Università Politecnica Delle Marche

Dipartimento di Ingegneria dell'Informazione

MASTER'S DEGREE IN BIOMEDICAL ENGINEERING

Preterm infants' limb-pose estimation: a dense-net approach

Relatore:

Prof. Emanuele Frontoni

Candidato:

Appugliese Simone

Correlatore:

Sara Moccia, PhD

Anno Accademico 2019/2020

Il fallimento è la nebbia attraverso la quale
scorgiamo il trionfo

Aldrich Killian

Contents

Useful links

- SINC project:

  https://vrai.dii.univpm.it/sinc

- Training detection preparation:

  https://colab.research.google.com/drive/1cnx4df03UVWA3wlR_UX9yGKI9IHd6XEi

- Training detection dense:

  https://colab.research.google.com/drive/1dMZ_c2-IibI4NALXij569IKca9AYRXRh?usp=sharing

- Prediction segmentation dense:

  https://colab.research.google.com/drive/1ChD4h9ybSCV2_wGpGLEA3bUADLEM7j-O#scrollTo=tOUQ9Dzlub7W

- BabyMovement dataset:

  https://drive.google.com/drive/folders/1cOiiyrgswQip-3zAXfhGfur_YFeKivXD?usp=sharing

Abstract

Preterm infants are babies born before the thirty-seven completed weeks of gestation. They are at higher risk developing an abnormal neuro-development outcome. Cognitive dysfunctions, motor impairments and behavioural disorders are among the major adverse implications of preterm birth. One of the most important problems concerning these disabilities is cerebral palsy (CP). Despite being recognized as a crucial clinical task, preterm-infants' movement evaluation is merely qualitative and episodic, and mostly based on clinicians' assessment. This kind of evaluation is time consuming and inaccurate due to clinicians' fatigue.

In literature two different approaches are proposed to analyse movement of preterm infants. The first one is based on wearable sensors, which are placed on infant limbs. These kinds of sensors are not used in the actual clinical practice because they are directly in contact with the infants, possibly causing discomfort, pain and skin damage while hindering infant's spontaneous movements. The second approach is based on camera sensors, which are more suitable than wearable ones as they are non-invasive. These sensors are installed to allow clinicians interacting with preterm infant's crib without any obstructions.

The work developed in this thesis is based on the generalization power of deep learning. It is proposed a 2D fully convolutional neural network (FCNN) with the addition of two dense block (DB) along the skip connections to estimate preterm infants' limb-pose. The analysed video sequences are obtained from RGB-D cameras and they composed the BabyPose dataset. Depth images are used to respect the privacy of the preterm infants.

The results obtained are of notable interest. It is worth observing that the proposed method was able to correctly detect visible joints when other ones were occluded. Indeed, the median Dice similarity coefficient (DSC) and recall (Rec) among all joints was equal to 0.854 and 0.839, respectively. Moreover, the testing of the model was performed using a mid-range laptop. It obtained an average detection time of 0.014 s per image. For this reason, it may be integrated in the domestic environment to estimate limb-specific pose of the infants in real-time.

Chapter 1

Introduction

This chapter is organized as follows: Section 1.1 reports the clinical background of preterm infants, Section 1.2 highlights the importance of monitoring limb-pose infant movement. In Section 1.3 the SINC project is described, Section 1.4 presents the overview of the thesis.

1.1 Clinical background on preterm infants

The World Health Organization defines *preterm infants* as infants born before the thirty-seven completed weeks of gestation. Every year there are more than fifteen million worldwide preterm births and the number of cases continues to increase [1]. In almost all high-income Countries, complications of preterm birth are the largest direct cause of neonatal deaths, accounting for the 35% of the world deaths a year. Today premature birth is the leading cause of mortality in the first year of life and the second cause of death in the first 5 years. Preterm infants are divided in three sub-groups, based on gestational age:

- Extremely preterm (less than 28 weeks)
- Very preterm (28 to 32 weeks)
- Moderate or late preterm (32 to 37 weeks)

The pathophysiology that leads to a preterm birth is mostly unknown, nevertheless, contributing maternal, foetal and placental predisposing factors have

been identified. Some of these are the antepartum haemorrhage, uterine overdistention, and bacterial infection and inflammation [2, 3]. Epidemiological studies have been conducted to determine the major risk factors that cause preterm birth. The most common amongst them include: an overweight pre-pregnancy body mass index, a maternal age of less than 17 years or more than 35 years, smoking and, physical and psychological stress. Preterm birth rates vary by geography and ethnicity, moreover, low and middle income countries (LMIC) have higher rates [3, 4].

When a preterm delivery is identified, an antenatally plan is arranged. This may be possible if the hospital has a tertiary level neonatal unit, otherwise, an antenatal transfer is needed. The clinicians have to provide a series of antenatal steroids to reduce the risk of death, intraventricular hemorrhage (IVH), and respiratory distress syndrome (RDS) in infants [5]. Furthermore, magnesium sulphate needs to be supplied since it is neurprotective to the newborn [6]. Subsequent to the labour and initial management, the preterm infants need to be monitored because of all organ systems are immature. The main complications of each system are:

- Respiratory system: Respiratory distress syndrome, Surfactant deficient lung disease, Chronic lung disease and recurrent apnoea.
- Cardiovascular system: Hypotension, perfusion abnormalities and Patent ductus arteriosus (PDA).
- Neurological system: Intraventricular haemorrhage, cerebral palsy,ventricular dilatation and neuro-developmental delay.
- Immune system: Sepsis
- Gastrointestinal system: Immature gut causing feed intolerance and necrotising enterocolitis (NEC).
- Metabolic system: Jaundice, hyperglycaemia and hypoglycaemia.
- Thermoregulation system: Immature thermoregulation.

- Visual system: Retinopathy of prematurity.

Cognitive and psychiatric dysfunctions, motor impairments and behavioural disorders are among the major adverse implications of preterm birth. Cognitive shortages happen in 25-50% of preterm infants, in particular if they were born with a weight less than 1500 g. One of the most important problems concerning these disabilities is cerebral palsy (CP). It is a permanent disorder in the development of movement and posture and is one of the major disabilities that affects up to 18% of infants who are born extremely preterm [7].
Preterm birth is a significant medical and social issue that needs to be monitored both promptly and in the first years of life of baby. Indeed, preterm infants born with a weight below the average represent 63% of the causes of mortality under 5 years of life.

1.2 The relevance of monitoring movement in preterm infants

Infants' spontaneous motility is a valuable diagnostic and prognostic index of infants' cognitive and motor development. For more than 25 years, the Prechtl General Movement Assessment (GMA) has been used as a non-intrusive and reliable technique. The GMA evaluates the functioning of the newborn's nervous system [8, 9]. The neonatal nervous system produces several motor patterns spontaneously. These involve twitches, stretching, yawing and general movements (GMs). The latter includes the movement of the whole body in a variable sequence of legs, trunk, arms and neck movements [8].

From early fetal life until the end of the second month after term GMs shows analogous aspects. They are called "writhing movements" from term age onward. This kind of movements gradually disappear from 6-9 weeks of postterm age, and GMs take on a fidgety character. The "fidgety movements" are little motion of neck, trunk and limbs in all directions with a variable acceleration that are noticeable from 3 to 5 months after term [9, 10].

GMs are produced by the central pattern generators (CPGs), which are

mainly located in the brainstem. Supraspinal projections take under control and modulate CPGs activity in order to regulate the motor output [11, 12]. Fetal or neonatal issue is indicated by a reduced modulation of the CPGs that shows a small range of motion (i.e. abnormal).
During preterm and term age, abnormal GMs are divided in three different categories:

- Poor repertoire GMs: the succession of movements is monotonous and the speed, intensity and variability of movement are different from a healthy infant.
- Cramped-synchronized GMs: the movements are less smooth and fluent, moreover, when the limb and trunk muscles both contract or relax simultaneously, the motion results rigid.
- Chaotic GMs: the movements are abrupt and quivering. The range of motion is wider and the speed is higher. This kind of GMs are usually related to the moderate preterm age.

Abnormal fidgety movements are overstated in amplitude, speed and jerkiness. When this kind of movement occurs occasionally or completely absents at 3-5 months, the likelihood of developing a neurological disease such as cerebral palsy is elevated [9, 10, 13]. The GMA has been growing exponentially as a prediction method for motor dysfunctions such as cerebral palsy, since its introduction 30 years ago [8]. Great indexes of prediction of cerebral palsy are cramped-synchronized GMs and the lack of fidgety movements, while lower neurological dysfunctions are related to poor repertoire GMs and abnormal fidgety movements [9, 12]. Furthermore, recent studies are focused on understanding whether or not GMA could also be related to cognitive and language development and, behavioral, mental, and genetic disorders [12, 13, 14].

To avoid preterm infants' to develop abnormal neuro-developmental outcomes, early detecting motor impairments is crucial to properly define intervention programs [15]. Clinicians in Neonatal Intensive Care Unit (NICU) have to pay particular attention to the monitoring of infants' limb so that is possible to notice an early issue such as cerebral palsy [16].

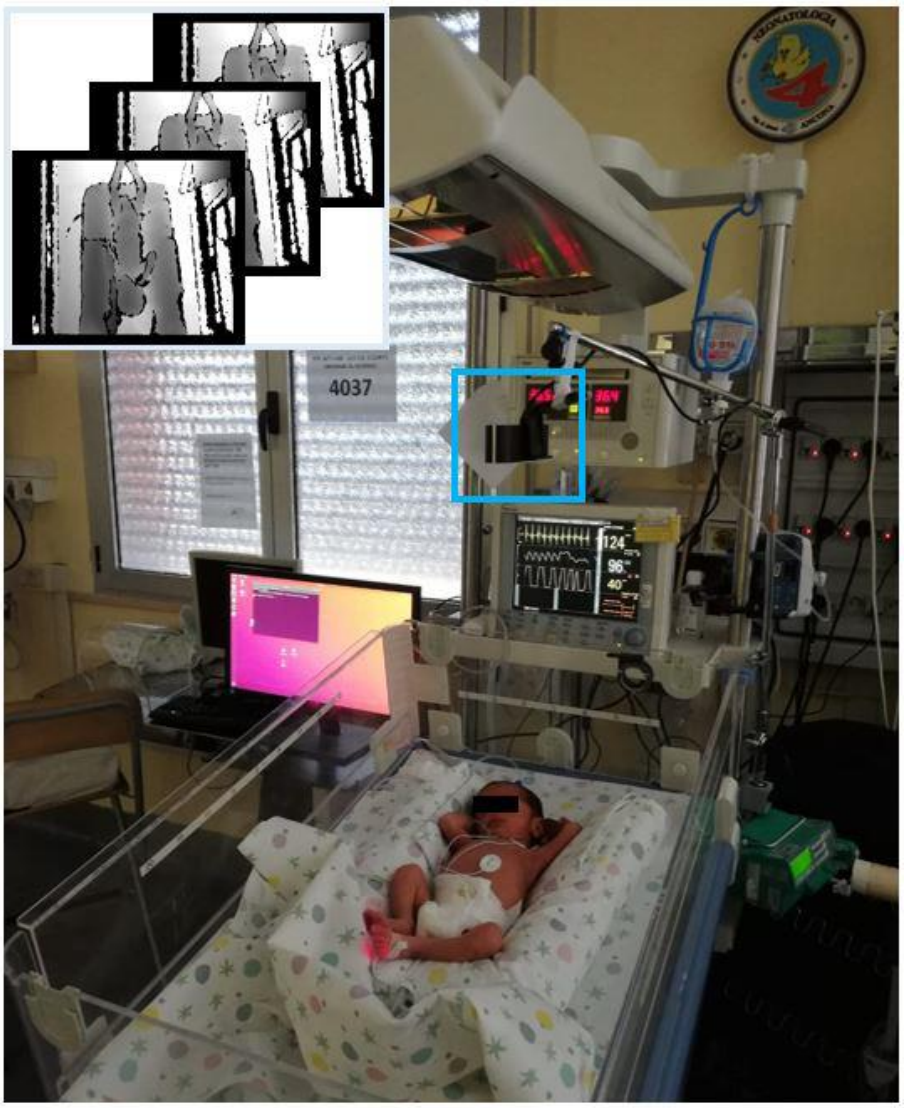

Figure 1.1: *Depth-image acquisition setup. The depth camera is positioned at about 40cm over the infant's crib and does not hinder health-operator movements.*

Despite being recognized as a crucial clinical task, preterm-infants' movement evaluation is merely qualitative and episodic, and mostly based on clinicians' assessment at the crib side in NICUs or review of infants' video-recordings. This kind of evaluation is time consuming, inaccurate due to clinicians' fatigue and susceptible to intra and inter-clinician variability [15]. Moreover, there are no standardized clinical guidelines to predict motor impairment of infants and so the recognition of that is based on the experience of healthcare professionals [17]. All this leads to a lack of quantitative parameters inherent on the matter.

