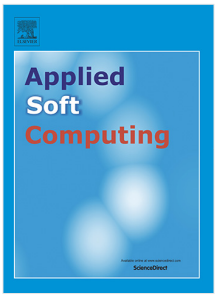
Accepted Manuscript



A convolutional neural network with feature fusion for real-time hand posture recognition

Sérgio F. Chevtchenko, Rafaella F. Vale, Valmir Macario, Filipe R. Cordeiro

PII: S1568-4946(18)30527-1
DOI: https://doi.org/10.1016/j.asoc.2018.09.010
Reference: ASOC 5089

To appear in: *Applied Soft Computing Journal*

Received date : 2 February 2018
Revised date : 2 September 2018
Accepted date : 12 September 2018

Please cite this article as: S.F. Chevtchenko, et al., A convolutional neural network with feature fusion for real-time hand posture recognition, *Applied Soft Computing Journal* (2018), https://doi.org/10.1016/j.asoc.2018.09.010

A Convolutional Neural Network with Feature Fusion for Real-Time Hand Posture Recognition

Sérgio F. Chevtchenko[a,*], Rafaella F. Vale[b], Valmir Macario[a], Filipe R. Cordeiro[a]

[a] *Departamento de Computação, Universidade Federal Rural de Pernambuco, Rua Dom Manoel de Medeiros, s/n, Dois Irmãos, 52.171-900, Brazil*
[b] *Centro de Informática, Universidade Federal de Pernambuco, Av. Jornalista Aníbal Fernandes, s/n, Cidade Universitária, 50.740-560, Brazil*

Abstract

Gesture based human-computer interaction is both intuitive and versatile, with diverse applications such as in smart houses, operating theaters and vehicle infotainment systems. This paper presents a novel architecture, combining a convolutional neural network (CNN) and traditional feature extractors, capable of accurate and real-time hand posture recognition. The proposed architecture is evaluated on three distinct benchmark datasets and compared with the state-of-the-art convolutional neural networks. Extensive experimentation is conducted using binary, grayscale and depth data, as well as two different validation techniques. The proposed feature fusion-based convolutional neural network (FFCNN) is shown to perform better across combinations of validation techniques and image representation. The recognition rate of FFCNN on binary images is equivalent to grayscale and depth when the aspect ratio of gestures is preserved. A real-time recognition system is presented with a demonstration video.

Keywords: hand postures, convolutional neural networks, deep learning, hyperparameter selection.

[☆]This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

[*]Corresponding author. Tel.: +55 81 3320-6491
Email addresses: sergio.chevtchenko@ufrpe.br (Sérgio F. Chevtchenko), rfv@cin.ufpe.br (Rafaella F. Vale), valmir.macario@ufrpe.br (Valmir Macario), filipe.rolim@ufrpe.br (Filipe R. Cordeiro)

1. Introduction

Human communication involves multiple elements that can work as channels for conveying information. It is argued that some of the most common aspects of human communication, including speech [1] and gestures [2, 3], are essential for a more natural interaction between humans and machines. With these aspects, human-computer interaction (HCI) can benefit from both verbal and non-verbal forms of communication. Static hand gestures, in particular, which are also known as hand postures, can be used to provide a more intuitive interaction with computers [2] and are the focus of this paper.

Hand gesture recognition can serve to accommodate groups with specific needs. Some works are concerned with the deaf or hearing impaired [4, 5], while others provide comments that address the issue of remote control of home appliances by elderly people and people who suffer from physical disabilities or are bedridden [6, 7, 8, 9]. Hand gestures in generic human-computer interfaces can also benefit these groups [10]. There is also evidence pointing to the feasibility of the use of touch and gesture input technologies by older people, provided that these interfaces are properly adapted to suit their needs and skills [11]. This could potentially include other gesture-based technologies such as hand posture recognition systems.

This type of system can also be useful in places with critical sanitary conditions, such as operating theaters and kitchens. Specifically in the former and other related situations, gesture control can reduce the duration of procedures that demand sterile conditions by minimizing the need for contact with non-sterile input devices like mice, keyboards or touch screens [12], lowering the probability of contamination. As suggested by Wachs et al. [13], a gesture interface in an operating theater can increase efficiency by preventing change of location and loss of focus due to ease of use, and by providing fast reaction, in addition to a sterile interaction. In the studies of Johnson et al. [12] and O'Hara et al. [14], the application of interaction methods without physical contact with control devices is investigated in the context of image-guided interventional radiology and surgical procedures, respectively.

Hand gestures have other potential applications. These include an interface for manipulating objects in virtual environments [15], controlling multimedia devices [16] and video games [17], as well as possibly acting as a less distracting user interface for infotainment systems in vehicles [18, 19]. The

last example is a consequence of the reduced hand-eye coordination required from the user compared to haptic controls [20]. Control of industrial and commercial robots brings other possible applications of such systems, e.g. using gestures to give directions to robots [21, 22] or to remotely control their movements [23].,

A good recognition method must aim for high accuracy and in most of the aforementioned applications, including others involving hand posture recognition, real-time capability is also desirable for providing fast feedback for the user. Deep neural networks, like convolutional neural networks (CNNs), have gained attention for performing well in pattern recognition compared to shallow networks and other methods [24]. While CNNs are a state-of-the-art method for several image recognition problems, they usually depend on massive parallel computation for training and deployment. Nevertheless, recently a smaller and less computationally expensive CNN has been proposed for object recognition on mobile devices [25]. Furthermore, it is possible to train a small convolutional neural network dedicated to a specific task, such as hand posture recognition, as described by Oyedotun and Khashman [26], Nasr et al. [27] and Ji et al. [28].

The purpose of this paper is to propose and evaluate a novel combination of classical feature descriptors and a convolutional neural network for fast and accurate hand posture recognition. Gabor features, Zernike moments, Hu moments and contour based descriptors are used to increase the diversity of information available in a convolutional neural network.

Extensive experimentation is conducted with three benchmark datasets: Massey (2,515 images) [29], JaRED (81,000 images) [30] and OUHANDS (3,000 images) [31]. The proposed novel classifier is compared with the state-of-the-art in terms of recognition accuracy and speed. It is shown to provide better accuracy across the datasets. Additionally, the aspect ratio of hand images is shown to impact recognition accuracy when binary representation is used. Finally, a real-time gesture recognition system based on the proposed scheme is implemented with a 3D RealSense^{TM} camera. While depth frames are used to obtain a robust segmentation, the proposed classifier is not dependent on a 3D camera, as it is trained on binary images. Furthermore, the proposed feature fusion-based architecture is shown to outperform state-of-the-art models on binary, depth and grayscale images. A demo video is provided in Appendix A.

The rest of this paper is structured as follows. Section 2 contextualizes the problem addressed in this study, and presents the related literature in . In

Section 2.2, the feature descriptors implemented in this work are introduced, along with the architecture of the proposed convolutional neural network. Experimental setup and evaluation details, including datasets and the design rationale behind implemented architecture models, are given in Section 4. Section 6 summarizes the main results and contributions of this paper and raises ideas for future work.

2. Background

This section presents a general introduction to hand posture recognition and convolutional neural networks.

2.1. Hand posture recognition

In a global overview of hand gesture recognition systems, the fundamental operational steps consist in acquiring the image, detecting and segmenting the hand, tracking the hand and, finally, classifying the gesture [3]. However, when the recognition system can operate at image acquisition rate, tracking is not necessary [3]. Thus, image acquisition, detection and segmentation, along with feature extraction and classification are the main steps that comprise a hand posture recognition system [32].

The hand segmentation task is challenging due to variability of the background, and is usually treated separately from feature extraction and posture classification. In order to mitigate this difficulty, some posture recognition systems rely on markers or specially colored gloves [28]. With advances in portable 3D cameras, such as Kinect™ and RealSense™, it is now possible to improve segmentation and recognition accuracy in uncontrolled environments, such as with unpredictable lightning or skin-colored objects [33, 34]. Since depth sensors are affected by lighting conditons, fusion of color and depth data can also improve segmentation by overcoming the limitations of each modality [35].

Following the acquisition and segmentation steps, the feature selection phase is in charge of extracting a representative feature set that will be fed to a classifier. While relevant features can be selected by hand, it is also possible to train the feature extractor simultaneously with the classifier. Such an approach has been successfully applied in hand posture recognition with convolutional neural networks, in which convolutional filters are trained in a similar manner to a multilayer perceptron [26, 27, 28, 36, 37].

2.2. Feature extraction

Some commonly used feature extraction methods are employed as shown in the following subsections. These features are integrated into a convolutional neural network in order to provide additional information about the image.

2.2.1. Hu moments

The set of Hu moment invariants [38] is one of the oldest and best-established image descriptors. They remain roughly constant under scale, rotation and translation. The seven moments used in this paper are extracted from a binary image and are defined in Equations (1)–(7):

$$I_1 = \eta_{20} + \eta_{02}, \tag{1}$$

$$I_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2, \tag{2}$$

$$I_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \tag{3}$$

$$I_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2, \tag{4}$$

$$\begin{aligned} I_5 = {} & (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\ & + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2], \end{aligned} \tag{5}$$

$$\begin{aligned} I_6 = {} & (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\ & + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}), \end{aligned} \tag{6}$$

$$\begin{aligned} I_7 = {} & (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\ & - (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 \\ & - (\eta_{21} + \eta_{03})^2] \end{aligned} \tag{7}$$

where η_{ij} are scale invariant moments, as defined by Hu [38].

2.2.2. Zernike moments

The Zernike polynomials were first proposed in 1934 by Frits Zernike [39]. The corresponding moments are known to be invariant to rotation, and can be modified to also be stable under scale and translation [40].

The complex ZM of a 2D image of order n and repetition m, over an intensity image $f(x, y)$, is

$$Z_{n,m} = \frac{n+1}{\pi} \sum_x \sum_y f(x, y) V_{nm}(x, y)^*,$$
$$n - |m| \text{ even}, \quad |m| \leq n, \tag{8}$$

where $x^2+y^2 \leq 1$, so the original image coordinates are scaled for a unit disk $x^2 + y^2 = 1$, and $*$ denotes the complex conjugate. The Zernike polynomial $V_{nm}(x, y)$ is defined as follows:

$$V_{nm}(x, y) = V_{nm}(r, \theta) = R_{nm}(r)e^{jm\theta}, \tag{9}$$

$$R_{nm}(r) = \sum_{s=0}^{\frac{n-|m|}{2}} (-1)^s \frac{(n-s)!}{s!\left(\frac{n+|m|}{2}-s\right)!\left(\frac{n-|m|}{2}-s\right)!} r^{n-2s}, \tag{10}$$

where j is the imaginary unit and $0 \leq r \leq 1$. The normalized image coordinates (x, y) transformation to the domain of the unit circle (r, θ) is given by the following equations:

$$r = \sqrt{x^2 + y^2}, \quad \theta = \tan^{-1}\left(\frac{y}{x}\right) \tag{11}$$

See Tahmasbi et al. [41] for a more detailed explanation on feature extraction using Zernike moments. Since $Z_{n,-m} = Z_{n,m}$ and therefore $|Z_{n,-m}| = |Z_{n,m}|$, only $|Z_{n,m}|$ is used for the feature vector. Also, $|Z_{0,0}|$ and $|Z_{1,1}|$ are the same for all normalized images and are not used. A Zernike feature vector of order n is formed by concatenating all moments from order 2 to n.

2.2.3. Gabor filter

The response of this filter is considered to be a representation of simple visual cells in mammals [42], which encode both spatial and frequency variables of an image. It is widely used to recognize texture features by convolving an image with the filter kernel [43]. A two-dimensional Gabor filter is given by:

$$G(x, y) = \exp\left(-\frac{1}{2}\left[\frac{x_\theta^2}{\sigma_x^2} + \frac{y_\theta^2}{\sigma_y^2}\right]\right)\cos\left(\frac{2\pi x_\theta}{\lambda} + \psi\right) \tag{12}$$

$$x_\theta = x\cos(\theta) + y\sin(\theta) \tag{13}$$

$$y_\theta = -x\sin(\theta) + y\cos(\theta) \tag{14}$$

$$\sigma_x = \sigma, \quad \sigma_y = \frac{\sigma}{\gamma} \tag{15}$$

The parameters of this filter are:

- standard deviation of the Gaussian envelope (σ);

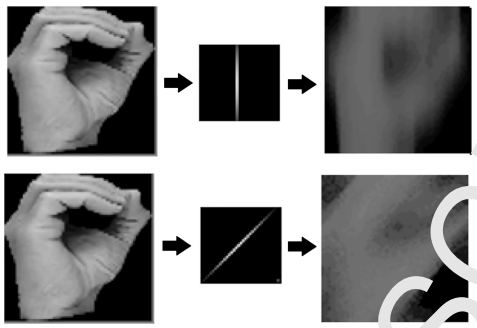


Figure 1: A Gabor filter with $\theta = 0°$ and $\theta = 45°$.

- Orientation of the normal to the parallel stripes of a Gabor function (θ);
- Wavelength of the sinusoidal factor (λ),
- Spatial aspect ratio (γ);
- Phase offset (ψ).

The Gabor filter performs a low pass filtering along the orientation of θ and a band pass filtering orthogonal to its orientation θ. Therefore, by choosing the parameters above it is possible to enhance visual properties of an image, such as spatial frequency and orientation, as illustrated in Figure 1.

