Sérgio Chevtchenko

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

Brasil

2018

Sérgio Chevtchenko

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

Dissertation presented to the graduate program in Applied Informatics of Federal Rural University of Pernambuco as partial fulfillment of the requirements for a Master of Science degree.

Universidade Federal Rural de Pernambuco

Programa de Pós-Graduação em Informática Aplicada

Supervisor: Filipe Cordeiro

Co-supervisor: Valmir Macário

Brasil

2018

This work is dedicated to my grandfather, electrical engineer Sergej Denisovitch Chevtchenko.

Acknowledgements

Special thanks go to my advisors, Filipe Cordeiro and Valmir Macário (UFRPE) for their patience, motivation and dedication. I would like to show my sincere gratitude to Oleg Krasilnikov (UFPE, *in memoriam*), Paulo Sérgio Brandão de Nascimento (IFPE), Diogo Roberto Raposo de Freitas (UNINASSAU) and several other teachers, both in schools and universities. Last but by no means least, thanks to my family and friends for support and encouragement!

Resumo

O uso de gestos de mão é uma maneira intuitiva e versátil para humanos interagirem com computadores. Este trabalho tem como foco o reconhecimento de gestos estáticos, também conhecidos como posturas de mão. Um bom sistema de reconhecimento de gestos deve suportar variações na imagem, como de escala, iluminação e rotação, além de ser capaz de funcionar em tempo real. Considerando o sucesso recente de redes neurais convolutivas e robustez de técnicas tradicionais, esta dissertação apresenta uma nova arquitetura baseada em redes convolutivas para reconhecimento de gestos com acurácia e em tempo real. A arquitetura proposta combina redes convolutivas com descritores de características tradicionais. Os hiperparâmetros que descrevem esta nova rede são selecionados de forma automática usando um algoritmo de otimização. As características tradicionais são extraídas da imagem usando momentos de Zernike, momentos de Hu, filtros de Gabor e propriedades de contorno da mão. Estas características são usadas para complementar o conjunto de informações disponível para a camada de classificação da rede convolutiva. A arquitetura proposta é comparada com modelos de redes convolutivas propostos recentemente. Para isso são usadas três bases de dados de gestos estáticos de mão. Para verificar como a representação da imagem pode influenciar nos classificadores considerados nesse trabalho, as bases de dados são subdivididas em representações por profundidade, escala de cinza e binárias. Além disso, as arquiteturas são comparadas em termos de velocidade e acurácia de classificação, usando reescalonamento com e sem preservação de *aspect ratio* e dois métodos de validação comumente empregados no contexto de reconhecimento de gestos: *holdout* e *leave-one-subject-out*. É demonstrado experimentalmente que a arquitetura proposta supera o estado da arte com reconhecimento de gestos em tempo real, sendo robusta em diferentes representações e escalas da imagem. Foi registrada uma melhora de até 5.93% em comparação ao melhor modelo existente em uma base de dados RGBD com 81,000 imagens e 27 classes de gestos. Além disso, é disponibilizado um vídeo demostrando reconhecimento em tempo real de até 27 gestos estáticos de mão a 30 quadros por segundo, usando uma câmera 3D.

Palavras-chave: posturas de mão, redes neurais convolutivas, seleção de hiperparâmetros.

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 work focuses on recognition of static hand gestures, also known as hand postures. A good hand posture recognition system has to be both robust to image variations and capable of real-time performance. Considering the recent success of convolutional neural networks (CNNs) and robustness of more traditional methods, this dissertation presents a novel architecture, combining a CNN and several traditional feature extractors, capable of accurate and real-time hand posture recognition. Several hyperparameters present in the proposed architecture are automatically selected by a model optimization algorithm. The traditional features are extracted from Zernike moments, Hu moments, Gabor filters and properties of the hand contour. This features are used to complement the information available to the classification layer of a CNN. Besides the proposed architecture, recent convolutional neural networks are evaluated on three distinct benchmarking datasets. This datasets are further divided in depth, binary and grayscale subsets in order to investigate the influence of image representation on recognition accuracy. Furthermore, architectures are compared in terms of speed and accuracy using rescaling with and without preserving aspect ratio and two common validation techniques: holdout and leave-one-subject-out. The proposed architecture is shown to obtain state-of-the art recognition rate in real-time, while being robust to different image representations and scalings. A recognition improvement of 5.93% on current best model is achieved on an RGBD dataset containing 81,000 images of 27 hand postures. A demo video is provided as supplementary material, containing real-time recognition by the proposed network of up to 27 gestures at 30 fps from a 3D camera.

Key-words: hand postures, convolutional neural networks, deep learning, hyperparameter selection.

List of Figures

List of Tables

List of abbreviations and acronyms

AI	Artificial intelligence
Alex-net	A convolutional neural network that won the 2012 ImageNet Large Scale Visual Recognition Challenge
ASL	American sign language
CNN	Convolutional neural network
DZM	Discriminative Zernike moment
FC	Fully connected layer of a convolutional neural network
FD	Fourier descriptor
FFCNN	Feature fusion-based convolutional neural network proposed in this work
G_i	i-th gesture among the 27 gestures from the LaRED dataset
GPU	Graphics processing unit
HCI	Human-computer interaction
HMM	Hidden Markov model
ImageNet	A large visual database
LaRED	A large RGB-D database of hand postures
LaRED-B	Subset of LaRED composed of binary images
LaRED-D	Subset of LaRED composed of depth images
LBP	Local binary patterns
LGN	Lateral geniculate nucleus
OUHANDS	An RGB-D dataset for recognition and segmentation of hand postures
OUHANDS-B	Subset of OUHANDS composed of binary images
OUHANDS-D	Subset of OUHANDS composed of depth images
OUHANDS-G	Subset of OUHANDS composed of grayscale images
Massey	A dataset of hand postures from ASL

Massey-B	Subset of Massey composed of binary images
Massey-G	Subset of Massey composed of grayscale images
MobileNet	A convolutional neural network proposed for recognition on mobile devices
PCA	Principal component analysis
RBF	Radial basis function
ReLU	Rectified linear unit
RGB	Color model with red, green and blue components
RGB-D	Depth component considered with RGB components
SAE	Stacked autoencoder
SDAE	Stacked denoising autoencoders
SP	Super pixels
SURF	Speeded up robust features
SVM	Support vector machine
TPE	Tree-structured Parzen estimator
V1	Primary visual cortex
VGG16	A deep convolutional neural network proposed in 2014
ZM	Zernike moment

Contents

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 (ROBERTSON; ZACHARY; BLACK, 1990) and gestures (WANG; WANG, 2008; RAUTARAY; AGRAWAL, 2015), 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. Even static hand gestures, also known as hand postures, can be used to provide an intuitive interaction with computers (WANG; WANG, 2008). Additionally, it has been shown that employing hand postures can accommodate the deaf or hearing impaired (BIRK; MOESLUND; MADSEN, 1997), who mainly use hands for communication, elderly people (STEFANOV; BIEN; BANG, 2004), and people who are bedridden (ICHIMURA; MAGATANI, 2015) or have physical disabilities (STEFANOV; BIEN; BANG, 2004; ICHIMURA; MAGATANI, 2015).

Hand gesture recognition can serve as an interface to facilitate interaction with components of smart houses. In particular, this type of interaction can be targeted at groups with specific needs, as shown by Stefanov et al. (2004), Ichimura and Magatani (2015) and Park et al. (2007). These works 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.

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 touchscreens (JOHNSON et al., 2011). This can increase efficiency and lower the probability of complications in medical procedures. In the studies of Johnson et al. (2011) and O'Hara et al. (2014), 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, such as providing an interface for manipulating objects in virtual environments (SOH et al., 2013), for controlling multimedia devices (MAIDI; PREDA, 2013) and video games (KANG; LEE; JUNG, 2004), as well as possibly acting as a less distracting user interface for infotainment systems in vehicles (ZOBL et al., 2003). Control of industrial and commercial robots (MALIMA; ÖZGÜR; ÇETIN, 2006; BERGH et al., 2011) are other possible applications of such systems.

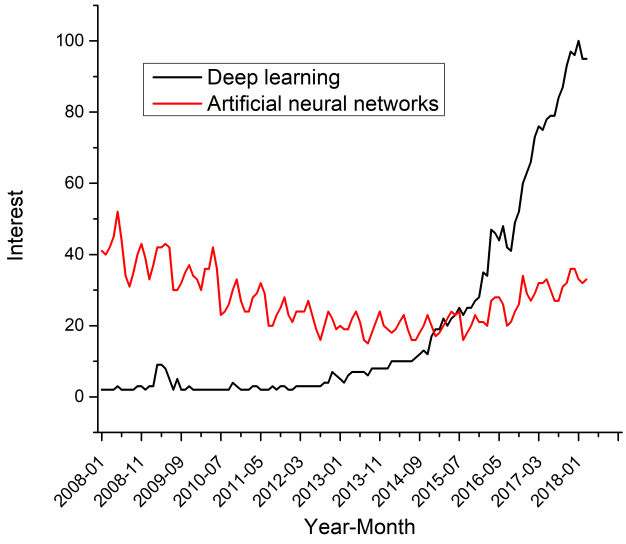
Machine learning is a major area of artificial intelligence (AI) that assists humans in a plethora of real-world problems, such as hand gesture recognition. This problems can be very complex and the design of a human-made solution for them may require strenuous efforts, especially when they have high abstraction or are influenced by large quantities of data (GOODFELLOW et al., 2016). Deep learning is a subfield of machine learning that has gathered significant attention. It can be difficult to find out what is relevant and what exerts little or no influence on each task, which is the kind of knowledge needed to design features to represent the data in question. That is why AI systems capable of learning from the data presented to them are so significant in contributing to redirecting human endeavor to other issues worthy of attention. Deep learning, on the other hand, requires no prior representation or description of the problem it is employed to solve. This is due to deep learning being an approach in representation learning, which is a set of techniques that learn the representations (or features) from the raw data.

Existing methods in deep learning seek to "understand the world in terms of a hierarchy of concepts, with each concept defined in terms of its relation to simpler concepts" (GOODFELLOW et al., 2016, p. 1). For instance, in tasks that deal with images, although all the information initially available to deep learning methods is a collection of pixels and their individual values, the network gradually learns from the input specific features that are related to properties (e.g., edges, corners, and even more advanced shapes) found in the image. The learned features grow in abstraction, with more abstract learned representations being described in terms of simpler, previously learned, representations. The combination of all learned features, in turn, describes the relevant objects in the image.

There is no easy way to mathematically conceptualize some real-world problems due to the many unknown variables and their interactions and, for this reason, statistical models may not be suitable as an approach to solving them. In numerous of these cases, deep learning architectures can be employed and also save time that would otherwise be spent in feature engineering. This characteristic also makes deep learning networks adaptable to various types of input, including images, videos, sound and text. Having said that, there are several elements that motivate working with deep learning in object recognition tasks. When compared to other machine learning methods, deep learning networks perform more intensive computations involving floating point operations, such as with matrices. Processing in deep networks, as with other neural networks, can be highly parallelized and, because of this, the market of graphics processing units (GPUs), with the decline of GPU prices, is making a positive impact in deep learning applications by offering enhanced performance necessary to train deep learning models (SCHMIDHUBER, 2015; DENG; YU, 2014; LECUN; BENGIO; HINTON, 2015).

A type of deep network that is commonly used in computer vision is the convolutional neural network (CNN). In CNN architectures, the combination of specialized layers

Figure 1 – Interest in artificial neural networks and deep learning over time in the last ten years



Source – Data acquired from Google Trends (<www.google.com/trends>)

is one of the important factors that influence the capabilities of these networks to learn features, but is not the only one responsible for their success. New training algorithms, better GPUs and other enhancements also make CNNs more competitive in machine learning tasks, contributing to state-of-the-art performances (SCHMIDHUBER, 2015; LECUN; BENGIO; HINTON, 2015).

In summation, the use of deep learning and its techniques is motivated by the many advantages they bring. The above points gain strength when considering the growth in popularity of deep learning networks when compared to classical artificial neural networks. As shown in the graph in Figure 1 with data acquired from Google Trends, in the last decade interest in classical neural networks has remained roughly constant, whereas interest in deep learning has risen from a plateau and currently greatly outmatches the former by a large margin.

Considering the success of CNNs in various applications involving image recognition, several works have applied them for recognition of static and dynamic gestures (HSIAO et al., 2014; BARROS et al., 2014; OYEDOTUN; KHASHMAN, 2016; SANCHEZ-RIERA et al., 2016; NASR-ESFAHANI et al., 2016; JI et al., 2016). This work focuses on static hand gestures and advances the current field by i) proposing a new architecture with fusion of classical image descriptors and convolutional filters, ii) thoroughly evaluating the proposed and related architectures on three distinct benchmarking datasets, considering various preprocessing parameters such as colour space and rescaling method, and iii) demonstrating real-time capability of the proposed system. The choice of classical image

descriptors is partially based on our previous work (CHEVTCHENKO; VALE; MACARIO, 2018), where we compare several classical feature descriptors for hand posture recognition and rank them by accuracy and speed.

