Neurocomputing 379 (2020) 64–73

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Bayesian capsule networks for 3D human pose estimation from single 2D images

Iván Ramírez, Alfredo Cuesta-Infante, Emanuele Schiavi, Juan José Pantrigo*

Universidad Rey Juan Carlos, C/ Tulipán s/n, Móstoles 28933, Spain

ARTICLE INFO

Article history:
Received 14 February 2019
Revised 5 September 2019
Accepted 25 September 2019
Available online 1 November 2019

Communicated by Dr. Liquan Shen

Keywords:
Deep bayesian networks
Capsule networks
3D Human pose estimation

ABSTRACT

Deep Bayesian Networks are a hot topic in Deep Learning because this approach makes it possible to minimize both the epistemic and the homoscedastic uncertainty at the same time self balancing multiple and complementary losses for a given task, simply by employing standard operations such as dropout, mean squared error or cross-entropy. On the other hand, Capsule networks are a novel DNN architecture that offer a richer representation because each concept is represented by a number of different vectors. The bayesian formulation of the Capsule networks is still an open problem that we address in this paper. We present a bayesian formulation of Capsule networks and compare its performance against the state-of-the-art for the ill-posed regression problem of estimating the 3D human pose from a single 2D image. The results show that our network is very competitive with a much more straightforward solution. To enable fair comparisons the source code is openly available at https://github.com/rollervan/BCN_3DPose/

1. Introduction

Deep neural networks (DNN) are possibly the Machine Learning (ML) technique with the fastest growing research effort in computer science, worldwide, ever. In less than a decade numerous large datasets have become readily available, also, computational power, open source libraries and a multitude of tutorials for and from the ML community of researchers, practitioners and academia. Many of the state-of-the-art DNN architectures are off-the-shelf, pretrained with huge datasets, lasting many hours and with powerful computers, so that with minimal changes and much less computational effort can be adapted to a specific task and carry out production quickly [1].

DNN as presented above are an archetype of a discriminative, black-box model: lacking in transparency and poor at representing uncertainty, with the additional shortcoming of being impossible to interpret. For instance, consider the simple task of single-label classification with a DNN. Given a new instance, the DNN will return a probability over all the possible classes, but we do not have a measure of how much uncertainty there is in every probability mass. In order to assess the quality of the estimators one can proceed in a frequentist way as in the *Bag of little Bootstraps* (BLB) [2]. But again, the quality of the estimator does not provide a reliable way to state the confidence in their estimations.

On the other hand, in the bayesian formulation of neural networks (NN), weights, biases, number of neurons and number of hidden layers are all considered sources of uncertainty, and therefore modeled by probability distributions that can be assumed a priori and updated with data a posteriori. Classification and regression tasks are cast as the problem of estimating the distribution over the network outputs given a dataset and a model. Additionally, it also makes it possible to estimate the distribution over the network outputs given the dataset but not necessarily conditioned on a particular model, and even over a set of models, given the data. Bayesian Neural Networks (BNN) were introduced early in [3] within a proposal to compute the posterior of the parameters algorithmically via the Laplace approximation, and more broadly in [4], with a quantitative and practical approach that tackles the architecture selection, the choice of weight decay terms and the uncertainty in network parameters and outputs. After some years of decreasing interest in NN, the arrival of DL has brought new results in BNN [5,6], and has sparked the Deep Bayesian Networks (DBN), [7–9]. Interestingly, dropout has recently been revisited as a bayesian approximation for representing and estimating the model uncertainty [10]; whereas stochastic gradient descent (SGD) has been investigated as an approximation of bayesian inference in [11].

Dealing with the interpretability of DNN models is much harder. Convolutional neural networks (CNN) attain excellent results in computer vision because images have strong local correlations which are exploited by the convolutional neurons. Within this scope it is possible to explain the contribution of each

* Corresponding author.
E-mail address: juanjose.pantrigo@urjc.es (J.J. Pantrigo).

https://doi.org/10.1016/j.neucom.2019.09.101
0925-2312/

convolutional neuron as a kernel that enhances some visual feature. But these good properties are not transferable to the general problem in which instances are described by an array of arbitrarily ordered attributes. A novel neural network architecture known as Capsule Networks (Capsnets), introduced in [12,13], transforms the feature space into a higher dimensional space in which each instance is represented by a set of vectors bounded to norm one. Each vector encodes the input, but having many of them also makes it possible to encode the variability of the abstract *concept* or general *entity*. This variability may include pose, illumination, texture, etc. depending on the data set. Besides, the norm of every vector is interpreted as the probability of such a vector representing a given *concept*. The intuition is that having one of these vectors close to norm one may be enough to classify the instance because it is very distinctive, so the representation obtained with Capsnets hierarchizes the features in some way.