A possible solution to overcome the issue of qualitative monitoring could be the utilization of an automatic video-based system for tracking the infants' limb pose. It can be easily integrated into standard clinical monitoring setup as this kind of sensors does not hamper movements of newborns and healthcare operators. The camera could be positioned on top of infants' crib, leaving health operators free to move and interact with the infants. (Fig. 1.1).

Some preliminary studies proposed by [18] and [19] have reached results for infant's whole-body segmentation with threshold-based algorithms. Nevertheless, during the health-assessment process, each limb has to be monitored individually to best aid clinicians. Furthermore, the estimation of the preterms' limb-pose is not a trivial task because of small and hardly visible joint, presence of limb occlusions, lighting changes, and variable distance between camera and infants. Methods for monitoring the movement of preterm infants are discussed more in details in Section 2.1

In order to tackle these drawbacks, it is implemented an automatic system that exploits Artificial Intelligence methods for the analysis of depth images. The aim is to improve the utilization of convolutional neural networks (CNN) for the estimation of infants limb pose. It is added two dense block (DB) to the 2D fully convolutional neural network (FCNN) in order to detect limb joints and joint connections. It has been decided to use depth images instead of RGB ones in order to respect the privacy of the patients.

1.3 The SINC project

The proposed algorithm estimates the pose of preterm infants using depth images acquired in NICU. The work of this thesis is part of the SINC project. The aim of the project is to develop an innovative integrated system that allows improving the neonatal ward through the application of new products and a new organizational model. The main purposes of SINC are the following:

- Integrate a new system on the crib in order to detect the main physiological parameters of the infant without physical contact with the subject.
- Produce a cloud service that makes available services and utilities in the Marche Region that handle and integrate the monitoring data acquired through these new devices to improve the diagnosis.
- Build a new hospital model for the management of newborn cribs and medical care between the several hospitals in the Marche Region, supported by tools and distributed data management systems.

The project is developed in Neonatal Intensive Care Unit in G. Salesi hospital. The proposed modernisation allows obtaining contact-less measurements for principal physiological parameters to monitor the state of health of preterm infants. The purpose of this thesis is to develop a system that automatically estimates limb-specific pose of the infant in order to integrate it in the domestic environment. This method allows evaluating the spontaneous motility of preterm infants, which is a predictor of cognitive disorders such as cerebral palsy.

1.4 Thesis overview

In this section a brief summary of the work proposed is reported. The thesis is organized as follows:

- Chapter 2 summarise the state of the art about monitoring of preterm infants' spontaneous limb-movement.

- Chapter 3 presents the essentials of Deep Learning, focusing on functions and algorithms applied during the developing of the model.
- Chapter 4 analyses more in details the architecture of the network utilised.
- Chapter 5 presents all the results obtained.
- Chapter 6 discusses the results presented in the previous chapter and it focuses on future developments.
- Chapter 7 presents the conclusions about the model.

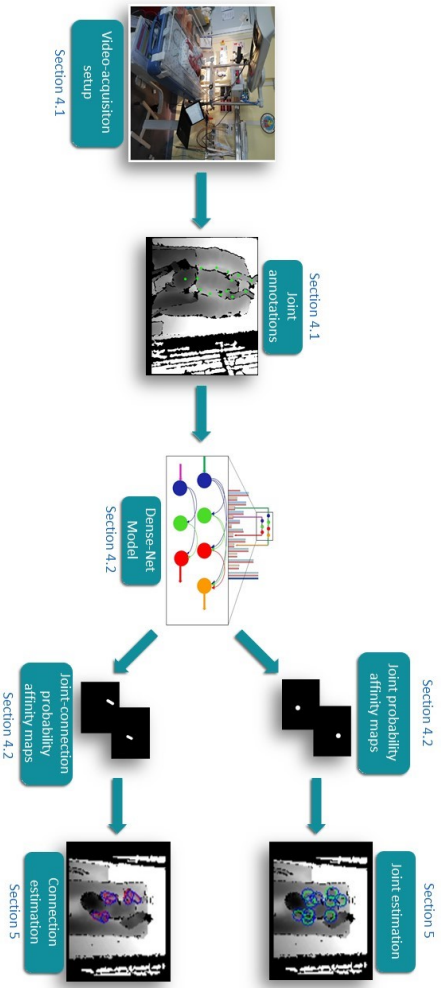

Figure 1.2: *Representation of the workflow of the thesis.*

The workflow of the thesis is reported in Fig. 1.2.

Chapter 2

State of the art

In the past decades many researches have been conducted to automatically monitor preterm infants' spontaneous limb-movement. In the following chapter different approaches proposed in the literature will be discussed in Section 2.1. Moreover, a brief introduction of the method will be exposed in Section 2.2.

2.1 Methods for monitoring the movement of preterm infants

In literature different approaches have been proposed to monitoring limb-movement of preterm infants. They are mainly divided in two broad categories: wearable sensors and camera sensors. The former are discussed in Section 2.1.1 and the latter in Section 2.1.2.

2.1.1 Wearable sensor-based approaches to monitor preterm infants' limb-movement

Wearable sensors are placed on infant limbs to detect his/her movement. In particular, Trujillo-Priego et al. [20] proposed an algorithm to determine the number of arm movement bouts an infant produces across a fully day in the natural environment. Wearable sensors placed on wrist record tri-axial accelerometer and gyroscope data at many samples per second. In this manner they acquire

movement data unobtrusively across many continuous hours.

Figure 2.1: *Representation of the wearable sensors presented by Smith et al. [21] (2015). In picture is shown three-month-old infant wearing sensors on the front of each ankle.*

Another method proposed by Smith et al. [21] is based on wearable sensors placed on knees. They use inertial movement sensors to record fully-day leg movement activity from infants. Their algorithm determines the number of leg motion that an infant performed during a day.
Other systems utilise wireless accelerometer [22, 23, 24] or electromagnetic sensors [25, 26], however, their utilization is soon ceased.

These kinds of sensors have some limitations because are time-consuming and need to be constantly recalibrated. Moreover, even though miniaturized, these sensors are directly in contact with the infants, possibly causing discomfort, pain and skin damage while hindering infant's spontaneous movements. Because of such drawbacks wearable sensors are not used in the actual clinical practice.

2.1.2 Camera sensors

Meinecke et al. [27] are among the first to exploit optical flow to estimate preterm infants' spontaneous movement in RGB-image streams.

Figure 2.2: *Representation of the tracking system presented by Meinecke et al. [27] (2006). In picture (a) is shown a photo of the 7 cameras positioned around the preterm infant. In figure (b) are shown the markers positioned on the body of the preterm infant.*

Their system (Fig. 2.2(a)) is based on seven infrared cameras that tracking twenty markers placed on the infants' body (Fig. 2.2(b)). After markers' calibration, knowing the position of them, it is possible to compute the rotation of head and trunk of the infant, and the movement of the limbs. However, the use of markers makes hard to apply this method in a clinical environment due to occlusion of markers in the limbs.

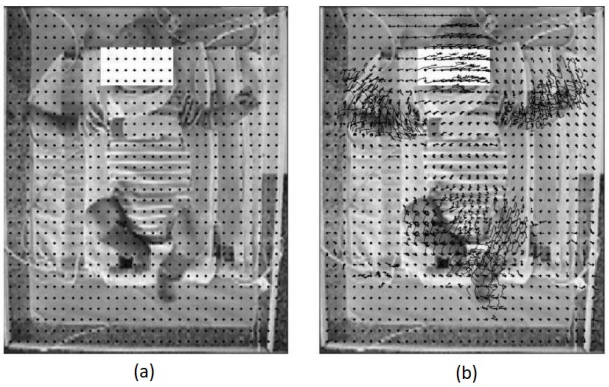

Figure 2.3: *Representation of the monitoring system presented by Stahl et al. [28] (2012). In figure (a) it is possible to notice the grid that is placed above the image. Figure (b) shows the variation of the points due to a movement of both the head and the limbs.*

A further method based on optical flow is proposed by Stahl et al. [28]. The image is divided in a grid of points (Fig. 2.3(a)) that follows the movement's trajectory of infant (Fig. 2.3(b)). The spots of hands, feet and head are manually selected and monitored in order to estimate the baby's pose. Shivakumur et al. [29] implemented a multiple view stereoscopic 3D vision system that provides better resolution than classical RGB cameras. Firstly the algorithm localizes the centre of body, subsequently, others parts of body infants such as hands and legs. All the selected regions are segmented based on a colour threshold. The evaluation of the system is done on sixty photograms manually annotated. However, the two approaches proposed in [28] and [29] are semi-supervised and computational expensive, hampering their use in the clinical practice.

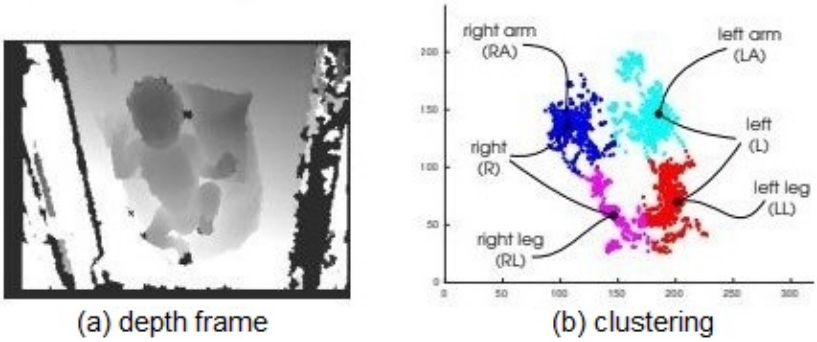

Figure 2.4: *Representation of the approach proposed by Cenci et al. [1] (2017). In figure (a) is shown a depth image, while in figure (b) the k-means classification of the limbs is represented.*

Cenci et al. [1] proposed an automatic threshold approach based on the difference between two consecutive depth frames (Fig. 2.4(a)) with a certain delay each other. The image obtained is processed through K-means clustering technique (Fig. 2.4(b)). The advantage of this method is that it is objective, contactless, non-invasive and suitable to be installed in an indoor environment like Neonatal Intensive Care Unit. Moreover, it preserves both infants' and operators' privacy. Nevertheless, it is high sensitive to noise and inter-subject variability due to the fact that the threshold value has to be set manually. In addition, there are others drawbacks when processing depth images such as variable distance between the crib and the camera, and variability in infants' movements.

Figure 2.5: *Representation of the approach proposed by Cao et al. [30] (2017). Multi-person pose estimation. Body parts belonging to the same person are linked.*

Inspired by recent considerations, a possible solution to overcome these challenges could be the utilisation of deep learning, mainly the exploitation of CNN. Cao et al. [30] suggested a deep-learning approach to efficiently detect the 2D pose of multiple people in an image (Fig. 2.5). This model consists of two CNNs, the first one is a detection FCNN to obtain joint probability maps and the second one is a regression CNN to improve estimate of joints' position. This method obtains a high accuracy despite of the number of people in the image.

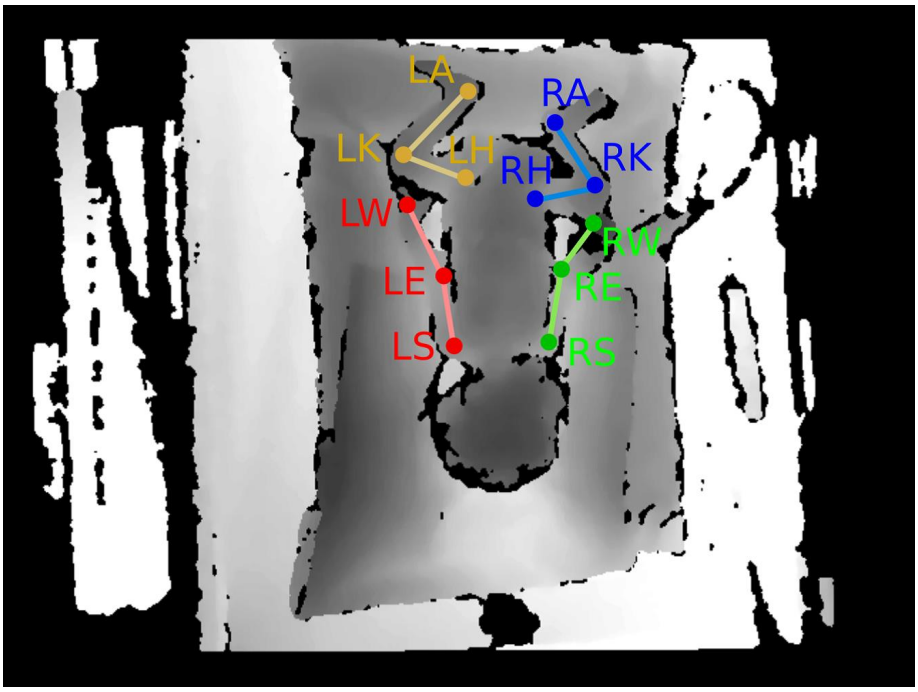

Figure 2.6: *Representation of the approach proposed by Moccia et al. [31] (2019). LS and RS: left and right shoulder, LE and RE: left and right elbow, LW and RW: left and right wrist, LH and RH: left and right hip, LK and RK: left and right knee, LA and RA: left and right ankle.*

Moccia et al. [31], inspired by [30], implemented a model based on two consecutive CNNs to estimate the pose of preterm infants (Fig. 2.6). The first CNN is used to detect limb joints and joint connections, while the second one is applied to predict accurately joint and joint-connection position. The approach used by [31] accomplishes comparable results for the detection of all joints, highlighting that it is able to process in parallel different joint probability maps. Moreover, it is capable to detect visible joints in the right manner, although the other ones are occluded.