1.1 Research problem

An effective hand posture recognition system faces several challenges. Hand detection should work with complex background and variations in illumination and scale. Although it is not the main focus of this work, a 3D camera is used in order to make hand detection more robust. However, the classifier proposed here does not rely on a depth sensor and could be used with a common camera. The influence of preprocessing methods, such as change of colour space, scale and aspect ratio, are experimentally evaluated on benchmarking datasets.

A good recognition method must also aim for high accuracy, but in the circumstances of hand posture recognition, real-time capability is also desirable. Deep neural networks, like convolutional neural networks (CNNs), have gained attention for performing well in pattern recognition compared to shallow networks and other methods (SCHMIDHUBER, 2015). 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 (HOWARD et al., 2017; IANDOLA et al., 2016). 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 (2016), Nasr et al. (2016) and Ji et al. (2016). In order to diversify the information from features, an architecture model of a convolutional neural network is proposed that does not rely only on feature learning, but also merges features output from classical image descriptors.

1.2 Research goals

1.2.1 Main goal

The objective of this research 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.

1.2.2 Specific goals

- Propose a real-time capable scheme for posture recognition. This is tested with a system implemented with a 3D RealSense™ camera.
- Achieve state-of-the-art performance and compare with other recently proposed models in terms of recognition accuracy and speed.

1.3 Outline

This remainder of this text is structured in the following way. Chapter 2 contextualizes the problem addressed in this study. Chapter 3 presents the related literature. Chapter 4 describes the proposed convolutional neural network. Experimental setup and evaluation details, including datasets and the design rationale behind implemented architecture models, are given in Chapter 5. Finally, Chapter 7 summarizes the main results and contributions of this work and raises ideas for future endeavors.

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 (RAUTARAY; AGRAWAL, 2015). However, when the recognition system can operate at image acquisition rate, tracking is not necessary (RAUTARAY; AGRAWAL, 2015). Sections 2.1.1 to 2.1.4 contain a brief introduction to the main steps involved in hand posture recognition.

2.1.1 Image acquisition

Generally speaking, image acquisition devices are colour or grayscale cameras, that may also contain a depth sensor. Some less common approaches may involve inference of depth data from multiple or dynamic light sources (SUAREZ; MURPHY, 2012). The speed and resolution of the camera should also be taken into account for real-world applications. A fast and high resolution camera may be beneficial for recognition of subtle variations between gestures, but may be limited by the speed of the image processing and recognition methods. On the other hand, fast recognition may allow multiple inferences to be made from a single frame, thus potentially improving recognition accuracy.

2.1.2 Hand detection and segmentation

As the cropped image in Figure 2 suggests, detection step is used to identify the region that contains the hand. In some cases, such as when depth data is used, it is possible to perform feature extraction and recognition directly after detection, without removal of the background. Next, segmentation completely separates the region that represents the hand from the background. The output of segmentation is a description of the pixels in a way that discriminates between the hand and everything else in the image. If the classification task is simple enough, this can generate a binary image (e.g., pixels that are part of the hand have value of 1 and the rest have value of 0). The example in Figure 2 corresponds to a binary output to the segmentation step. In more complex situations, especially when the shape of the gesture does not provide sufficient discrimination between some gestures, depth images can be used. In this case, each pixel

has a depth value that is proportional to its distance from the camera. A threshold is established to distinguish between hand and background, but depth information may or may not be kept for classification.

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 (JI et al., 2016) to maximize contrast between hand and background. With recent advances in portable 3D cameras, such as Kinect™ for Windows® and Intel® RealSense™, it is now possible to significantly improve segmentation and recognition accuracy even in uncontrolled environments (SUAREZ; MURPHY, 2012; PISHARADY; SAERBECK, 2015; D'ORAZIO et al., 2016) using depth information.

2.1.3 Extraction of hand features

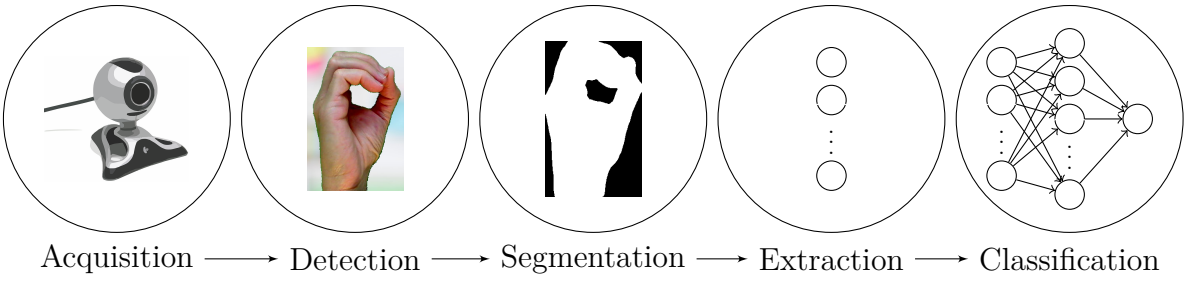
Following the acquisition and segmentation steps, the feature extraction phase is in charge of obtaining a representative feature set that will be fed to a classifier. It is generally desirable to have features that are as stable as possible to variations in illumination, scale, translation and rotation. Some commonly used invariant features are described in Section 2.2. 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 several convolutional filters are trained in a similar manner to a multilayer perceptron (MLP) (OYEDOTUN; KHASHMAN, 2016; NASR-ESFAHANI et al., 2016; JI et al., 2016; SANCHEZ-RIERA et al., 2016; BARROS et al., 2014). Convolutional neural networks are explained in more details in Section 2.3.

2.1.4 Hand posture classification

The classifier in Figure 2 depicts an MLP, but as long as appropriate features are extracted, any classifier can be used. Besides artificial neural networks, some common classifiers include k-nearest neighbors, random forests, Support Vector Machine, Dynamic time warping and finite state machines (RAUTARAY; AGRAWAL, 2015). As well as with other components of the hand posture recognition system depicted in Figure 2, it is usually necessary to consider a trade-off between accuracy and speed of the classifier.

In conclusion, image acquisition, detection and segmentation, along with feature extraction and classification are the main steps that comprise a hand posture recognition system (RASINES; REMAZEILLES; BENGOA, 2014). These steps are outlined in Figure 2.

Figure 2 – A diagram of the main steps of a hand posture recognition system



Source – The author

2.2 Feature extraction

Some commonly used feature extraction methods are employed in this work, as shown in the following subsections. These features are integrated into a classifier in order to provide additional information about the image.

As mentioned in Subsection 2.1.3, image descriptors are numerical representations of certain elementary characteristics of the image. This descriptors can synthetically represent color, shape, texture, among other common features of an image. Some descriptors, such as Hu and Zernike moments are specifically designed to be robust to image variations, such as scale and rotation. Besides providing some key information about an image, this invariant features can be expected to provide similar result to the same image under some distortions.

2.2.1 Hu moments

The set of Hu moment invariants (HU, 1962) 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 work are extracted from a binary image and are defined in Equations (2.1)–(2.7):

$$I_1 = \eta_{20} + \eta_{02}, \tag{2.1}$$

$$I_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2, \tag{2.2}$$

$$I_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2, \tag{2.3}$$

$$I_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2, \tag{2.4}$$

$$\begin{aligned} I_5 = {} & (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\ & + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2], \end{aligned} \tag{2.5}$$

$$\begin{aligned} I_6 = {} & (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\ & + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}), \end{aligned} \tag{2.6}$$

$$\begin{aligned} I_7 = {} & (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\ & - (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 \\ & - (\eta_{21} + \eta_{03})^2], \end{aligned} \tag{2.7}$$

where η_{ij} are scale invariant moments, as defined by Hu (1962).

2.2.2 Zernike moments

The Zernike polynomials were first proposed in 1934 by Frits Zernike (ZERNIKE, 1934). The corresponding moments are known to be somewhat invariant to rotation, and can be modified to also be stable under scale and translation (TEH; CHIN, 1988).

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

$$\begin{aligned} Z_{n,m} = \frac{n+1}{\pi} \sum_x \sum_y f(x, y) V_{nm}(x, y)^*, \\ n - |m| \text{ even}, \quad |m| \leq n, \end{aligned} \tag{2.8}$$

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:

$$V_{nm}(x, y) = V_{nm}(r, \theta) = R_{nm}(r) e^{jm\theta}, \tag{2.9}$$

$$R_{nm}(r) = \sum_{s=0}^{\frac{n-|m|}{2}} (-1)^s \frac{(n-s)!}{s! \left(\frac{n+|m|}{2} - s\right)! \left(\frac{n-|m|}{2} - s\right)!} r^{n-2s}, \tag{2.10}$$

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:

$$r = \sqrt{x^2 + y^2}, \quad \theta = tan^{-1}\left(\frac{y}{x}\right) \tag{2.11}$$

See Tahmasbi et al. (2011) 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 mathematical representation of simple visual cells in mammals (MARĈELJA, 1980), that encode both spatial and spatial frequency variables of an image. It is widely used to recognize texture features by convolving an image with the filter kernel (JAIN; FARROKHNIA, 1991). A two-dimensional Gabor filter is given by:

$$G(x, y) = \exp\left(-\frac{1}{2}\left[\frac{x_\theta^2}{\sigma_x^2} + \frac{y_\theta^2}{\sigma_y^2}\right]\right)\cos\left(\frac{2\pi x_\theta}{\lambda} + \psi\right) \tag{2.12}$$

$$x_\theta = x\cos(\theta) + y\sin(\theta) \tag{2.13}$$

$$y_\theta = -x\sin(\theta) + y\cos(\theta) \tag{2.14}$$

$$\sigma_x = \sigma, \quad \sigma_y = \frac{\sigma}{\gamma} \tag{2.15}$$

The parameters of this filter are:

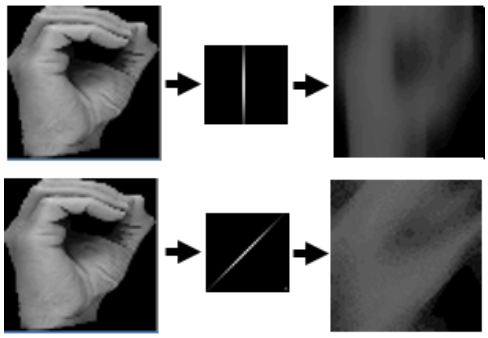
- Standard deviation of the Gaussian envelope (σ);
- Orientation of the normal to the parallel stripes of a Gabor function (θ);
- Wavelength of the sinusoidal factor (λ);
- Spatial aspect ratio (γ);
- Phase offset (ψ).

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 3.

2.2.4 Contour properties

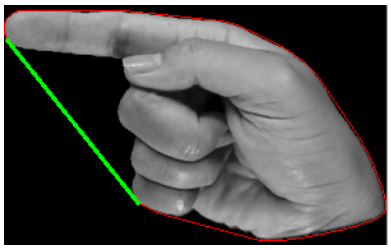
Several simple properties can be extracted just from the image contour. Figure 4 shows a hand gesture with a corresponding convex hull and a convexity defect. The features below can be used to rapidly distinguish between simple gestures, such as open and closed hand.

Figure 3 – A Gabor filter with $\theta = 0°$ and $\theta = 45°$.



Source – The author

Figure 4 – A convex hull (red line) with a convexity defect (thick green line).



Source – The author

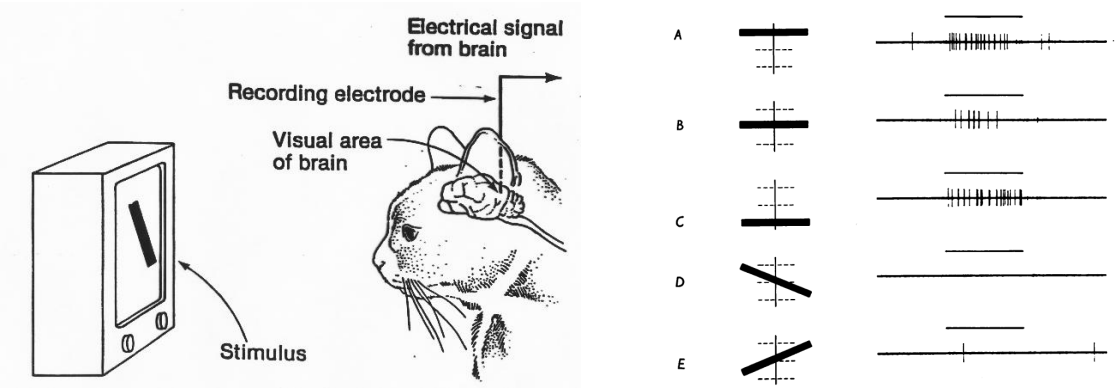
- Solidity: the ratio of contour area to its convex hull area;
- Extent: the ratio of contour area to bounding rectangle area;
- Convexity defects: a list of the five largest convexity defects is used as a feature vector.

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) (SCHMIDHUBER, 2015; DENG; YU, 2014; LECUN; BENGIO; HINTON, 2015), 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 (SCHMIDHUBER, 2015; LECUN; BENGIO; HINTON, 2015).

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 (LECUN; BENGIO; HINTON, 2015). They

Figure 5 – An illustration of the 1961 experiment by Hubel and Wiesel.