2.2.4. Contour properties

Some simple properties can be extracted just from the image contour. Figure 2 shows a hand gesture with a corresponding convex hull and a convexity defect. The features below can be used to rapidly distinguish between simple gestures, such as open and closed hand.

- Solidity: the ratio of contour area to its convex hull area;
- Extent: the ratio of contour area to bounding rectangle area;
- Convexity defects: a list of the five largest convexity defects is used as a feature vector.

The computational cost of the above features are compared in Figure 3 as a function of image size. The feature extraction time is averaged across

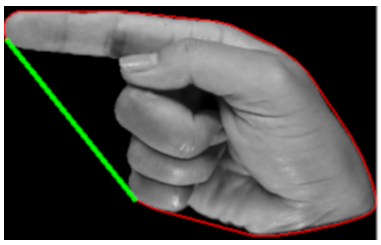
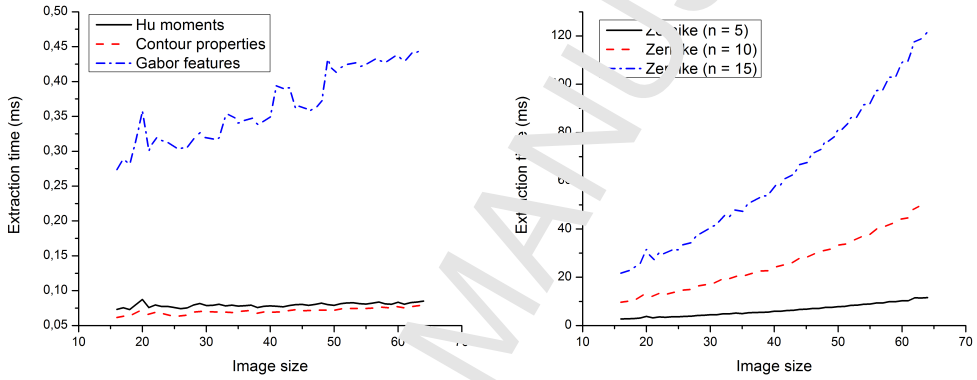


Figure 2: A convex hull (red line) with a convexity defect (black green line).



(a) (b)

Figure 3: Comparison of feature extraction time for Gabor features, Hu moments and contour properties (left) and Zernike features of maximum order 5, 10 and 15 (right).

100 images for each image size and the image size is varied from 16×16 to 64×64 pixels. As can be seen from the graph in Figure 3b, Zernike moments have a significantly higher computational cost than other features.

2.3. Convolutional neural networks

Being a popular class of machine learning techniques, deep learning has greatly profited from technological advances in graphical processing units (GPUs) [24, 44, 45], and this in turn made its extensive use possible. Faster GPUs meant lower network training time. In addition to this, the effectiveness of deep network models was another factor that led to their quick gain in popularity. This class of techniques has achieved very good results in many important problems, outperforming other traditional techniques [24, 45].

Convolutional neural networks (CNNs) are deep learning models with many layers that comprise multiple levels, each of which is responsible for transforming the input into a more generic representation of itself [45]. Successive transformations highlight features present in the input pattern, and are fundamental in the learning process of deep neural networks. Compared to fully-connected networks, CNNs are easiear to train and have higher capability for generalization [45]. Consequently, CNNs can be employed more easily than feature extractor-based networks, the latter being potentially specific to the problem domain. This makes them suitable for application in a wide range of problems, such as pattern recognition (e.g., image and speech) and natural language processing tasks (e.g., machine translation).

2.3.1. Main components

Broadly speaking, CNNs consist of multiple layers, each with a particular function, responsible for nonlinearly transforming the input and forwarding the output to the next layer. From these transformations, CNNs abstract away from details considered irrelevant and concentrate on invariant properties present in the data [44, 45].

Two types of layer worth mentioning are the convolutional layer and the subsampling layer. The units in these layers form feature maps, and to each feature map there is an associated kernel of fixed size. In a convolutional layer, the convolution kernel passes horizontally and vertically throughout the input map, convolving with each unit. The output of the convolution with each kernel is fed to a nonlinear activation function, such as a sigmoid function or the popular ReLU (rectified linear unit) [24, 45].

Subsampling methods are used to reduce the volume of data during training [44]. A technique known as max pooling is one of the most commonly employed in this context, as it can, in conjunction with other components like convolutional layers in GPU-based CNNs, significantly help achieve good results in benchmark datasets [24]. The max pooling layer divides an input feature map in equally sized rectangles and computes the maximum unit for each partition. The results are then forwarded to the next layer, so that a new feature map is built with the maximum units replacing each rectangle.

Typically, both types of layers alternate multiple times and are followed by one or more fully connected (FC) layers. To introduce the last set of feature maps outputted by a CNN to an FC layer, the flattening operation turns the grids into a one-dimensional array of units that are then treated as input neurons to the FC layer. Adding to the set of essential tools for deep

learning networks, a regularization technique called dropout can be used to prevent overfitting and further increase a network's performance [24, 44]. Dropout acts by eliminating units during training with a certain probability and can be applied to any layer of the CNN. Lastly, combinations of the described components are successfully used in state-of-the-art CNNs, which can efficiently reap the benefits of modern GPUs.

2.3.2. Considerations about real-time performance and training

One of the drawbacks of CNNs is the amount of memory required to store the parameters and its real-time performance. For example, a classical network for image recognition, Alex-net [46], requires about 250 MB of RAM and 1.5 million floating-point operations. Training also usually takes longer than other machine learning methods, although this can be mitigated by parallel processing on a GPU.

Another important concern when training CNNs is how to compromise between training time and the amount of data used for the task. The Faster R-CNN model [47], using region proposal to help locate objects, is state-of-the-art in object detection, despite being trained for the small dataset PASCAL VOC 2007 (9,963 images of 20 categories split in half for training and testing). The network contains shared convolutional layers that are initialized via transfer learning from a model pre-trained on ImageNet. Another example of network trained for a relatively smaller dataset (129,450 images of 2,032 classes with 127 463 training and validation images and 1,942 test images) is a CNN to diagnose skin cancer [48], achieving accuracy matching the dermatologists that participated in the tests. Transfer learning was used with a GoogLeNet Inception v3 model pre-trained on ImageNet.

The two previous examples reinforce one of the advantages of employing transfer learning in training. In the case of insufficient data for training, transfer learning can be used to obtain previous knowledge from larger datasets to new tasks, even if the domains are different, thus avoiding effort to expand smaller datasets [49]. In the present work, transfer learning is evaluated on two state-of-the-art large CNNs.

2.4. Related works

Among traditional feature extraction techniques, Aowal et al. [50] propose and evaluate discriminative Zernike moments (DZM) and compared them with standard Zernike moments (ZM), principal component analysis (PCA) and Fourier descriptors (FDs). The best recognition rates are

obtained with Zernike and discriminative Zernike moments. Kumar et al. [51] compared Local Binary Patterns (LBP), Speeded Up Robust Features (SURF), and Super Pixels (SP) on a proprietary dataset. SURF features were found to yield better recognition rate, although the feature extraction time is not considered. A real-time system that uses Fourier descriptors to represent hand blobs is developed by Ng and Ranganath [52]. An RBF network is used for hand pose classification. Hu moments are compared with Zernike moments for general object recognition [53] as well as for static hand gestures [54]. In general, Zernike moments were found to be more accurate than Hu moments [53, 54]. Wang C. and Wang K. [55] also used Hu moments, along with valley circle features for real-time recognition of static gestures from a 2D camera. An improved version of Zernike moments is proposed by Guo et al. [56], measuring accuracy as well as computational cost on a common static hand gesture database, also used in this paper. A combination of graph-based characteristics and appearance-based descriptors such as detected edges for modeling static gestures is proposed by Avraam [57]. This method is tested on ten numerical gestures from the Massey database [29]. An average recognition rate of 83% with a standard deviation of 10% is achieved.

A hand gesture recognition system is proposed by Yang [58], using a Kinect™ depth sensor for increased robustness. Dynamic gestures were recognized as a sequence of static hand postures. Depth data is also used by Palacios et al. [59] to improve segmentation of hands. The proposed segmentation works for multiple hands, even when the subject's face and hands are at the same depth. A simple decision tree is used for gesture recognition and the system is able to work with 25 frames per second on an Intel® Core™ i7 processor. A Kinect™ sensor is also used by Plouffe and Cretu [60] for recognition of static and dynamic gestures. Prior to gesture recognition, the positions of the palm and the fingertips are detected and used as features for classification. A dynamic time warping matrix is used for recognition, where the current gesture is compared with data stored in a database. The gestures are recognized with an average delay of 100 ms.

A convolutional neural network with three channels is proposed by Barros et al. [37]. Besides a grayscale image, convolution is also applied to an image filtered by vertical and horizontal Sobel operators. Images are reduced to 28×28 pixels, allowing real-time recognition. Inspired by the aforementioned work, a multichannel CNN is also used in the present paper. The proposed network has two channels: a grayscale image and the same image convolved

by a Gabor filter. The Gabor filter is tuned by a hyperparameter selection algorithm.

A large RGB-D hand posture dataset [30] is used by Sanchez-Riera et al. [36] to compare different forms of fusion between standard and depth frames. Several classifiers were used for comparison, including Support Vector Machine (SVM), Convolutional Neural Networks (CNN) and Stacked Autoencoders (SAE). CNNs achieved the best accuracy with a concatenation of depth and color data.

Huang et al. [61] explored an alternative approach with depth sensors, comparing depth data obtained from Kinect™ with finger joints data obtained from a RealSense™ camera. Similarly, Dinh et al. [62] used a hand model for smart home appliances with four distinct gestures. Although recognition rate with finger joints is in most cases better than with depth data, it should be noted that the hand model is extracted from depth data. Thus there is a processing step before recognition, where the raw data from a RealSense™ camera is converted into a 3D hand model. This step is expected to have some additional error as well as computational cost [3, 34].

3D hand gesture acquisition and recognition techniques were surveyed by Cheng et al. [63], considering Kinect™ and Leap Motion sensors. Possible applications of such systems are identified in contexts such as gaming, sign language, virtual manipulation, daily assistance, human-robot interaction, among others. Distinction between similar hand gestures is identified as a challenging task. The presented work is evaluated on benchmark datasets and special attention is given to the recognition of similar gestures.

Three different configurations of CNNs and Stacked Denoising Autoencoders (SDAE) are evaluated by Oyedotun and Khashman [26] on a static hand posture dataset [4], considering accuracy and recognition time. Recognition rates of 92.83% and 91.33% are obtained by models called SDAE3 and CNN1, respectively. The model CNN1 is evaluated in the present study and compared against other state-of-the-art models.

Ji et al. [28] combine five CNNs by voting. Gestures are segmented with an aid of a colored glove and images are resized to 28×28 pixels. This CNN architecture is thoroughly evaluated in the present study, confirming that combination of CNNs results in increased accuracy, although with a penalty on real-time performance.

A small convolutional neural network with two layers of filters is evaluated by Nasr et al. [27] on a public dataset with 1,400 images. A depth camera is used for gesture segmentation, but recognition is made from a 2D

binary image for lower computational cost. The present study implements this model. Furthermore, recognition rates are compared for classification from depth, grayscale and binary images.

Cambuim et al. [64] propose and efficient hand posture recognizer implemented on an FPGA. As in Nasr et al. [27], gestures are converted into binary images and classified by an Extreme Learning Machine (ELM) neural network. On an FPGA, the system achieves 97% recognition rate at 6.5 ms per image. The present work also explores binary representation of images as a more robust and computationally efficient method. Emphasis is also given to efficiency, as classifiers are compared by speed with and without use of a GPU.

A random forest classifier is used by Nai et al. [65] to recognize hand postures in real time. Once the hand is located, four features are extracted on depth data from line segments located near the hand. The system runs at 600 fps with a Kinect™ sensor. Leave-one-subject-out cross validation is used and a 89.6% recognition rate is obtained from a dataset with ten postures corresponding to digital numbers. A sign language dataset with 24 postures is also evaluated, but the recognition rate is not shown to be better than prior methods.

A 3D convolutional network is proposed by Kopinski et al. [66] for recognition of static hand gestures from depth data. The depth image is perceived as a point cloud, subdivided into cubes of fixed size. This network is evaluated on a large dataset, containing 10 postures and 20 subjects. The generalization performance is measured by leave-one-person out validation technique. Similarly, the present paper uses leave-one-subject-out validation on various datasets with 10, 27 and 36 postures. The classifier proposed in the present study has the advantage of working well with depth, grayscale and binary data without any adjustment.

A multi-objective optimization method for feature selection and classifier optimization is proposed in Chevtchenko et al. [67]. Several feature descriptors are compared on the Massey dataset, obtaining up to 97.63% of accuracy. The study suggests combining Zernike moments, Hu moments and Gabor filters for increased accuracy. An optimized multilayer perceptron is used as a classifier. A combination of Gabor and Hu features is also used for real-time recognition with feature extraction time of less than 2 ms on an Intel® Core™ i5 processor. The Massey database is also used in the present work and the results are presented for the entire set of gestures, including 26 letters and 10 digits.

