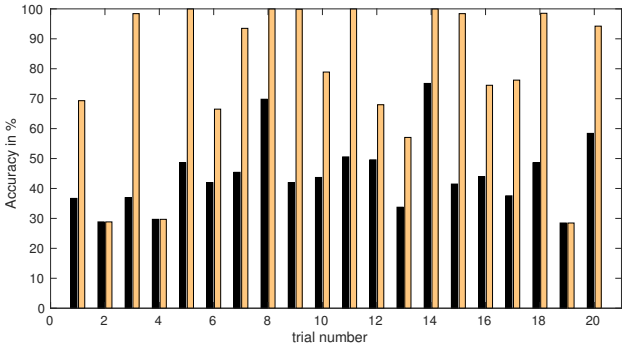

Fig. 11. Comparison of performance results within trials for parameters $\lambda = 0.001$, $\beta = 1$, and $\rho = 0.2$ for the NCD set and $L1$ design. The black bars show the accuracy obtained for training only AE+softmax, the orange bars depict the performance results for the whole network. Trial number 2,4, and 19 show a negative impact of failed autoencoder learning on the final classification.

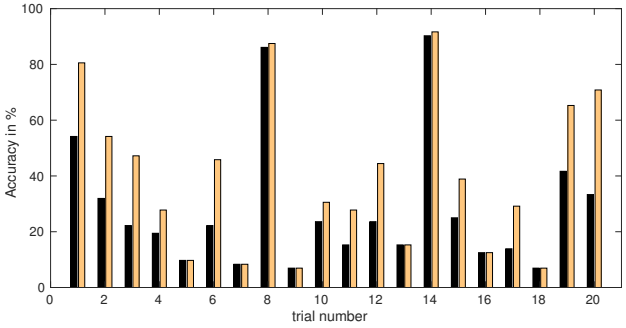

Fig. 12. Comparison of performance results for the JTD-W subset of the individual trials for the $L2$ design. The black bars show the accuracy obtained for training only AE+softmax, the orange bars depict the performance results for the whole network. The graphs depict high variations across trials and 6 cases where the autoencoder stage negatively impacts subsequent network finetuning.

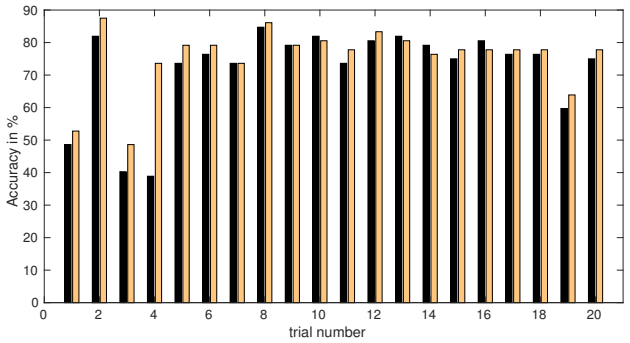

Fig. 13. Comparison of performance results for the JTD-B subset within the individual trials for the $L2$ design. The black bars show the accuracy obtained for training only AE+softmax, the orange bars depict the performance results for the whole network.

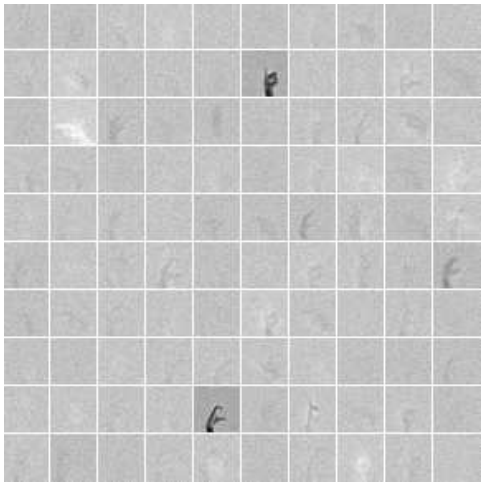

Fig. 14. Visualization of neuronal responses for β =1 and ρ= 0.1 for JTD-B.