The model presented in this thesis automatically estimates limb-specific pose of the infant. It is proposed a 2D fully convolutional neural network (FCNN) with the addition of two dense block (DB) in the skip connections to improve the detection of joints and joint connections.

2.2 Overcoming the state of the art

The approaches reported in the state of the art show some differences. Wearable sensors exhibit various obstacles such as irritating the skin of the infant and hindering his spontaneous movements. In addition, these sensors have to be constantly recalibrated. Camera-based approaches are more suitable than wearable ones, because these sensors are non-invasive as they are not in direct contact with the preterm infant's body. Moreover, camera sensors are installed to allow clinicians interacting with preterm infant's crib without any obstructions.

The following work wants to exploit the generalization power of deep learning, starting from the method proposed by Moccia et al. [31]. The difference between the implemented model and the approaches previously proposed in the literature is based on the addition of the two DB along the skip connections. This has led to an improvement in the detection of joints and joint connections. More in details, the system has to extract frames from the video sequence and each frame is passed to the model in order to estimate preterm infants' limb-pose.

Chapter 3

Deep Learning

Deep Learning is an evolution of traditional machine learning approaches. It is an Artificial Intelligence technique based on different levels of representation. Each layer uses the output of the previous one as input in order to obtain a progressively abstract representation of the data. This model replicates the behaviour of brain cells. Indeed, each level matches to a different layer of the cerebral cortex.
The following chapter presents the essential of Deep Learning, describing more in deep the characteristic of the model used.

3.1 Convolutional neural network

Convolutional Neural Networks are based on Artificial Neural Networks, which are inspired by biological neural networks. An ANN is composed of nodes, called artificial neurons, which model the neurons in a biological brain. Each connection between nodes transmits a signal that is a real number. Neurons and connections are adjusted during learning procedure by a weight. The nodes constitute layers, the first one is known as *input layer* while the last one as *output layer*. Among of them there are one or two intermediate layers called *hidden layer*.

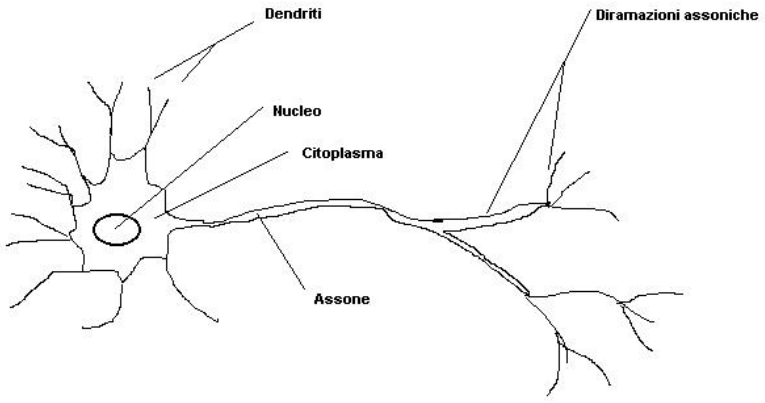

Figure 3.1: *Biological representation of neuron.*

The simplest form of ANN is the *perceptron* proposed by Frank Rosenblatt [32] and inspired by the function of the neuron. A neuron is the main component of nervous tissue. Human nervous system is made of about 86 billion of neurons, which communicates with other cells through specialized connections called synapses. A biological representation of neuron is shown in Fig. 3.1. Each of them takes inputs from its dendrites and inside the soma a weighted sum of these inputs is performed. The signal arrives to the axon hillock. If the result is higher than the threshold limit, the signal propagates along the axon and it is run to the successive neurons through synapses.

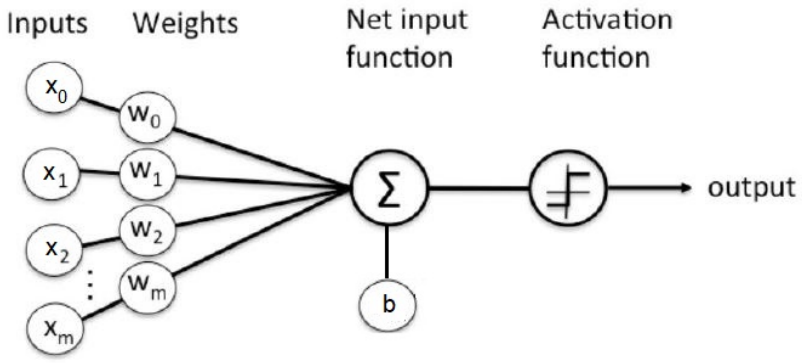

Figure 3.2: *Representation of the Rosenblatt perceptron [32]. It receives M value inputs x_m and each of them is multiplied by the corresponding weight w_m. Subsequently all the products are added together and passed to the activation function.*

The perceptron, shown in Fig. 3.2, mathematically represents the biological activity of the neuron. The input signals (x_m) communicate with the synapses (w_m) and give ($w_m * x_m$) as product. The synaptic weights have different values because some inputs influence the output more than others. Moreover, they can be negative and therefore have an inhibitory influence. All the products are summated, if the result overcomes threshold (bias b) the neuron fires. The weighted sum is modelled by an activation function g. The sigmoid function is usually applied ($\sigma(x) = \frac{1}{(1+e^x)}$).
The output can be expressed in the following way:

$$output = \hat{y} = g(\sum_{m=1}^{M} w_m x_m + b) \tag{3.1}$$

Where M represents the number of inputs.

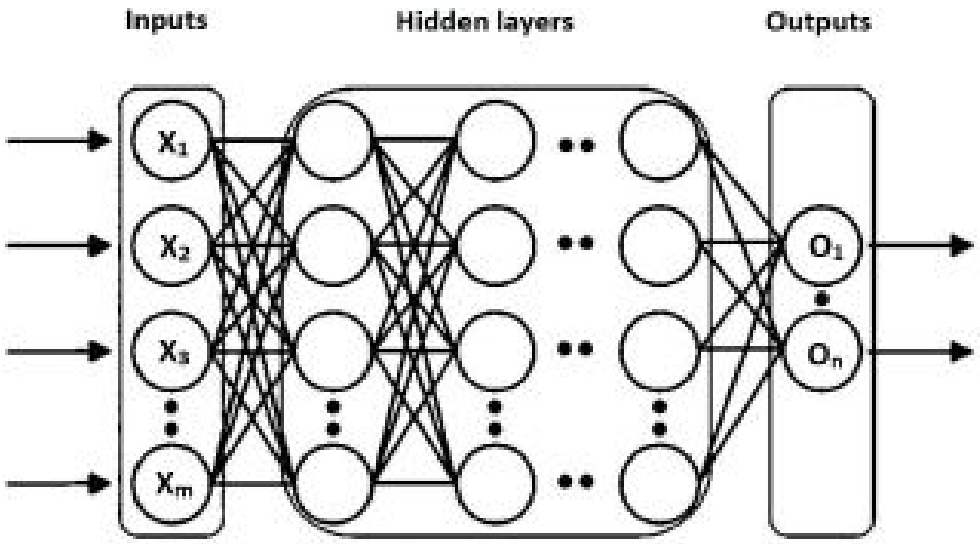

Figure 3.3: *Representation of a generic Deep Neural Network with m input* ($x_i, i \in [1, m]$) *n outputs* ($O_i, i \in [1, n]$) *and different hidden layers.*

The problem of the perceptron is that it is a linear classifier and thus it is never used individually. Its main application is to be used in more complex ANN, called Deep Neural Network (DNN). It is an ANN with multiple layers between the input and output layers. An example of DNN is shown in Fig. 3.3.

The main advantage of this architecture is to be able to implement DNNs with a much lower number of nodes than ANNs with only one hidden layer. Hence, a model with fewer weights to learn requests a smaller dataset for training and the size of dataset is often an issue [33]. However, the problem of overfitting and computation time is shown by the use of DNNs. This is due to the high number of hidden layers and training parameters.

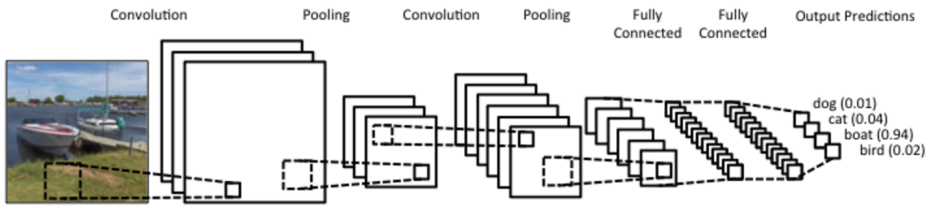

Figure 3.4: *Representation of a generic CNN architecture having convolutional, pooling and fully connected layers.*

A different kind of ANN is represented by Convolutional Neural Network, introduced by YannLeCun [34] in 1990. An example of it is shown in Fig. 3.4. In the following subsections are described the different layers utilised in the architecture of the model.

Convolutional layer

The first difference between CNNs and DNNs is constituted by the presence of convolutional layer. It consists of a set of learnable filters, called *kernel*. It is similar to a matrix of small spatially dimensions (height and width) and as deep as the input image. The application of kernel reduces the number of parameters because the number of weights is independent of the size of the input image.

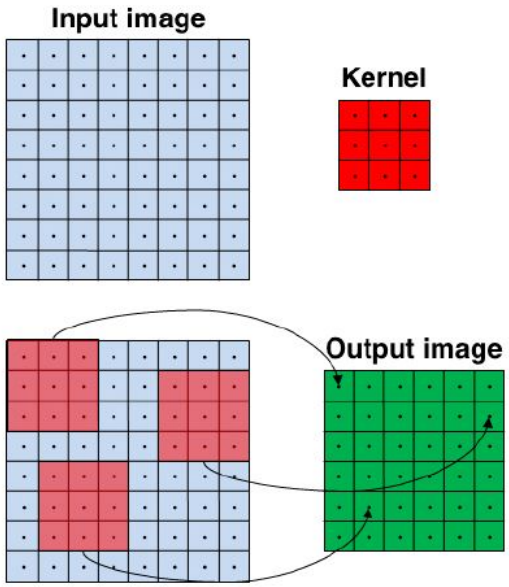

Figure 3.5: *Representation of a 3x3 kernel that slides along the input image making the weighted sum at each step.*

The convolution consists of the sliding of each filter along the width and height of each channel of the input volume. It computes the dot product between the values of the kernel and the portion of the overlapping image. An example of convolution is shown in Fig. 3.5.
The convolution can be mathematically seen as a multiplication of matrices:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} * \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} = (1 \cdot a) + (2 \cdot b) + (1 \cdot c) + ... + (9 \cdot i) \tag{3.2}$$

The result is a weighted sum that gives pixel value of the output image. Therefore for each convolutional layer the input image is convolved with a set of kernel $W = \{W_1, W_2, ..., W_K\}$. The result is sum with bias vector $B = \{b_1, b_2, ..., b_K\}$ in order to obtain a feature map X_K. Subsequently, all the feature maps are modelled by the activation function g and the result for the k-th kernel of the convolution layer l is:

$$X_k^l = g(W_k^l * X^{l-1} + b_k^l) \tag{3.3}$$

The size of the output volume of each layer is controlled by the *depth*, *stride* and *zero-padding*. The first one corresponds to the number of kernels applied during the convolution, each learning to look something different in the input. For example, if the first convolutional layer takes as input the raw image, then different neurons along the depth dimension may activate in presence of various oriented edges, or blobs of colour. The *stride* is the step size where the kernel is slid along the image. Thus if the stride is 1, the filter moves one pixel at a time. This parameter controls the output volume along its height and width. The last one is the *zero-padding*, a technique that allows maintaining the original input size.

The aim of the convolutional operation is to extract the *high-level features* from the input image. In a CNN more convolutional layers are applied. The first ones capture *low-level features* such as edges, colour and gradient orientation, while the deeper layers extract *high-level features*.