Table 1: Comparative summary of related works

Work	Classifier	Color space	Evaluated datasets	Real-time
Palacios et al. [59]	Heuristic	RGB, Depth	1	Yes
Dinh et al. [62]	RF, heuristic	Depth	1	Yes
Hsiao et al. [30]	CNN	Depth	1	No
Barros et al. [37]	CNN	Gray	2	No
Sanchez-Riera et al. [36]	CNN	Gray, Depth	1	No
Oyedotun and Khashman [26]	CNN, SDAE	Gray	1	No
Cambuim et al. [64]	ELM	Binary	1	Yes
Nasr et al. [27]	CNN	Binary	1	No
Wang C. and Wang K. [55]	Heuristic	Binary	1	No
Ji et al. [28]	CNN	Gray	1	No
Nai et al. [65]	RF	Depth	1	Yes
Kopinski et al.[66]	CNN	Depth	1	Yes
Chevtchenko et al. [67]	MLP	Gray	1	Yes
This paper	CNN	Gray, Binary, Depth	3	Yes

Table 1 contains a comparative summary of the state-of-the-art. The 'Color space' column contains the color space used for feature extraction, and the 'Real-time' column indicates whether the related work contains a demonstration of real-time recognition or information about the speed of recognition per image. The current work is unique in the sense that it proposes a combination of traditional and convolutional features for hand posture recognition. Additionally, it evaluates different combinations of validation methods, color space and rescaling. A real-time recognition system is also demonstrated using the proposed classifier.

As can be seen from related works, convolutional neural networks and traditional feature descriptors have obtained good results for hand posture recognition. Drawing inspiration from both, this paper proposes a novel architecture, where a convolutional neural network is combined with a feature vector obtained from classical image descriptors. The key difference to a classical CNN is that the fully connected layer in our model receives information from both convolutional and classical features. This feature fusion-based architecture is thoroughly compared on three hand posture datasets with recent convolutional neural networks with respect to accuracy and real-time capability. Furthermore, it is shown to perform just as well with binary images, allowing a more robust classifier, independent from the type of camera. Our model recognizes the hand directly from depth or 2D camera frames, without

need for additional computation.

3. The proposed convolutional neural network with feature fusion

In a traditional convolutional neural network, a sequence of convolutional and subsampling layers is followed by a fully connected layer—a multilayer perceptron. As explained in Section 2.3, the convolutional layers act as feature extractors while the last fully connected layers are responsible for delivering the final recognition result. In our previous study [67], we have investigated combinations of different image descriptors for hand posture recognition, including Hu moments, Gabor filters and Zernike moments. More specifically, the combination of Hu and Zernike moments was shown to provide a statistically significant increase in accuracy compared to just using Zernike moments. The idea behind the proposed architecture is to make use of common feature descriptors to complement the information available to the last fully connected layers. For example, since the biologically inspired Gabor filter is shown to increase recognition rate of hand postures in Chevtchenko et al. [67], in the proposed architecture a second convolutional channel is fed with an image previously enhanced by this filter. Other properties based on shape and contour are also used to diversify information about the hand posture prior to classification. Note that although CNNs with feature fusion have not been used before in the context of hand posture recognition, similar architectures have been successfully applied to problems of different domains, such as pedestrian identification [68] and text classification [69].

The architecture of this feature fusion-based convolutional neural network (FFCNN) is depicted in Figure 4. The input is a single grayscale or binary image of a hand, rescaled to 32×32 pixels. The grayscale image can also represent the depth channel of an RGB-D camera. How this and other networks perform on binary, grayscale and depth data is evaluated further in Section 4. The input image is fed to a convolutional channel represented on the bottom of Figure 4. This channel contains two layers of convolution, concatenated with max pooling layers. The number of layers was adjusted experimentally. As Oyedotun and Khashman [26], we found that two layers perform better than three, four or five on evaluated datasets.

The same image is also fed to a Gabor filter and then to a second convolutional channel, shown above the first one in Figure 4. This channel has the same number of layers as the previous one. The outputs of the final max pooling layers of both channels are concatenated into a one-dimensional

feature vector. Some neurons from this single feature vector can be disabled during training, i.e. the dropout regularization technique is applied with a certain probability.

In order to further increase the diversity of information, an additional set of fully connected neurons receives features that describe the shape and contour of a hand. This additional feature vector is obtained from the concatenation of Zernike moments, Hu moments and contour properties, described in Section 2.2. As a set of Zernike moments can be of a variable size, this auxiliary feature vector is also of a variable size M, as represented in Figure 4.

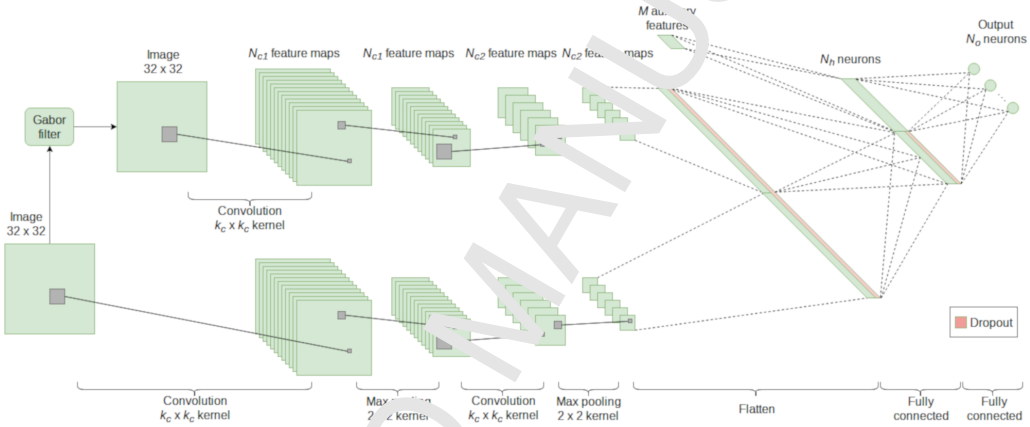


Figure 4: A diagram of the proposed feature fusion-based convolutional network.

Both feature vectors are then applied to the input of a fully connected neural network. Note that dropout is used only on the features extracted by convolution, as they are more numerous and more likely to contain redundant information than the manually extracted features. The output layer contains the same number of neurons as classes in the dataset. The final classification is obtained by selecting the output neuron with the highest activation. The above process is further illustrated by Figure 5, where activations of input, output and some convolutional layers are depicted for a simple gesture.

4. Experimental evaluation

Extensive experimentation is conducted in order to compare the proposed feature fusion-based convolutional architecture with other models. This section introduces validation methods and system setup, as well as describes

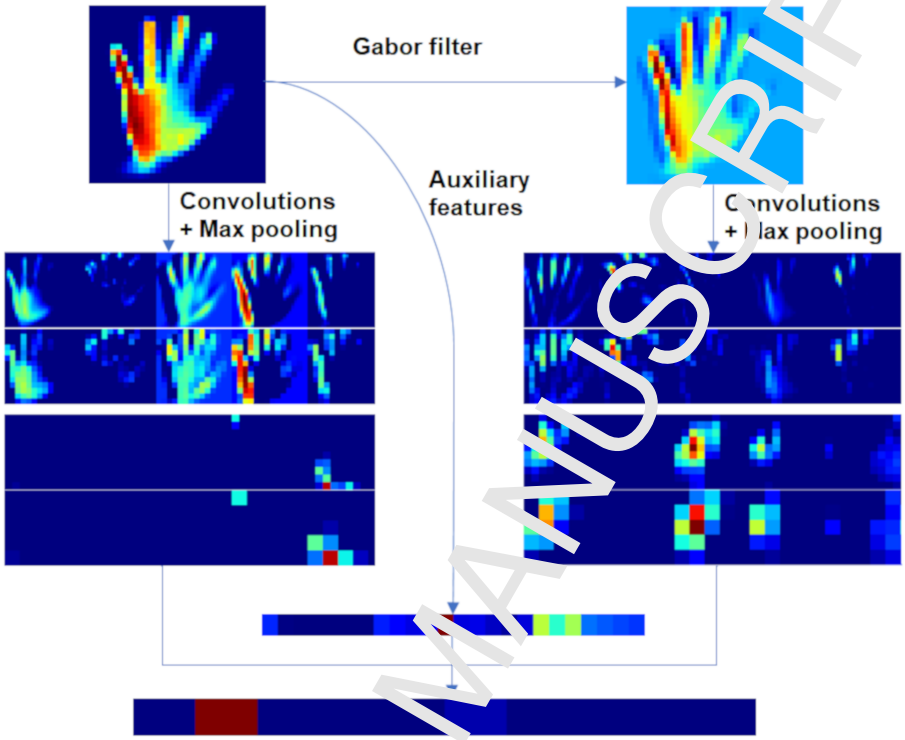


Figure 5: An illustration of activation of layers in the FFCNN architecture. A depth image is used as an input.

the implemented architectures. A real-time hand posture recognition system, based on the proposed architecture is also presented.

4.1. Validation technique

The datasets used in this paper contain gestures from more than one subject [29, 30, 31]. This makes it possible to use two different validation techniques:

- Holdout: 80% of the dataset is used for training and 20% is separated for testing. The selection is random and gestures from any subject in the dataset can appear in both training and testing subsets.
- Leave-one-subject-out cross-validation: consider S as the total number of subjects in the dataset. One of the subjects is separated for testing

and the rest $(S-1)$ is used for training. This process is repeated S times, one for each subject.

While holdout is a more common validation technique, leave-one subject-out is more challenging as the classifier is tested with a set of gestures from a new person. Thus it is useful to assess how the recognition would perform when a new subject is using a previously trained system, i.e. the classifiers generalization capability. The results presented in this paper are averaged across ten repetitions for both validation methods.

All models are limited to train for a maximum of 2,000 epochs. The classification process is interrupted if the training accuracy does not change for 100 epochs.

4.2. System setup

The experiments were conducted on the same computer. Relevant configuration characteristics are listed below.

- Hardware:
  - Processor: AMD FX™-8320E Eight-Core
  - Installed RAM: 16 GB
  - GPU model: GeForce® GTX 1070
- Software:
  - Operational system: Linux Mint 18.1
  - Python version: 3.5
  - OpenCV library version: 3.2
  - TensorFlow library version: 1.0.1
  - Keras library version: 2.0.3

4.3. Benchmark datasets

Three different hand posture datasets are used in this paper. They are summarized in Table 2 and described in detail below.

Table 2: A summary of the datasets used in this paper

Dataset	Year	Channels	Subjects	Gestures	Images
Massey [29]	2011	3	5	36	2,515
LaRED [30]	2014	4	10	27	81,000
OUHANDS [31]	2016	4	20	10	3,000

4.3.1. Massey

The Massey dataset [29] has 2,515 images, illustrated in Figure 6. The database contains variations in scale, illumination and rotation, as illustrated in Figure 7, which shows three similar gestures: 'a', 's' and 't'. There are five subjects, allowing to use both the holdout and leave-one-subject-out validation techniques. The hand postures are grouped in 36 classes, corresponding to digits and letters of an ASL alphabet. The segmentation of this database is also straightforward, since the background is already removed. This dataset is subdivided in grayscale and binary subsets and the classification methods implemented in this paper are evaluated on this subsets. The two subsets are named Massey-G and Massey-B, corresponding to grayscale and binary data respectively.

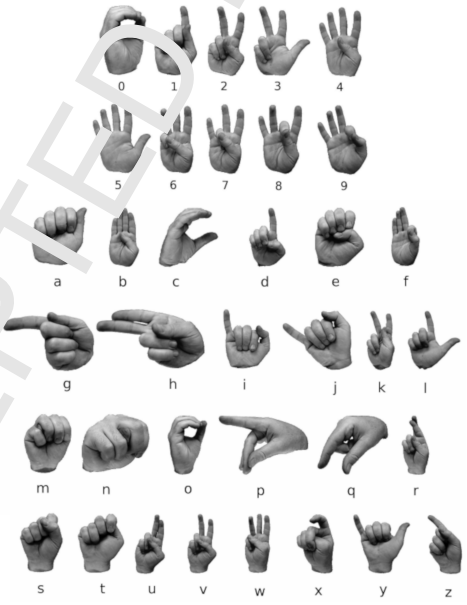


Figure 6: 36 gestures from the Massey dataset [29].

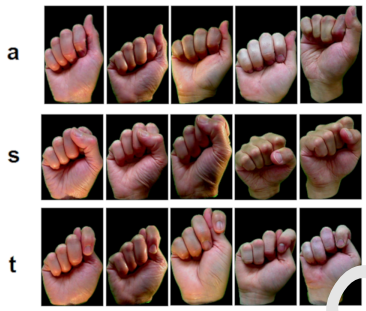


Figure 7: Three similar letters from the Massey dataset, showing variations in scale, tone, illumination and slight rotation within the same class.

4.3.2. OUHANDS

This dataset [31] is aimed for evaluation of both, classification and segmentation methods. It contains manually segmented binary masks, as well as aligned depth and color frames. There are 10 classes of gestures, performed by 20 subjects, as seen in Figure 8. The images are obtained by a hand-held Realsense® camera, similar to the one used in this work. Although the overall number of images is comparable with the Massey dataset, there are more images per gesture, since this database contains 26 fewer classes. The classes are also simpler, since the main difference between gestures is the number of raised fingers. Similar to the Massey subsets, there are three datasets based on OUHANDS, namely OUHANDS-G, OUHANDS-B and OUHANDS-D, corresponding to grayscale, binary and depth data respectively.