14 a value of 90.28% and 91.67%. In other words, there is no statistically significant performance difference between the classifications. Evaluating the classifier performances between the two training stages for $L2$, the performances for 14 out of 20 trials were not significantly different. Considering also the representations, posture contours are more visible in Figure 15 than Figure 14. Our evaluation of these specific trials supports the assumption: while for $\beta = 1$ the AE+softmax achieved only 43.06% accuracy but 72.22% in the final classification, the average performance for $\beta = 4$ increases to 80.56 for AE+softmax and 83.33% when finetuning the network, showing no statistical significant different performance between both learning stages. We find this fact supports that reasonable performance is achievable with shallow autoencoders even for small datasets but with rich classes and omitting heavy label learning.

The differences in the posture set are revealed by the corresponding feature representations: while for the NCD set with only 4 postures the weights show rather clear and uniform representations, it is more challenging to capture all diversity characteristics comprised in the JTD posture set.

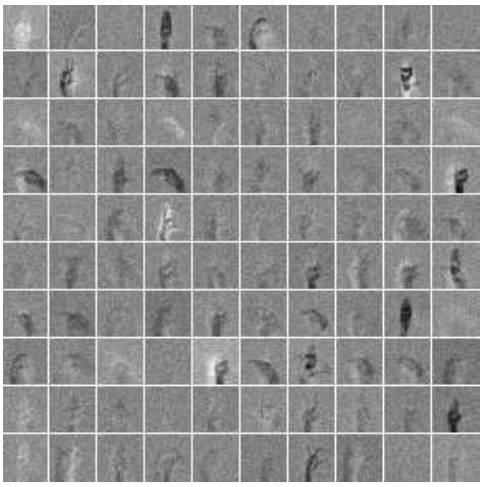

Fig. 15. Visualization of neuronal responses when β= 4 and ρ= 0.1 (JTD-B).

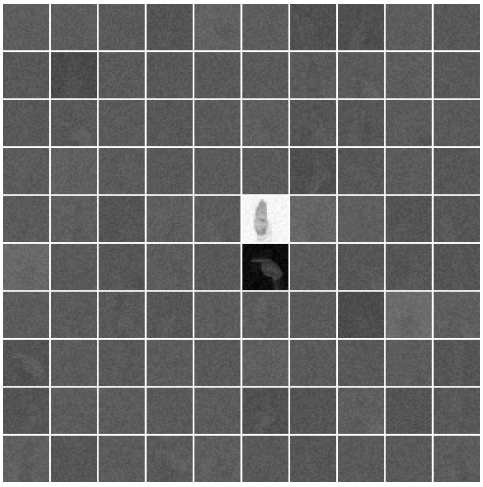

Fig. 16. Example of neuronal responses sampled from a training trial using the JTD-W subset which show insufficient image representations with negative impact on the accuracy in the subsequent classification.

Thus, we hypothesize that the NCD set relies more on a suitable preprocessing as both postures and background are simplistic and only the variations in the image plane must be considered. Our results show, that good performance can be achieved with a simplistic neural architecture.

Our study further demonstrates that much performance power can be squeezed out from the finetuning, which can obfuscate the learning itself in the sense of "what has been learned". This way, the learning reduces to enforcing label matching and supports critical voices in the neural network community that deep neural networks are merely blackboxes.

VI. Discussion and Future Work

Deep neural models have shown superior performance for a number of benchmark datasets for vision and audio data. However, architectural variants, constraints and requirements for successfully training those models boil down the learning part to a network engineering task. Still, training DNNs is

data-hungry, time-consuming, and often deep learning models are treated rather as black boxes. We were questioning this approach and explored instead the power of autoencoders for two specific but distinct posture datasets under the hypothesis that "lightweight" architectures potentially perform equally well and may simplify image feature learning.