Pooling Layer

The other difference between CNNs and DNNs is due to the pooling layer (Fig. 3.4). It is a kind of down-sampling that progressively reduces the spatial size of the feature maps. This is done to decrease the amount of parameters and computational power required to process the data. In this way it is possible to control overfitting. There are two types of pooling: *max pooling* and *average pooling*. The difference among them is how the output value is computed. In the first case it returns the maximum value from the portion of the image covered by the window, generally of size 2 x 2 or 3 x 3. On the other hand, average pooling returns the average of all the values. The window is scrolled along the spatial dimension of the feature map by a fixed stride.

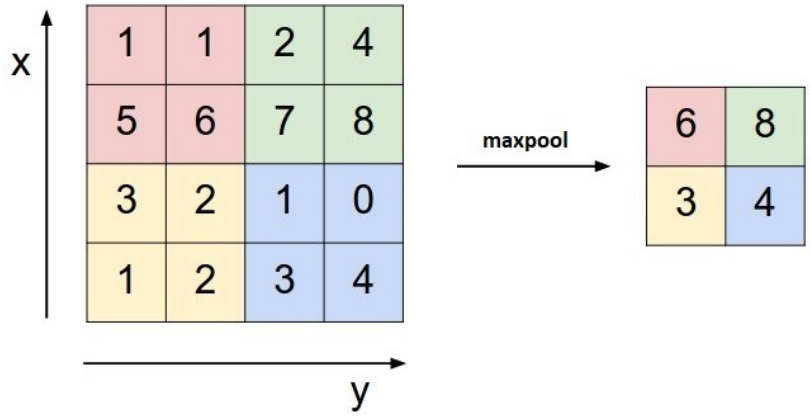

Figure 3.6: *Representation of the Max-pooling operation with a window of size 2x2 and a stride of 2 pixels.*

An example of max pooling is shown in Fig. 3.6.

3.2 Minimum's research

The aim of a CNN is to find a function $f(x, W)$ where $\mathbf{x}$ is the input and

$$W = [w_1, w_2, w_3, ..., w_n]$$

Is the vector that represents the values of kernels called *weights* $(w_1, w_2, .., w_n)$. Each one of these values is computed during *training* in order to minimize the loss function *f*. The minimization of loss function is mathematically expressed as:

$$min\left\{L(W)\right\} = min\left\{\frac{1}{N}\sum_{i=1}^{N}(Li(f(x_i, W), y_i))) + \lambda R(W)\right\} \tag{3.4}$$

where the first part of the sum represents the loss function on the training set data, $R(W)$ and λ indicate, respectively, the regularization's term and its parameter.

The training phase is divided in different steps. The first one is the initialization of weights with random values. Afterwards, all the data that constitute the training set pass through the layers of the architecture. The output from the

network is the *predicted label*. In this way, the prediction error is determine by comparing the predicted label with the reference one. The error is computed applied the *loss function* in order to update the weights. The training process ends when the value of loss function is minimized.

3.2.1 Loss function

The loss function allows defining a value that indicates the difference between the predicted label and reference one.
In general it can be expressed in the following way:

$$L(W) = \frac{1}{N}\sum_{i=1}^{N}(Li(f(x_i, W), y_i)) \tag{3.5}$$

Where:

- N: is the number of sample in the training set
- W: is the weights' matrix
- $f(x_i, W)$: is the model prediction based on the sample x_i
- y_i: is the reference label of the sample x_i
- $Li(f(x_i, W), y_i)$: is the prediction error

An example of loss function is the **Binary Cross-Entropy** loss. It is a Sigmoid activation plus a Cross-Entropy loss. It is independent for each vector component, thus the loss value computed for every CNN output vector component is not influenced by the other ones. It is mainly used in binary classification tasks and multi-label classification.
The Binary Cross-Entropy can be expressed as:

$$H(p) = \frac{1}{N}\sum_{i=1}^{N}(y_i log(p(y_i)) + (1 - y_i) log(1 - p(y_i))) \tag{3.6}$$

Where:

- N is the number of sample in the training set
- y_i is the target label
- $p(y_i)$ is the prediction probability

3.2.2 Regularization

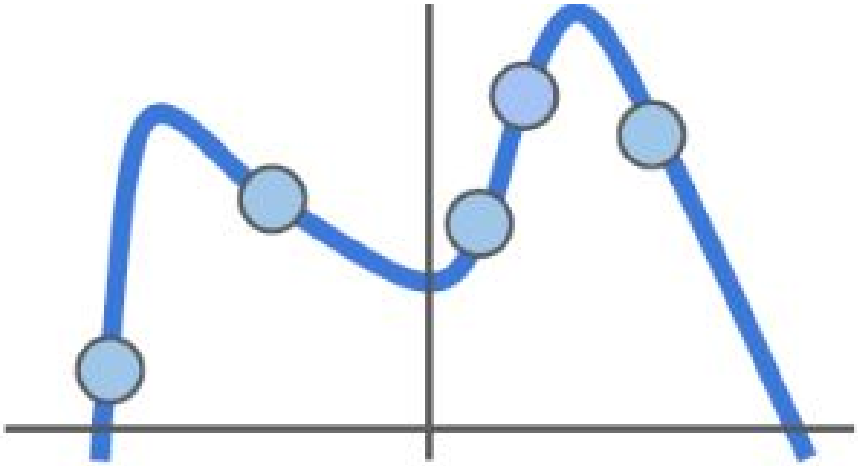

Figure 3.7: *Representation of the overfitting problem. The f function, represented as the blue curve, is adapted to the features of the input data.*

When the model learned patterns specific to the training data, it is occurred **overfitting**. A classic example of overfitting is shown in Fig. 3.7.
The blue points represent the data of the training phase. From the interpolation of these points, an approximation of the loss function f is obtained.
The consequence is that the model looks for the best fit for training data and therefore the training metric improves, meanwhile the test metric stops the enhancement and begins to decrease.

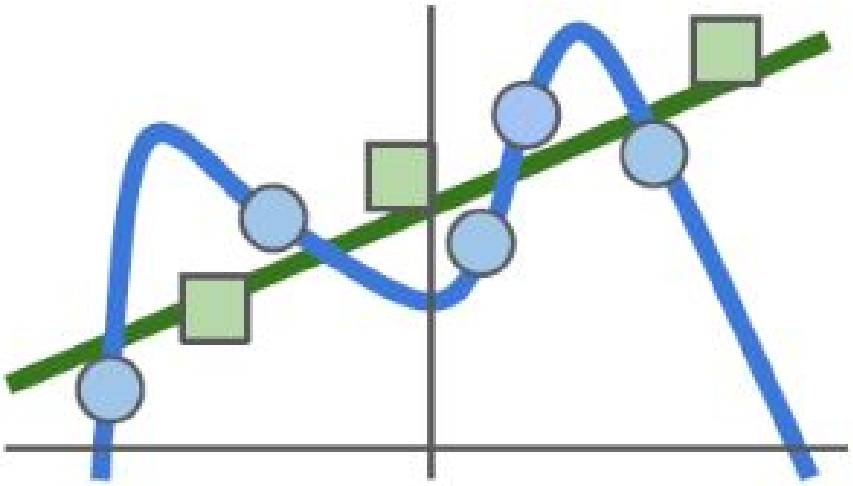

Figure 3.8: *Comparison between overfitting, represented by the blue curve, and generalization, represented by the green line.*

The desirable graph is illustrated in Fig. 3.8. Also in this case the blue points represent the data of the training phase while the green ones the prediction over the testing data. The approximation of loss function f is represented by the green line. The consequence is that the prediction is close to the training data and hence, the network is able to generalize. The regularization process allows overcoming the overfitting. Mathematically it is expressed by the positive term $R(W)$ and is summed to the loss function (formula (3.5)) in order to obtain:

$$L(W) = \frac{1}{N} \sum_{i=1}^{N} \left(Li(f(x_i, W), y_i)\right) + \lambda R(W)$$

The regularization term $R(W)$ penalizes the weights' matrix of the nodes.

An example of regularization process is the **Batch Normalization (BN)**. It is a method applied on ANNs to make them faster and more stable. The utilization of this technique is necessary to cope with the problem of **internal covariate shift**. It is the change in the distribution of network activations due to the change in network parameters during training.

The BN allows normalizing each output layer, in order to overcome the problem of internal covariate shift. The first stage is the computation of both the mean and the variance of a batch (B) of N samples (x):

$$\mu_B = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{3.7}$$

$$\sigma_B^2 = \frac{1}{N}\sum_{i=1}^{N} (x_i - \mu_B) \tag{3.8}$$

Afterwards, each sample (x) is normalized by subtracting the batch mean and dividing by the batch standard deviation. The sample normalized is symbolized by $\hat{x}$:

$$\hat{x_i} = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{3.9}$$

is a constant positive value, such as 0.001, that gives more stability. The last step is to perform a scale and shift. Therefore, the output layer is:

$$y_i = \gamma\hat{x_i} + \beta \tag{3.10}$$

Where γ represents the "standard deviation" parameter and β is the "mean parameter". y_i are the normalized feature maps that will then be activated.

During the training phase the *backpropagation* algorithm is applied. It computes the gradient of loss function with respect to weights in order to obtain β and γ values. The testing phase has some differences from the training one. The testing network has only one input that gives a single output. Thus, it is useless to apply the batch normalization method. For this reason, during the testing phase the model used both the batch mean and the batch variance determined in the training phase.

3.2.3 Optimization

The aim of the optimization procedure is to find the values of the vector weights W that minimizes the loss value (Formula (3.4)).
This is done mathematically through the gradient descent method. It is used to minimize the loss function by iteratively moving in the direction of steepest descent as defined by the negative of the gradient.
The gradient is the vector of the partial derivatives of the loss function along each dimension w_i. Formally is expressed as:

$$\nabla_w L(W) = [\frac{\partial L}{\partial w_1}, \frac{\partial L}{\partial w_2}, ..., \frac{\partial L}{\partial w_n}]$$

Given the formula:

$$W_{new} = W_{old} + \Delta w \tag{3.11}$$

where Δw is the term that update the weights. It is the product between the learning rate (η) and the negative gradient ($-\nabla_w L(W)$), as reported:

$$\Delta w = \eta(-\nabla_w L(W))$$

From which it is obtained:

$$W_{new} = W_{old} - \eta \nabla_w L(W) \tag{3.12}$$

In matrix form is expressed as:

$$\begin{bmatrix} w_1 \\ w_2 \\ .. \\ w_n \end{bmatrix}_{new} = \begin{bmatrix} w_1 \\ w_2 \\ .. \\ w_n \end{bmatrix}_{old} - \eta \begin{bmatrix} \frac{\partial L}{\partial w_1} \\ \frac{\partial L}{\partial w_2} \\ .. \\ \frac{\partial L}{\partial w_n} \end{bmatrix}$$

The *backpropagation* technique is based on it for updating the weights.

In literature there are various algorithms applied to find the descent of the gradient such as **Stochastic Gradient Descendent (SGD)**. It is mainly used for dataset having a huge amount of data. The gradient descent method is applied for each training sample because is batch size is equal to one. The **batch size** is the number of training samples utilized in one iteration. An **iteration** corresponds to the number of batches needed to complete one epoch that is when the entire dataset passes forward and backward through the neural network only once.

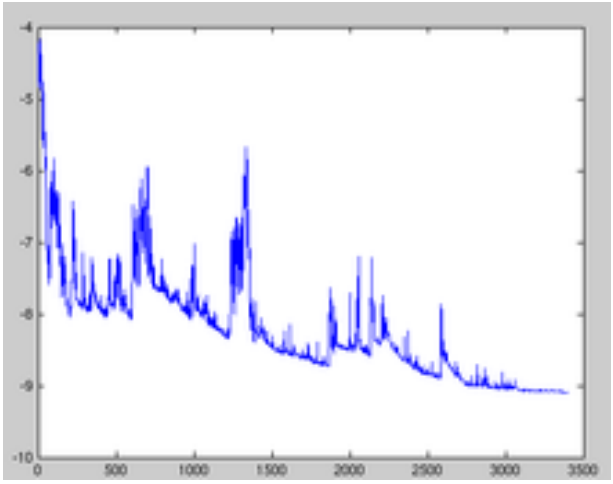

Figure 3.9: *Fluctuations in the descent of the gradient with the use of SGD.*

The pivot of this approach is to randomize the training dataset so as to mix up the order of the updating of the weights. This involves a high variance that produces intense oscillations in the descent of the gradient as can be seen in Fig. 3.9. In order to cope with oscillations a **momentum** is added.