Figure 8: Ten gestures with binary mask, depth data and RGB frames from the OUHANDS dataset [31].

4.3.3. LaRED

LaRED [30] is a large dataset with 81,000 images, containing color, depth and segmentation data. It contains 27 static gestures obtained from 10 subjects (five males and five females), illustrated in Figure 9. This database can be further extended to 243,000 images by applying rotation. Differently from other datasets used in this paper, LaRED is automatically segmented based on depth data. This means that sometimes segmentation is not ideal, but is closer to a realistic application, where segmentation error are expected. Also note that there are small difference between gestures G_1 and G_3, G_4 and G_5, G_{14} and G_{15}, G_{23} and G_{27}, among others. The depth and binary data are used to generate LaRED-D and LaRED-B subsets. RGB channel is not aligned with depth channel in this database, and so, a subset with grayscale images was not generated. This is not a significant drawback, since grayscale and binary images are already compared using Massey and OUHANDS databases.

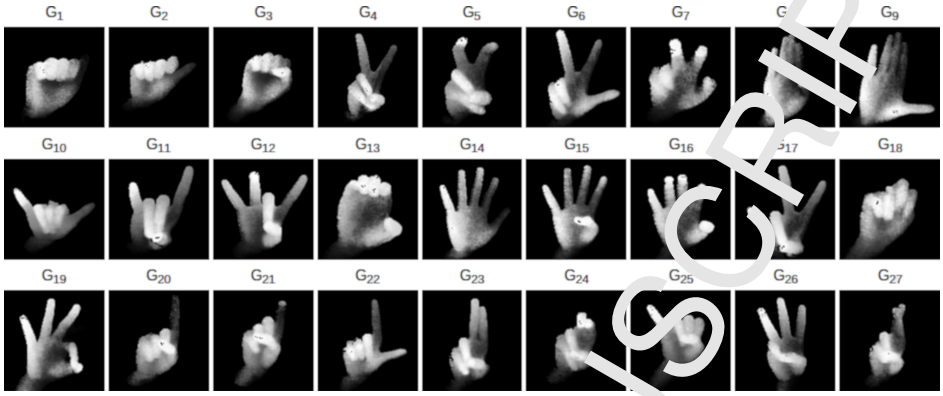


Figure 9: 27 gestures from the LaRED dataset [30].

4.4. Hyperparameter selection

Even a small convolutional network has a large number of parameters associated with its architecture, such as the number of layers, size of filtering windows, number of fully connected neurons, etc. Changes in these parameters may have a significant effect on the overall performance of the network.

The hyperparameter selection task usually has high dimensions and small fitness evaluation budget (high computational cost per evaluation). In order to avoid manual tuning, we use the Tree-of-Parzen-Estimators (TPE) algorithm, introduced by Bergstra et al. [70], to search the parameter space. Given that $y = f(x)$ is an observation of the model with hyperparameters x, the TPE algorithm is used to iteratively generate samples of y based on previous observations [71, 72].

The feature fusion-based convolutional neural network described in this paper has several parameters that can be adjusted by a search algorithm—TPE. These parameters are given below. Note that *uniform* search space means that the sample values are drawn uniformly, constrained to a two-sided interval. Space exploration values are based on experimental evaluation and similar applications found in literature, such as Chevtchenko et al. [67], Oyedotun and Khashman [26], Nasr et al. [27] and Yamashita and Watasue [73]. The parameters are:

- Activation function for fully connected layers: sigmoid or softmax.
- Size of a convolutional kernel ($k_c \times k_c$): 3×3 or 5×5.

- Number of convolutions in the first layer (N_{c_1}): {4, 8, 16, 32}.
- Number of convolutions in the second layer (N_{c_2}): {4, 8, 16, 32}.
- Hidden neurons in the fully connected layers (N_h): {50, 100, 150, 200, 300}.
- Dropout rate before the fully connected layer: continuous, between 0 and 100%.
- Dropout rate after the fully connected layer: continuous, between 0 and 100%.
- Zernike moments order (n): {0, 5, 10, 15, 20, 25}. Order 0 means that the Zernike moments are not extracted. This is permitted due to higher computational cost of this features.
- Gabor filter parameters:
  - σ: uniform, between 0.001 and 1.
  - θ: uniform, between 0.001 and π.
  - λ: uniform, between 0.001 and π.
  - γ: uniform, between 0.001 and 1.
  - ψ: uniform, between 0.001 and π.

The Massey-G subset is used to evaluate each model proposed by the TPE algorithm. More specifically, gestures from the subject 'hand5' are separated as validation, while the other subjects are used for training. The recognition rate on subject 'hand5' is used as fitness function because this subject is the most challenging from the dataset and thus provides better margin for optimization.

The hyperparameter selection loop is executed for 150 iterations, thus evaluating 150 different models. Each model returns validation accuracy on 'hand5' and average classification time per image. Figure 10 shows the average and best recognition rates obtained during this process. As new models are evaluated, the TPE algorithm becomes more likely to generate better models.

Although only validation accuracy is used as fitness function, classification time is also recorded and is used to support the selection of a final model.

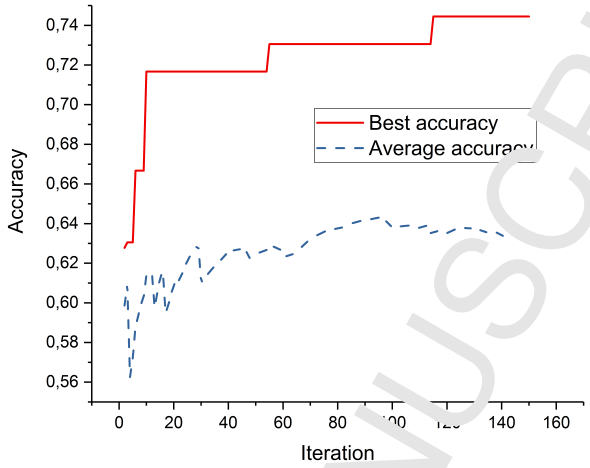


Figure 10: Average and best recognition rate obtained during the hyperparameter selection loop.

From 150 architectures proposed by TPE, we select the one with best recognition rate and with real-time capability of at least 30 fps (or 33 ms). The following subsection provides detailed information about the configuration of this model as well as other state-of-the-art convolutional neural networks.

4.5. Implemented models

This section details the deep learning architectures considered for comparison and benchmark. Five models are implemented following recent papers. These models are denoted as CNN1, CNN2 and CNN3, based on Oyedotun and Khashman [20], Nasr et al. [27] and Ji et al. [28] respectively. Two pretrained convolutional networks are also used: VGG16 [74] and MobileNet [25]. As an additional baseline, the TPE algorithm is also used to select hyperparameters for a simple convolutional network with a single channel, denoted as CNN4. The architecture proposed in this paper is denoted as FFCNN, which stands for feature fusion-based convolutional neural network. The hyperparameters of FFCNN were selected through TPE search, as described in Section 4.4, and are summarized in Table 3.

Table 3: Hyperparameters of the proposed FFCNN architecture

	Hyperparameters
Activation function for conv. layers	ReLU
Activation function for FC layers	Sigmoid
Size of convolutional kernel	5×5
Number of convolutions in 1st layer	32
Number of convolutions in 2nd layer	16
Hidden neurons in FC layers	200
Dropout rate before FC layer	61%
Dropout rate after FC layer	76%
Hidden neurons in auxiliary FC layer	200
Order of Zernike moments	5
Gabor filter parameters	σ: 0.35
	θ: 0.67
	λ: 0.57
	γ: 0.23
	ψ: 0.37

4.5.1. CNN1

Proposed by Oyedotun and Khashman [26], this is a single-channel convolutional neural network. A 5×5 kernel is used for the convolution operations. First convolutional layer receives a single-channel image and convolves it through 6 kernels, resulting in 6 maps of size 28×28. Then a pooling windows with size 2×2 is used for subsampling, generating six feature maps of size 14×14. A second convolution with 12 kernels and another pooling layer generate 12 feature maps of size 5×5. Finally, the 12×5×5 feature map is flattened to a vector with 300 scalars. A fully connected multilayer perceptron with 400 neurons in the hidden layer is used to classify the image. The number of neurons in the output layer is the same as number of classes in the dataset– N. A log-sigmoid activation function is used for all layers. This network is depicted in Figure 11.

4.5.2. CNN2

Proposed by Nasr et al. [27], compared to CNN1, this network employs more filters in convolutional layers but less neurons in the fully connected layer. The input image size is set to 50×50. The first convolutional layer has 30 filters of size 5×5 and the second has 20 kernels with size 3×3. The

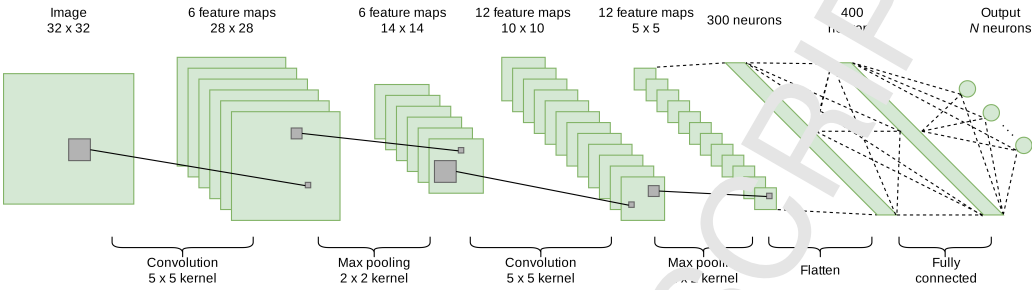


Figure 11: A convolutional network, denoted as CNN1, proposed by Oyedotun and Khashman [26]. N is the number of classes in the dataset.

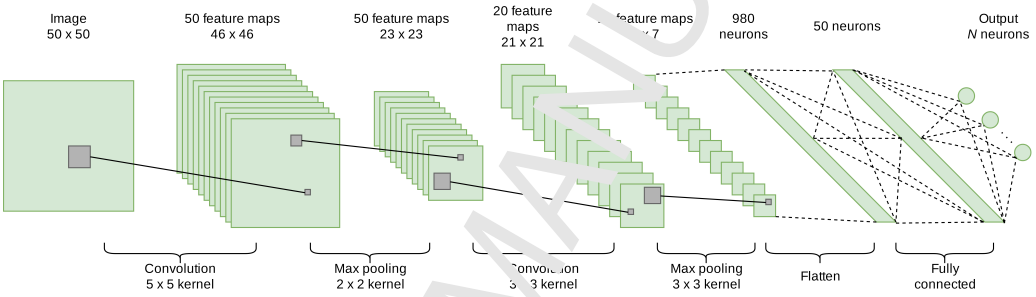


Figure 12: A convolutional network, denoted as CNN2, proposed by Nasr et al. [27].

result of the second pooling layer is flattened into a vector with 980 scalars. The fully connected layer has 50 hidden neurons. This network is illustrated in Figure 12.

4.5.3. CNN3

Proposed by Ji et al. [28], this is an ensemble of five convolutional neural networks. The architecture of each network is similar to CNN1, with input image of size 28×28. The final classification is made by assigning the most voted class by all networks. Figure 13 depicts the structure of this network.

4.5.4. CNN4

In order to provide an additional baseline for the proposed FFCNN, we apply TPE on a single-channel CNN. This is equivalent to the proposed FFCNN without the additional channel and the auxiliary feature vector. As with FFCNN, the TPE algorithm was used to evaluated 150 configurations of this network on Massey dataset. The hyperparameter selection process and search space for this network are the same as described in Section 4.4,

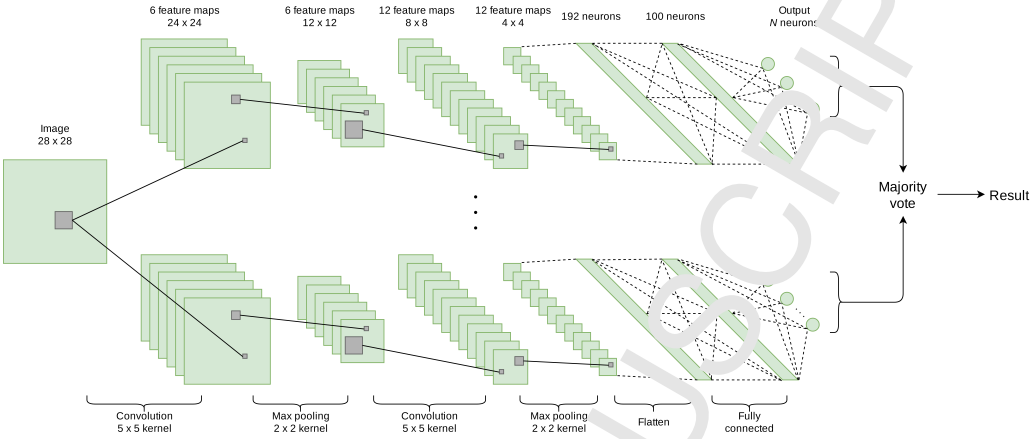


Figure 13: An ensemble of 5 convolutional networks, denoted as CNN3, with majority voting proposed by Ji et al [28].