(a) (b)

Source – Hubel (1995)

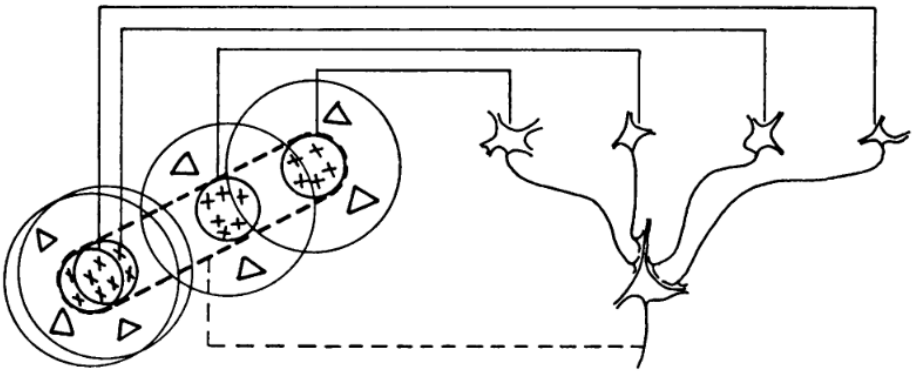
are specialized networks that work with data in a grid-like topology, such as images (GOODFELLOW et al., 2016), and employ the convolution operation in at least one layer. Successive transformations highlight features present in the input pattern without the need for feature engineering, and are fundamental in the learning process of deep neural networks. Two important properties found in CNNs are their ease of training and high capability for generalization (LECUN; BENGIO; HINTON, 2015). 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 Biological inspiration

In their famous 1961 experiment, Hubel and Wiesel (1962) studied how neurons in a cat's primary visual cortex (V1) respond to images of certain shapes and orientations. In their previous experiments, Hubel and Wiesel found that simple neurons closer to the retina are responsible for detection of spots of light. The information from this neurons is relayed mostly to V1 where it is refined for other layers in the visual cortex. The experiment is illustrated on Figure 5a and the response of one neuron in V1 is shown on Figure 5b, this particular neuron is more responsive to a black horizontal rectangle placed anywhere in the receptive field. Rectangles with other orientations do not produce the same stimulus in this neuron.

In oder to explain the organization of this receptive fields in V1, Hubel and Wiesel propose a scheme illustrated in Figure 6. In this scheme, simple neurons from Lateral Geniculate Nucleus (LGN), that are sensitive to spots of light, are connected to a single neuron in V1, which in turn can react to (encode) more complex shapes. Additionally,

Figure 6 – Possible connectivity scheme between simple neurons in LGN and a neuron in V1.



Source – Hubel and Wiesel (1962)

complex cells response is more invariant to the position of the encoded shape in the visual field.

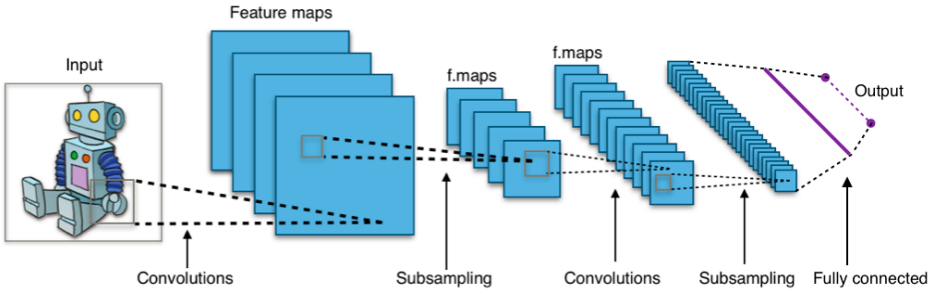
While the mechanism of biological neurons are an ongoing research topic, some simplified stationary models, such as multilayer perceptions and convolutional neural networks have been successfully employed in several applications. For instance, both of the properties of visual neurons described above, i.e. hierarchical feature encoding and spatial invariance of filters are present in most of modern convolutional neural networks.

2.3.2 Main components

Broadly speaking, CNNs consist of multiple layers of a neural network, each with a particular function, responsible for nonlinearly transforming the input and forwarding the output to the next layer. From these transformations, CNNs abstract from details considered irrelevant and concentrate on invariant properties present in the data (DENG; YU, 2014; LECUN; BENGIO; HINTON, 2015). For this reason, the need for data preprocessing is minimized when using CNNs. In other words, this type of network dispenses from feature descriptors in favor of learning its own filters during training. Figure 7 showcases the general configuration of a convolutional neural network, with its convolutional and subsampling layers and, lastly, a fully connected layer.

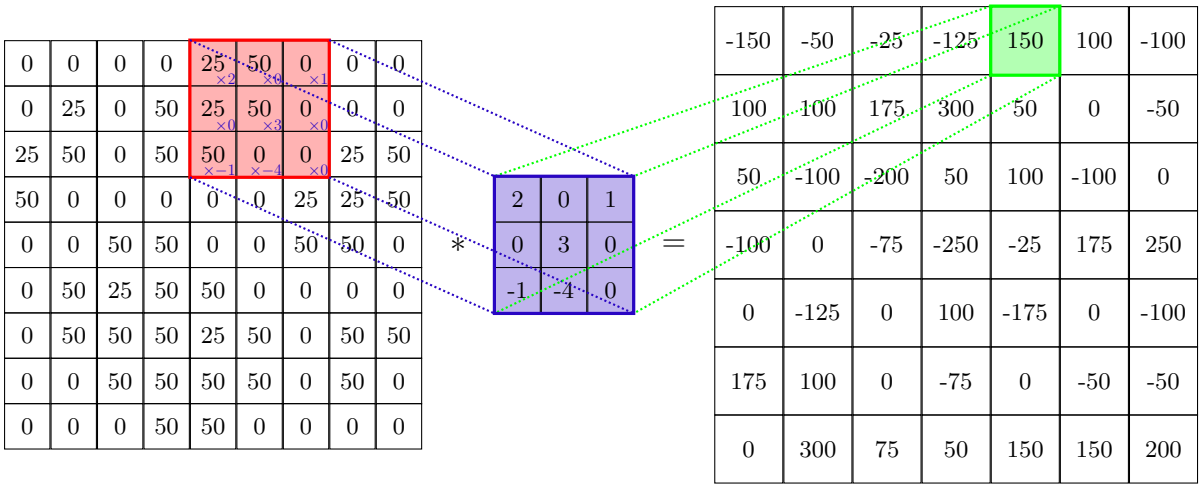
Two types of layer worth mentioning are the convolutional layer and the subsampling layer. The units in these layers form feature maps, and to each feature map there is an associated kernel (or filter) of fixed size. In a convolutional layer, the convolution kernel passes horizontally and vertically throughout the input map, convolving with each unit, as depicted in Figure 8. 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) (SCHMIDHUBER, 2015; LECUN; BENGIO; HINTON, 2015). Examples of common

Figure 7 – A general scheme of convolutional neural networks



Source – Aphex34 [CC BY-SA 4.0 (<https://creativecommons.org/licenses/by-sa/4.0>)], from Wikimedia Commons

Figure 8 – An example of convolution. During this process, the $k \times k$ kernel is convolved with the $n \times n$ input image. To produce the output, the dot product between the kernel and the $k \times k$ blocks of the input (such as the one in red) is computed by passing the filter horizontally and vertically across the entire input map. The results are sequentially included in the $n - k + 1 \times n - k + 1$ output map.



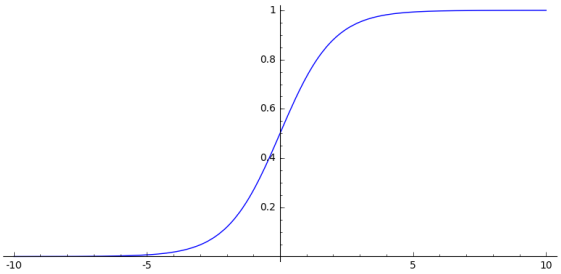
Source – The author

functions used in activation layers are shown in Figure 9.

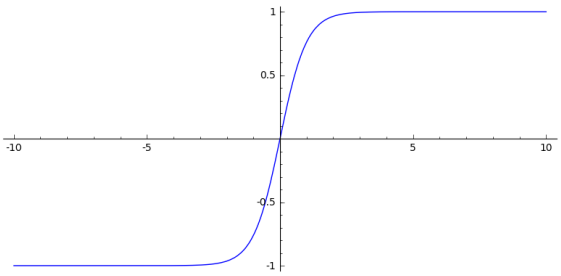
In particular, ReLUs and its variations have been shown to improve the discrimination capabilities of artificial neural networks, including deep learning networks (JARRETT et al., 2009; NAIR; HINTON, 2010; DAHL; SAINATH; HINTON, 2013), and the choice for a rectifying nonlinearity is one of the most important decisions when designing a deep architecture for accurate object recognition (JARRETT et al., 2009). The ReLU, $f(x) = \max\{0, x\}$, is a nonlinear function with two linear parts, and therefore it preserves properties of linear models that allow them to generalize well and that facilitate optimization with gradient methods (GOODFELLOW et al., 2016).

Subsampling methods are used to reduce the volume of data during training (DENG; YU, 2014). A technique known as max pooling is one of the most commonly

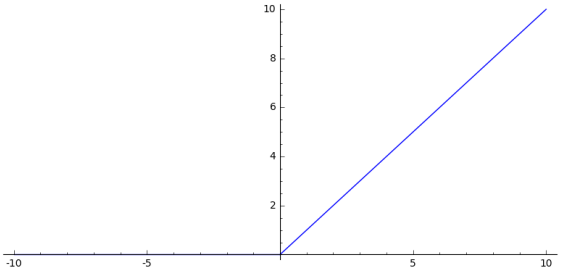
Figure 9 – Some commonly employed activation functions.



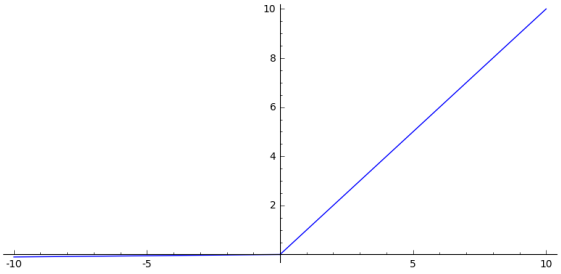
(a) Sigmoid



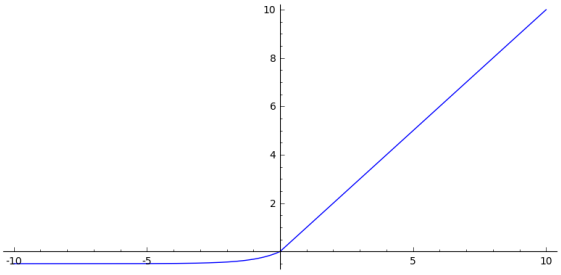
(b) Hyperbolic tangent



(c) ReLU



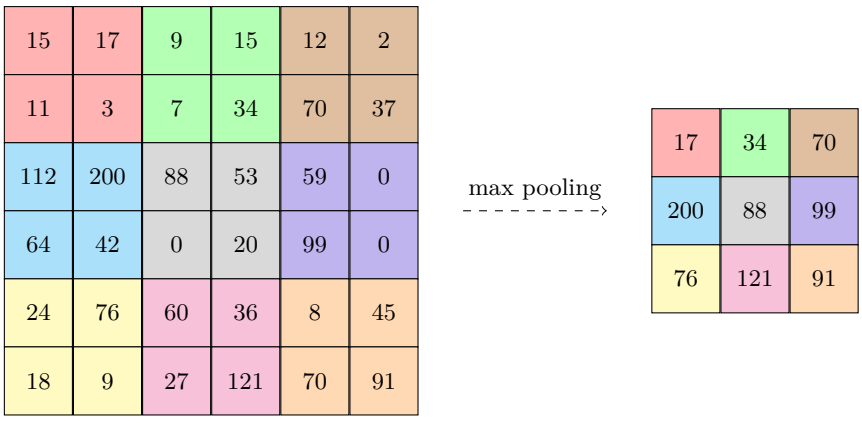
(d) Leaky ReLU



(e) ELU

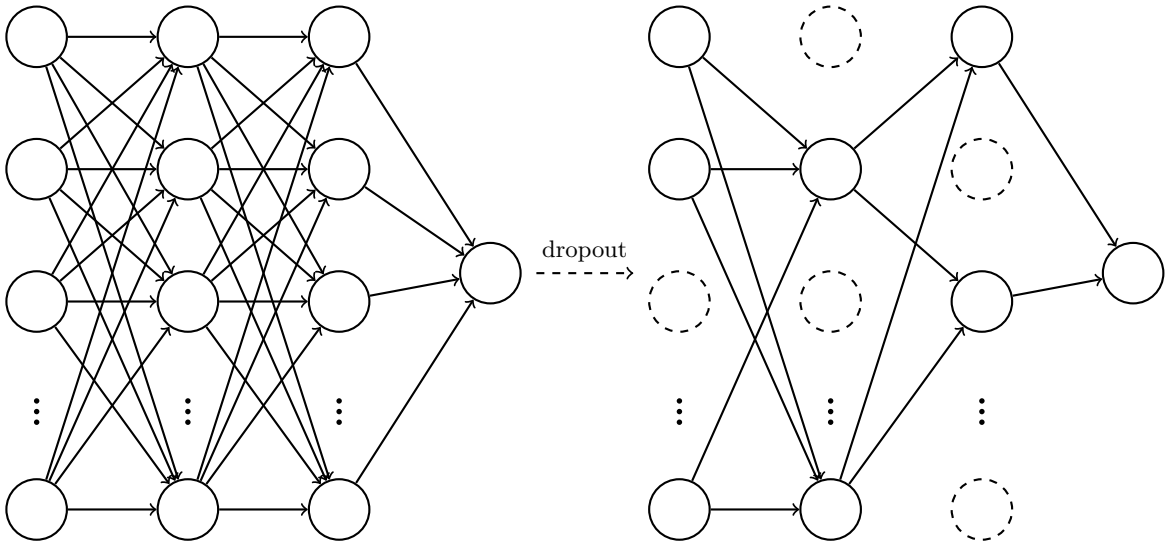
Source – The author

Figure 10 – The max pooling operation. In this depiction, the non-overlapping rectangles are of size 2×2, which results in a feature map half the original size.