In the following, we will highlight both the advantages and downsides of our approach within the context of the work who have introduced the JTD and NCD datasets ([11], [13]) to identify alternatives and improvements on the topic of posture recognition. As the two approaches (EGM and MCNN) differ in their methodology, a direct comparison with the approach presented here would not be legitimate. An introduction of the NCD posture set and the usage of the JTD data for experimental evaluation was performed with a multichannel CNN using both a 2D kernel and a 3D kernel [13]. The authors reported an F1-score on the JTD postures of 77% for both kernel designs. As no further details on how the JTD dataset was splitted, we contrast the results with the accuracy we gained using an $L3$ network on JTD-A of approx. 81%. Although the results improved to over 90% when considering all channels in the MCNN network architecture [13], we find our networks more comparative in the following sense: autoencoders, as used here in the study, do not need a predetermined number of filters and their corresponding sizes, which greatly reduces the number of parameters. Instead, the feature encoding is done in an unsupervised fashion and thus gives an insight into the corresponding input representation as we have exemplary shown in this paper. Also, we showed that for the NCD and the JTD-B as well as JTD-A subset, the performance is competitive to the convolutional approaches. Moreover, a statistical test reveals equal performance capabilities of the AE+softmax network stage as finetuning for the JTD-B set. The extension of traditional CNNs with multiple channels was proposed to enhance the image features [13]. Our evaluation showed that the sparse autoencoders performed superior to the MCNN for the NCD set even with a shallow network design (median accuracy 100%). Our analysis on the reported performances revealed only slight recognition improvements between the 2D and the 3D kernel, which speaks rather in favor of the edge enhancement of the independent channels than the necessity of stackwise image learning. In the light of our present study results, we conclude that, for datasets similar to the NCD dataset, sparse autoencoder networks have the potential to be powerful alternatives to convolutional approaches. A critical discussion point remains open regarding the sparsity firing ρ, which is usually claimed to be small. In our study we could not obtain a certain value for ρ, presumably due to data variations for only a few classes. A deeper analysis and further experiments on this particular dataset remains necessary. Our evaluation on the JTD subsets reveal worse performance for postures with white background than reported for hand model graphs [11]. We assume that low contrasts between the hand and the background negatively influence the learning of image features.

Until today, only a few studies in the domain of posture recognition focus on learning with autoencoders. Studies described in our paper exploited depth images [9] or compared both CNNs and stacked denoising AE demonstrating competitive capability of the latter [10]. In our paper, we showed the influence of different parameter configurations from both the qualitative and quantitative perspective on the performance. By separating the autoencoder with subsequent classifier from the backpropagation training along the whole network, we demonstrated that the first training phase is able to achieve competitive results to network finetuning for certain input data. For specific datasets we reported the optimal parameters and demonstrate their influence on the image feature representations and the classification performance. Here, we also looked into cases where autoencoders show low accuracy. Although this seems counterintuitive to the usual procedure of reporting best results only, our motivation was to investigate also certain network designs and parameter configurations connected to an impact on also poor performance in order to better understand the methodology. Together with the visualization of learnt autoencoder weights (feature representations) we showed both cases when but also when not an autoencoder learns useful representations. This may guide future applications on autoencoder network design on image data.

The evaluation of our study let us promote the integration of (even shallow) autoencoder networks in a modular fashion into larger gesture recognition systems would be highly beneficial when combining static with dynamic gestures as to enlarge the gesture vocabulary. The ease of training and testing of network parameters omitting labels and different filter sizes in contrast to convolutional approaches may be potential candidates to substitute those approaches when processing time is constrained as in HRI scenarios with humanoid robots. Notably, we restricted our experiments to a small image size and achieved superior results on the NCD posture set, which was specifically designed for NAO robot interactions [13]. Our results on this dataset underpin our suggestion employing "lightweight" computational models.

Although we showed that small datasets can yield reasonable performance, our study is limited in the sense that we always assume a hand in the scene. Additional image data to the current datasets to distinguish between presence and absence of a hand per se is desirable to avoid false positive results on the classification.

From our present study, we suggest two major further research directions: first, incorporating easy trainable and slim network architectures for postures into systems for dynamic gestures, which would increase the gesture vocabulary due to the additional access to hand shape and finger configurations. To distinguish both the hand movement and the hand shape in a modular fashion with fast yet robust computational tools would enhance sensible gesture HRI scenarios.

Secondly, it is interesting to unveil the potential of pretraining or transfer learning also for large-scale datasets. Recent advances in the area of generative models (e.g. variational autoencoders) may provide a way to learn variations across postures to account for inter-subject variability without the

effort of creating larger datasets. A special challenge for a deep learning model is evident when considering only images from the JTD dataset with complex backgrounds: it contains too few examples for the number of classes to reasonably train, e.g., CNNs, as originally the evaluation of hand graphs was in the focus. To still benefit from such data, we hypothesized that training autoencoders on the uniform backgrounds would provide us with weights coding the image details, which then blends out the background when applying the trained autoencoders on test images with complex background. Our initial experiments evaluated in a qualitative manner showed indeed segmentation-like effects, and we are currently running experiments to include a quantitative measure on this topic.