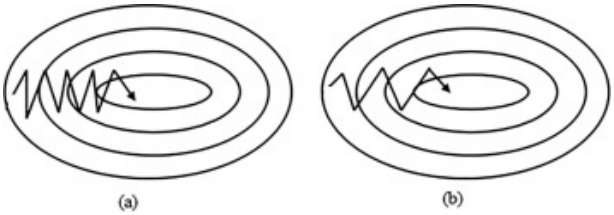

Figure 3.10: *Example of descent of the gradient considering only two weights. In figure (a) descent of the gradient without momentum. In figure (b) descent of the gradient considering the momentum*

Considering the example of Fig. 3.10 (a) is possible to observe that there is a fast variation along the vertical direction whereas the variation along the horizontal one is slow. This occurs when SGD is applied.
The desired effect is the opposite one, that is a slow variation along the vertical direction and a quick variation along the horizontal direction, as shown in Fig. 3.10 (b).

The introduction of the momentum smooths the direction in which the descent of the gradient occurs. In this way the *global minimum* is achieved in fewer epochs. It is mathematically expressed as:

$$\begin{aligned} v_t &= \rho v_{t-1} + \nabla L(W_{t-1}) \\ W_t &= W_{t-1} - \eta v_t \end{aligned} \tag{3.13}$$

Comparing (3.12) with the last one it is worth nothing that the learning rate (η) multiplies v_t, called the velocity, and not directly the gradient. The velocity allows to make the training quicker and to avoid the gradient's descent from being blocked in a local minimum or in a saddle point.
The parameters used to compute the velocity are:

- v_{t-1}: that is the retained gradient from previous t iterations
- ρ: is the "Coefficient of Momentum" that determines the percentage of the gradient retained every iteration. During the update of the weights the contribution of the accumulated gradients is prevalent, if ρ is much greater than η. Consequently, the gradient at step t changes the direction slowly. If it occurs the opposite condition, the accumulated gradient produces a kind of attenuation. Typically ρ has a value between 0.9 and 0.99. If it is set to zero, it returns to classic SGD algorithm.

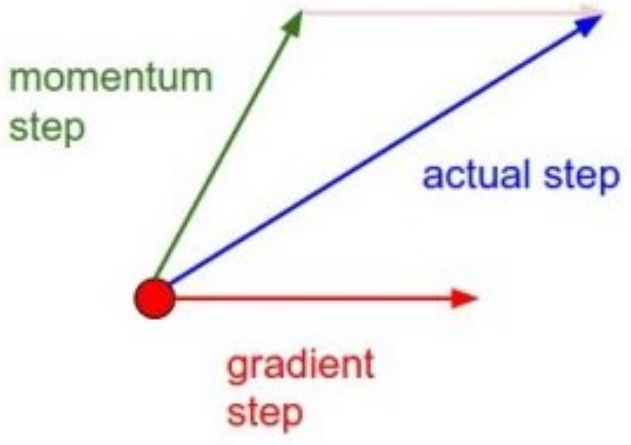

Figure 3.11: *Physical interpretation of the Momentum update rule.*

A physical explanation of (3.13) is shown in Fig. 3.11. The red point corresponds to the point that is achieved during the gradient's descent at the end of the step t. Firstly, it is estimated the velocity of the current step. It is indicated by the green vector *(Momentum step)* and it is obtained by the product between ρ and v_{t-1}. The gradient of the step t is got by computing $\nabla L(W_{t-1})$ and it is represented by the red vector *(Gradient step)*.
The velocity v_t is indicated by the blue vector *(Actual step)* and it is the result of the summation of the two previous vectors. From this it is obtained:

$$v_t = \rho v_{t-1} + \nabla L(W_{t-1}).$$

3.2.4 Activation function

The activation function allows obtaining an output value from an input one. It is a kind of gate that decides whether a neuron should be "fired" or not by computing a weighted sum and further adding bias with it. Activation function introduces *non-linearity* into the output allowing the neural networks to learn and perform more complex tasks.

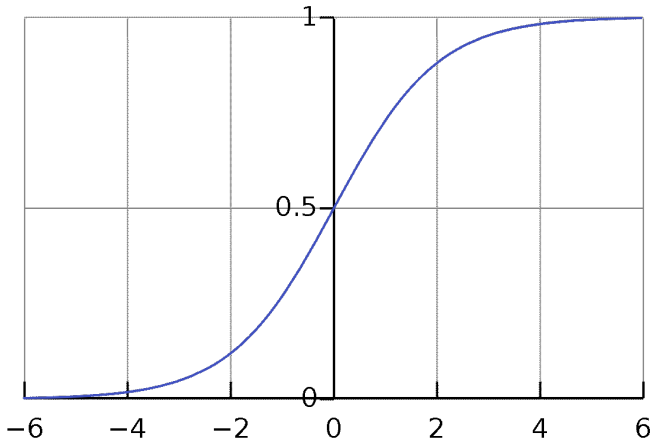

Figure 3.12: *Representation of the sigmoid function*

One of the most used activation function is the **sigmoid**, also called the logistic function. It transforms the input values into a value ranges from 0.0 to 1.0, as shown in Fig. 3.12.

If the input is a value much greater than 1.0 the corresponding output is equal to 1.0. On the other hand, if the input has a negative value, the matching output is equal to 0.0. Hence, it is mainly applied to models that have to predict the probability as an output.

The sigmoid is mathematically expressed as:

$$\sigma(x) = \frac{1}{1 + e^x} \tag{3.14}$$

During the early 1990s the sigmoid was the most used function in neural networks. However, it was discovered that it presents some problems during the training such as saturation. Indeed, the high values output to 1.0, while the low ones output to 0.0. Moreover, it is only strongly sensitive to change around its mid-point of its input.

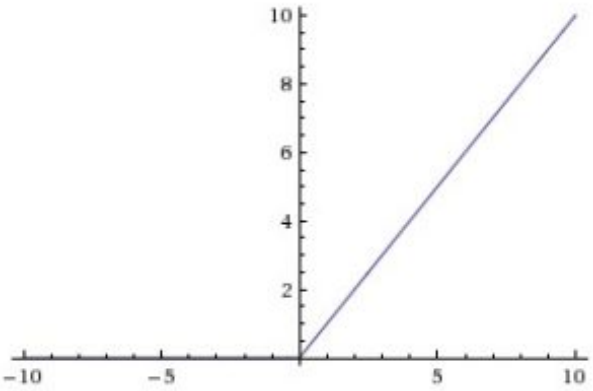

Figure 3.13: *Representation of the Rectified Linear Units (ReLU) function*

Nowadays the most commonly used activation function is the **Rectified Linear Units (ReLU)**. It is shown in Fig. 3.13.
For any positive value x the function returns the same input value, while for any negative input it returns 0. It can be mathematically expressed as:

$$ReLU(x) = max(0, x) \tag{3.15}$$

An advantage of this function is that it can solve the *vanish gradient problem*. This is achievable because ReLU's derivative is 1 for positive inputs, while is 0 for negative ones as reported in the following formula:

$$\frac{\partial ReLU(x)}{\partial x} = \begin{cases} 1 & x > 0 \\ 0 & otherwise \end{cases} \tag{3.16}$$

The introduction of ReLU has become necessary to apply SGD with backpropagation of errors during the training phase of deep neural networks. Therefore, during the backpropagation phase, if the local derivative is equal to 1, the gradient is allowed to move.

3.3 Densely Connected Convolutional Networks

In the last decades the use of CNNs has grown more and more to become the most used technique for visual object recognition. However, the improvements of computer hardware allowed implementing deeper CNNs. This has led to the development of a new research problem: the *vanish gradient problem*.
The gradient information moves through many layers because of deeper networks, and so it can vanish when gets to the end or the beginning of the architecture. For this reason, Huang et al. [35] proposed a model to overcome this issue called *Dense Convolutional Network* (DenseNet). They maintain the feed-forward nature of the network; each layer receives inputs from all preceding ones and passes its feature maps to all subsequent layers. The feature maps received from other layers are concatenated and not summated. All layers are directly connected with each other to ensure maximum information flow among them.

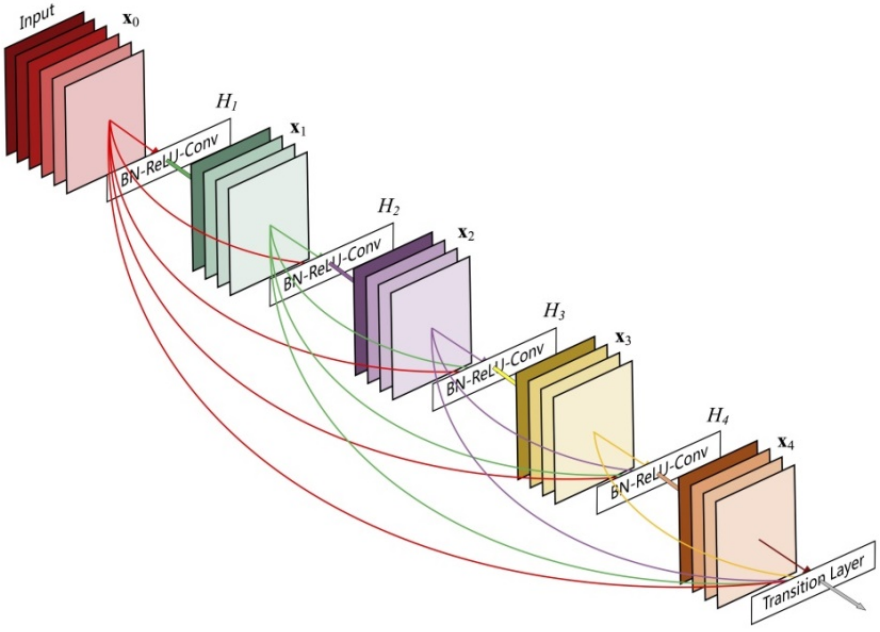

Figure 3.14: *A schematic layout of Dense Convolutional Network.*

A schematic example is reported in Fig. 3.14. DenseNet requires fewer parameters than traditional convolutional networks due to these dense connections, since there is no need to learn redundant feature maps. As a consequence, the

fewer parameters to learn make the training process easier and prevent the model from overfitting.

Dense Block

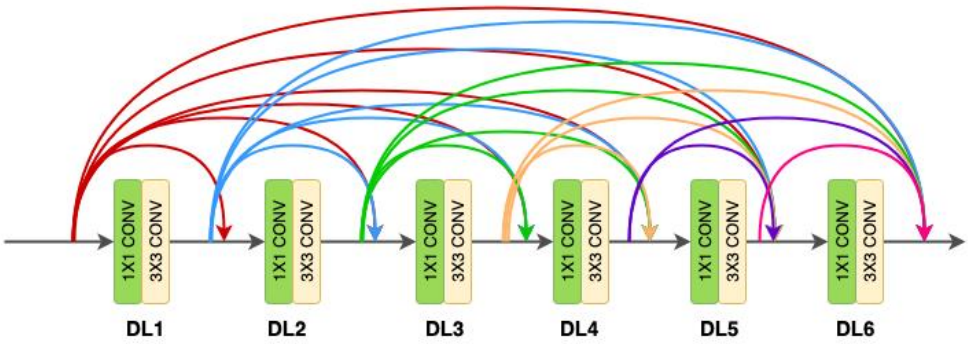

Figure 3.15: *A schematic layout of Dense Block. DL: dense layer. CONV: convolutional layer.*

The method proposed by [35] introduces the concept of *Dense Block*. It is made of N dense layers, each of which receives the feature maps from all previous dense layers. Its feature maps are passed to all successive layers. Concatenation applied in a dense block cannot occur if the size of the feature maps is not the same. An example of dense block is illustrated in Fig. 3.15.

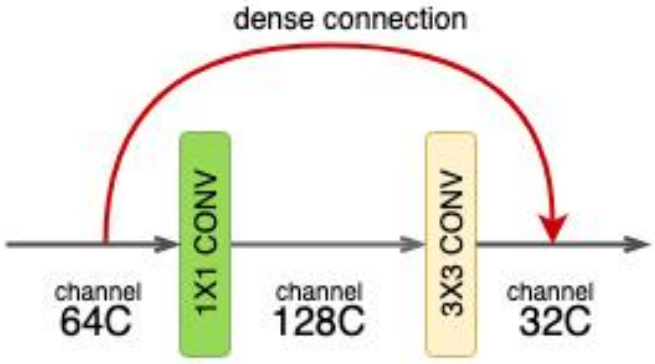

Figure 3.16: *A schematic example of Dense Layer. C: number of channels. CONV: convolutional block.*

Each *dense layer* is composed by 2 *convolutional blocks*. The former is a 1x1 convolution utilized for extracting features while the latter is a 3x3 convolution that decreases feature channels. Each convolutional block is made of three consecutive operations: *batch normalization (BN)*, *rectified linear unit (ReLU)* and either a 1x1 convolution or a 3x3 convolution. An example of single dense layer is shown in Fig. 3.16.