Source – The author

Figure 11 – An example of fully connected network before and after the dropout regularization technique.



Source – The author

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 (SCHMIDHUBER, 2015) by adding robustness to small distortions (JARRETT et al., 2009; GOODFELLOW et al., 2016). 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. This process can be seen in Figure 10. Another known type of pooling is the average pooling, which instead of taking the maximum element of each window calculates the average between all window elements and forwards to the next feature map (JARRETT et al., 2009). Typically, both types of layers alternate multiple times and are followed by one or more fully connected (FC) layer. 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, by adding a random element to training, can be used to prevent overfitting and further increase a network's performance (SRIVASTAVA et al., 2014; SCHMIDHUBER, 2015; DENG; YU, 2014). Figure 11 depicts the effect of this technique on a network. Dropout acts by disabling units during training with a certain probability, the network hyperparameter p, so that incoming and outgoing connections to these units are removed. It can be applied to any layer of the CNN. It has been shown that the dropout regularization technique, when compared to other regularization methods, can lower error rates in several classification domains (SRIVASTAVA et al., 2014). Lastly,

combinations of the described components are successfully used in state-of-the-art CNNs, which can efficiently reap the benefits of modern GPUs. For instance, despite dropout causing an increase in training time, it can be combined with ReLU in order to lessen this penalty (DAHL; SAINATH; HINTON, 2013).

Batch normalization 2015 is another technique for improving training time and stability of deep neural networks. It works by normalizing the activations of a layer per batch, keeping the mean and standard deviation close to 0 and 1 respectively. The model proposed in this work does not employ batch normalization, mainly because it would increase the number of parameters and this method has not been shown to increase performance of convolutional networks with small number of layers. However, future works could consider evaluating this technique also.

2.3.3 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 (KRIZHEVSKY; SUTSKEVER; HINTON, 2012), 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. High computational requirements of modern general purpose CNNs do not apply to smaller convolutional networks, designed for specific purposes. For example, this work compares two state-of-the-art general purpose CNNs with other networks proposed for hand posture recognition, suggesting that smaller architectures are more suitable for specific problems. Image size and representation should also be considered for real-time applications, as smaller and more compact image representations (such as binary) are potentially faster and can be done by a simpler and more robust convolutional network (NASR-ESFAHANI et al., 2016; YAMASHITA; WATASUE, 2014; CAMBUIM et al., 2016).

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 (REN et al., 2015), 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 (ESTEVA et al., 2017), 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 (PAN; YANG et al., 2010). In the present work, transfer learning is evaluated on two state-of-the-art large CNNs.

3 Related Works

3.1 Traditional feature descriptors

Among traditional feature extraction techniques, Aowal et al. (2014) proposes and evaluates 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. (2017) 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 (2002). An RBF network is used for hand pose classification. Hu moments are compared with Zernike moments for general object recognition (SABHARA; LEE; LIM, 2013) as well as for static hand gestures (OTINIANO-RODRÍGUEZ; CÁMARA-CHÁVEZ; MENOTTI, 2012). In general, Zernike moments were found to be more accurate than Hu moments (SABHARA; LEE; LIM, 2013; OTINIANO-RODRÍGUEZ; CÁMARA-CHÁVEZ; MENOTTI, 2012). Wang C. and Wang K. (2016) 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. (2015), measuring accuracy as well as computational cost on a common static hand gesture database, also used in the present work. A combination of graph-based characteristics and appearance-based descriptors such as detected edges for modeling static gestures is proposed by Avraam (2014). This method is tested on ten numerical gestures from the Massey database (BARCZAK et al., 2011). An average recognition rate of 83% with a standard deviation of 10% is achieved.

Huang et al. (2015) 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. (2014) 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 (RAUTARAY; AGRAWAL, 2015; PISHARADY; SAERBECK, 2015).

Several 3D hand gesture acquisition and recognition techniques were surveyed by Cheng et al. (2016), 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 present work is evaluated on several datasets and special attention is given to the recognition of similar gestures.

A hand gesture recognition system is proposed by Yang (2016), 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. (2013) 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 (2016) 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.

Cambuim et al.(2016) propose and efficient hand posture recognizer implemented on an FPGA. As in Nasr et al. (2016), 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. (2017) 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. Similarly, the present work uses leave-one-subject-out validation on three datasets with 10, 27 and 36 postures. The classifier proposed in this 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. (2018). 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. A 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.

3.2 Convolutional neural networks

A convolutional neural network with three channels is proposed by Barros et al. (2014). Besides a grayscale image, convolution is also applied to an image filtered by a Sobel operator on a vertical and horizontal directions. Images are reduced to 28×28 pixels, allowing real-time recognition. A multichannel CNN is also used in this work, this network has two channels: 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 (HSIAO et al., 2014) is used by Sanchez-Riera et al. (2016) 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 concatenation of depth and color data.

Three different configurations of CNNs and Stacked Denoising Autoencoders (SDAE) are evaluated by Oyedotun and Khashman (2016) on a static hand posture dataset (BIRK; MOESLUND; MADSEN, 1997), considering accuracy and inference 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 models on several datasets.

Ji et al. (2016) 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. (2016) 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.

3.3 Comparison with the present work

Table 1 contains a comparative summary of the state-of-the-art. 'Colour space columns' contains the colour space used for feature extraction and 'Real-time' column indicated 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, colour space and rescaling. A real-time recognition system is also demonstrated using the

Table 1 – Comparative summary of related works

Work	**Classifier**	**Color space**	**Evaluated datasets**	**Real-time**
Palacios et al. 2013	Heuristic	RGB, Depth	1	Yes
Dinh et al. 2014	RF, heuristic	Depth	1	Yes
Hsiao et al. 2014	CNN	Depth	1	No
Barros et al. 2014	CNN	Gray	2	No
Sanchez-Riera et al. 2016	CNN	Gray, Depth	1	No
Oyedotun and Khashman 2016	CNN, SDAE	Gray	1	No
Cambuim et al. 2016	ELM	Binary	1	Yes
Nasr et al. 2016	CNN	Binary	1	No
Wang C. and Wang K. 2016	Heuristic	Binary	1	No
Ji et al. 2016	CNN	Gray	1	No
Nai et al. 2017	RF	Depth	1	Yes
Chevtchenko et al. 2018	MLP	Gray	1	Yes
This work	CNN	Gray, Binary, Depth	3	Yes

proposed classifier.

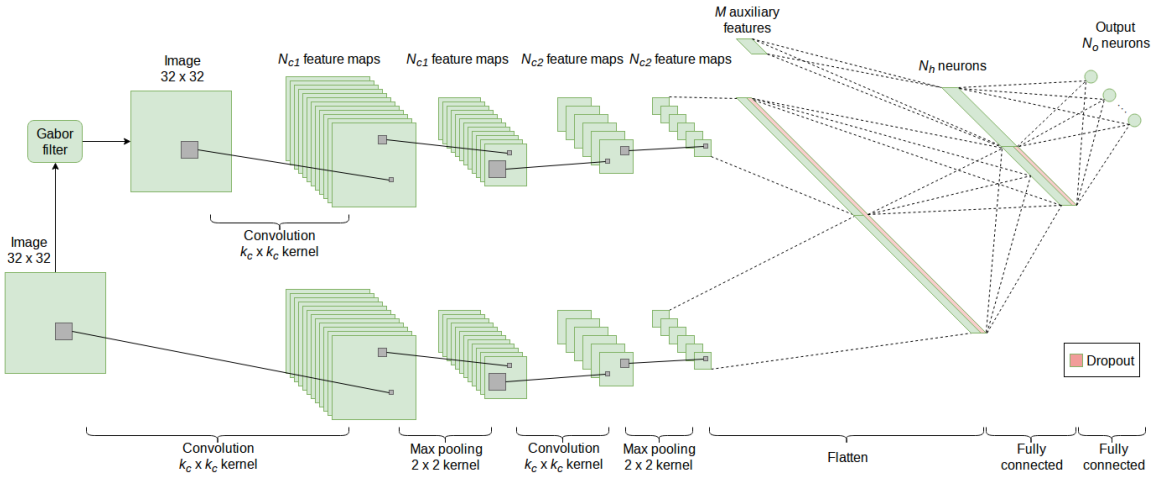
As can be seen from related works, convolutional neural networks and traditional feature descriptors have obtained good results for hand posture recognition. Drawing inspiration from both, this work proposes a novel architecture, where a convolutional neural network is combined with a feature vector 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.

4 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. 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. (2018), in the proposed architecture a second convolutional channel is fed with an image previously enhanced by this filter. Such architecture would be consistent with what is thought to happen in the visual system of mammals, where the Gabor filter is considered a good representation of simple cells in the visual cortex (TURNER, 1986) and further conventional layers are used to extract higher level features. Other properties based on shape and contour are also used to diversify information about the hand posture prior to classification.

The architecture of this feature fusion-based convolutional neural network (FFCNN) is depicted in Figure 12. 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 Chapter 5. The input image is fed to a convolutional channel represented on the bottom of Figure 12. This channel contains two layers of convolution, concatenated with max pooling layers. The number of layers and the input image size were adjusted experimentally based on related works. As Oyedotun and Khashman (2016), we found that two layers perform better than three, four or five on evaluated datasets.

Figure 12 – A diagram of the proposed feature fusion-based convolutional network.



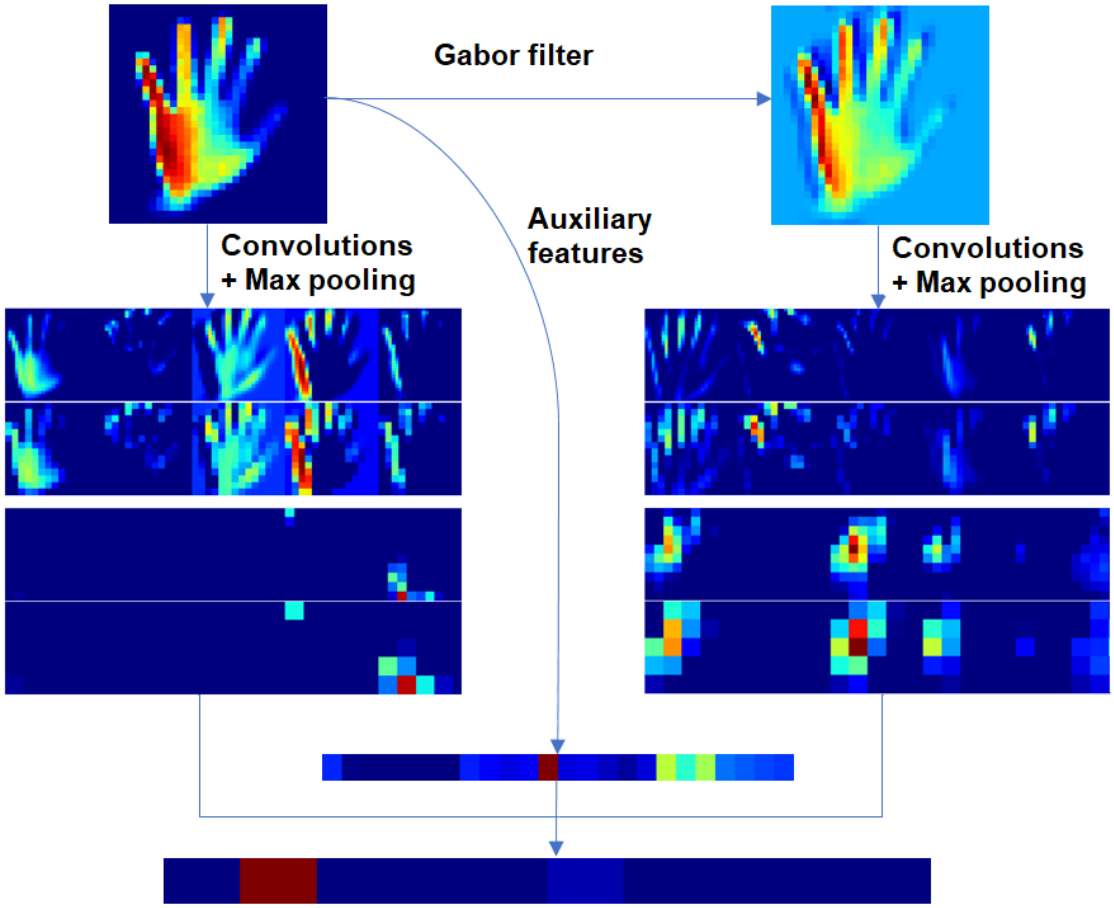
Source – The author

The same image is also fed to a Gabor filter and then to a second convolutional channel, shown above the first one in Figure 12. 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 12.

