Learning Siamese Features for Finger Spelling Recognition

Bogdan Kwolek[1] and Shinji Sako[2]

[1] AGH University of Science and Technology, 30 Mickiewicza Av.,
30-059 Krakow, Poland
http://home.agh.edu.pl/~bkw/contact.html
[2] Frontier Research Institute for Information Science, Nagoya Institute of Technology
466-8555, Gokiso-cho, Showa-ku Nagoya, Japan
sako@msp.nitech.ac.jp

Abstract. This paper is devoted to finger spelling recognition on the basis of images acquired by a single color camera. The recognition is realized on the basis of learned low-dimensional embeddings. The embeddings are calculated both by single as well as multiple siamese-based convolutional neural networks. We train classifiers operating on such features as well as convolutional neural networks operating on raw images. The evaluations are performed on freely available dataset with finger spellings of Japanese Sign Language. The best results are achieved by a classifier trained on concatenated features of multiple siamese networks.

Keywords: Finger spelling recognition, Siamese neural networks, CNNs

1 Introduction

Hand gesture based Human-Computer-Interaction (HCI) is one of the most intuitive and versatile ways to achieve user-friendly communication between people and machines [17], similarly to how human interact with each other. Gesture recognition is becoming a central key in HCI due to its widespread applications in virtual reality [18], game control, robotics and assistive technologies for the handicapped and the elderly. Touchless gesture recognition systems are becoming important in automotive user interfaces as they have strong potential to improve safety and comfort of travel. Despite considerable amount of previous work [16], building a robust hand gesture recognition system that is useful for real-life applications remains a challenging problem. Gesture recognition on images from a single camera is very useful, yet difficult task, due to occlusions, differences in hand anatomy, variations of posture appearance, etc. Typically it involves many challenging sub-tasks such as detection of hands and other body parts, motion modeling and tracking, pattern recognition and classification.

While gesture communication may involve motion of all parts of the body, most of the research focuses on arms and hands, which are the most essential in nonverbal communication. Hand gestures can be classified into static hand

gestures that are characterized by hand posture with a particular finger-thumb-palm configuration, and dynamic hand gestures, which can be characterized by their shape, position and movement. Depending on the type of the input data, interpreting hand gestures could be achieved in several ways. This work is devoted to static gesture recognition on images taken by a single camera. In the last decade, several approaches to static gesture recognition on color images were proposed [15]. Earlier it was thought that single camera may not be as effective as stereo or depth cameras. Recent advances in machine learning open new possibilities to achieve robust hand posture and finger spelling recognition using a single color camera. In this work we show that by the use of state-of-the-art methods in machine learning and pattern recognition it is possible to achieve promising results. We demonstrate that high classification rates can be obtained on a dataset consisting of a few thousands of gray images with hand poses.

Recently, Convolutional Neural Networks (CNNs) have become the state-of-the-art tool for object recognition on the basis of color images [10]. Despite high potential of CNNs that has been demonstrated in object recognition, only few papers reports promising results in hand gesture recognition, e.g. a recent survey on hand gesture recognition [15] reports only one significant work, namely [21]. One of the obstacles to broader use of CNNs for gesture recognition is lack of properly aligned datasets of sufficient size as well as absence of robust real-time hand detectors. In [14], a CNN has been utilized in classification of six hand gestures to control robots using colored gloves. In more recent work [1], a CNN has been implemented in Theano and then executed on the Nao humanoid robot. In recently published work [9], a CNN has been trained on one million of exemplars. However, only a subset of data, namely 3361 manually labeled frames into 45 classes of real sign language has been made publicly available.