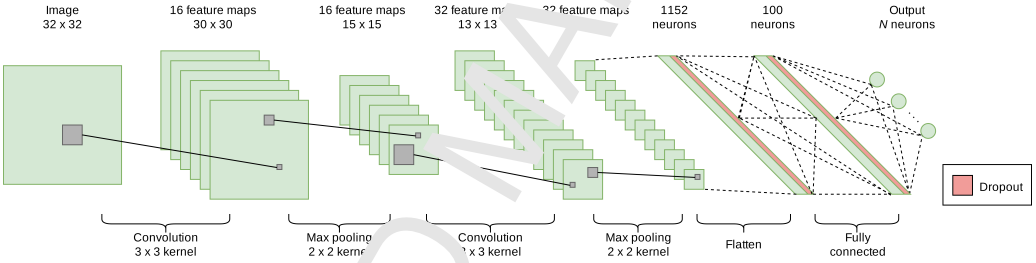


Figure 14: An additional baseline network, denoted as CNN4. As for the proposed FFCNN, the hyperparameters are automatically adjusted using the TPE algorithm.

excluding Zernike moments order and Gabor filter parameters. The resulting network is depicted on Figure 14. The input image is resized to 32×32 pixels. There are two convolutional layers with 16 and 32 kernels of size 3×3. The selected dropout rate is 75% before the fully connected hidden layer and 63% after. Convolutional and fully connected layers use ReLu and sigmoid activations respectively.

4.5.5. VGG16

A famous convolutional network, proposed by Simonyan and Andrew [74]. The input image is resized to 48×48 pixels. The top, fully connected layers are removed and replaced by a fully connected layer with 50 neurons and an output layer with the same number of neurons as classes. Only this last two layers have their weights adjusted during the training step. The weights

of other layers are obtained from the original network, trained on ImageNet dataset.

4.5.6. MobileNet

Proposed in 2017 by Howard et al. [25], this is a new model of convolutional networks that aim to be employed on mobile devices. The model used in this paper is denoted as '1.0 MobileNet-224' in the original article. As with VGG16, the input image is resized to 48×48 pixels and the top fully connected layers are replaces by a fully connected layer with 50 neurons, followed by an output layer. The weights of this network are also pretrained on the ImageNet dataset.

4.6. Experimental results

The objective of this section is to evaluate the implemented architectures on Massey, OUHANDS and LaRED datasets, using validation methods presented in Section 4.1.

The experiments are averaged across 10 repetitions and are performed using holdout and leave-one-subject-out validations. For each repetition using holdout validation the dataset is randomly divided in training (80%) and testing (20%), as described in Section 4.1. For leave-one-subject-out cross-validation each repetition is averaged across all subjects in the dataset. For instance, Massey dataset contains five subjects, thus for each repetition, a different subject is selected for testing. Considering five subjects and ten repetitions, a single result for leave-one-subject-out cross-validation, such as a cell in Table 5 is obtained from fifty experiments. Average recognition rates are presented with standard deviation. Furthermore, we have used a non-parametric Wilcoxon statistical test with a 5% significance to compare results [75]. For any table, the best result in terms of accuracy is compared against others in the same table. The best result is highlighted along with the statistically equivalent ones.

4.6.1. Massey dataset

Initially all architectures described in Section 4.5 are compared using Massey dataset with holdout and leave-one-subject-out validations. Table 4 compares recognition rates of different models on Massey-G (grayscale) and Massey B (binary) subsets. The highlighted result was found to be statistically better than the other results in this table.

Table 4: Comparison of accuracy on Massey subsets using holdout validation.

Architecture model	Massey-G	Massey B
VGG16	92.20 (1.09)	86.62 (2.01)
MobileNet	93.89 (1.59)	93.73 (1.74)
CNN1	96.81 (0.73)	95.90 (1.42)
CNN2	96.97 (0.93)	96.16 (1.05)
CNN3	96.97 (0.45)	95.89 (0.81)
CNN4	96.79 (0.92)	95.42 (1.36)
FFCNN	**98.05 (0.90)**	96.93 (0.70)

Table 5 contains a similar comparison using leave-one-subject-out validation. The highlighted result—FFCNN on the Massey-G subset—is significantly better than other results in the same table.

Table 5: Comparison of accuracy on Massey subsets using leave-one-subject-out validation

Architecture model	Massey-G	Massey-B
VGG16	61.42 (0.82)	57.47 (1.68)
MobileNet	62.53 (1.36)	70.40 (1.60)
CNN1	73.86 (1.04)	76.61 (0.40)
CNN2	79.04 (0.85)	77.62 (1.26)
CNN3	78.51 (0.70)	79.88 (0.72)
CNN4	82.93 (0.61)	80.57 (0.55)
FFCNN	**84.02 (0.59)**	82.58 (0.81)

While accuracy is critical for a good model for posture recognition, it also has to perform in real time. Thus, Table 6 compares the performance of the models for training and recognition of a single image. This experiment is conducted with and without a GPU and the models are trained and evaluated on Massey-G subset. The training time is averaged across 10 repetitions and the recognition time is averaged across 503 images.

Table 6: Comparison of training and recognition times on the Massey dataset with and without a GPU

Architecture model	With GPU		Without GPU	
	Training per epoch (s)	Recognition time (ms)	Training per epoch (s)	Recognition time (ms)
VGG16	0.9	6.0	89.8	55.7
MobileNet	2.0	22.6	57.5	19.3
CNN1	0.2	1.7	1.2	1.5
CNN2	0.3	1.7	10.3	2.9
CNN3	0.9	8.4	3.8	4.9
CNN4	0.2	1.7	1.7	1.3
FFCNN	0.3	9.2	1.2	7.8

Despite GPU providing an obvious improvement in training time, there is usually no benefit in using a GPU for real-time recognition. For a network to recognize a specific gesture, both the gesture data and a network have to be transfered to a GPU shared memory. This process creates an overhead that makes smaller architectures, such as CNN1, CNN3 and FFCNN faster to process on a CPU. A large network such as VGG16 has to be used on a computer with a GPU for real-time applications. While MobileNet is designed to be used on devices with limited processing capabilities, it is still a large network and is slower to train in both cases.

4.6.2. OUHANDS dataset

Recognition rates of the models are evaluated on OUHANDS subsets using holdout validation in Table 7. The highlighted results are statistically equivalent and significantly outperform other results in the same table.

Table 7: Comparison of accuracy on OUHANDS subsets using holdout validation

Architecture model	OUHANDS-G	OUHANDS-B	OUHANDS-D
VGG16	90.00 (1.57)	89.50 (2.73)	90.80 (1.29)
MobileNet	94.60 (0.68)	95.80 (0.75)	95.81 (0.65)
CNN1	97.50 (0.67)	97.90 (0.50)	97.55 (0.50)
CNN2	96.83 (0.83)	97.68 (0.70)	97.48 (0.64)
CNN3	97.25 (0.64)	98.01 (0.51)	98.06 (0.46)
CNN4	**98.70 (0.33)**	98.06 (0.47)	**98.35 (0.32)**
FFCNN	**98.75 (0.48)**	**98.68 (0.33)**	**98.48 (0.63)**

Table 3 compares the same architectures using leave-one-subject-out val-

idation. The FFCNN model on the grayscale subset is shown to be statistically better than other combinations of models and subsets.

Table 8: Comparison of accuracy on OUHANDS subsets using leave-one-subject-out validation. As previously mentioned, the Wilcoxon statistical test was performed to compare the results. According to the test, with a 5% level of significance, the highlighted result is statistically superior to other results in this table.

Architecture model	OUHANDS-G	OUHANDS-B	OUHANDS-D
VGG16	88.01 (0.71)	88.92 (0.64)	88.46 (0.52)
MobileNet	92.67 (0.65)	96.66 (0.47)	95.67 (0.40)
CNN1	97.04 (0.27)	97.80 (0.21)	97.03 (0.23)
CNN2	97.40 (0.25)	98.11 (0.18)	97.52 (0.39)
CNN3	97.20 (0.25)	98.33 (0.17)	98.20 (0.17)
CNN4	98.80 (0.14)	98.62 (0.20)	98.79 (0.18)
FFCNN	**99.20 (0.12)**	99.03 (0.07)	99.00 (0.14)

It is worth noting that this dataset contains 20 subjects, which means that when using leave-one-subject-out cross validation each model is trained on 19 subjects, or 95% of data. This probably accounts for better performance of some models and overall lower standard deviation when comparing both validation methods in Table 7 and 8. The proposed feature fusion-based architecture consistently provides better recognition rate, regardless of validation method or subset (binary, grayscale or depth).

Furthermore, there is significant difference in performance of large and small networks. Based on the experiments above we conclude that there is no advantage in using large multipurpose networks for a more specific task, such as hand posture recognition, as this networks did note achieve good recognition rate or real-time performance. Note that the smaller architectures could be used on a wider variety of platforms, without the need of a GPU.

4.6.3. LaRED dataset

We further compare the networks proposed by related works, namely, CNN1, CNN2 and CNN3, against the proposed FFCNN architecture on a dataset with 81,000 images. The LaRED dataset is divided in two subsets (binary and depth) and is evaluated using the holdout and leave-one-subject-out validation methods.

The results for holdout validation are presented in Table 9. The highlighted results are statistically equivalent to the best result in this table.

Table 9: Comparison of accuracy on LaRED subsets using holdout validation

Architecture model	LaRED-B	LaRED-D
CNN1	**99.60 (0.05)**	97.62 (0.21)
CNN2	99.52 (0.07)	97.01 (0.36)
CNN3	**99.57 (0.04)**	98.44 (0.16)
CNN4	98.74 (0.06)	98.15 (0.06)
FFCNN	**99.59 (0.04)**	95.50 (0.59)

Table 10 compares models using leave-one-subject-out validation. The combination of FFCNN and the depth subset obtains the best accuracy.

Table 10: Comparison of accuracy on LaRED subsets using leave-one-subject-out validation

Architecture model	LaRED-B	LaRED-D
CNN1	83.33 (0.61)	80.53 (0.44)
CNN2	84.44 (0.54)	83.08 (0.38)
CNN3	86.60 (0.28)	85.49 (0.30)
CNN4	90.41 (0.14)	92.07 (0.16)
FFCNN	91.23 (0.17)	**92.53 (0.24)**

As seen in Table 10, leave-one-subject-out cross-validation reduces the performance of all networks, compared to holdout validation. According to the Wilcoxon test, the proposed architecture still achieves a statistically better recognition rate of 92.53% on LaRED-D. In the following section, another factor that can significantly affect recognition rate is investigated. We also verify whether the difference between binary, depth and gray representations continues to be significant.

4.6.4. Impact of preserving aspect ratio

Figure 15: A gesture from the Massey dataset resized wit and without aspect ratio preservation.

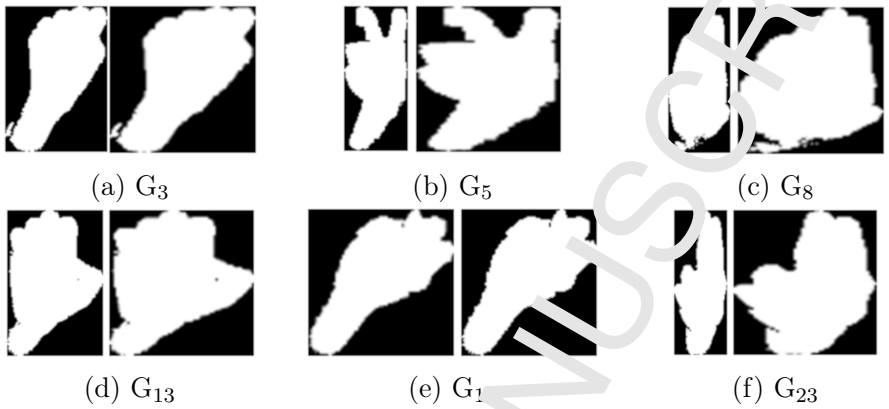
The images in a dataset are resized pr or to feature extraction and training. This is done to speed-up computations a d to make the image compatible with feature extraction methods, uch as convolutional networks. An image can be resized with or witho t pre ervation of the aspect ratio of the gesture, as illustrated in Figure 15. Al prior experiments resize the image without preserving the aspect ra o, as commonly done in the related literature [26, 27, 28, 50, 73]. The influen e of the aspect ratio on the recognition rate of the FFCNN architect re is evaluated in Table 11.

Table 11: Evaluation of the impa of as ect ratio on the FFCNN architecture using leave-one-subject-out validation. A Wilcoxo test is applied in order to verify the significance of the improvement in accu acy

Dataset	Subs t	Without aspect ratio	With aspect ratio	Significant improvement?
Massey	Gra le	84.02 (0.59)	84.27 (0.60)	No
Massey	B nar	82.58 (0.81)	83.82 (0.73)	Yes
OUHANDS	Gra ale	99.20 (0.12)	99.18 (0.16)	No
OUHANDS	Binary	99.03 (0.07)	99.45 (0.09)	Yes
OUHANDS	D epth	99.00 (0.14)	99.45 (0.14)	Yes
LaRED	Bina y	91.23 (0.17)	93.06 (0.18)	Yes
LaRED	Depth	92.53 (0.24)	93.11 (0.16)	Yes

Co sidering that the best recognition rates from Tables 8 and 10 are impr ved in Table 11 we conclude that: 1) preserving aspect ratio while rescali g p ovides a significant advantage to the FFCNN architecture and 2) th re i ignificant difference between depth and binary data. This conclu-

sions are supported by comparisons in Figures 17, 18 and 16, obtained from ten repetitions using leave-one-subject-out validation.