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 13, where activations of input, output and some convolutional layers are depicted for a simple gesture.

Figure 13 – An illustration of activations of layers in the FFCNN architecture. A depth image is used as an input.



Source – The author

The proposed network is used for real-time hand posture recognition. We used a 3D RealSense^{TM} 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 in the Results section, the proposed recognition method is tolerant to faulty segmentation, as long as it is trained on a database that contains such cases. As described in the flowchart in Figure 14a, the system consists of the following steps:

1. Load database: in order to sped-up the training process the database 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 can be converted to a binary representation.

6. Crop hand image: only the hand is left for further processing.

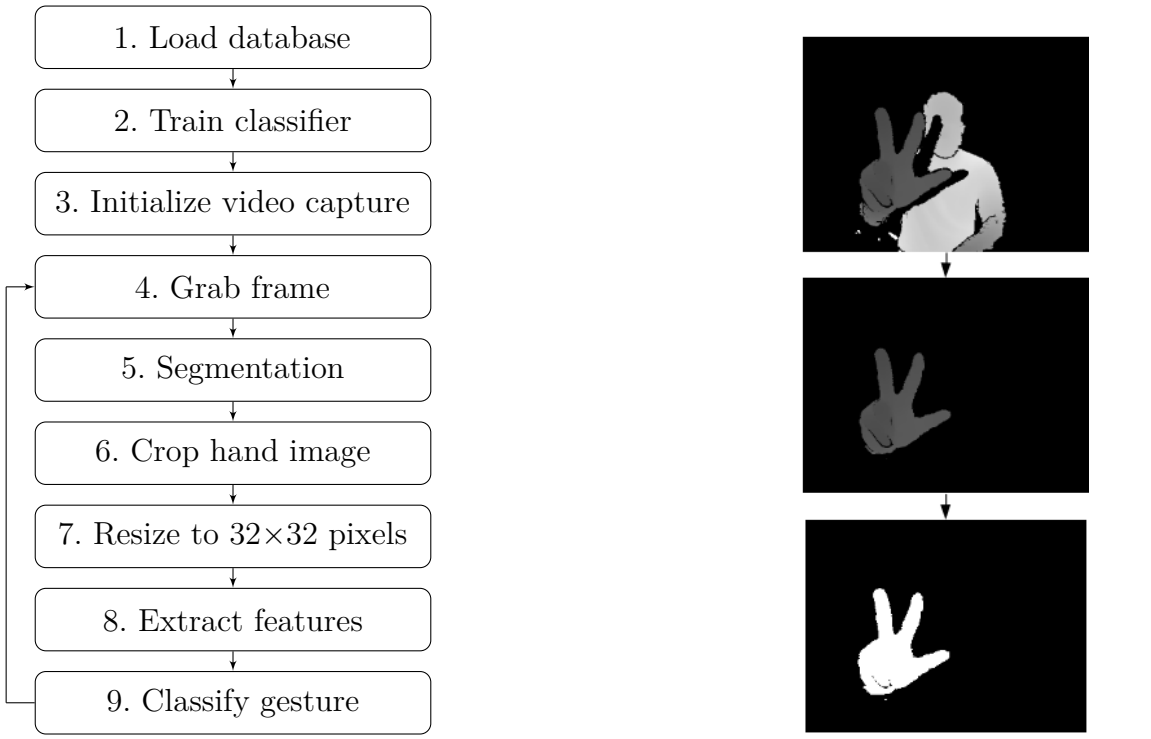
7. Resize to 32×32 pixels: the segmented image is rescaled to 32×32 pixels, preserving the aspect ration.

8. Extract features: Hu and Zernike moments along with contour features are extracted from the binary or grayscale image, which is also passed through the Gabor filter for the second convolutional channel of FFCNN.

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.

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

Figure 14 – Real-time gesture recognition system. During experimentation, steps 4–9 required 26 ms on average.



(a) Flowchart of the system. (b) Illustration of steps 4 and 5.

Source – The author

It is worth noting that while segmentation relies on the depth sensor, the recognition can be made from a binary image. Thus the proposed system can also work with common RGB cameras, requiring only the segmentation steps to be adjusted.

5 Methodology

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

5.1 Benchmarking datasets

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

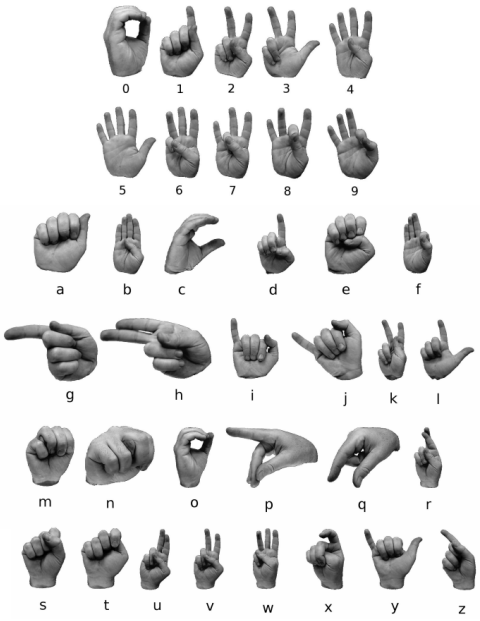
Table 2 – A summary of the datasets used in this work

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

5.1.1 Massey

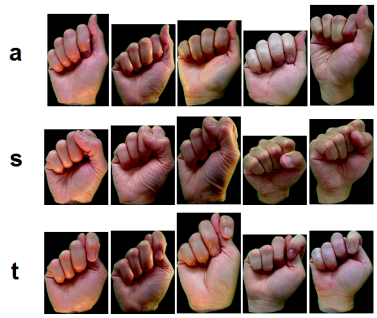
The Massey dataset (BARCZAK et al., 2011) has 2,515 images, illustrated in Figure 15. The database contains variations in scale, illumination and rotation, as illustrated in Figure 16, which shows three similar gestures: ‘a’, ‘s’ and ‘t’. There are five subjects, allowing to use both the holdout and 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 work are evaluated on this subsets. The two subsets are named Massey-G and Massey-B, corresponding to grayscale and binary data respectively.

Figure 15 – 36 gestures from the Massey dataset.



Source – Barczak et al. (2011)

Figure 16 – Three similar letters from the Massey dataset, showing variations in scale, tone, illumination and slight rotation within the same class.



Source – Barczak et al. (2011)

5.1.2 OUHANDS

This dataset (MATILAINEN et al., 2016) 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 17. 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.

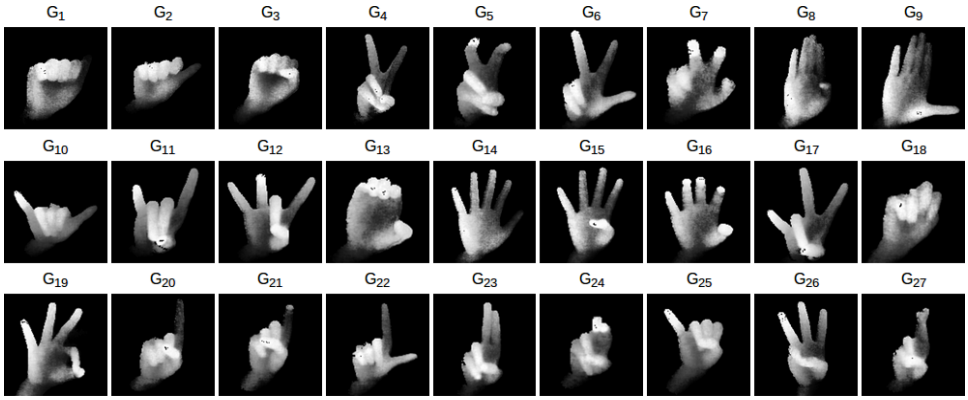
Figure 17 – Ten gestures with binary mask, depth data and RGB frames from the OUHANDS dataset.

Source – Matilainen et al. (2016)

5.1.3 LaRED

LaRED (HSIAO et al., 2014) 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 18. This database can be further extended to 243,000 images by applying rotation. Differently from other datasets used in this work, 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 18 – 27 gestures from the LaRED dataset.



Source – Hsiao et al. (2014)

5.2 Validation technique

The datasets used in this work contain gestures from more than one subject (BARCZAK et al., 2011; HSIAO et al., 2014; MATILAINEN et al., 2016). 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 (although not the same image) 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 work are averaged across ten repetitions for both validation methods.

5.3 Statistical test for comparisons of results

As suggested in Demšar (2006), the Wilcoxon signed-rank test is used to compare results in terms of accuracy. The signed-rank test is a non-parametric hypothesis test for two samples. In the context of this work, the test is always applied with a level of significance of 5% considering samples of n accuracy results of two classifiers. The null

and alternative hypotheses of the test are:

$$H_0 : \text{the two classifiers are equivalent} \tag{5.1}$$

$$H_1 : \text{the two classifiers are not equivalent} \tag{5.2}$$

This test ranks the absolute value of the differences between (related) measurements of two samples. The differences of the paired measurements are calculated and, subsequently, the absolute values of the results are ranked and grouped as positive and negative according to the signs of the respective differences. Null differences can be ignored. The minimum between the sum of ranks in the positive group and the sum of ranks in the negative group, both expressed in Equation (5.3), is the computed test statistic. The signed-rank test contains a table of critical values which is used to assess if the null hypothesis is rejected.

$$R^{+\,(-)} = \sum_{d_i>0\,(d_i<0)} \text{rank}(d_i) \tag{5.3}$$

5.4 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

5.5 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

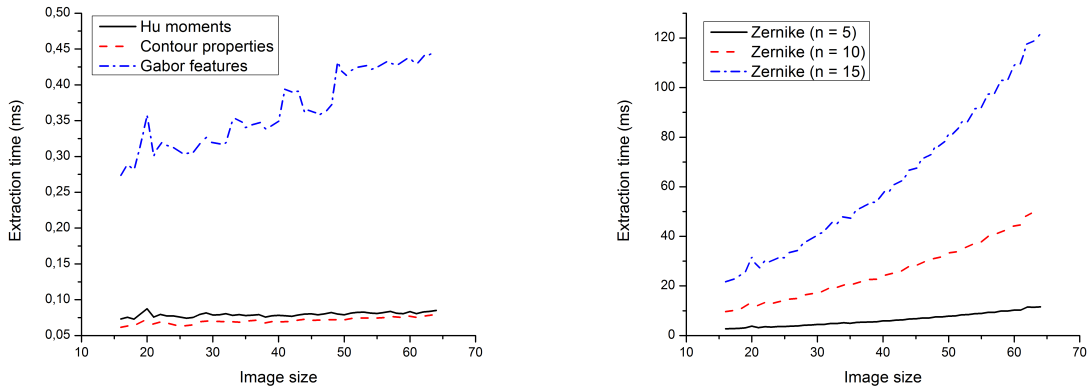
fully connected neurons, etc. Changes in these parameters may have a significant effect on the overall performance of the network.

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. (2011), to search the parameter space. Given that $y = f(x)$ is an observation of the model with hyperparameters x, TPE algorithm is used to iteratively generate samples of y based on previous observations (DOMHAN; SPRINGENBERG; HUTTER, 2015; BERGSTRA; YAMINS; COX, 2013).

The feature fusion-based convolutional neural network described in this work has several parameters that can be adjusted by a search algorithm—TPE. These parameters are given below. Space exploration values are based on experimental evaluation and similar applications found in literature, such as Chevtchenko et al. (2018), Oyedotun and Khashman (2016), Nasr et al. (2016) and Yamashita and Watasue (2014). 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 these features (See Figure 19b).
- Gabor filter parameters:
  - σ: continuous, between 0.001 and 1.
  - θ: continuous, between 0.001 and π.
  - λ: continuous, between 0.001 and π.
  - γ: continuous, between 0.001 and 1.
  - ψ: continuous, between 0.001 and π.

Figure 19 – 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).



(a) (b)

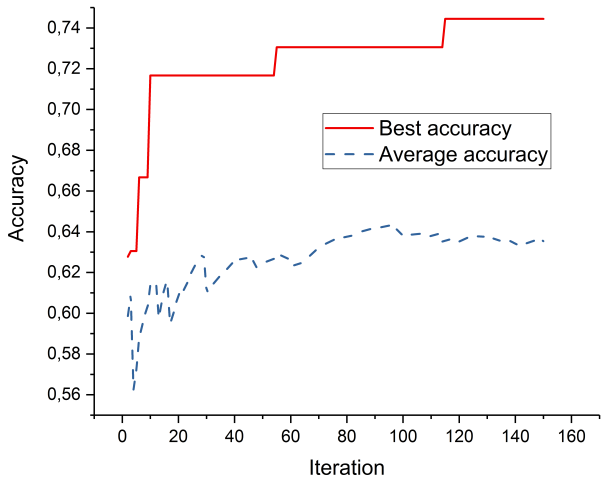
Source – The author

The computational costs of the classical features considered in this work are compared in Figure 19 as a function of image size. The feature extraction time is averaged across 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 19b, Zernike moments have a significantly higher computational cost than other features.