Doing bayesian inference on Capsnets is still an open issue. Given the qualities of both lines, the hypothesis is that Bayesian Capsnets generalize better while allowing the uncertainty to be modelled.

Besides, this novel architecture, while promising, has been only tested in simple classification tasks based on toy datasets such as MNIST [14]. Thus, we test a Bayesian Capsnet architecture in a more challenging field: 3D human pose estimation from 2D images. The 3D human pose estimation problem can trace its origins back to the early works of [15] and [16], where the goal was to "understand" the motion of an object (person) through a sequence of images. Such a problem requires possibly the integration of different techniques such as time analysis, image processing, objects tracking, human-based modelling priors, multi-modalities image acquisition, etc. Scene understanding, activity recognition or bio-metrical estimation are a few examples of motivation interests for predicting 3D human poses from 2D images. Other approaches restrict the input, commonly a 2D RGB image, to a binary silhouette image [17,18], or a 2D coordinates matrix [19,20] where the goal is to infer the third depth coordinate. Recently, with the rise of deep learning and machine learning development, a great effort have been carried out to abstract and learn a meaningful representation of scenes and human poses, that, in the particular problem of 3D coordinates prediction can be cast in mainly two approaches as described in [21]: 1) In a direct way (in this paper understood as 'end-to-end'), in-which 3D coordinates are inferred directly from 2D RGB images using machine learning systems or, 2) through modular systems that integrates separable and independent parts that are ultimately combined to predict the coordinates. The former, as in [22,23] for 3D pose prediction and [24,25] for 2D, mainly makes use of CNNs and MLP neural networks. On the contrary, the latter, although shown in literature as a win strategy in terms of metric results [26–29], lacks of a well-defined latent and generalizable representation. For this reason, we explore the ability of the recent proposed Capsule Network architecture [12] to predict 3D human pose coordinates in a straightforward global architecture.

Thus, in this paper we present the following contributions: 1) We introduce a fully bayesian formulation of Capsule Networks. 2) We show its potential by solving an ill-posed problem such as the human 3D pose estimation from a single 2D image. To this end, we propose a NN architecture. 3) We attain the results with less uncertainty and less error, outranking the best solution under the same conditions so far.

The rest of the paper is organized as follows. In Section 2 we present the basic theoretical background on BNN and Capsnets. In Section 3 we introduce the bayesian formulation of Capsule Networks. In Section 4 we present the network architecture for the case study of human 3D pose estimation. Experiments, results and comparisons with the state-of-the-art are shown in Section 5. Finally we discuss the results and give conclusions in Section 6.

2. Preliminaries

Before presenting our contributions and results, we introduce the notation used throughout the paper as well as recapping basic background on BNN and Capsule Networks.

Notations

Throughout this paper we mark with a *hat* those variables that are estimations obtained by means of a NN, e.g. $\hat{y}$, with a *star* those which are the solution of an optimization problem, e.g. w^*, and with an accent the samples from a probability distribution, e.g. $\tilde{x} \sim p(x)$. The 2-norm of a vector x is expressed as $\|x\|_2$, whereas $\|X\|_F$ represents the Frobenius norm of a matrix X. Finally, $\mathbb{I}_n$ is the identity matrix of size $n \times n$

Let $\mathbf{X}$ be a data set of N instances $\{x^{(i)}\}$, and let $\mathbf{Y}$ be a set of the corresponding labels, or a ground truth, $\{y^{(i)}\}$, for $i = 1 \dots N$. We will also assume that the pairs $(x^{(i)}, y^{(i)})$ are i.i.d. Let f: $\mathbf{X} \rightarrow \mathbf{Y}$ be the function implemented by a given NN, and let $\hat{y} = f(x; \mathbf{W})$ be the output of such a NN for a given input x and a given configuration of the weights and biases of the NN, jointly represented by $\mathbf{W}$ and referred to simply as the *weights*.

2.1. Bayesian neural networks

The bayesian approach is two-fold. On one hand it is the maximum a posteriori estimation of the NN weights. On the other hand, it is the bayesian inference of the target output according to a prediction distribution. Below, we next present the basic background of both together with the issues raised and the state-of-the-art solutions.