Finger spelling is a form of sign language, where each sign corresponds to a letter of the alphabet. Tabata and Kuroda [19] proposed a system to recognize hand shapes for finger spelling in Japanese Sign Language (JSL), which they call "Stringlove". Their system uses a custom-made glove, equipped with sensors to capture finger features. The glove contains nine contact sensors and 24 inductcoders that jointly estimate several descriptors such as: adduction/abduction angles of fingers, thumb and wrist rotations, the joint flexion/extension of the fingers and the contact positions among the fingertips of the fingers. In [7], a modified form of the shape matrix that is capable of capturing salience of the finger spelling postures on the basis of precise sampling of contours and regions has been proposed. In [6] recognition of JSL finger spellings is achieved by a CNN, which has been trained on a dataset consisting of 5000 real images, 1000 augmented images and 2000 synthetic images. According to recently published survey [18], the commonly used datasets for hand gestures and postures recognition are: (i) the dataset available in Nanyang Technological University that consists of 200 images of size 352×288, with five gestures and with ten variations for each gesture, (ii) dataset available at National University of Singapore, which consists of 450 data samples. To best of our best knowledge, see also list of available datasets [18], except for our publicly available dataset [6], there is no finger spelling dataset, which could be suitable for learning of deep architectures.

2 Our Approach to Finger Spelling Recognition

In this work, gesture recognition has been realized using both hand-crafted features as well as on the basis of features extracted by the proposed neural network-based algorithm. Both learning of the models as well as experimental evaluation of the proposed algorithm were realized on dataset introduced in [6]. The dataset contains 41 different finger spellings from Japanese Sign Language. The recognition of JSL finger spellings has been performed on gray images of size 64×64.

The hand-crafted features were extracted through vectorization, i.e. converting the input images into column vectors and then reduction of the dimensionality of the vectorized input data. The dimensionality of the vectorized input data has been reduced using the Principal Components Analysis (PCA). The resulting vectors of size 128, consisting of principal components have then been used in training of Random Forests (RF) and Support Vector Machines (SVM).

In experimental evaluations, various convolutional neural networks have been trained on gray images of size 64×64 to select a neural architecture that achieves best classification performance. We investigated relatively simple convolutional neural networks with two convolutional layers as well as more complex architectures with several convolutional layers. A comparison of the learned RF, SVM and CNN multi-class classifiers has been performed in 5-fold cross-validation with respect to classification accuracy, precision, recall and F1-score.

In order to improve the classification performance in the performer independent scenarios, i.e. scenarios in which the recognition of finger spelling is done using models that had been learned without data samples of evaluating performer, we employed siamese neural networks. Siamese neural networks have been used to learn low dimensional embedding of the input data. The embedded data were stored in vectors of size 128 and then used to learn RF and SVM classifiers. CNN-based siamese neural networks were learned and evaluated in performer independent scenarios with respect to classification performance.

Next, low-dimensional embeddings, that were determined by two independently trained siamese networks were concatenated, and then used to train a SVM classifier. We also trained and evaluated SVM classifiers using low-dimensional features that were produced by three independently trained siamese networks. The experimental results showed that a SVM-based classifier operating on the concatenated low-dimensional embeddings outperforms both the CNNs operating on raw images as well as classifiers operating on embeddings from single siamese neural network.

The Siamese neural network (SNN) has been proposed in [4] to obtain an embedding of the input data so that Euclidean proximity between features in the embedding space implies that the represented data are semantically similar. In [3], a non-linear similarity metric has been learned using a siamese neural network in order to improve the recognition of gestures. The recognition has been realized on the basis of data acquired by an inertial measurement unit (IMU). The authors demonstrated experimentally that the SNN outperforms conventional MLP and Dynamic Time Warping-based similarity metrics on a dataset with 18 gestures, which were performed by 22 individuals.

3 Siamese Neural Networks

Siamese neural network is a special type of neural network that consists of two identical sub-networks sharing the parameters [4]. In contrast to classical MLPs that use loss functions comparing the outputs with the target values, the SNNs use objectives that compare the feature vectors of pairs of the exemplars. Moreover, in contrast to ordinary MLPs, in which the number of outputs is typically the same as the number of the classes in the classification task, in the siamese network the dimension of the target space can be specified with respect to the considered problem. Sharing weights across the sub-networks results in smaller number of learned parameters, which in turn means less data required and less tendency to overfitting.