VII. Conclusion

Our study contributes to the area of posture recognition by evaluating the performance of differently designed sparse autoencoders for two datasets with distinct characteristics. Although in standard deep learning tasks large datasets are needed, our evaluation revealed good to superior performance in even shallow networks for specific image data. A statistical test on classifier performance demonstrated the effectiveness of a "lightweight" learning scheme for a subset of images. Notably, our results were achieved on a rather small image size, which fosters the robustness of the networks and their applicability in time-critical scenarios. Thus, autoencoders may even substitute past approaches promoting filter learning as in convolutional networks. We explored different parameter configurations and their impact on the recognition performance (quantitative analysis) and the learned representations (qualitative analysis). The corresponding parameter values may guide other researchers conducting similar experiments. Consequently, sparse autoencoders qualify to be an integral part in successive applications in the domain of gesture recognition extended to dynamic hand gestures.

References

[1] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[2] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by V1?" *Vision Research*, vol. 37, no. 23, pp. 3311 – 3325, 1997.

[3] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09. New York, NY, USA: ACM, 2009, pp. 609–616.

[4] L. Bo, X. Ren, and D. Fox, "Unsupervised feature learning for rgb-d based object recognition," in *Experimental Robotics: The 13th International Symposium on Experimental Robotics*, J. P. Desai, G. Dudek, O. Khatib, and V. Kumar, Eds. Heidelberg: Springer International Publishing, 2013, pp. 387–402.

[5] M. Liang and X. Hu, "Recurrent convolutional neural network for object recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[6] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, Jan 2013.

[7] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 568–576.

[8] G. I. Parisi, J. Tani, C. Weber, and S. Wermter, "Lifelong learning of human actions with deep neural network self-organization," *Neural Networks*, vol. 96, no. Supplement C, pp. 137 – 149, 2017.

[9] A. Tang, K. Lu, Y. Wang, J. Huang, and H. Li, "A real-time hand posture recognition system using deep neural networks," *ACM Trans. Intell. Syst. Technol.*, vol. 6, no. 2, pp. 21:1–21:23, 03 2015.

[10] O. K. Oyedotun and A. Khashman, "Deep learning in vision-based static hand gesture recognition," *Neural Computing and Applications*, Apr 2016.

[11] J. Triesch and C. von der Malsburg, "Classification of hand postures against complex backgrounds using elastic graph matching," *Image and Vision Computing*, vol. 20, no. 13, pp. 937 – 943, 2002.

[12] J. P. Jones and L. A. Palmer, "An evaluation of the two-dimensional gabor filter model of simple receptive fields in cat striate cortex." *J Neurophysiol*, vol. 58, no. 6, pp. 1233–1258, Dec 1987.

[13] P. Barros, S. Magg, C. Weber, and S. Wermter, "A multichannel convolutional neural network for hand posture recognition," in *Artificial Neural Networks and Machine Learning - ICANN 2014 - 24th International Conference on Artificial Neural Networks, Hamburg, Germany, September 15-19, 2014. Proceedings*, 2014, pp. 403–410.

[14] K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "Fast inference in sparse coding algorithms with applications to object recognition," *CoRR*, vol. abs/1010.3467, 2010.

[15] D. Arpit, Y. Zhou, H. Ngo, and V. Govindaraju, "Why regularized auto-encoders learn sparse representation?" in *Proceedings of The 33rd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. F. Balcan and K. Q. Weinberger, Eds., vol. 48. New York, New York, USA: PMLR, 20–22 Jun 2016, pp. 136–144.

[16] M. F. Møller, "A scaled conjugate gradient algorithm for fast supervised learning," *Neural Networks*, vol. 6, no. 4, pp. 525 – 533, 1993.

[17] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: The rprop algorithm," in *International Conference on Artificial Neural Networks*, 1993, pp. 586–591.