2.1.1. Maximum a posteriori estimation of the weights

One bayesian approach to NN consists of setting $p(\mathbf{W})$, a *prior* probability distribution over the weights of the NN, and then look for $\mathbf{W}^*$, the maximum a posteriori (MAP) weights given the data $\mathbf{X}$ and $\mathbf{Y}$. According to the Bayes theorem and taking into account that the input is independent of the weights, the MAP problem is formalized as

$$\mathbf{W}^* = \arg\max_{\mathbf{W}} \{p(\mathbf{W}|\mathbf{X}, \mathbf{Y})\} = \arg\max_{\mathbf{W}} \{p(\mathbf{W})\, p(\mathbf{Y}|\mathbf{X}, \mathbf{W})\}, \tag{1}$$

where $p(\mathbf{Y}|\mathbf{X}, \mathbf{W})$ is the *likelihood* and $p(\mathbf{W}|\mathbf{X}, \mathbf{Y})$ is the *posterior*. The proposal in [30] is to assume that both the likelihood and the prior are Normal distributions. Specifically, the prior over the weights is a multivariate normal centred in 0 with covariance matrix $\Sigma = \sigma_w \mathbb{I}$. For the sake of clarity, we use σ_w.

$$p(\mathbf{W}) = \frac{1}{\sqrt{2\pi}\sigma_w} \exp\left(-\frac{\|\mathbf{W}\|_2^2}{2\sigma_w^2}\right). \tag{2}$$

The likelihood of a pair $(x^{(i)}, y^{(i)})$ is centered on the output of the NN $f(x^{(i)}; \mathbf{W})$ with standard deviation σ,

$$p(y^{(i)}|x^{(i)}, \mathbf{W}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\|y^{(i)} - f(x^{(i)}; \mathbf{W})\|_2^2}{2\sigma^2}\right). \tag{3}$$

The standard deviations introduced in the last two expressions are associated to different types of uncertainty in bayesian modeling. According to [30], σ_w is the Epistemic uncertainty of the model, and σ is the Aleatoric uncertainty associated to the difficulty on the particular task. If the aleatoric uncertainty is the same for all the inputs, it is said to be homoscedastic. On the contrary, if it depends on the input data it is known as heteroscedastic. In this paper we assume homoscedasticity.

Taking into account that $p(\mathbf{Y}|\mathbf{X},\mathbf{W}) = \prod_{i=1}^{N} p(y^{(i)}|x^{(i)},\mathbf{W})$, introducing (2) and (3) into (1), and taking logarithms we have

$$\{\mathbf{W}^*, \sigma^*, \sigma_w^*\} = \arg\min_{\mathbf{W},\sigma,\sigma_w} \{\mathcal{L}_{\text{MSE}} + \mathcal{L}_{\text{H}} + \mathcal{L}_{\text{T}} + \mathcal{L}_{\text{E}}\}, \tag{4}$$

$$\text{where} \quad \mathcal{L}_{\text{MSE}} = \frac{1}{2\sigma^2}\frac{1}{N}\sum_{i=1}^{N}\left(\|y^{(i)} - f(x^{(i)};\ \mathbf{W})\|_2^2\right), \tag{5}$$

$$\mathcal{L}_{\text{H}} = \log\sigma, \tag{6}$$

$$\mathcal{L}_{\text{T}} = \frac{1}{2\sigma_w^2}\|\mathbf{W}\|_2^2, \tag{7}$$

$$\mathcal{L}_{\text{E}} = \log\sigma_w. \tag{8}$$

In the functional above, (5) is the Mean Squared Error (MSE) or L_2 loss function and (7) is the Tikhonov regularization over the weights which in NN is usually introduced in the minimization step referred to as *weight decay*. Additionally, the normality assumptions have brought the homoscedastic term (6) and the epistemic term (8). Otherwise these two terms can be renamed into a constant λ, recovering the loss function $\mathcal{L} = \text{MSE} + \lambda\|\mathbf{W}\|_2^2$ for non bayesian NN.

Notice that $\lim_{\sigma\to 0}\log\sigma = \lim_{\sigma_w\to 0}\log\sigma_w = -\infty$; so both (6) and (8) may dominate the minimization (4). Pushing σ towards 0 means attaining 100% accuracy, but σ_w towards 0 carries $\|\mathbf{W}\|_2^2$ to 0 and viceversa. We deal with this issue in the *Practical BNN* section, below.

2.1.2. Bayesian inference

Another bayesian approach to NN is through inference, predicting the target y for a new given input x according to the prediction distribution

$$p(y|x,\mathbf{X},\mathbf{Y}) = \int p(y|x,\mathbf{W})\,p(\mathbf{W}|\mathbf{X},\mathbf{Y})\,d\mathbf{W}. \tag{9}$$

Since integrating over all possible weight configurations is computationally intractable, the posterior is approximated by a family of probability distributions $q_\theta(\mathbf{W})$. The parameter θ is found looking for the one that minimizes the KL divergence with the actual posterior distribution. This is known as the variational approach, which leads to maximizing the *Evidence Lower Bound* (ELBO) which, after manipulation, can be expressed in the two terms below.

$$\begin{aligned}\theta^* &= \arg\min_\theta \left\{\text{KL}\big(q_\theta(\mathbf{W})\big)\|\,p(\mathbf{W}|\mathbf{X},\mathbf{Y})\right\} = \arg\max_\theta \{\text{ELBO}\} \\ &= \arg\min_\theta \left\{-\mathbb{E}_{q_\theta}\big(\log p(\mathbf{Y}|\mathbf{X},\mathbf{W})\big) + \text{KL}\big(q_\theta(\mathbf{W})\|\,p(\mathbf{W})\big)\right\}.\end{aligned} \tag{10}$$