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. Note that only subject 'hand5' is used for optimization and thus the resulting model is not biased towards other subjects in the Massey-G subset. There is obviously no bias towards other OUHANDS not LaRED datasets, since these are not considered for optimization. Finally, TPE algorithm could also be used on other datasets, which would possibly result in a different set of hyperparameters.

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 20 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.

Figure 20 – Average and best recognition rates obtained during the hyperparameter selection loop.



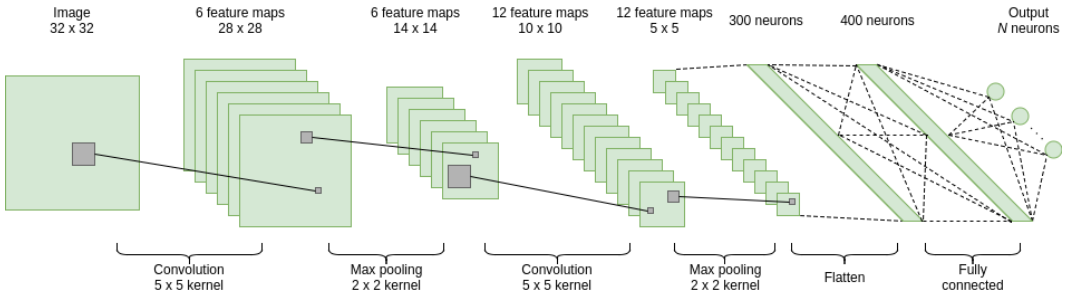
Source – The author

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. 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). Table 3 contains the list of hyperparameters selected for the proposed FFCNN architecture.

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

Figure 21 – A convolutional network, denoted as CNN1, proposed by Oyedotun and Khashman (2016). N is the number of classes in the dataset.



Source – The author

5.6 Implemented models

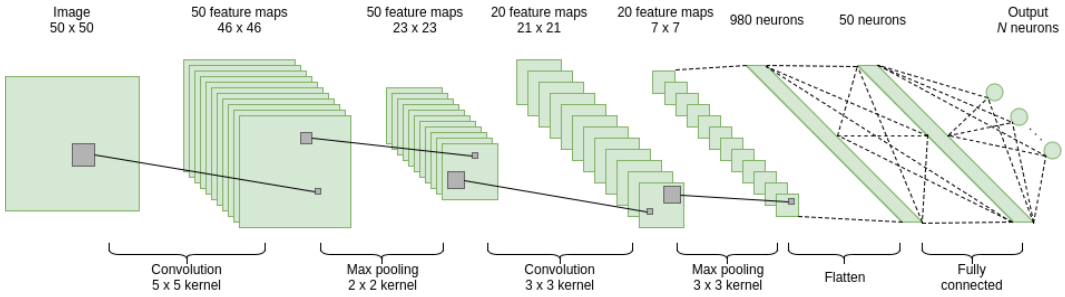
This section details the deep learning architectures considered in this work. For comparison and benchmarking, we have selected recently proposed convolutional models that could be applied to 2D images. These models are denoted as CNN1, CNN2 and CNN3, based on Oyedotun and Khashman (2016), Nasr et al. (2016) and Ji et al. (2016), respectively. Two commonly referred large convolutional networks are also used: VGG16 (SIMONYAN; ZISSERMAN, 2014) and MobileNet (HOWARD et al., 2017).

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. Nesterov Adam optimizer (DOZAT, 2016), following parameters provided in the paper, was experimentally selected for CNN1, CNN2, CNN3 and FFCNN architectures. Due to large number of trainable parameters, VGG16 and MobileNet architectures were found to converge better using a stochastic gradient descent (SGD) optimizer with constant learning rate of 0.001, 0.9 momentum and without use of Nesterov momentum.

5.6.1 CNN1

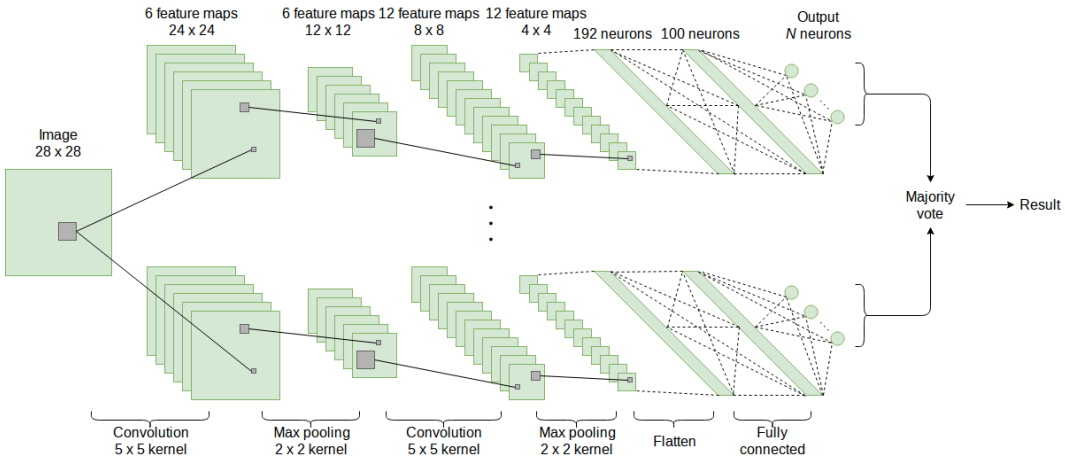
Proposed by Oyedotun and Khashman (2016), this is a single-channel convolutional neural network. A 5×5 kernel is used for the convolution operations. First convolutional layer receives a single-channel image and convolves it through 6 kernels, resulting in 6 maps of size 28×28. Then a pooling windows 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 21.

Figure 22 – A convolutional network, denoted as CNN2, proposed by Nasr et al. (2016).



Source – The author

Figure 23 – An ensemble of 5 convolutional networks, denoted as CNN3, with majority voting proposed by Ji et al (2016).



Source – The author

5.6.2 CNN2

Proposed by Nasr et al. (2016), 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 50 filters of size 5×5 and the second has 20 kernels with size 3×3. The 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 22.

5.6.3 CNN3

Proposed by Ji et al. (2016), 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 23 depicts the structure of this network.

5.6.4 VGG16

A famous convolutional network, proposed by Simonyan and Andrew (2014). 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.

5.6.5 MobileNet

Proposed in 2017 by Howard et al. (2017), this is a new model of convolutional networks that aim to be employed on mobile devices. The model used in this work 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.

6 Experimental results

The objective of this chapter is to evaluate the convolutional architectures CNN1 (OYEDOTUN; KHASHMAN, 2016), CNN2 (JI et al., 2016), CNN3 (JI et al., 2016), VGG16 (SIMONYAN; ZISSERMAN, 2014), MobileNet (HOWARD et al., 2017) and the proposed FFCNN on Massey (BARCZAK et al., 2011), OUHANDS (MATILAINEN et al., 2016) and LaRED (HSIAO et al., 2014) datasets, using validation methods presented in Section 5.2.

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 5.2. 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 and standard deviation recognition rates are presented. Furthermore, we have used the Wilcoxon statistical test with a 5% level of significance to compare results (DEMŠAR, 2006). 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.

6.1 Massey dataset

Initially all architectures described in Section 5.6 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 is statistically better than 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)	95.86 (0.81)
FFCNN	**98.05 (0.90)**	96.93 (0.70)

Figure 24 – 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.

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

Source – Barczak et al. (2011)

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. The usage of grayscale images results in better accuracy, this can be attributed to a number of complex gestures in this dataset. For instance, the gestures 'a' and 's' are very similar, as illustrated in Figure 24, and are better distinguished by grayscale than binary images.

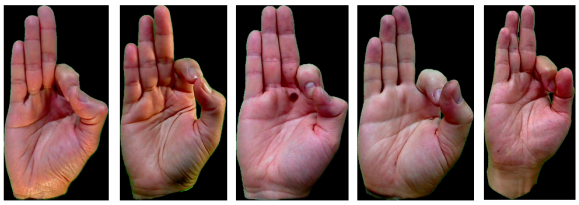
Table 5 – Comparison of accuracy on Massey subsets using leave-one-subject-out validation

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)
FFCNN	**84.02 (0.59)**	82.58 (0.81)

The proposed FFCNN architecture obtains better results than state-of-the-art architectures. Also, 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' hands, as shown in Figure 25. In our experiments, subject 5 was the most challenging when selected for the Test subset.

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 inference time is averaged across 503 images.

Figure 25 – Gesture 'f' from the Massey dataset performed by each of the five subjects.



Source – Barczak et al. (2011)

Table 6 – Comparison of training and inference times on the Massey dataset with and without a GPU

	With GPU		Without GPU	
Architecture model	**Training per epoch (s)**	**Recognition time (ms)**	**Training per epoch (s)**	**Inference 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
FFCNN	0.3	9.2	5.2	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 on both cases.

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)
FFCNN	**98.75 (0.48)**	**98.68 (0.33)**	**98.48 (0.63)**

Table 8 compares the same architectures using leave-one-subject-out validation. The FFCNN model on the grayscale subset is shown to be significantly better than other combinations of models and subsets.

Table 8 – Comparison of accuracy on OUHANDS subsets using leave-one-subject-out validation

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)	98.53 (0.17)	98.20 (0.17)
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 Tables 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.

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)	97.04 (0.36)
CNN3	**99.57 (0.04)**	96.44 (0.16)
FFCNN	**99.59 (0.04)**	95.30 (0.59)

Considering holdout validation in Table 9, most models obtained close to 100% accuracy. In the same experiment the binary subset is significantly better than the depth, this suggests that gestures in LaRED dataset are recognizable only by the binary mask and depth data does not provide additional information, this hypothesis will be tested further.

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)
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. The proposed architecture still achieves a significantly 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.

6.4 Impact of preserving aspect ratio

Finally, the images in a dataset are resized prior to feature extraction and training. This is done to speed-up computations and to make the image compatible with feature extraction methods, such as convolutional networks. An image can be resized with or without preservation of the aspect ratio of the gesture, as illustrated in Figure 26. All prior experiments resize the image without preserving the aspect ratio, as is done by several related works (OYEDOTUN; KHASHMAN, 2016; NASR-ESFAHANI et al., 2016;

Figure 26 – A gesture from the Massey dataset resized with and without aspect ratio preservation.

Source – Barczak et al. (2011)

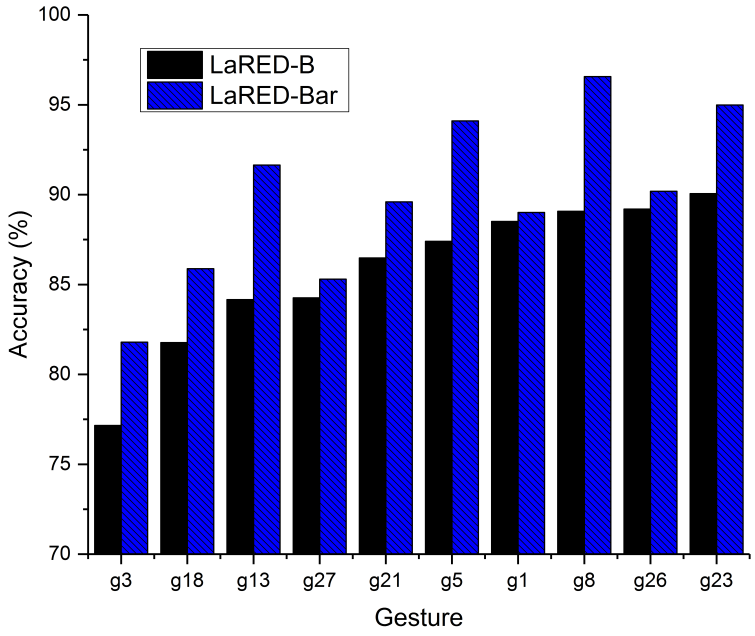
Table 11 – Evaluation of the impact of aspect ratio on the FFCNN architecture using leave-one-subject-out validation. A Wilcoxon test is applied in order to verify the significance of the improvement in accuracy.