The central idea behind SNNs is to learn an embedding, where similar image pairs (i.e. images representing the same hand posture) are close to each other and dissimilar image pairs (i.e. images representing different hand postures) are separated by a distance that depends on a parameter called margin. Let us assume that we want to learn a SNN on the basis of a training set consisting of images with hands $\{x_i, y_i\}_{i=1}^{N}$, where $x_i \in \mathbb{R}^n$ and y_i are class labels. The siamese network produces a feature embedding $f(x, \theta_f)$ that is defined as $f : \mathbb{R}^n \times \mathbb{R}^k \to \mathbb{R}^m$, where $\theta_f \in \mathbb{R}^k$ stands for parameters of the network. The parameters θ_f consists of all the weights and biases of the network. For deep convolutional neural networks with several layers the number of parameters can be up to 100 M values. The goal of the learning is to seek for the parameter vector θ_f such that the embedding produced through f has desirable properties, and in particular it places similar examples nearby. The SNN can extract information from the available data and determine such an embedding without requiring specific information about the object or image categories. Thus, learning siamese network is achieved in a weakly-supervised scheme using only pairs of data instances labeled as matching or non-matching.

Let us consider a pair of images (I_p, I_q) that contains views of the same object, and a pair of images (I_p, I_r), which are views of different object categories. The siamese neural network maps such image pairs into embedded representations (x_p, x_q, x_r) such x_p and x_q are close, whereas x_p and x_r are further apart. Embedding with such properties can be achieved using a contrastive loss function L, which expresses how well the function f is capable of placing similar image representations in the close proximity and keep dissimilar image representations distant. The contrastive loss function $L(\theta)$ has the following form:

$$L(\theta) = \underbrace{\sum_{(x_p, x_q)} L_q(x_p, x_q)}_{\substack{\text{penalty term} \\ \text{for similar images}}} + \underbrace{\sum_{(x_p, x_r)} L_r(x_p, x_r)}_{\substack{\text{penalty term} \\ \text{for dissimilar images}}} \tag{1}$$

where

$$\begin{aligned} L_q(x_p, x_q) &= \|x_p - x_q\|_2^2 \\ L_r(x_p, x_r) &= \max\left(0, m - \|x_p - x_r\|_2\right)^2 \end{aligned} \tag{2}$$

The penalty term L_q penalizes the pair (x_p, x_q) that is too apart, whereas L_r penalizes the pair (x_p, x_r) that is closer than a margin m. Thus, dissimilar image pairs contribute to the loss function if their distance is within the margin m. Given the label y indicating whether the images are similar or dissimilar the margin-based loss function can be expressed as follows:

$$L(\theta, x_i, x_j) = y \left\| f(x_i) - f(x_j) \right\|_2^2 + (1 - y) \max(0, m - \left\| f(x_i) - f(x_j) \right\|_2)^2 \quad (3)$$

where function f performs a forward propagation through one of the siamese sub-networks. The loss function penalizes positive pairs by the squared Euclidean distances and negative pairs by the squared difference between the margin m and Euclidean distance for pairs having distance less than the margin m.

In addition to the above discussed contrastive loss function, relevant functions employ non-hinge based pairwise loss functions such as log-sum-exp and cross-entropy on the similarity values, which softly reinforce similarities to be high for positive values and low for negative values, e.g. [20,22]. In order to prevent overfitting as a result of collapsing all positive pairs, a double-margin contrastive loss, which drops to zero for positive pairs as long as their distances fall beyond the second (smaller) margin, has been proposed in [13].

Siamese neural networks were used in several applications. In [22], a siamese convolutional neural network architecture for metric learning has been presented. In the discussed work, person re-identification is achieved on the basis of three independent convolutional networks, which act on three overlapping parts of the input image pairs. Each part-specific network has two convolutional layers with max pooling, followed by a fully connected layer. The fully connected layer yields an output vector for each input image, and two output vectors are compared using a cosine loss function. The cosine outputs for each of the three sub-images are then combined to get a final similarity score. The original siamese network was extended in a recent work [2]. In the extended network the softmax loss functions are also incorporated to predict the object categories of the input images. It has been shown that such an additional softmax loss can also help to prevent undesirable overfitting that may take place in case of using only the contrastive loss function.