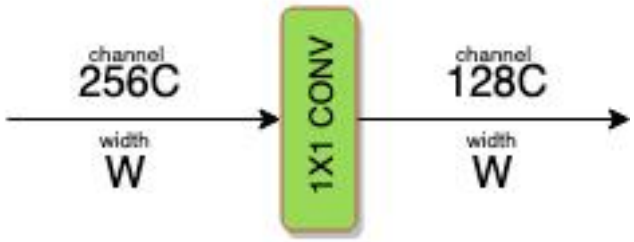

Figure 3.17: *A schematic example of Transition Layer. W: width dimension. C: number of channels. CONV: convolutional block.*

The number of channels output by a dense layer is called *growth rate (k)*. Each layer receives feature maps from all preceding layers of the dense block and thus has the "collective knowledge" of the network. For this reason, a small growth rate is sufficient to improve the efficiency. In addition, a *transition layer* is added between two dense blocks (Fig. 3.17). It consists of a 1x1 convolution to reduce the number of feature channels by half.

Chapter 4

Model architecture

This chapter is organized as follow: Section 4.1 describes the model utilised to estimate preterm infants' limb-pose and Section 4.2 presents the experimental protocol.

4.1 Architecture of the model

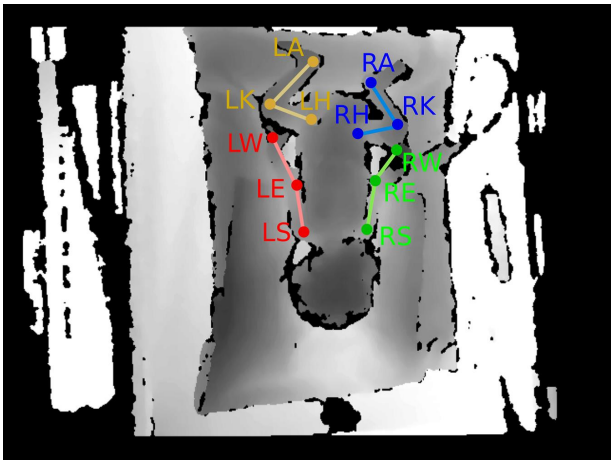

Figure 4.1: *Representation of the infant model. LS and RS: left and right shoulder, LE and RE: left and right elbow, LW and RW: left and right wrist, LH and RH: left and right hip, LK and RK: left and right knee, LA and RA: left and right ankle.*

In the current study each limb of the infants is modelled as a set of three connected joints: wrist, elbow and shoulder for arms, and ankle, knee and hip

for legs. This is shown in Fig. 4.1. The architecture implemented is inspired by the work of Moccia et al. [31]. It consists of two consecutive CNNs in order to estimate preterm infants' limb pose. The former detects joints and joint connections and the latter is applied for the regression of joints position. The joints belonging to the same limb are then connected using bipartile graph matching.

For the development of the FCNN detection, it is made multiple binary-detection operations to manage possible uncertainties due to the fact that multiple joints and joint connections could cover the same portion of the image, such as a limb self-occlusion. 20 separate ground-truth binary detection maps are produced for each video frame, the first twelve for the joints and the remaining eight for the joint connections. In the detection network the joint and the joint-connection branches gives as output joint and joint-connection confidence maps, respectively.

The joint mask consists of all pixels that lie inside the circle of a specific radius that is centred in the centre of the joint. In the same way, the ground truth for the joint connections are generated. The difference is that the region of interest is a rectangular of a given thickness and centrally aligned with the joint-connection line.

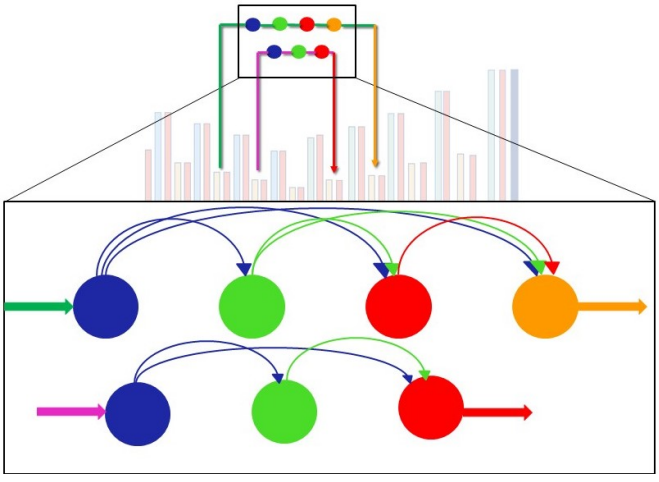

Figure 4.2: *Representation of the FCNN with Densely Connected skip connections.*

The architecture of the detection network is based on the classic encoder-decoder architecture of U-Net made by 8 blocks: 4 for the encoding path and 4 for the decoding path (Fig. 4.2).

Table 4.1: *Representation of the detection architecture to estimate the pose of preterm infants' limbs.*

Name	Kernel (Size / Stride)	Channels
Downsampling path		
Input	–	1
Convolutional layer - Common branch	3x3 / 1x1	64
Block 1 - Branch 1	2x2 / 2x2	64
	3x3 / 1x1	64
Block 1 - Branch 2	2x2 / 2x2	64
	3x3 / 1x1	64
Block 1 - Common branch	1x1 / 1x1	128
Block 2 - Branch 1	2x2 / 2x2	128
	3x3 / 1x1	128
Block 2 - Branch 2	2x2 / 2x2	128
	3x3 / 1x1	128
Block 2 - Common branch	1x1 / 1x1	256
Block 3 - Branch 1	2x2 / 2x2	256
	3x3 / 1x1	256
Block 3 - Branch 2	2x2 / 2x2	256
	3x3 / 1x1	256
Block 3 - Common branch	1x1 / 1x1	512
Block 4 - Branch 1	2x2 / 2x2	512
	3x3 / 1x1	512
Block 4 - Branch 2	2x2 / 2x2	512
	3x3 / 1x1	512
Block 4 - Common branch	1x1 / 1x1	1024
Upsampling path		
Block 5 - Branch 1	2x2 / 2x2	256
	3x3 / 1x1	256
Block 5 - Branch 2	2x2 / 2x2	256
	3x3 / 1x1	256
Block 5 - Common branch	1x1 / 1x1	512
Block 6 - Branch 1	2x2 / 2x2	128
	3x3 / 1x1	128
Block 6 - Branch 2	2x2 / 2x2	128
	3x3 / 1x1	128
Block 6 - Common branch	1x1 / 1x1	256
Block 7 - Branch 1	2x2 / 2x2	64
	3x3 / 1x1	64
Block 7 - Branch 2	2x2 / 2x2	64
	3x3 / 1x1	64
Block 7 - Common branch	1x1 / 1x1	128
Block 8 - Branch 1	2x2 / 2x2	32
	3x3 / 1x1	32
Block 8 - Branch 2	2x2 / 2x2	32
	3x3 / 1x1	32
Block 8 - Common branch	1x1 / 1x1	64
Output	1x1/1x1	20

Each block is divided in two branches, the former for joints and the latter for connections.The outputs of these branches are concatenated in a single one that enters in the next block (Table 4.1). It has been demonstrated that using a bi-branch structure provides higher detection performance because joint-probability and joint-connection affinity maps are separately processed.

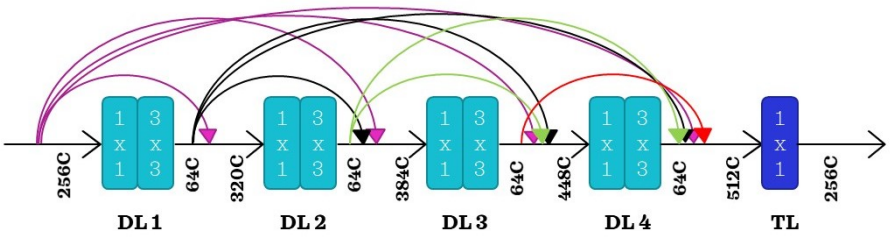

Figure 4.3: *Representation of the Dense Block applied on the Encoder 2. DL: dense layer. TL: transition layer. C: number of channels.*

Inspired by Huang et al. [35] two **dense block** are added in the detection network along the path of the second and the third encoder. The first one (Fig. 4.3) is constituted of 4 *dense layers* with a *growth rate* of 64. Each dense layer is composed by 2 convolutional blocks. The former is a 1x1 convolution utilized for extracting features while the latter is a 3x3 convolution that decreases feature channels. In addition, a *transition layer*, which consists of 1x1 convolution, is applied before the concatenation between the Encoder 2 and the Decoder 2.

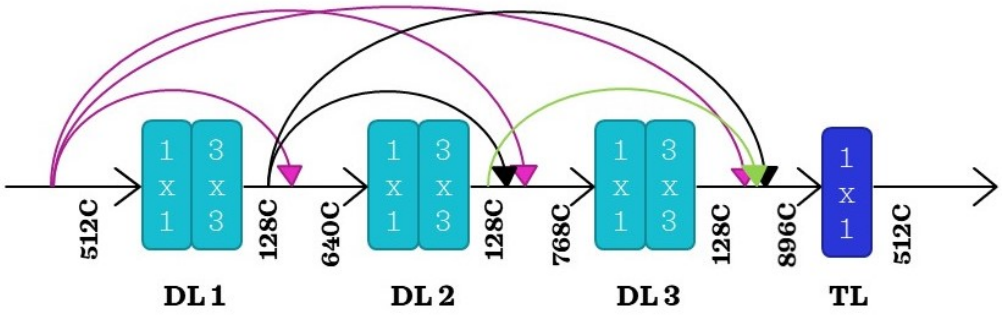

Figure 4.4: *Representation of the Dense Block applied on the Encoder 3. DL: dense layer. TL: transition layer. C: number of channels.*

The second dense block (Fig. 4.4) is constituted of 3 *dense layers* with a *growth rate* of 128. Each dense layer is composed as the previous ones. Also in this case, a *transition layer* is applied before the concatenation between the Encoder 3 and the Decoder 1.
Batch normalization and activation with the *ReLU* is performed after each convolution. The detection model is trained using the per-pixel *binary cross entropy* as loss function, and *SGD* as optimizer.

4.2 Experimental protocol

The following section presents the image dataset, the training settings and the metrics evaluated to test the model.

4.2.1 Dataset

The dataset used in the thesis is the BabyPose one, which was employed by Moccia et al. [31]. It consisted of 22 depth videos of 22 preterm infants that were acquired in the NICU of the G.Salesi Hospital in Ancona. The choice of acquiring depth frames over RGB frames was made to protect the patient's privacy. The infants who breathed spontaneously were identified by the clinicians. Video-acquisition setup, as shown in Fig. 1.1, was organized so as not to hamper movements of healthcare operators.

Every video recording was acquired using the Astra Mini S-Orbbec, with a frame rate of 30 frames per second and image size of 640x480 pixels. Joints annotation was done under the supervision of clinicians and for each video, 1000 frames were annotated. The annotation was manually obtained every 5 frames. This sampling can be assumed a good compromise considering the average number of movements of a preterm infant.

Table 4.2: *The division of dataset in training, validation and testing set.*

	Frames each video	Number of infants	Total samples
Training set	500	22	1100
Validation set	250	22	5500
Testing set	250	22	5500

These 1000 frames were divided into training and testing data in the following way: 750 frames for training the network and 250 frames to test it. In this way it was obtained a training set of 16500 samples (22 infants x 750 frames) and a testing set of 5500 samples (22 infants x 250 frames). From the training samples were kept 250 frames for each video as validation set, resulting in a total of 5500 frames (22 infants x 250 frames). The division of dataset is reported in Table 4.2.

Figure 4.5: *BabyPose Dataset Challenges.*

In Fig. 4.5 some challenges of the BabyPose dataset are shown: some of them are the presence of limb occlusions (both self-occlusion and due to healthcare operators), different number of visible joints in the camera field, variable distance between camera sensors and infants.

Training settings

Images were resized to 128x96 pixels in order to smooth noise and reduce both training time and memory requirements. It was selected a joint radius of 6 pixels to build the ground truth masks. For training the detection network it was applied an initial learning rate of 0.01 with a learning decay of 10% every 10 epochs, and a momentum of 0.98. It was used a batch size of 16 and a number of epochs equal to 100.

Performance metrics

It was calculated the *Dice similarity coefficient (DSC)* and *recall (Rec)* in order to measure the performance of our detection network.
They are defined as:

$$DSC = \frac{2\,X\,TP}{2\,X\,TP + FP + FN} \tag{4.1}$$

$$Rec = \frac{TP}{TP + FN} \tag{4.2}$$

Where TP represents true positive, FP represents false positive and FN indicated false negative. It was measured the segmentation time for the network.