(a) G_3 (b) G_5 (c) G_8

(d) G_{13} (e) G_1 (f) G_{23}

Figure 16: Gestures from the LaRED-B subset with the most improved recognition by preserving aspect ratio. For quantitative comparison see Figures 17 and 18.

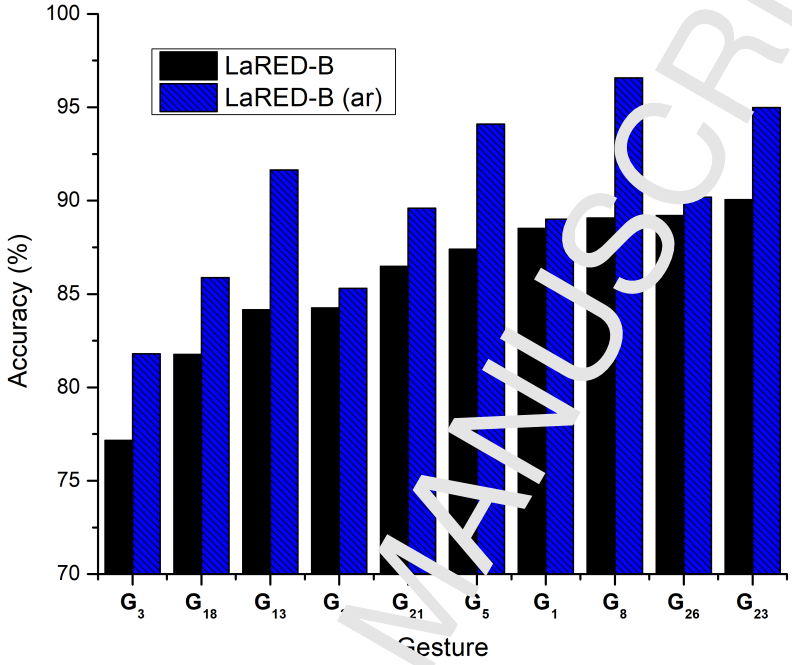
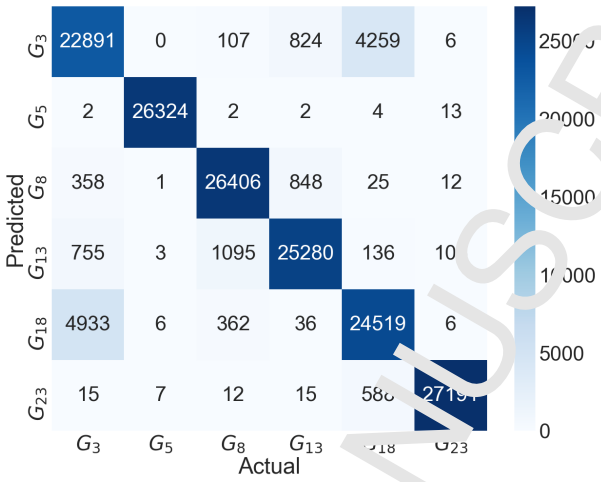


Figure 17: Accuracy of the most misclassified gestures by the FFCNN architecture in the LaRED-B subset (in black) and respective accuracy improved by preserving aspect ratio (in blue). For qualitative comparison see Figure 16.

For instance, consider recognition rate of gesture G_8, illustrated in Figure 16. The average recognition rate of this gesture is improved by 7.5% with preservation of the aspect ratio, otherwise G_8 is confused with gesture G_{13}. A confusion matrix of most misclassified gestures from LaRED-B subset is presented in Figure 18. Considering now all gestures in the LaRED-B subset, by preserving the aspect ratio the average precision is increased by 1.96% and the average recall is increased by 2.03%. In the next section, a real-time recognition system based on FFCNN using a binary mask is presented.



(a) FFCNN on LaRED-L with simple rescaling

(b) FFCNN on LaRED-B with aspect ratio preservation

Figure 18: The partial confusion matrices for a subset containing some of the most misclassified gestures by the FFCNN architecture without (18a) and with (18b) preservation of aspect ratio. Note that gestures G_8 and G_{13} are less confused in Figure 18b.

4.7. Real-time recognition

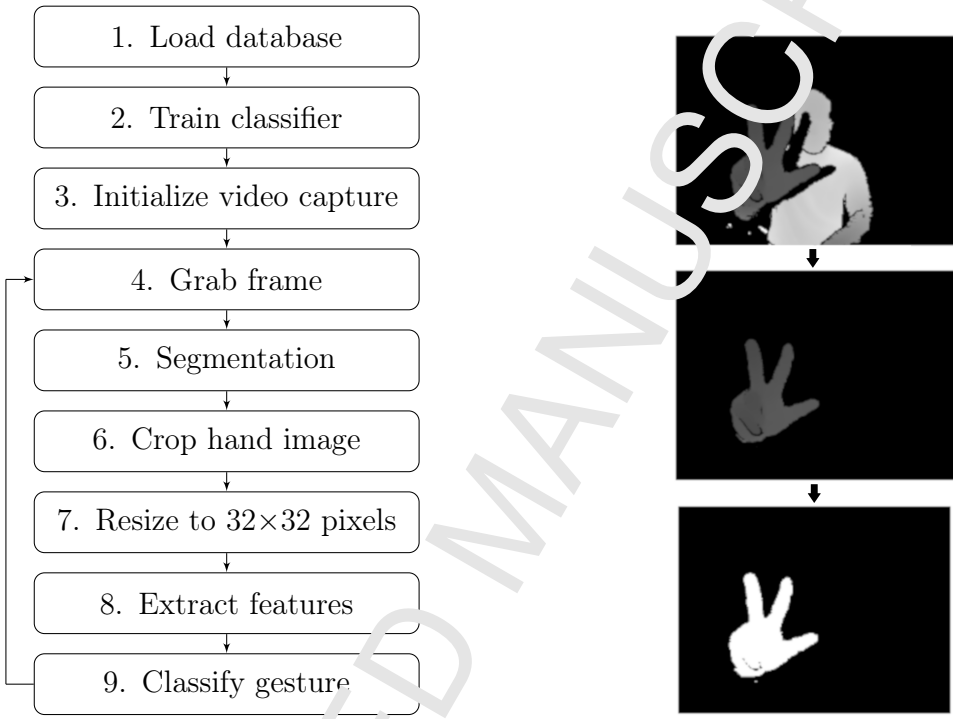
From experiments in Section 4.6, considering accuracy as well as speed, we selected FFCNN as a viable model for real-time recognition. In this section we present a system based on this model that recognizes gestures from OUHANDS and LaRED datasets in real-time.

Hand segmentation is a difficult problem and is usually treated separately from recognition. We used a 3D RealSense™ camera in order to make segmentation simpler and independent from background. The hand is assumed to be the closest object to the camera at depth d. The segmentation then consists of erasing all object that are further than $d + 10$ cm. This has the downside of the wrist occasionally being segmented along with the hand. As will be shown, the proposed recognition method is tolerant to faulty segmentation, as long as it is trained on a database that contains such cases. As illustrated in Figure 19a, the system consists of the following steps:

1. Load database: in order to speed up the training process the database (OUHANDS or LaRED) is loaded to the main memory prior to training.
2. Train classifier with random distortions: the proposed FFCNN architecture is trained on the entire dataset. The dataset is extended ten times using random rotations with an amplitude of 20° and random rescaling with an amplitude of 10%.
3. Initialize video capture: the *PyRealSense*[1] library is used for depth frame capture from the camera. Real-time recognition starts here.
4. Grab frame: a single depth frame from the camera is obtained.
5. Segmentation: the hand is assumed to be closest to the camera. Objects further than 10 cm from the tip of the hand are considered background and removed. From this point the depth image is converted to a binary image.
6. Crop hand image: only the hand is left for further processing.
7. Resize to 32×32 pixels: the segmented binary image is rescaled to 32×32 pixels, preserving the aspect ration.
8. Extract features: Hu and Zernike moments along with contour features are extracted from the binary image, which is also passed through the Gabor filter for the second convolutional channel of FFCNN.

[1] https://github.com/toinsson/pyrealsense

9. Classify gesture: the binary image and the auxiliary feature vector are presented to the previously trained network and the recognized gesture corresponds to the most activated output neuron.



(a) Flowchart of the system.

(b) Illustration of steps 4 and 5.

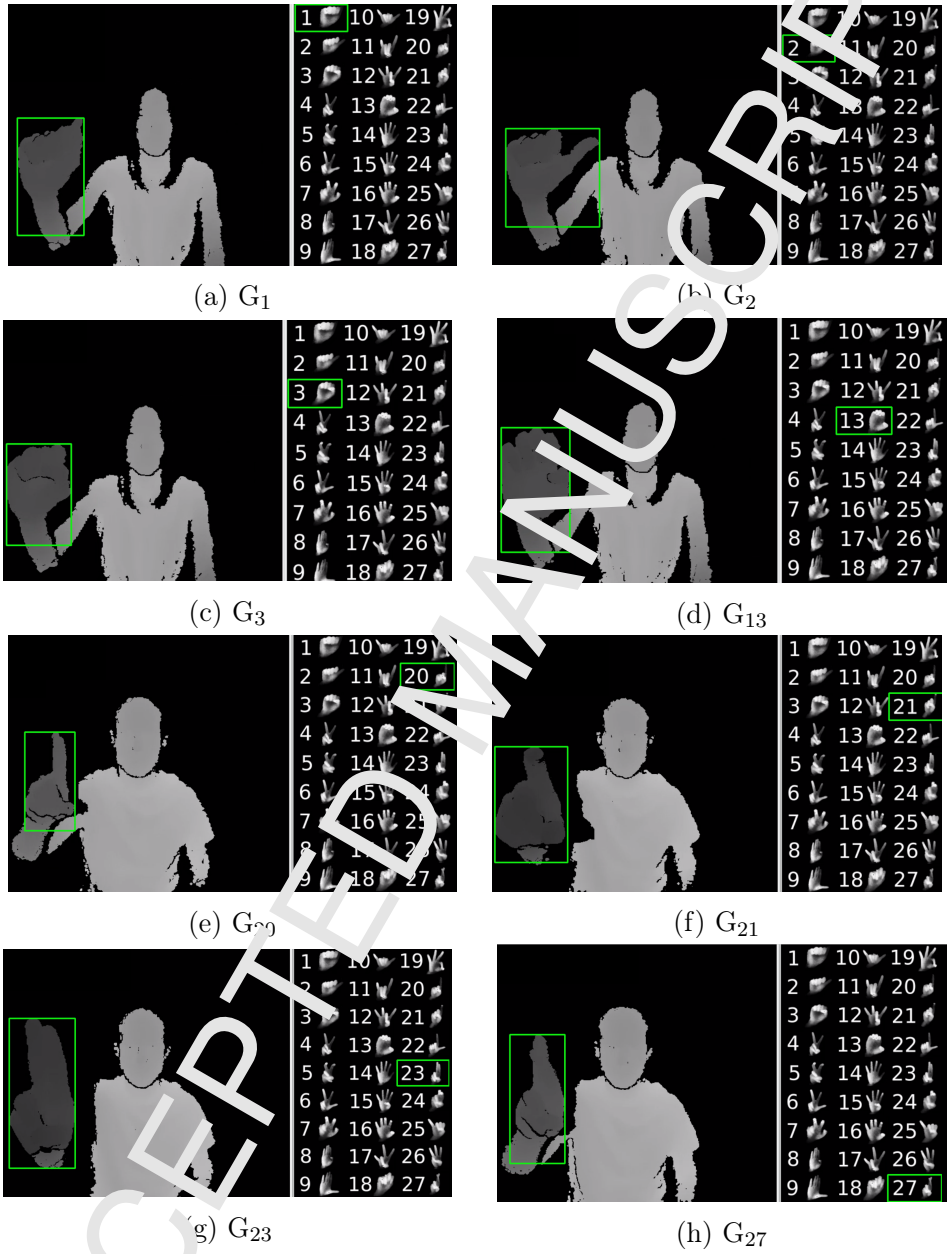
Figure 19: Real-time gesture recognition system. On average, steps 4–9 require about 26 ms.

It is worth noting that while segmentation relies on the depth sensor, the recognition is made from a binary image. Thus the proposed system can also work with common RGB cameras, requiring only the segmentation steps to be modified. The average time required for steps 5–9 can be seen in Table 12. The total time required to process image, extract features and recognize gestures is 23.02 ms, which is more than enough for recognition in real time at 30 fps. In fact the frame rate is limited by the camera, as new frames are obtained every 33.33 ms.

A video of real-time recognition of 10 postures from OUHANDS dataset and 24 postures from LaRED dataset is provided as supplementary material.

From this video it can be seen that the proposed system recognizes a variety of gestures with sufficient robustness and accuracy. Recognition of the most challenging cases from LaRED dataset is also presented in Figure 20.

The most common reason for incorrect classification is presented in Figure 21, where the forearm is segmented along with the hand. As explained in Section 4.3.2, OUHANDS dataset is also intended for evaluation of hand detection and is manually segmented. On the other hand, LaRED dataset is automatically segmented from depth images and can be used to train a more robust classifier.