Once the SNN is trained, we have in disposal a feature vector representing a similarity measure of a set of samples. Any classifier can be utilized on these feature vectors. In Section 4 we overview the classifiers that were used to classify such embedded feature vectors as well as vectors with hand-crafted features. The convolutional neural networks have achieved state-of-the-art results in a variety of tasks. One possible reason is that their hierarchical, layered structure may allow them to capture the geometric and structural regularities of commonplace data [12,11]. Having the above on regard, the siamese architectures for recognition of finger spellings were built on multi-layer CNN networks. The utilized siamese architecture along with the utilized CNN are described in Section 5.

4 Feature Classification

4.1 k-Nearest Neighbors

The principle behind k-Nearest Neighbors (k-NN) methods is to seek a predefined number of training samples closest in a distance to the classified example, and then to predict the label from these. This means that the k-NN algorithm does not learn anything from the training data and just utilizes the training data itself for the classification.

4.2 Random Forests

A random forest (RF) is an ensemble of unpruned decision trees. A RF model is usually made up of tens or hundreds of decision trees. Each decision tree is built on the basis of a random subset of the training dataset. The random forests algorithm is very much like the bagging algorithm, which constructs a large number of trees using bootstrap samples from a dataset. Unlike in bagging, in RF only a subset of k features is selected at random out of the total K features to split each node in a tree, unlike in bagging where all K features are considered in each node for a split. The randomness introduced by the random forest model builder makes the resulting algorithm robust to noise, outliers, and over-fitting, when compared to a single tree classifier. Such model can deliver competitive results in regression or classification tasks, particularly in case of very large training datasets and a very large number of input variables (hundreds or even thousands of input variables). It can be very competitive with nonlinear classifiers such as neural nets and support vector machines. It is worth noting that its performance is often dataset dependent. Taking into account the discussed advantages of RF-based classifiers, we employed them to recognize finger spellings on JSL dataset, which consists of considerable number of data samples.

4.3 Support Vector Machines

The basic idea of Support Vector Machines (SVM) based classification is to determine a separating hyperplane that corresponds to the largest possible margin between the points of different classes [5]. The optimal hyperplane for an SVM means the one with the largest margin between the two classes, so that the distance to the nearest data point of both classes is maximized. Such a largest margin means the maximal width of the tile parallel to the hyperplane that contains no interior data points and thus incorporating robustness into decision making process. Given a set of datapoints $\mathcal{D}$: $\mathcal{D} = \{(x_i, y_i)|x_i \in \mathbb{R}^p, y_i \in \{-1, 1\}\}_{i=1}^n$, where each example x_i is a point in p-dimensional space and y_i is the corresponding class label, we search for vector $\omega \in \mathbb{R}^n$ and bias b, forming the hyperplane H: $\omega^T x + b = 0$ that seperates both classes so that: $y(\omega^T x + b) \geq 1$. The optimization problem that needs to be solved is: $min_{\omega,b} \frac{1}{2}\omega^T\omega$ subject to: $y(\omega^T x + b) \geq 1$. The problem consists in optimizing a quadratic function subject to linear constraints, and can be solved with an off-the-shelf QP solver.

5 Neural Networks and Learning

At the beginning we present convolutional neural network that has been used for classification of the finger spellings as well as a base network of the siamese neural network. Afterwards, we present the siamese neural network that has been employed for learning of the low-dimensional embeddings. Finally, we describe how low-dimensional embeddings were employed to train the finger spelling classifiers.