Dataset	**Subset**	**Without aspect ratio**	**With aspect ratio**	**Significant improvement?**
Massey	Grayscale	84.02 (0.59)	84.27 (0.60)	No
Massey	Binary	82.58 (0.81)	83.82 (0.73)	Yes
OUHANDS	Grayscale	99.20 (0.12)	99.18 (0.16)	No
OUHANDS	Binary	99.03 (0.07)	99.45 (0.09)	Yes
OUHANDS	Depth	99.00 (0.14)	99.45 (0.14)	Yes
LaRED	Binary	91.23 (0.17)	93.06 (0.18)	Yes
LaRED	Depth	92.53 (0.24)	93.11 (0.16)	Yes

JI et al., 2016; AOWAL et al., 2014; YAMASHITA; WATASUE, 2014). When aspect ratio is preserved, the remaining pixels are filled with zero in order to still produce an image of size 32×32.$The influence of the aspect ratio on the recognition rate of the FFCNN architecture is evaluated di$

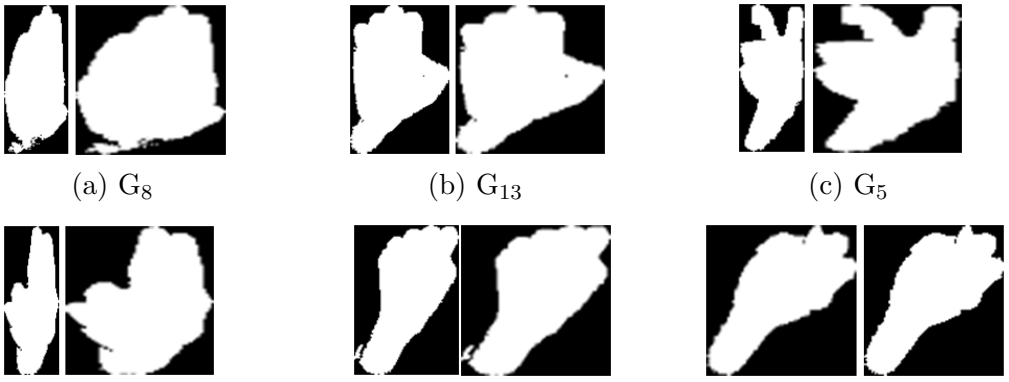
Considering that the best recognition rates from Tables 8 and 10 are improved in Table 11 we conclude that: 1) preserving aspect ratio while rescaling provides a significant advantage to the FFCNN architecture and 2) there is significant difference between depth and binary data. This conclusions are supported by comparisons in Figures 27 and 28, obtained from ten repetitions using leave-one-subject-out validation. For example, recognition rate of gesture G_8 is improved 7.5% by preserving the aspect ratio, otherwise G_8 can be easily confused with gestures G_1 and G_3, presented in Figure 18. Furthermore, the usage 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 (NASR-ESFAHANI et al., 2016; YAMASHITA; WATASUE, 2014; CAMBUIM et al., 2016). Note that Massey dataset contains more complex gestures, which seem to be better recognized from grayscale images. In the next section, a real-time recognition system based on FFCNN using a binary mask is presented.

Figure 27 – 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 28.



Source – The author

Figure 28 – Gestures from the LaRED-B subset with the most improved recognition by preserving aspect ratio. For quantitative comparison see Figure 27.



(a) G_8 (b) G_{13} (c) G_5

(d) G_{23} (e) G_3 (f) G_{18}

Source – The author

6.5 Real-time recognition

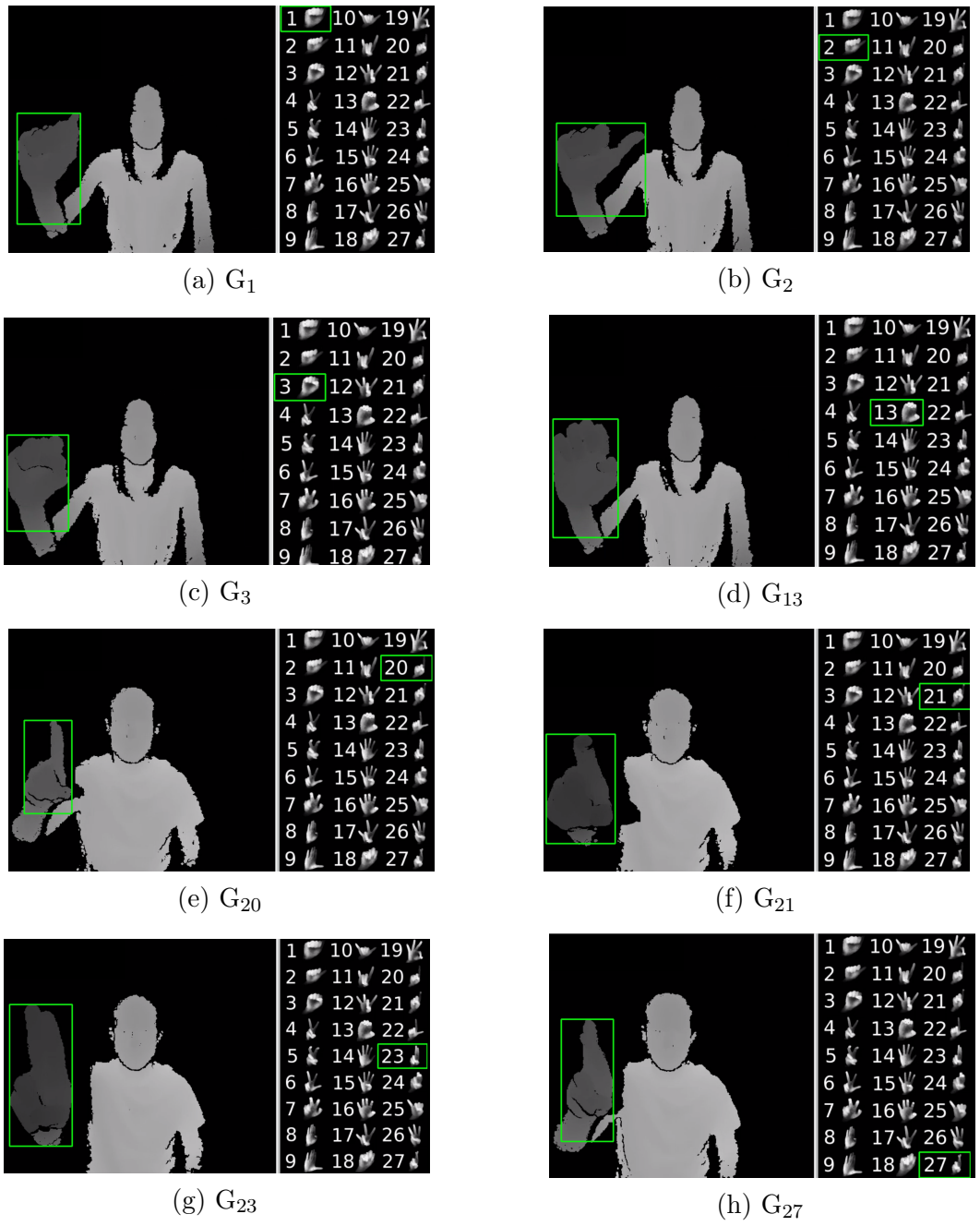
From experiments in Section 6, considering accuracy as well as speed, we demonstrated that FFCNN as a viable model for real-time recognition. In this section we present a system based on this model that recognizes gestures from OUHANDS and LaRED datasets in real-time.

The main steps of the recognition system are described in Figure 14a. Images are converted to binary representation during the segmentation step. 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 27 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 29.

The most common reason for incorrect classification is presented in Figure 30, where the forearm is segmented along with the hand. As explained in Section 5.1.2, 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.

Figure 29 – 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) G_1 (b) G_2

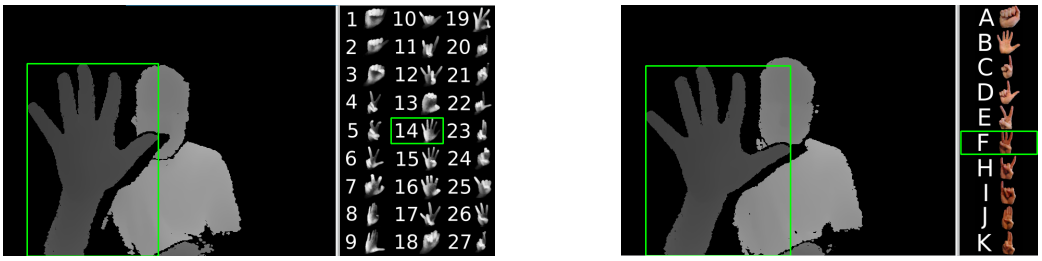
(c) G_3 (d) G_{13}

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

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

Source – The author

Figure 30 – 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'.



(a) LaRED (b) OUHANDS

Source – The author

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

7 Conclusion and final remarks

A novel architecture is proposed, based on a convolutional neural network and classical feature descriptors. This feature fusion-based convolutional network (FFCNN) is thoroughly evaluated, outperforming state-of-the-art models on different benchmarking datasets. Based on performed experiments we can highlight the most important findings:

- The proposed FFCNN obtained state-of-the-art recognition rate on three datasets and image representations.
- For smaller problems, such as hand posture recognition, it is preferable to use smaller, custom built convolutional networks, rather than using pretrained large ones, such as VGG16.
- On 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.
- Leave-one-subject-out is a more challenging validation technique, but as suggested in related works, it better represents the performance of the system in real scenarios. It simply may not be possible to retrain the system for each new user.
- The aspect ratio of segmented hand should be preserved during rescaling, this is shown to significantly improve the performance of our classifier.

7.1 Contributions

The main contributions of this work are summarized as follows:

- Recent convolutional neural networks are evaluated on three hand posture datasets: Massey (2,515 images) (BARCZAK et al., 2011), LaRED (81,000 images) (HSIAO et al., 2014) and OUHANDS (3,000 images) (MATILAINEN et al., 2016). 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 gesture recognition system based on the proposed scheme is implemented with a 3D RealSense^{TM} camera. A demo video is provided (see Appendix A).

7.2 Future work

As future endeavors we suggest to investigate other methods for hyperparameter selection and optimization, such as multi-objective algorithm. 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. Fusion in different stages could be evaluated in terms of accuracy and speed benefits. Ensemble of the proposed classifiers 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 several 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.

7.3 Publications

The following works have been published or submitted for publishing during this graduate program:

- "Multi-objective optimization for hand posture recognition." — Expert Systems with Applications (2018): pp 170–181.
- "Detecting people from beach images" — Proceedings of the IEEE conference on Tools with Artificial Intelligence (2017): pp 636–643.

- "A convolutional neural network with feature fusion for real-time hand posture recognition" — preprint submitted to Applied Soft Computing on February, 2018.
- "Deep learning for people detection from beach images" — submitted to BRACIS 2018 on May, 2018.

Bibliography

AOWAL, M. A. et al. Static hand gesture recognition using discriminative 2D Zernike moments. In: IEEE. *TENCON 2014-2014 IEEE Region 10 Conference*. [S.l.], 2014. p. 1–5. Cited 3 times in pages 31, 55, and 56.

AVRAAM, M. Static gesture recognition combining graph and appearance features. *International Journal of Advanced Research in Artificial Intelligence (IJARAI)*, v. 3, n. 2, 2014. Cited in page 31.

BARCZAK, A. et al. A new 2D static hand gesture colour image dataset for ASL gestures. Massey University, 2011. Cited 9 times in pages 31, 40, 41, 43, 51, 52, 53, 56, and 61.

BARROS, P. et al. A multichannel convolutional neural network for hand posture recognition. In: SPRINGER. *International Conference on Artificial Neural Networks*. [S.l.], 2014. p. 403–410. Cited 4 times in pages 15, 19, 33, and 34.

BERGH, M. Van den et al. Real-time 3D hand gesture interaction with a robot for understanding directions from humans. In: IEEE. *RO-MAN, 2011 IEEE*. [S.l.], 2011. p. 357–362. Cited in page 13.

BERGSTRA, J.; YAMINS, D.; COX, D. Making a science of model search: hyperparameter optimization in hundreds of dimensions for vision architectures. In: *International Conference on Machine Learning*. [S.l.: s.n.], 2013. p. 115–123. Cited in page 45.

BERGSTRA, J. S. et al. Algorithms for hyper-parameter optimization. In: *Advances in Neural Information Processing Systems*. [S.l.: s.n.], 2011. p. 2546–2554. Cited in page 45.

BIRK, H.; MOESLUND, T. B.; MADSEN, C. B. Real-time recognition of hand alphabet gestures using principal component analysis. In: PROCEEDINGS PUBLISHED BY VARIOUS PUBLISHERS. *Proceedings of the Scandinavian Conference on Image Analysis*. [S.l.], 1997. v. 1, p. 261–268. Cited 2 times in pages 13 and 33.

CAMBUIM, L. F. et al. An efficient static gesture recognizer embedded system based on elm pattern recognition algorithm. *Journal of Systems Architecture*, Elsevier, v. 68, p. 1–16, 2016. Cited 4 times in pages 29, 32, 34, and 56.

CHENG, H.; YANG, L.; LIU, Z. Survey on 3D hand gesture recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, IEEE, v. 26, n. 9, p. 1659–1673, 2016. Cited in page 31.

CHEVTCHENKO, S. F.; VALE, R. F.; MACARIO, V. Multi-objective optimization for hand posture recognition. *Expert Systems with Applications*, Elsevier, v. 92, p. 170–181, 2018. Cited 5 times in pages 16, 32, 34, 35, and 45.

DAHL, G. E.; SAINATH, T. N.; HINTON, G. E. Improving deep neural networks for LVCSR using rectified linear units and dropout. In: IEEE. *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. [S.l.], 2013. p. 8609–8613. Cited 2 times in pages 26 and 29.

DEMŠAR, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, v. 7, n. Jan, p. 1–30, 2006. Cited 2 times in pages 43 and 51.