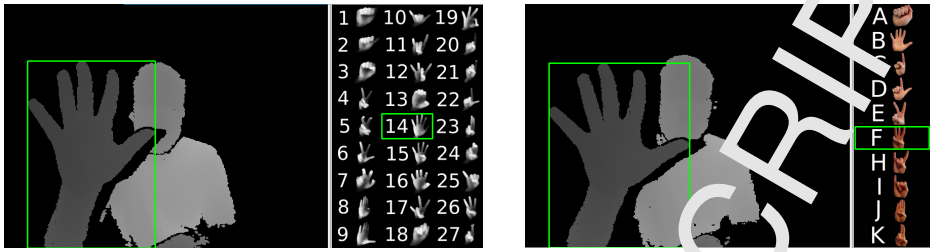
(a) G_1 (b) G_2

(c) G_3 (d) G_{13}

(e) G_{20} (f) G_{21}

(g) G_{23} (h) G_{27}

Figure 20. Some of the most similar gestures from the LaRED dataset being correctly recognized in real-time. The recognized class is highlighted from a list of possible gestures.



(a) LaRED (b) OUHANDS

Figure 21: A gesture with faulty segmentation correctly recognized from the LaRED dataset and incorrectly recognized from the OUHANDS dataset. The classifier trained on the OUHANDS dataset mistakes the actual gesture 'B' for a similar gesture 'F'.

Table 12: Average speed measured for each main component of real-time recognition.

Step	Time (ms)	%
(5) Segmentation	6.26	27
(6) Crop hand image	1.45	6
(7) Resize to 32×32	0.53	2
(8) Extract features	7.73	34
(9) Recognition	7.04	31
Total	**23.02**	100

5. Discussion

The proposed FFCNN architecture is shown to provide statistically better results to the state-of-the-art on most evaluated datasets. In this Section, the results are examined considering the influence of color space, rescaling technique and classification architecture.

On Massey dataset, the usage of grayscale images results in better accuracy, this can be attributed to a number of complex gestures. For instance, the gestures 'a' and 's' are very similar, as illustrated in Figure 22, and are better distinguished by grayscale than binary images.

As can be seen by comparing results for any model in Tables 4 and 5, leave-one-subject-out validation is more challenging for this dataset. This is probably due to a small number of subjects in the Massey dataset (5 subjects) combined with a large number of gestures (36 gestures). This can be difficult because there are variations in proportions across subjects'

(a) Gesture 'a' (b) Gesture 's'

Figure 22: Two similar gestures from the Massey ASL dataset. There is significant loss of relevant information after conversion from grayscale to binary image, which in this case may cause the two gestures to be indistinguishable from one another.

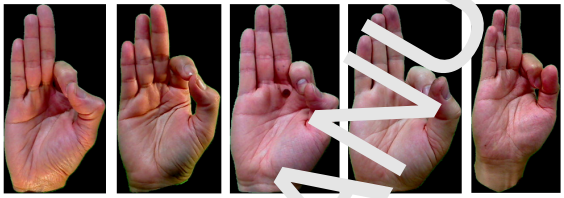


Figure 23: Gesture 'f' from the Massey dataset performed by each of the five subjects.

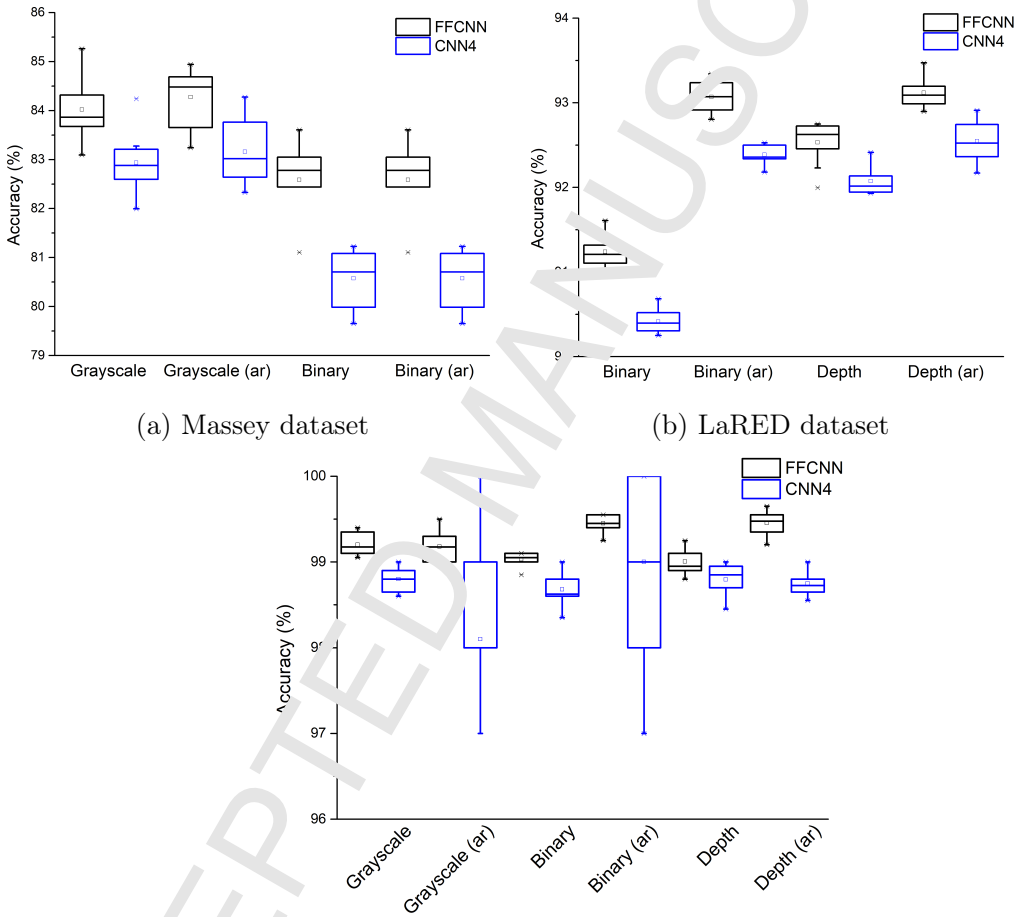
hands, as shown in Figure 23. In our experiments, subject 5 was the most challenging when selected for the Test subset. This results suggest that the Massey dataset should be extended with more subjects, which would potentially allow the classifier to perform better with new users.

Holdout validation is also less challenging than leave-one-subject-out on LaRED dataset. For instance, most models obtained close to 100% accuracy using holdout (Table 6). In the same experiment the binary subset is usually better than the depth. Unlike with Massey dataset, this suggests that gestures in LaRED dataset are recognizable only by the binary mask and depth data does not provide additional information. It is also demonstrated that if rescaling is performed with preservation of aspect ratio, FFCNN obtains statistically equivalent result for both binary and depth subsets (Table 11).

Note that use of binary data provides some benefits over depth and grayscale: 1) a binary image can be obtained from any camera, while depth image requires an RGB-D camera such as RealSense™, 2) feature extraction from binary data is potentially faster and can be done by a simpler and more robust convolutional network [27, 73, 64].

The two best performing models evaluated in this paper are CNN4 and FFCNN. Their accuracy on all subsets is compared on Figure 24. Note that

both models take advantage of preservation of aspect ratio, suggesting that this could apply to other convolutional classifiers. While FFCNN consistently provides better accuracy, CNN4 is expected to be more suitable for applications with limited computation. An optimization of FFCNN in terms of recognition time is left for future works.



(a) Massey dataset

(b) LaRED dataset

(c) OUHANDS dataset

Figure 24: Box plot comparison of FFCNN and CNN4 over datasets considered in this paper – (ar) suffix indicates that the aspect ratio has been preserved during rescaling.

Finally, FFCNN and CNN4 are also qualitatively compared against state-of-the-art methods on Table 13. Note that models by Nai et al. [65] and

Kopinski et al. [66] are proposed only for depth sensors and thus not evaluated in this work. In this case, the comparisons are based on the results provided in the papers. Overall, our experiments suggest that use of classical features contribute to increasing recognition rate of a convolutional neural network. Due to the success of deep learning, fusion of CNNs and classical features may have been overlooked for hand posture recognition. This paper advances the field by providing experimental evidence that encourages further research on the combination of classical hand posture descriptors and deep learning methods.

Table 13: Comparison with state-of-the-art methods

Method	Strengths	Weaknesses
Transfer learning with VGG16 and MobileNet	Straightforward implementation	Lower overall accuracy Slower to train and deploy
CNN1 [26]	Real-time without a GPU Straightforward implementation	Low accuracy on most evaluated subsets
CNN2 [27]	Real-time without a GPU Straightforward implementation	Low accuracy on most evaluated subsets
CNN3 [28]	Real-time without a GPU Straightforward implementation	Low accuracy on most evaluated subsets Slower than a single CNN
Depth features [65]	Very fast Good reported accuracy on a dataset with 10 classes	Applicable only on depth data
3D CNN on point clouds [66]	Good reported accuracy on a large dataset	Applicable only on depth data Recognition time not available
CNN4 (Present work)	Real-time without a GPU Good overall accuracy Straightforward implementation	Lower accuracy than FFCNN
FFCNN (Present work)	Consistently better accuracy Real-time without a GPU	Slower than CNN4 Manual feature extraction

6. Conclusion and final remarks

A novel architecture is proposed, based on a convolutional neural network and classical feature descriptors. This feature fusion-based network is thoroughly evaluated, outperforming state-of-the-art models on different benchmark datasets.

The main contributions of this paper are summarized as follows:

- Recent convolutional neural networks are evaluated on three hand posture datasets. The datasets are further divided in depth, binary and grayscale subsets.

- The comparisons are made using two validation techniques encountered in literature—holdout and leave-one-subject-out.
- A novel architecture is proposed and compared against state-of-the-art methods, considering accuracy and recognition speed. The proposed architecture is shown to outperform related methods across different datasets and any of three representations: binary, depth or grayscale.
- Resizing of gestures with constant aspect ratio is shown to have significant influence on recognition rate.
- A real-time posture recognition system is implemented with a depth camera. A demo video is provided (see Appendix A).

Based on performed experiments we can also highlight the most important findings:

- For smaller problems, such as hand posture recognition, it is preferable to use shallower, custom built convolutional networks, rather than using pretrained large ones (transfer learning), such as VGG16.
- On most of the evaluated datasets, binary images can provide a recognition rate equivalent to depth or grayscale representations. An additional advantage is the potential economy in size and computational cost of the recognition system based on binary images.
- The aspect ratio of segmented hand should be preserved during rescaling, this is shown to significantly improve the performance of our classifier (Table 11).

6.1. Future work

As future endeavours we suggest to investigate other methods for hyperparameter selection and optimization, such as multi-objective algorithms. We also intend to evaluate a greater number of hyperparameters related to micro and macro properties of the architecture, such as number of layers and methods of feature fusion. Ensemble of classifiers, such as CNN4 and FFCNN, is also expected to improve recognition results and could be evaluated on some applications. Note that an ensemble of FFCNNs only requires a single feature extraction step, as the same features would be fed to multiple convolutional channels.

A hand detection step can be introduced in order to make segmentation more robust to the position of hand and possibly eliminate the need of a depth camera. Also, hand segmentation used in this work is fairly simple and could be greatly improved.

Although the focus of this work is static hand gestures, the proposed method can be extended to recognize a wide range of dynamic gestures. This can be done by adding dynamic information, such as position, orientation and velocity to the classified posture. A sequence of postures can be recognized as a dynamic gesture by a probabilistic method, such as the Hidden Markov Model (HMM).

Finally, the FFCNN classifier trained on the OUHANDS dataset is available upon request, in order to encourage potential applications in smart houses, vehicle infotainment systems, operating theaters, among others. An embedded system for out-of-the-box hand posture and gesture recognition is also being developed.

Appendix A. Supplementary material

A demo video with real-time recognition can be found at:
https://youtu.be/kW3nnw2xrAw.

[1] S. P. Robertson, W. Zachary, J. B. Black, Cognition, computing, and cooperation, volume 2, Intellect Books, 1990.

[2] C.-C. Wang, K.-C. Wang, Hand posture recognition using Adaboost with SIFT for human robot interaction, in: Recent progress in robotics: viable robotic service to human, Springer, 2008, pp. 317–329.

[3] S. S. Rautaray, A. Agrawal, Vision based hand gesture recognition for human computer interaction: a survey, Artificial Intelligence Review 43 (2015) 1–54.

[4] H. Birk, T. B. Moeslund, C. B. Madsen, Real-time recognition of hand alphabet gestures using principal component analysis, in: Proceedings of the Scandinavian Conference on Image Analysis, volume 1, Proceedings published by various publishers, pp. 261–268.

[5] S. Vutinuntakasame, V.-r. Jaijongrak, S. Thiemjarus, An assistive body sensor network glove for speech- and hearing-impaired disabilities, in:

2011 International Conference on Body Sensor Networks, IEEE, pp. 7–12.

[6] D. H. Stefanov, Z. Bien, W.-C. Bang, The smart house for older persons and persons with physical disabilities: structure, technology arrangements, and perspectives, IEEE transactions on neural systems and rehabilitation engineering 12 (2004) 228–250.

[7] K.-H. Park, Z. Bien, J.-J. Lee, B. K. Kim, J.-T. Lim, J.-O. Kim, H. Lee, D. H. Stefanov, D.-J. Kim, J.-W. Jung, et al., Robotic smart house to assist people with movement disabilities, Autonomous Robots 22 (2007) 183–198.