5.1 Convolutional Neural Network

The convolutional neural network processes gray images of size 64 x 64. The first convolutional layer consists of 16 filters of size 5 x 5. The second convolutional layer comprises 32 filters of size 3 x 3 and is followed by maxpool layer. The next two layers are convolutional layers with 32 and 64 filters of size 3 x 3, respectively, which are followed by maxpool layer. This layer is followed by a convolutional layer with 64 filters of size 3 x 3, which in turn is followed by a maxpool layer. The flattened output of this layer is fed to three fully connected layers. The number of neurons in the first two fully connected layers is equal to 128. Between the fully connected layers the dropout is executed. The last layer is soft-max with 41 outputs. The convolution and dense layers apply ReLU activation function. The number of trainable parameters of the discussed network is equal to 279 945. The categorical crossentropy objective has been optimized using stochastic optimization [8].

5.2 Siamese Neural Network

The siamese neural network consists of two branches of convolutional neural networks. The CNNs have the same structure as CNN presented in Section 5.1, except that the last output layer has been excluded. The number of neurons in the output layer has been set to 128. This means that the siamese neural network produces 128-dimensional embeddings of the input data. Such low-dimensional embeddings were calculated for every image from the training part of the dataset. They were then utilized to train classifiers responsible for recognition of JSL finger spellings.

5.3 Recognition of JSL Finger Spellings on the Basis of Low-dimensional Embeddings

The low-dimensional embeddings were used to train the classifiers discussed in Section 4. Since the SVM classifier achieved superior results in comparison to remaining classifiers, it has been selected to perform recognition of JSL finger spellings on the basis of low-dimensional embeddings. The first SVM classifier has been trained on low-dimensional embeddings, which were determined by single siamese neural network. The second SVM classifier has been trained on

concatenated low-dimensional embeddings, which were calculated by the siamese neural networks, and which were learned on the same data with unlike initializations. This means that the SVM has been trained on feature vectors of size 2×128. We also trained a SVM classifier on embeddings determined by three siamese networks.

6 Gesture Dataset

In general, in order to learn high-quality data model and to achieve promising results on the basis of a convolutional neural network it is necessary to prepare a consistent dataset consisting of sufficient number properly selected and aligned images. For training of siamese neural networks, the labeled image pairs are needed. Collection of such set of image pairs is a non-trivial task. In this work, both training of the neural networks as well as the experimental evaluations have been performed on JSL finger spelling dataset [6]. The JSL words are articulated by five fingers of the single hand and the direction of the hand. Most of finger spellings are expressed through static postures. In addition, dullness, half dullness, and long sound are expressed by dynamic hand postures. In this work we focus on 41 static finger spellings from the JSL, see Fig. 1 that depicts the example JSL finger spellings.

Fig. 1: Sample images form JSL dataset.

Training and evaluation of the convolutional neural networks as well as the classifiers operating on hand-crafted features has been performed on 5000 real images from the JSL dataset. In the person independent scenario the dataset has been divided into training and test part. The test part consists of 579 images with the finger spellings, which were expressed by person #10. All experiments were performed on gray images of size 64×64.

7 Experimental Results

At the beginning of this Section we overview the measures of the performance. Afterwards, we present results that were obtained in the experimental evaluation.

7.1 Performance Measures

The performance of the finger spelling classifier has been evaluated with respect to precision, recall, F1-score and accuracy. Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. It has been calculated as follows:

$$precision = \frac{TP}{TP + FP} \tag{4}$$

where TP denotes number of correctly predicted positive values (true positives), whereas FP (false positive) denotes number of incorrectly identified examples. High precision relates to the low false positive rate. A precision score of 1.0 for a class C indicates that every sample labeled as belonging to class C does indeed belong to the class C. Recall and F1-score were calculated in the following manner:

$$recall = \frac{TP}{TP + FN} \tag{5}$$

$$F1\text{-}score = \frac{2 * recall * precision}{recall + precision} \tag{6}$$

where FN stands for the number of incorrectly classified as not the class of interest. The F1-score reaches best value at 1 and worst score at 0 and can be interpreted as a weighted harmonic mean of the precision and recall. The accuracy is defined as the proportion of true responses (both TN - number of examples that are correctly classified as negative and TP) among the total number of cases examined. It has been calculated as follows:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}$$

It expresses the ratio of correctly classified instances.