Chapter 5

Results

The results obtained using the network architecture explained in the Section 4.1 are reported in the following chapter. The performance metrics of the Section 4.2.1 are calculated for joints and joint connections detection.

Table 5.1: *Joint-detection performance in terms of median Dice similarity coefficient (DSC) and recall (Rec). The metrics are reported separately for each joint. LS and RS: left and right shoulder, LE and RE: left and right elbow, LW and RW: left and right wrist, LH and RH: left and right hip, LK and RK: left and right knee, LA and RA: left and right ankle.*

	Right arm			Left arm			Right leg			Left leg		
	RW	RE	RS	LS	LE	LW	RA	RK	RH	LH	LK	LA
DSC	0.828	0.858	0.857	0.844	0.862	0.840	0.853	0.858	0.870	0.863	0.850	0.866
Rec	0.805	0.841	0.850	0.836	0.841	0.823	0.832	0.858	0.850	0.841	0.832	0.858

Table 5.2: *Joint-connection detection performance in terms of median Dice similarity coefficient (DSC) and recall (Rec). The metrics are reported separately for each joint connection. LS and RS: left and right shoulder, LE and RE: left and right elbow, LW and RW: left and right wrist, LH and RH: left and right hip, LK and RK: left and right knee, LA and RA: left and right ankle.*

	Right arm		Left arm		Right leg		Left leg	
	RW-RE	RE-RS	LS-LE	LE-LW	RA-RK	RK-RH	LH-LK	LK-LA
DSC	0.837	0.856	0.846	0.845	0.844	0.847	0.845	0.850
Rec	0.812	0.852	0.838	0.840	0.833	0.844	0.835	0.851

The values of median *DSC* and *Rec* for the 12 joints are shown in Table 5.1. A median *DSC* of 0.854 (IQR=0.126) was obtained, and a median *Rec* equal to 0.839 (IQR=0.134) was achieved among all joints. The highest median *DSC* was reached for the right hip joint and it was equal to 0.870 with an IQR of 0.114. The highest median *Rec* was reached for right knee and left ankle joint, and it was equal to 0.858 with an IQR of 0.115. The interquartile range (IQR) for *DSC* and *Rec* of joints was always lower than 0.160 and 0.167, respectively. The boxplots of *DSC* and *Rec* for joints are shown in Fig. 5.1 and Fig. 5.2, respectively.

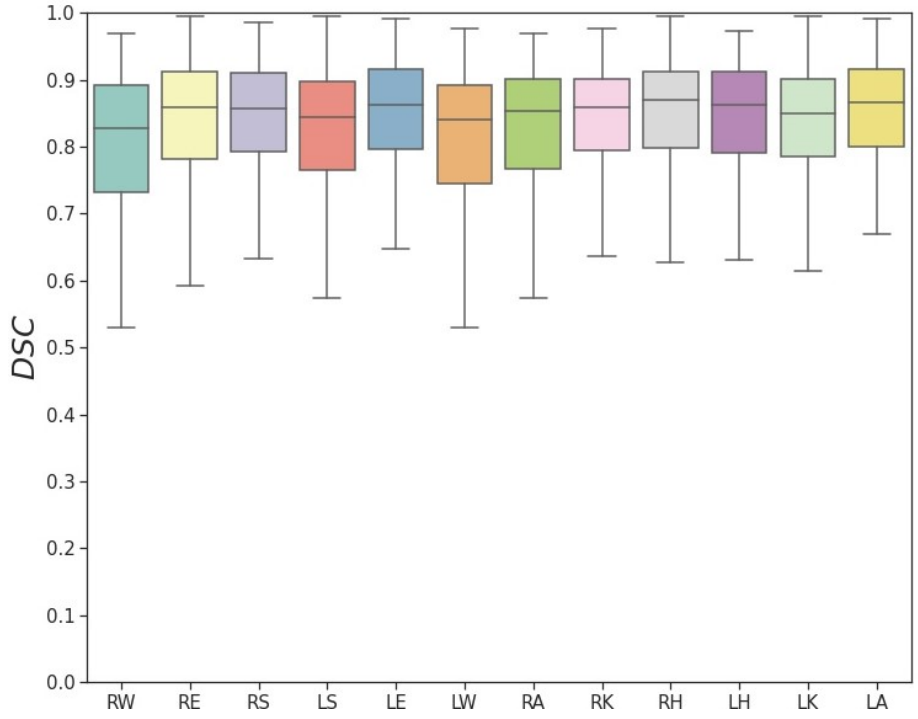

Figure 5.1: *Boxplots of the Dice similarity coefficient (DSC) for joint detection achieved with the proposed dense-net neural network.*

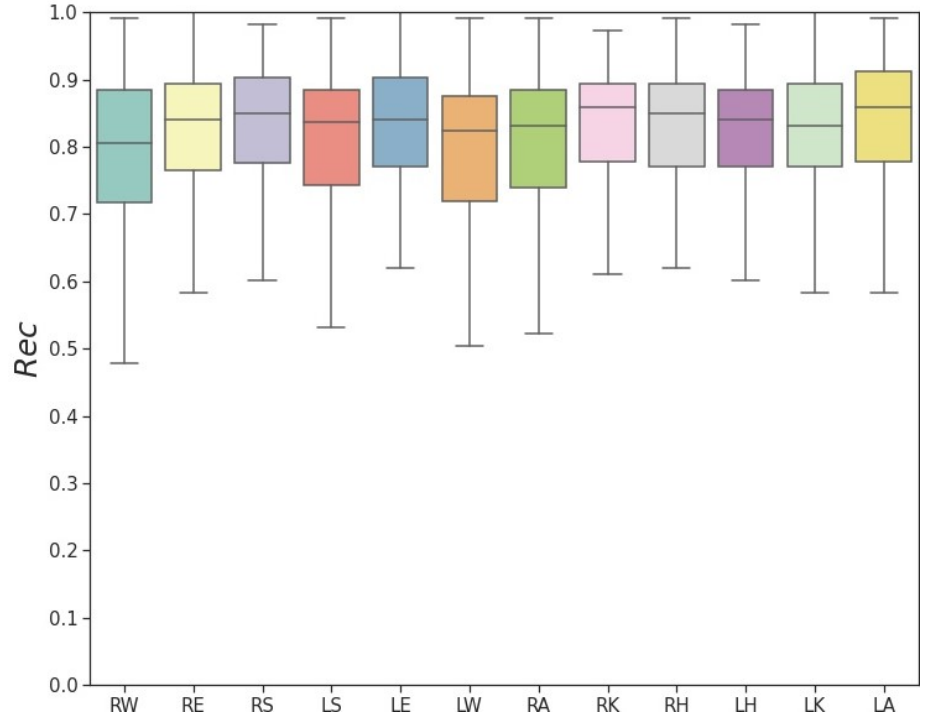

Figure 5.2: *Boxplots of the recall (Rec) for joint detection achieved with the proposed dense-net neural network.*

The values of median *DSC* and *Rec* for the 8 joint connections are shown in Table 5.2. A median *DSC* of 0.846 (IQR=0.105) was obtained, and a median *Rec* equal to 0.838 (IQR=0.137) was achieved among all joint connections. The highest median *DSC* was reached for the connection between the right elbow joint and the right shoulder joint. It was equal to 0.856 with an IQR of 0.089. The highest median *Rec* was reached for the connection between the right elbow joint and the right shoulder joint. It was equal to 0.852 with an IQR of 0.124. The interquartile range (IQR) for *DSC* and *Rec* of joint connections was always lower than 0.120 and 0.159, respectively. The boxplots of *DSC* and *Rec* for joint connections are shown in Fig. 5.3 and Fig. 5.4, respectively.

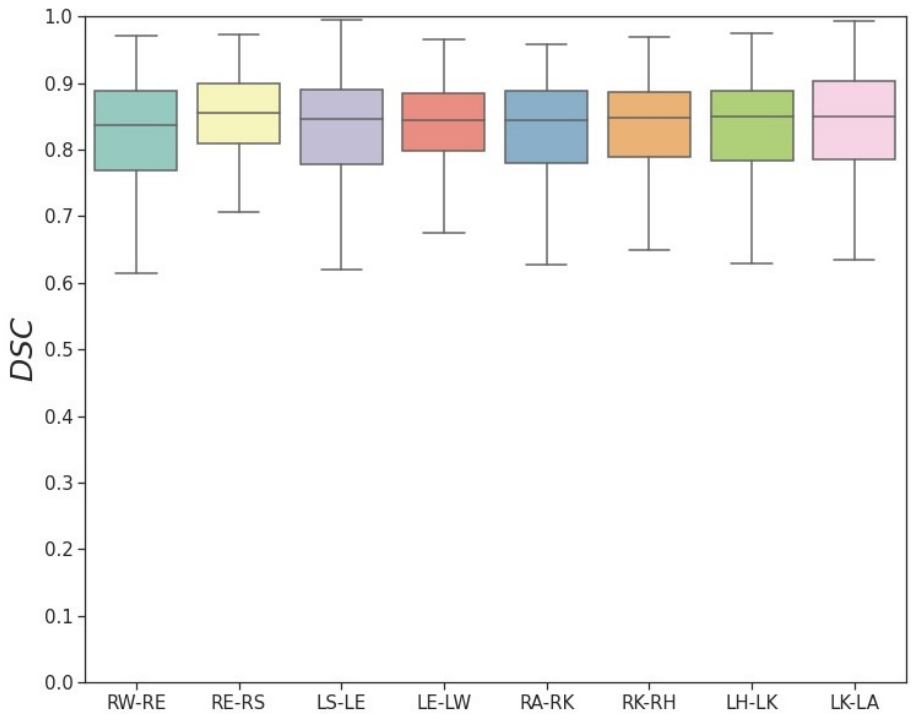

Figure 5.3: *Boxplots of the Dice similarity coefficient (DSC) for joint-connection detection achieved with the proposed dense-net neural network.*

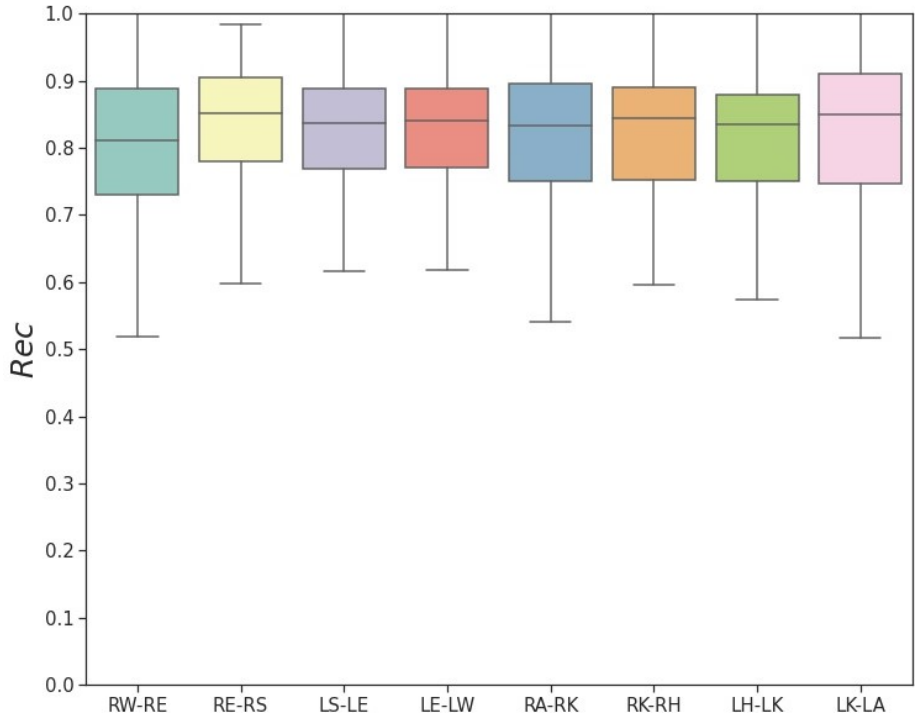

Figure 5.4: *Boxplots of the recall (Rec) for joint-connection detection achieved with the proposed dense-net neural network.*

It is worth observing that when some joints were occluded due to interaction of the healthcare-operator with the infant, as shown in Fig. 5.5, the model was able to correctly detect the visible joints. Detection time was on average 0.014 s per image.