DENG, L.; YU, D. Deep learning: methods and applications. *Foundations and Trends® in Signal Processing*, Now Publishers, Inc., v. 7, n. 3–4, p. 197–387, 2014. Cited 5 times in pages 14, 23, 25, 26, and 28.

DINH, D.-L.; KIM, J. T.; KIM, T.-S. Hand gesture recognition and interface via a depth imaging sensor for smart home appliances. *Energy Procedia*, Elsevier, v. 62, p. 576–582, 2014. Cited 2 times in pages 31 and 34.

DOMHAN, T.; SPRINGENBERG, J. T.; HUTTER, F. Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves. In: *IJCAI*. [S.l.: s.n.], 2015. p. 3460–3468. Cited in page 45.

D'ORAZIO, T. et al. Recent trends in gesture recognition: how depth data has improved classical approaches. *Image and Vision Computing*, Elsevier, v. 52, p. 56–72, 2016. Cited in page 19.

DOZAT, T. Incorporating nesterov momentum into adam. 2016. Cited in page 48.

ESTEVA, A. et al. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, Nature Publishing Group, v. 542, n. 7639, p. 115, 2017. Cited in page 29.

GOODFELLOW, I. et al. *Deep learning*. [S.l.]: MIT press Cambridge, 2016. v. 1. Cited 4 times in pages 14, 24, 26, and 28.

GUO, Y.; LIU, C.; GONG, S. Improved algorithm for Zernike moments. In: IEEE. *Control, Automation and Information Sciences (ICCAIS), 2015 International Conference on*. [S.l.], 2015. p. 307–312. Cited in page 31.

HOWARD, A. G. et al. MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. Cited 4 times in pages 16, 48, 50, and 51.

HSIAO, Y.-S. et al. LaRED: a large RGB-D extensible hand gesture dataset. In: ACM. *Proceedings of the 5th ACM Multimedia Systems Conference*. [S.l.], 2014. p. 53–58. Cited 7 times in pages 15, 33, 34, 42, 43, 51, and 61.

HU, M.-K. Visual pattern recognition by moment invariants. *Information Theory, IRE Transactions on*, IEEE, v. 8, n. 2, p. 179–187, 1962. Cited 2 times in pages 20 and 21.

HUANG, J. et al. Sign language recognition using real-sense. In: IEEE. *Signal and Information Processing (ChinaSIP), 2015 IEEE China Summit and International Conference on*. [S.l.], 2015. p. 166–170. Cited in page 31.

HUBEL, D. H. *Eye, brain, and vision*. [S.l.]: Scientific American Library/Scientific American Books, 1995. Cited in page 24.

HUBEL, D. H.; WIESEL, T. N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, Wiley Online Library, v. 160, n. 1, p. 106–154, 1962. Cited 2 times in pages 24 and 25.

IANDOLA, F. N. et al. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016. Cited in page 16.

ICHIMURA, K.; MAGATANI, K. 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)*. [S.l.: s.n.], 2015. p. 4550–4553. ISSN 1094-687X. Cited in page 13.

IOFFE, S.; SZEGEDY, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. Cited in page 29.

JAIN, A. K.; FARROKHNIA, F. Unsupervised texture segmentation using Gabor filters. *Pattern recognition*, Elsevier, v. 24, n. 12, p. 1167–1186, 1991. Cited in page 22.

JARRETT, K. et al. What is the best multi-stage architecture for object recognition? In: IEEE. *Computer Vision, 2009 IEEE 12th International Conference on*. [S.l.], 2009. p. 2146–2153. Cited 2 times in pages 26 and 28.

JI, P. et al. Vision-based posture recognition using an ensemble classifier and a vote filter. In: INTERNATIONAL SOCIETY FOR OPTICS AND PHOTONICS. *International Symposium on Optoelectronic Technology and Application 2016*. [S.l.], 2016. p. 101571J–101571J. Cited 11 times in pages 7, 15, 16, 19, 33, 34, 48, 49, 51, 55, and 56.

JOHNSON, R. et al. Exploring the potential for touchless interaction in image-guided interventional radiology. In: ACM. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. [S.l.], 2011. p. 3323–3332. Cited in page 13.

KANG, H.; LEE, C. W.; JUNG, K. Recognition-based gesture spotting in video games. *Pattern Recognition Letters*, v. 25, n. 15, p. 1701 – 1714, 2004. ISSN 0167-8655. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167865504001576>. Cited in page 13.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. ImageNet classification with deep convolutional neural networks. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2012. p. 1097–1105. Cited in page 29.

KUMAR, B. P.; MANJUNATHA, M. A hybrid gesture recognition method for American sign language. *Indian Journal of Science and Technology*, v. 10, n. 1, 2017. Cited in page 31.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *Nature*, Nature Research, v. 521, n. 7553, p. 436–444, 2015. Cited 5 times in pages 14, 15, 23, 24, and 25.

MAIDI, M.; PREDA, M. Interactive media control using natural interaction-based Kinect. In: IEEE. *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. [S.l.], 2013. p. 1812–1815. Cited in page 13.

MALIMA, A. K.; ÖZGÜR, E.; ÇETIN, M. A fast algorithm for vision-based hand gesture recognition for robot control. In: *2006 IEEE 14th Signal Processing and Communications Applications*. [S.l.: s.n.], 2006. p. 1–4. Cited in page 13.

MARĈELJA, S. Mathematical description of the responses of simple cortical cells. *JOSA*, Optical Society of America, v. 70, n. 11, p. 1297–1300, 1980. Cited in page 22.

MATILAINEN, M. et al. OUHANDS database for hand detection and pose recognition. In: IEEE. *Image Processing Theory Tools and Applications (IPTA), 2016 6th International Conference on*. [S.l.], 2016. p. 1–5. Cited 5 times in pages 41, 42, 43, 51, and 61.

NAI, W. et al. Fast hand posture classification using depth features extracted from random line segments. *Pattern Recognition*, Elsevier, v. 65, p. 1–10, 2017. Cited 2 times in pages 32 and 34.

NAIR, V.; HINTON, G. E. Rectified linear units improve restricted boltzmann machines. In: *Proceedings of the 27th international conference on machine learning (ICML-10)*. [S.l.: s.n.], 2010. p. 807–814. Cited in page 26.

NASR-ESFAHANI, E. et al. Hand gesture recognition for contactless device control in operating rooms. *arXiv preprint arXiv:1611.04138*, 2016. Cited 13 times in pages 7, 15, 16, 19, 29, 32, 33, 34, 45, 48, 49, 55, and 56.

NG, C. W.; RANGANATH, S. Real-time gesture recognition system and application. *Image and Vision computing*, Elsevier, v. 20, n. 13, p. 993–1007, 2002. Cited in page 31.

O'HARA, K. et al. Touchless interaction in surgery. *Communications of the ACM*, ACM, v. 57, n. 1, p. 70–77, 2014. Cited in page 13.

OTINIANO-RODRÍGUEZ, K.; CÁMARA-CHÁVEZ, G.; MENOTTI, D. Hu and Zernike moments for sign language recognition. In: *Proceedings of international conference on image processing, computer vision, and pattern recognition*. [S.l.: s.n.], 2012. p. 1–5. Cited in page 31.

OYEDOTUN, O. K.; KHASHMAN, A. Deep learning in vision-based static hand gesture recognition. *Neural Computing and Applications*, Springer, p. 1–11, 2016. Cited 12 times in pages 7, 15, 16, 19, 33, 34, 35, 45, 48, 51, 55, and 56.

PALACIOS, J. M. et al. Human-computer interaction based on hand gestures using RGB-D sensors. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 13, n. 9, p. 11842–11860, 2013. Cited 2 times in pages 32 and 34.

PAN, S. J.; YANG, Q. et al. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, Institute of Electrical and Electronics Engineers, Inc., 345 E. 47 th St. NY NY 10017-2394 USA, v. 22, n. 10, p. 1345–1359, 2010. Cited in page 30.

PARK, K.-H. et al. Robotic smart house to assist people with movement disabilities. *Autonomous Robots*, Springer, v. 22, n. 2, p. 183–198, 2007. Cited in page 13.

PISHARADY, P. K.; SAERBECK, M. Recent methods and databases in vision-based hand gesture recognition: A review. *Computer Vision and Image Understanding*, Elsevier, v. 141, p. 152–165, 2015. Cited 2 times in pages 19 and 31.

PLOUFFE, G.; CRETU, A.-M. Static and dynamic hand gesture recognition in depth data using dynamic time warping. *IEEE Transactions on Instrumentation and Measurement*, IEEE, v. 65, n. 2, p. 305–316, 2016. Cited in page 32.

RASINES, I.; REMAZEILLES, A.; BENGOA, P. M. I. Feature selection for hand pose recognition in human-robot object exchange scenario. In: IEEE. *Emerging Technology and Factory Automation (ETFA), 2014 IEEE*. [S.l.], 2014. p. 1–8. Cited in page 19.

RAUTARAY, S. S.; AGRAWAL, A. Vision based hand gesture recognition for human computer interaction: a survey. *Artificial Intelligence Review*, Springer, v. 43, n. 1, p. 1–54, 2015. Cited 4 times in pages 13, 18, 19, and 31.

REN, S. et al. Faster r-cnn: Towards real-time object detection with region proposal networks. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2015. p. 91–99. Cited in page 29.

ROBERTSON, S. P.; ZACHARY, W.; BLACK, J. B. *Cognition, computing, and cooperation*. [S.l.]: Intellect Books, 1990. v. 2. Cited in page 13.

SABHARA, R. K.; LEE, C.-P.; LIM, K.-M. Comparative study of Hu moments and Zernike moments in object recognition. *SmartCR*, v. 3, n. 3, p. 166–173, 2013. Cited in page 31.

SANCHEZ-RIERA, J. et al. A comparative study of data fusion for RGB-D based visual recognition. *Pattern Recognition Letters*, Elsevier, v. 73, p. 1–6, 2016. Cited 4 times in pages 15, 19, 33, and 34.

SCHMIDHUBER, J. Deep learning in neural networks: An overview. *Neural networks*, Elsevier, v. 61, p. 85–117, 2015. Cited 6 times in pages 14, 15, 16, 23, 25, and 28.

SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. Cited 3 times in pages 48, 50, and 51.

SOH, J. et al. 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*. New York, NY, USA: ACM, 2013. (VRCAI '13), p. 231–234. ISBN 978-1-4503-2590-5. Disponível em: <http://doi.acm.org/10.1145/2534329.2534338>. Cited in page 13.

SRIVASTAVA, N. et al. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, JMLR. org, v. 15, n. 1, p. 1929–1958, 2014. Cited in page 28.

STEFANOV, D. H.; BIEN, Z.; BANG, W.-C. The smart house for older persons and persons with physical disabilities: structure, technology arrangements, and perspectives. *IEEE transactions on neural systems and rehabilitation engineering*, IEEE, v. 12, n. 2, p. 228–250, 2004. Cited in page 13.

SUAREZ, J.; MURPHY, R. R. Hand gesture recognition with depth images: a review. In: IEEE. *Ro-man, 2012 IEEE*. [S.l.], 2012. p. 411–417. Cited 2 times in pages 18 and 19.

TAHMASBI, A.; SAKI, F.; SHOKOUHI, S. B. Classification of benign and malignant masses based on Zernike moments. *Computers in biology and medicine*, Elsevier, v. 41, n. 8, p. 726–735, 2011. Cited in page 22.

TEH, C.-H.; CHIN, R. T. On image analysis by the methods of moments. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, IEEE, v. 10, n. 4, p. 496–513, 1988. Cited in page 21.

TURNER, M. R. Texture discrimination by gabor functions. *Biological cybernetics*, Springer, v. 55, n. 2-3, p. 71–82, 1986. Cited in page 35.

WANG, C.-C.; WANG, K.-C. Hand posture recognition using Adaboost with SIFT for human robot interaction. In: *Recent progress in robotics: viable robotic service to human*. [S.l.]: Springer, 2008. p. 317–329. Cited in page 13.

WANG, M.; CHEN, W.-Y.; LI, X. D. Hand gesture recognition using valley circle feature and Hu's moments technique for robot movement control. *Measurement*, Elsevier, v. 94, p. 734–744, 2016. Cited 2 times in pages 31 and 34.

YAMASHITA, T.; WATASUE, T. Hand posture recognition based on bottom-up structured deep convolutional neural network with curriculum learning. In: IEEE. *Image Processing (ICIP), 2014 IEEE International Conference on*. [S.l.], 2014. p. 853–857. Cited 4 times in pages 29, 45, 55, and 56.

YANG, S. *Robust human computer interaction using dynamic hand gesture recognition*. Tese (Doutorado) — School of Electrical, Computer and Telecommunications Engineering, University of Wollongong, 2016. Cited in page 32.

ZERNIKE, F. Beugungstheorie des Schneidenverfahrens und seiner verbesserten Form, der Phasenkontrastmethode. *Physica*, Elsevier, v. 1, n. 7-12, p. 689–704, 1934. Cited in page 21.

ZOBL, M. et al. A real-time system for hand gesture controlled operation of in-car devices. In: IEEE. *Multimedia and Expo, 2003. ICME'03. Proceedings. 2003 International Conference on*. [S.l.], 2003. v. 3, p. III–541. Cited in page 13.

Appendix

APPENDIX A – Supplementary material

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