7.2 Evaluation of Classification Performance

At the beginning we evaluated the classification performance of k-NN, RF, and SVM classifiers on hand-crafted features. The classification of the finger spellings has been realized on features extracted by PCA. The number of the principal components has been set to 128. Table 1 presents experimental results that were obtained on the JSL dataset in the 5-fold cross-validation. As we can notice, the

Table 1: Classification performance measures achieved by PCA+RF, PCA+SVM and CNN in 5-fold cross-validation.

	Accuracy	Precision	Recall	F1-score
PCA+k-NN	0.852	0.889	0.852	0.861
PCA+RF	0.882	0.921	0.882	0.893
PCA+SVM	0.901	0.938	0.901	0.912
CNN	0.961	0.944	0.867	0.892

SVM classifier achieved the best classification performance. The parameters of RF and SVM were determined in a grid search. Afterwards, we trained a CNN (c.f. Section 5.1). As we can observe in Tab. 1, it achieves the best classification performance.

In the second stage of the experiments we conducted evaluations in performer independent scenarios. As we can notice in Tab. 2, the classification performance achieved by the convolutional neural network on such data is worse in comparison to classification performance achieved in the 5-fold cross-validation by the CNN, see Tab. 1. An ensemble of three CNNs with the same architecture gives slightly better results, cf. classification performance in the second row of Tab. 2.

Table 2: Classification performance in performer independent experiment: first row - performance measures obtained by CNN, second row - performance attained by ensemble of three CNNs, third row - performance achieved by SVM operating on embedded features, which were obtained by single siamese CNN, fourth row - performance obtained by SVM operating on embedded features that were extracted by two siamese CNNs.

	Accuracy	Precision	Recall	F1-score
CNN	0.753	0.792	0.754	0.730
CNN - ensemble	0.769	0.800	0.771	0.742
one Siamese+SVM	0.743	0.736	0.746	0.722
two Siamese+SVM	0.786	0.800	0.791	0.781

The results obtained by the SVM classifier operating on the embeddings, which were extracted by single CNN-based siamese network are slightly worse in comparison to results achieved by the CNN as well as the ensemble of the CNNs. The SVM classifier operating on concatenated features from two siamese convolutional networks achieves better results with respect to results attained

by the CNN as well as the ensemble of CNNs. The classification accuracy achieved by the SVM operating on features obtained by three siamese convolutional networks was better about 0.5% with regard to classification performance attained by the SVM operating on concatenated features from two siamese convolutional networks. The discussed results are the averages of the results that were obtained in five trials. The siamese neural network has been trained on 4421×2 pairs of images. For every training image we sampled the positive image from the same class as well negative image belonging to unlike class. The trained model and source code for evaluating of the classification accuracy of the discussed finger spelling classifiers is available at: `http://home.agh.edu.pl/~bkw/research/data/acivs`.

The data preprocessing has been realized in Matlab. The data preprocessed in such a way were imported into python. The learning of the neural network has been performed in python using theano/keras framework. The learning of the neural network has been accomplished with GPU support using tensorFlow backend. All computations were realized on a PC equipped with an NVIDIA graphics card and running Windows 7.

8 Conclusions

In this work we proposed an approach for finger spelling recognition on the basis of learned low-dimensional embeddings. The embeddings are obtained by siamese-based convolutional neural network. We presented experimental results that were attained on freely available JSL dataset using both hand-crafted and learned features. The experimental results show that a SVM classifier operating on low-dimensional embeddings, which were obtained by two and three siamese neural networks achieves better results in comparison to SVM operating on features obtained by single siamese neural network, convolutional neural network operating on raw images as well as ensemble of three CNNs. Future work will focus on extending the JSL dataset about new data samples as well as data augmentation.