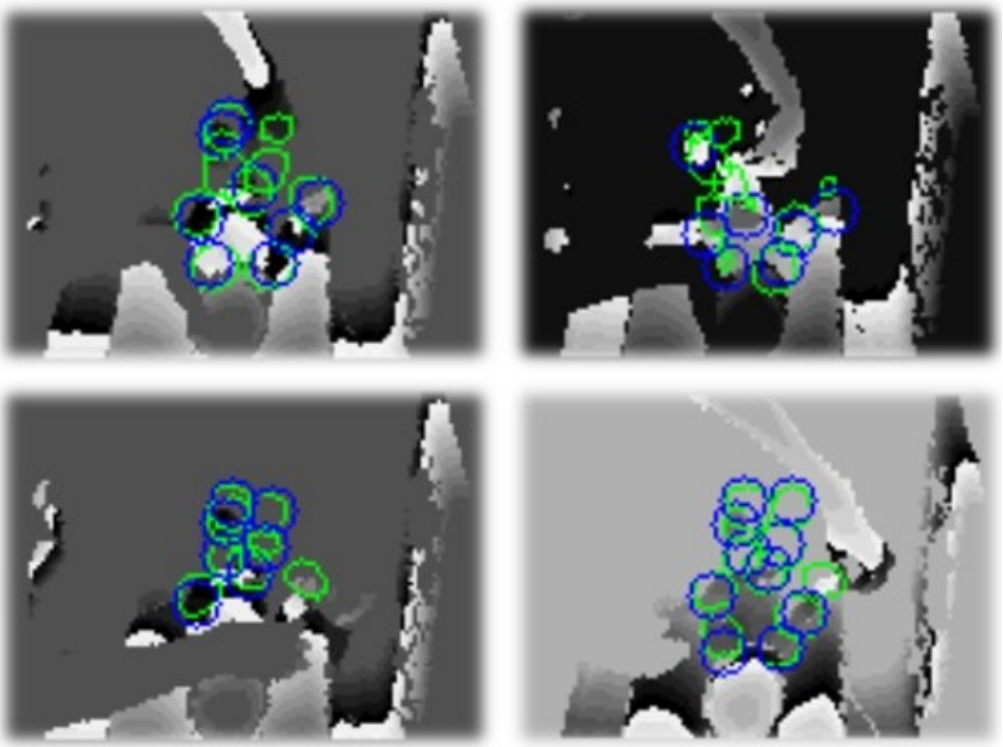

Figure 5.5: *Result of joints detection when other joints are occluded. Blue blobs represent ground-truth joint detection. Green blobs represent achieved joint detection.*

Chapter 6

Discussion and future work

The proposed model accomplished similar results for the detection of all joints as reported in Table 5.1, highlighting that it was able to process in parallel different joint probability maps [31]. Moreover, the detection network was able to correctly detect visible joints when the other ones were occluded, as shown in Fig. 5.5.

The addition of the two DB along the skip connections of the model allowed to reach great results in terms of joints detection and joint-connections detection performance. Indeed the proposed network achieved for joints a median *DSC* value equal to 0.854 with an IQR of 0.126, and a median *Rec* value equal to 0.839 with an IQR of 0.134. The same occurred in terms of joint connections detection performance, indeed the model reached a median *DSC* value equal to 0.846 with an IQR of 0.105, and a median *Rec* value equal to 0.838 with an IQR of 0.137.

A key result was reached when testing the model on the CPU of an ordinary laptop. Hence with an Intel-Core i3-6006U @ 2.00GHz it was possible to achieve an average detection time of 0.014 s per image. The proposed algorithm was consistent with real-time infants' monitoring and therefore, it could be used in the domestic environment to supervise newborns. Moreover, the integration of infant-specific measures acquired in the actual clinical practice, such as height and limbs length, may improve the limb-pose estimation. This way, it could be

possible to provide the best support for the diagnosis of cerebral dysfunctions of preterm infants.

The approach obtained great results in the prediction of limb-specific pose and it overcame the majority of the literature issues, such as computational cost and invasive monitoring of the infants. However, some improvements could be done to increase the performance of the model. Because of lack of available dataset in this field, an enhancement could be the increase of dataset when the healthcare operator interacts with the infant in the crib as it occurs joints' occlusion. Indeed the dataset dimension is limited, since the testing set has a narrow number of frames (5500).

Furthermore, the proposed method could be integrated in a clinical environment. In this setting, the higher computational efficiency could be exploited to implement more complex architectures based on detection and regression networks. Moreover, the addition of temporal information would allow to detect the movement of preterm infants with optimal results, as previously showed in et al. [36].

Chapter 7

Conclusion

In this thesis was proposed a deep learning method based on a 2D dense-net to estimate preterm infants' limb-pose. The addition of the two DB along the skip connections allowed the model to achieve great performance in the detection of preterm infants' limb-joints and joint-connections.

Currently, the utilisation of wearable sensors based-approaches have been lessening as they may cause stress and discomfort to the infant, and hindering the clinician. Indeed the application of camera sensors to monitor preterm infants' limbs has been significantly increasing. These sensors do not hamper the spontaneous movement of the infant and allow the healthcare operator to interact with the infant without any obstructions. This thesis provides further suggestions for replacing wearable sensors with camera sensors in a clinical environment.

It is worth noting that the testing of the model was performed using a mid-range laptop. A median *DSC* of 0.854 and a median *Rec* equal to 0.839 was achieved among all joints. The detection time obtained was on average 0.014 s per image. Thus the method proposed allows an estimate of preterm infants' pose in real-time and automatically, without the aid of a high-performance computer. Starting from this work, it may be suitable for being integrated in a domestic environment in order to provide assistance for early diagnosis of brain and cognitive dysfunctions such as cerebral palsy.

List of Figures

List of Tables

Bibliography

[1] E. F. P. Z. Annalisa Cenci, Daniele Liciotti and V. P. Carnielli, "Movements analysis of preterm infants by using depth sensor." ACM, 2017, p. 9.

[2] K. H. G. M. I. J. W. S. Villar J, Papageorghiou AT, "The preterm birth syndrome: a prototype phenotypic classification." *Am J Obstet Gynecol*, 2012.

[3] D. J. B. J. S. M. Wijnans L, de Bie S, "Safety of pandemic h1n1 vaccines in children and adolescents," *Vaccine*, 2011.

[4] D. G. G. M. Simmons LE, Rubens CE, "Preventing preterm birth and neonatal mortality: exploring the epidemiology, causes, and interventions," *Semin Perinatol*, 2010.

[5] D. S. Roberts D, "Antenatal corticosteroids for accelerating fetal lung maturation for women at risk of preterm birth." *Cochrane Library*, 2006.

[6] L. W. D. R. R. H. A. C. T. o. M. S. A. S. C. G. Caroline A Crowther, Janet E Hiller, *Effect of magnesium sulfate given for neuroprotection before preterm birth: a randomized controlled trial.*, 2003, vol. 20.

[7] I. Kr¨ageloh-Mann and C. Cans, *Cerebral palsy update.*, 2009, vol. 31.

[8] H. Prechtl, *Qualitative changes of spontaneous movements infetus and preterm infants are a marker of neurological dysfunction.*, 1990, vol. 23.

[9] ——, *State of the art of a new functional assessment of the young nervous system. A nearly predictor of cerebral palsy.*, 1997, vol. 50.

[10] C. Einspieler and H. Prechtl, *Prechtl's assessment of general movements: a diagnostic tool for the functional assessment of the young nervous system.*, 2005, vol. 11.

[11] P. H. B. A. F. F. Einspieler, C. and G. Cioni, *Prechtl's assessment of general movements: a diagnostic tool for the functional assessment of the young nervous system.*, 2004, vol. 167.

[12] C. Einspieler and P. Marschik, *Central pattern generators and their significance for the foetal motor function*, 2012, vol. 43.

[13] M. P. P. J. S. A. K. M. Y. H. Einspieler, C., "The general movement optimality score: a detailed assessment of general movements during preterm and term age." *Dev. Med. Child Neurol.*, 2015.

[14] v. K. B. A. E. C. S. E. Butcher, P.R. and A. Bos, *The quality of preterm infants' spontaneous movements: an early indicator of intelligence and behavior at school age.*, 2009, vol. 50.

[15] I. Bernhardt, M. Marbacher, R. Hilfiker, and L. Radlinger:, "Inter and intra-observer agreement of prechtl's method on the qualitative assessment of general movements in preterm, term and young infants." *Early Human Development*, 2011.

[16] F. F. G. C. A. F. B. Christa Einspieler, Heinz F R Prechtl, "The qualitative assessment of general movements in preterm, term and young infants - review of the methodology," *Early Human Development*, vol. 50, pp. 47–60, 1997.

[17] N. D. E. M. T. Claire Marcroft, Aftab Khan and T. Pl¨otz., "Movement recognition technology as a method of assessing spontaneous general movements in high risk infants," *Frontier in neurology*, vol. 5, p. 284, 2015.

[18] E. F. A. M. A. Cenci, D. Liciotti and P. Zingaretti, "Non-contact monitoring of preterm infants using rgb-d camera," *International Design Engineer-*

ing Technical Conferences and Computers and Information in Engineering Conference. American Society of Mechanical Engineers, vol. 09, 2015.

[19] C. R. S. D. M. P. C. V. S. M. L. S. Orlandi, K. Raghuram and T. Chau, "Detection of atypical and typical infant movements using computer-based video analysis," *IEEE*, p. 3598–3601, 2018.

[20] D. V. W. D. G. L. J. S. Trujillo-Priego, C. Lane and B. Smith, "Development of a wearable sensor algorithm to detect the quantity and kinematic characteristics of infant arm movement bouts produced across a full day in the natural environment," *Technologies*, vol. 5, p. 39, 2017.

[21] C. L. J. F. B. Smith, I. Trujillo-Priego and F. Horak, "Daily quantity of infant leg movement: wearable sensor algorithm and relationship to walking onset," *Sensors*, vol. 15, p. 19, 2015.

[22] M. Singh and D. J. Patterson, "Involuntary gesture recognition for predicting cerebral palsy in high-risk infants," in *International Symposium on Wearable Computers (ISWC) 2010*. IEEE, 2010, pp. 1–8.

[23] D. Gravem, M. Singh, C. Chen, J. Rich, J. Vaughan, K. Goldberg, F. Waffarn, P. Chou, D. Cooper, D. Reinkensmeyer *et al.*, "Assessment of infant movement with a compact wireless accelerometer system," *Journal of Medical Devices*, vol. 6, no. 2, p. 021013, 2012.

[24] M. Fan, D. Gravem, D. M. Cooper, and D. J. Patterson, "Augmenting gesture recognition with erlang-cox models to identify neurological disorders in premature babies," in *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*. ACM, 2012, pp. 411–420.

[25] D. Karch, K.-S. Kang, K. Wochner, H. Philippi, M. Hadders-Algra, J. Pietz, and H. Dickhaus, "Kinematic assessment of stereotypy in spontaneous movements in infants," *Gait & posture*, vol. 36, no. 2, pp. 307–311, 2012.

[26] D. Karch, K.-S. Kim, K. Wochner, J. Pietz, H. Dickhaus, and H. Philippi, "Quantification of the segmental kinematics of spontaneous infant movements," *Journal of biomechanics*, vol. 41, no. 13, pp. 2860–2867, 2008.

[27] L. Meinecke, N. Breitbach-Faller, C. Bartz, R. Damen, G. Rau, and C. Disselhorst-Klug, "Movement analysis in the early detection of newborns at risk for developing spasticity due to infantile cerebral palsy," *Human movement science*, vol. 25, no. 2, pp. 125–144, 2006.

[28] A. Stahl, C. Schellewald, Ø. Stavdahl, O. M. Aamo, L. Adde, and H. Kirkerod, "An optical flow-based method to predict infantile cerebral palsy," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 20, no. 4, pp. 605–614, 2012.

[29] S. S. Shivakumar, H. Loeb, D. K. Bogen, F. Shofer, P. Bryant, L. Prosser, and M. J. Johnson, "Stereo 3d tracking of infants in natural play conditions," in *2017 International Conference on Rehabilitation Robotics (ICORR)*. IEEE, 2017, pp. 841–846.

[30] S.-E. W. Z. Cao, T. Simon and Y. Sheikh, "Real time multi-person 2d pose estimation using part affinity fields," *Conference on Computer Vision and Pattern Recognition*, pp. 7291–7299, 2017.

[31] R. P. S. Moccia, L. Migliorelli and E. Frontoni, "Preterm infants' limb-pose estimation from depth images using convolutional neural networks," *IEEE*, 2019.

[32] F. Rosenblatt:, "The perceptron: A probabilistic model for information storage and organization in the brain." *Psychological review*, vol. vol. 65, p. p.386, 1958.

[33] G. Schwarz *et al.*, "Estimating the dimension of a model," *The annals of statistics*, vol. 6, no. 2, pp. 461–464, 1978.

[34] Y. LeCun, Y. Bengio *et al.*, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.

[35] L. v. d. M. K. W. G.Hung, Z.Liu, "Densely connected convolutional networks," *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[36] V. C. Sara Moccia, Lucia Migliorelli and E. Frontoni, "Preterm infants' pose estimation with spatio-temporal features," *IEEE*, 2019.