[8] J. Zeng, Y. Sun, F. Wang, A natural hand gesture system for intelligent human-computer interaction and medical assistance, in: Intelligent Systems (GCIS), 2012 Third Global Congress on, IEEE, pp. 382–385.

[9] K. Ichimura, K. Magatani, Development of the bedridden person support system using hand gesture, in: 2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pp. 4550–4553.

[10] M. Teimourikia, H. Saudi, S. Comai, F. Salice, Personalized hand pose and gesture recognition system for the elderly, in: International Conference on Universal Access in Human-Computer Interaction, Springer, pp. 191–202.

[11] C. Stößel, H. Wandke, L. Blessing, Gestural interfaces for elderly users: help or hindrance?, in: International Gesture Workshop, Springer, pp. 269–280.

[12] R. Johnson, K. O'Hara, A. Sellen, C. Cousins, A. Criminisi, Exploring the potential for touchless interaction in image-guided interventional radiology, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ACM, pp. 3323–3332.

[13] J. P. Wachs, H. I. Stern, Y. Edan, M. Gillam, J. Handler, C. Feied, M. Smith, A gesture-based tool for sterile browsing of radiology images, Journal of the American Medical Informatics Association 15 (2008) 321–323.

[14] K. O'Hara, G. Gonzalez, A. Sellen, G. Penney, A. Varnavas, H. Mentis, A. Criminisi, R. Corish, M. Rouncefield, N. Dastur, et al., Touchless interaction in surgery, Communications of the ACM 57 (2014) 70–77.

[15] J. Soh, Y. Choi, Y. Park, H. S. Yang, User-friendly 3D object manipulation gesture using Kinect, in: Proceedings of the 12th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and its Applications in Industry, VRCAI '13, ACM, New York, NY, USA, 2013, pp. 231–234.

[16] M. Maidi, M. Preda, Interactive media control using natural interaction-based Kinect, in: Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on, IEEE, pp. 1812–1815.

[17] H. Kang, C. W. Lee, K. Jung, Recognition-based gesture spotting in video games, Pattern Recognition Letters 25 (2004) 1701 – 1714.

[18] M. Geiger, M. Zobl, K. Bengler, M. Lang, Intermodal differences in distraction effects while controlling automotive user interfaces, in: Proc. Int. Conf. on Human-Computer Interaction HCI# 2001, New Orleans, Louisiana, USA.

[19] M. Zobl, M. Geiger, B. Schuller, M. Lang, G. Rigoll, A real-time system for hand gesture controlled operation of in-car devices, in: Multimedia and Expo, 2003. ICME'03. Proceedings. 2003 International Conference on, volume 3, IEEE, pp. III–541.

[20] S. Loehmann, M. Knobel, M. Lamara, A. Butz, Culturally independent gestures for in-car interactions, in: IFIP Conference on Human-Computer Interaction, Springer, pp. 538–545.

[21] A. K. Mazima, E. Özgür, M. Çetin, A fast algorithm for vision-based hand gesture recognition for robot control, in: 2006 IEEE 14th Signal Processing and Communications Applications, pp. 1–4.

[22] M. Van den Bergh, D. Carton, R. De Nijs, N. Mitsou, C. Landsiedel, K. Kuehnlenz, D. Wollherr, L. Van Gool, M. Buss, Real-time 3D hand gesture interaction with a robot for understanding directions from humans, in: RO-MAN, 2011 IEEE, IEEE, pp. 357–362.

[23] K. Qian, J. Niu, H. Yang, Developing a gesture based remote human-robot interaction system using kinect, International Journal of Smart Home 7 (2013).

[24] J. Schmidhuber, Deep learning in neural networks: An overview, Neural networks 61 (2015) 85–117.

[25] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, MobileNets: Efficient convolutional neural networks for mobile vision applications, arXiv preprint arXiv:1704.04861 (2017).

[26] O. K. Oyedotun, A. Khashman, Deep learning in vision-based static hand gesture recognition, Neural Computing and Applications (2016) 1–11.

[27] E. Nasr-Esfahani, N. Karimi, S. Soroushmehr, M. H. Jafari, M. A. Khorsandi, S. Samavi, K. Najarian, Hand gesture recognition for contactless device control in operating rooms, arXiv preprint arXiv:1611.04138 (2016).

[28] P. Ji, C. Wu, X. Xu, A. Song, H. Li, Vision-based posture recognition using an ensemble classifier and a vote filter, in: International Symposium on Optoelectronic Technology and Application 2016, International Society for Optics and Photonics, pp. 101571J–101571J.

[29] A. Barczak, N. Reyes, M. Abastillas, A. Piccio, T. Susnjak, A new 2D static hand gesture colour image dataset for ASL gestures (2011).

[30] Y.-S. Hsiao, J. Sanchez-Riera, T. Lim, K.-L. Hua, W.-H. Cheng, LaRED: a large RGB-D extensible hand gesture dataset, in: Proceedings of the 5th ACM Multimedia Systems Conference, ACM, pp. 53–58.

[31] M. Matilainen, P. Sangi, J. Holappa, O. Silvén, OUHANDS database for hand detection and pose recognition, in: Image Processing Theory Tools and Applications (IPTA), 2016 6th International Conference on, IEEE, pp. 1–5.

[32] I. Rasines, A. Remazeilles, P. M. Iriondo Bengoa, Feature selection for hand pose recognition in human-robot object exchange scenario, in:

Emerging Technology and Factory Automation (ETFA), 2014 IEEE, IEEE, pp. 1–8.

[33] J. Suarez, R. R. Murphy, Hand gesture recognition with depth images: a review, in: Ro-man, 2012 IEEE, IEEE, pp. 411–417.

[34] P. K. Pisharady, M. Saerbeck, Recent methods and databases in vision-based hand gesture recognition: A review, Computer Vision and Image Understanding 141 (2015) 152–165.

[35] T. D'Orazio, R. Marani, V. Renó, G. Cicirelli, Recent trends in gesture recognition: how depth data has improved classical approaches, Image and Vision Computing 52 (2016) 56–72.

[36] J. Sanchez-Riera, K.-L. Hua, Y.-S. Hsiao, T. Lim, S. C. Hidayati, W.-H. Cheng, A comparative study of data fusion for RGB-D based visual recognition, Pattern Recognition Letters 73 (2016) 1–6.

[37] P. Barros, S. Magg, C. Weber, S. Wermter, A multichannel convolutional neural network for hand posture recognition, in: International Conference on Artificial Neural Networks, Springer, pp. 403–410.

[38] M.-K. Hu, Visual pattern recognition by moment invariants, Information Theory, IRE Transactions on 8 (1962) 179–187.

[39] F. Zernike, Beugungstheorie des Schneidenverfahrens und seiner verbesserten Form, der Phasenkontrastmethode, Physica 1 (1934) 689–704.

[40] C.-H. Teh, R. T. Chin, On image analysis by the methods of moments, Pattern Analysis and Machine Intelligence, IEEE Transactions on 10 (1988) 496–513.

[41] A. Tahmasbi, F. Saki, S. B. Shokouhi, Classification of benign and malignant masses based on Zernike moments, Computers in biology and medicine 41 (2011) 726–735.

[42] S. Marčelja, Mathematical description of the responses of simple cortical cells, JOSA 70 (1980) 1297–1300.

[43] A. K. Jain, F. Farrokhnia, Unsupervised texture segmentation using Gabor filters, Pattern recognition 24 (1991) 1167–1186.

[44] L. Deng, D. Yu, et al., Deep learning: methods and applications, Foundations and Trends® in Signal Processing 7 (2014) 197–387.

[45] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (2015) 436–444.

[46] A. Krizhevsky, I. Sutskever, G. E. Hinton, ImageNet classification with deep convolutional neural networks, in: Advances in neural information processing systems, pp. 1097–1105.

[47] S. Ren, K. He, R. Girshick, J. Sun, Faster r-cnn: Towards real-time object detection with region proposal networks, in: Advances in neural information processing systems, pp. 91–99.

[48] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, S. Thrun, Dermatologist-level classification of skin cancer with deep neural networks, Nature 542 (2017) 115.

[49] S. J. Pan, Q. Yang, et al., A survey on transfer learning, IEEE Transactions on knowledge and data engineering 22 (2010) 1345–1359.

[50] M. A. Aowal, A. S. Zaman, S. Mahbubur Rahman, D. Hatzinakos, Static hand gesture recognition using discriminative 2D Zernike moments, in: TENCON 2014-2014 IEEE Region 10 Conference, IEEE, pp. 1–5.

[51] B. P. Kumar, M. Manjunatha, A hybrid gesture recognition method for American sign language, Indian Journal of Science and Technology 10 (2017).

[52] C. W. Ng, S. Ranganath, Real-time gesture recognition system and application, Image and Vision computing 20 (2002) 993–1007.

[53] R. K. Sabhara, C.-P. Lee, K.-M. Lim, Comparative study of Hu moments and Zernike moments in object recognition, SmartCR 3 (2013) 166–173.

[54] K. Cabiniano-Rodríguez, G. Cámara-Chávez, D. Menotti, Hu and Zernike moments for sign language recognition, in: Proceedings of international conference on image processing, computer vision, and pattern recognition, pp. 1–5.

[55] M. Wang, W.-Y. Chen, X. D. Li, Hand gesture recognition using valley circle feature and Hu's moments technique for robot movement control, Measurement 94 (2016) 734–744.

[56] Y. Guo, C. Liu, S. Gong, Improved algorithm for Zernike moments, in: Control, Automation and Information Sciences (ICCAIS), 2015 International Conference on, IEEE, pp. 307–312.

[57] M. Avraam, Static gesture recognition combining graph and appearance features, International Journal of Advanced Research in Artificial Intelligence (IJARAI) 3 (2014).

[58] S. Yang, Robust human computer interaction using dynamic hand gesture recognition, Ph.D. thesis, School of Electrical, Computer and Telecommunications Engineering, University of Wollongong, 2016.

[59] J. M. Palacios, C. Sagüés, E. Montijano, S. Llorente, Human-computer interaction based on hand gestures using RGB-D sensors, Sensors 13 (2013) 11842–11860.

[60] G. Plouffe, A.-M. Cretu, Static and dynamic hand gesture recognition in depth data using dynamic time warping, IEEE Transactions on Instrumentation and Measurement 65 (2016) 305–316.

[61] J. Huang, W. Zhou, H. Li, W. Li, Sign language recognition using real-sense, in: Signal and Information Processing (ChinaSIP), 2015 IEEE China Summit and International Conference on, IEEE, pp. 166–170.

[62] D.-L. Dinh, J. T. Kim, T.-S. Kim, Hand gesture recognition and interface via a depth imaging sensor for smart home appliances, Energy Procedia 62 (2014) 576–582.

[63] H. Cheng, L. Yang, Z. Liu, Survey on 3D hand gesture recognition, IEEE Transactions on Circuits and Systems for Video Technology 26 (2016) 1659–1673.

[64] L. F. Camboim, R. M. Macieira, F. M. Neto, E. Barros, T. B. Ludermir, C. Zanchettin, An efficient static gesture recognizer embedded system based on elm pattern recognition algorithm, Journal of Systems Architecture 68 (2016) 1–16.

[65] W. Nai, Y. Liu, D. Rempel, Y. Wang, Fast hand posture classification using depth features extracted from random line segments, Pattern Recognition 65 (2017) 1–10.

[66] T. Kopinski, F. Sachara, A. Gepperth, U. Handmann, A deep learning approach for hand posture recognition from depth data, in: International Conference on Artificial Neural Networks, Springer, pp. 179–186.

[67] S. F. Chevtchenko, R. F. Vale, V. Macario, Multi-objective optimization for hand posture recognition, Expert Systems with Applications 92 (2018) 170–181.

[68] S. Wu, Y.-C. Chen, X. Li, A.-C. Wu, J.-J. You, W.-S. Zheng, An enhanced deep feature representation for person re-identification, in: Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on, IEEE, pp. 1–8.

[69] S. P. Suggu, K. N. Goutham, M. K. Chinnakotla, M. Shrivastava, Deep feature fusion network for answer quality prediction in community question answering, arXiv preprint arXiv:1606.07103 (2016).

[70] J. S. Bergstra, R. Bardenet, Y. Bengio, B. Kégl, Algorithms for hyper-parameter optimization, in: Advances in Neural Information Processing Systems, pp. 2546–2554.

[71] T. Domhan, J. T. Springenberg, F. Hutter, Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves, in: IJCAI, pp. 3460–3468.

[72] J. Bergstra, D. Yamins, D. Cox, Making a science of model search: hyperparameter optimization in hundreds of dimensions for vision architectures, in: International Conference on Machine Learning, pp. 115–123.

[73] T. Yamashita, T. Watasue, Hand posture recognition based on bottom-up structured deep convolutional neural network with curriculum learning, in: Image Processing (ICIP), 2014 IEEE International Conference on, IEEE, pp. 853–857.

[74] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556 (2014).

[75] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine learning research 7 (2006) 1–30.

Recent
datasets
Different image representations and validation techniques are used for evaluation
A novel architecture is proposed and compared against state-of-the-art methods
The proposed model outperforms most ot compared methods in accuracy and recognition speed
A real-time posture recognition system is implemented with a dept' camera