Acknowledgment. This work was supported by Polish National Science Center (NCN) under a NCN research grant 2014/15/B/ST6/02808 as well as JSPS KAKENHI Grant Number 17H06114 and 15KK0008.

References

1. Barros, P., Magg, S., Weber, C., Wermter, S.: A Multichannel Convolutional Neural Network for Hand Posture Recognition, pp. 403–410. Springer, Cham (2014)
2. Bell, S., Bala, K.: Learning visual similarity for product design with convolutional neural networks. ACM Trans. Graph. 34(4), 98:1–98:10 (2015)
3. Berlemont, S., Lefebvre, G., Duffner, S., Garcia, C.: Siamese neural network based similarity metric for inertial gesture classification and rejection. In: 11th IEEE Int. Conf. and Workshops on Automatic Face and Gesture Recognition (FG). pp. 1–6 (2015)

4. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: Proc. of the IEEE Computer Society Conf. on Computer Vision and Pattern Rec. pp. 539–546 (2005)
5. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. 20(3), 273–297 (1995)
6. Hosoe, H., Sako, S., Kwolek, B.: Recognition of JSL finger spelling using convolutional neural networks. In: Fifteenth IAPR Int. Conf. on Machine Vision Applications (MVA). pp. 85–88. IEEE, Nagoya, Japan (2017)
7. Kane, L., Khanna, P.: A framework for live and cross platform fingerspelling recognition using modified shape matrix variants on depth silhouettes. Comput. Vis. Image Underst. 141, 138–151 (2015)
8. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. CoRR (2014)
9. Koller, O., Ney, H., Bowden, R.: Deep hand: How to train a CNN on 1 million hand images when your data is continuous and weakly labelled. In: IEEE Conf. on Comp. Vision and Pattern Rec. pp. 3793–3802 (2016)
10. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS, pp. 1097–1105 (2012)
11. Kwolek, B.: Face Detection Using Convolutional Neural Networks and Gabor Filters, pp. 551–556. Lecture Notes in Computer Science, vol. 3696, Springer (2005)
12. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. In: Proc. of the IEEE. pp. 2278–2324 (1998)
13. Lin, J., Morère, O., Chandrasekhar, V., Veillard, A., Goh, H.: Deephash: Getting regularization, depth and fine-tuning right. CoRR (2015)
14. Nagi, J., Ducatelle, et al., F.: Max-pooling convolutional neural networks for vision-based hand gesture recognition. In: IEEE ICSIP. pp. 342–347 (2011)
15. Oyedotun, O.K., Khashman, A.: Deep learning in vision-based static hand gesture recognition. Neural Computing and Applications pp. 1–11 (2016)
16. Pisharady, P., Saerbeck, M.: Recent methods and databases in vision-based hand gesture recognition. Comput. Vis. Image Underst. 141, 152–165 (2015)
17. Rautaray, S.S., Agrawal, A.: Vision based hand gesture recognition for human computer interaction: A survey. Artif. Intell. Rev. 43(1), 1–54 (2015)
18. Sagayam, K.M., Hemanth, D.J.: Hand posture and gesture recognition techniques for virtual reality applications: a survey. Virtual Reality 21(2), 91–107 (2017)
19. Tabata, Y., Kuroda, T.: Finger spelling recognition using distinctive features of hand shape. In: Int. Conf. on Disability, Virtual Reality and Associated Technologies with Art Abilitation. p. 287292 (2008)
20. Taigman, Y., Yang, M., Ranzato, M., Wolf, L.: Deepface: Closing the gap to human-level performance in face verification. In: Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition. pp. 1701–1708. CVPR '14 (2014)
21. Tompson, J., Stein, M., LeCun, Y., Perlin, K.: Real-time continuous pose recovery of human hands using convolutional networks. ACM Trans. Graph. 33(5) (2014)
22. Yi, D., Lei, Z., Li, S.Z.: Deep metric learning for practical person re-identification. CoRR (2014)