This is tractable via Monte Carlo integration. Let $\{\widetilde{\mathbf{W}}_t\}_{t=1\ldots T} \sim q_\theta(\mathbf{W})$ be a set of T samples drawn from the distribution q_θ. Then,

$$-\mathbb{E}_{q_\theta}\big(\log p(\mathbf{Y}|\mathbf{X},\mathbf{W})\big) \approx -\frac{1}{T}\sum_{t=1}^{T}\log p\big(\mathbf{Y}|\mathbf{X},\widetilde{\mathbf{W}}_t\big), \tag{11}$$

$$\text{KL}\big(q_\theta(\mathbf{W})\|\,p(\mathbf{W})\big) \approx \frac{1}{T}\sum_{t=1}^{T}\log\widetilde{\mathbf{W}}_t - \frac{1}{T}\sum_{t=1}^{T}\log p(\widetilde{\mathbf{W}}_t). \tag{12}$$

Once θ^* is estimated, $\mathbb{E}_{q_\theta^*}\big(p(y|x,\mathbf{W})\big)$ is also approximated via Monte Carlo.

2.1.3. Practical BNN

The choice of the prior is a key of the Bayesian approach. We have shown that the assumption of normality leads to terms in the loss function that may make training difficult, whereas choosing another prior should also take into account how easy it is to sample from. The solution to the first issue, proposed in [10], is to remove (7) and (8) from the functional in (4) and use dropout instead. Dropout was firstly introduced in [31] as a way of preventing overfit in DNN. Later in [30], it was explained as if every weight in a NN was multiplied by a binary random variable z with probability π, i.e. $z \sim \text{Ber}(\pi)$. Hence, doing dropout in every layer is a practical way of getting samples $\widetilde{\mathbf{W}}_t \sim p(\mathbf{W})$, not necessarily normally distributed. By doing so, and defining $s = \log\sigma^2$, (4) is rewritten as.

$$\{\mathbf{W}^*, s^*\} = \arg\min_{\mathbf{W},s}\left\{e^{-s}\mathcal{L}_{\text{MSE}} + s\right\}. \tag{13}$$

Moreover, if we consider $\mathcal{L}_{\text{MSE}}$ as the loss of a given task, we can extend (13) for multiple tasks carried out by the same NN as follows:

$$\{\mathbf{W}^*, s_1^* \ldots s_\tau^*\} = \arg\min_{\mathbf{W},s_1\ldots s_\tau}\left\{e^{-s_1}\mathcal{L}_1 + s_1 + \cdots e^{-s_\tau}\mathcal{L}_\tau + s_\tau\right\}. \tag{14}$$

In summary, dropout in every layer is an easy way towards BNN in practice and provides two extra advantages: it makes it possible to minimize the uncertainty in the model by using (13); and it makes it possible to self-balance multiple losses in the same NN because $s_1, \ldots, s_\tau$ are optimized at the same time in (14). Notice that (14) requires the loss functions to be MSE. Kendall et al. show in [32] that it is also possible to use softmax as the activation of the last layer and then cross-entropy for the loss. Thus, BNN can be applied both in regression and classification tasks.

2.2. Capsule networks

The so-called Capsule Network architecture has been released recently [12], initially with the purpose of image recognition, introducing some variations with respect to the Convolutional Neural Networks (CNN). For the sake of completeness, we summarize here the most relevant facts of them as presented in [12], although using a simplified notation. The Capsule network architecture can be divided in 5 steps: Input, Encapsulation, Inverse graphics, Routing and Output, depicted in Fig. 1.

The feed-forward step during training is detailed in Algorithm 1. Notice that the coefficients $c_{k,j}$ are updated in this step, whereas the matrices $A_{k,j}$ are updated in the backpropagation.

3. Bayesian capsule networks

In this paper we introduce a Bayesian formulation of Capsnets. To this end, we propose introducing prior knowledge as restrictions over the matrices $A_{k,j}$ related to their geometric interpretation. In [12] authors justify these matrices as the inverse of computer graphics, in which a 3D scene is shown in a 2D screen, thus reducing the dimensionality.

Here, on the contrary, every capsule $\mathbf{u}_j \in \mathbb{R}^S$ is mapped K times into a higher dimensional space by means of linear transformations $A_{k,j} \in \mathbb{R}^{S'\times S}$, to obtain $\mathbf{v}_{k,j} = A_{k,j}\cdot\mathbf{u}_j$, with $\mathbf{v}_{k,j} \in \mathbb{R}^{S'}$ and $S' > S$. Following on with the intuition behind Inverse graphics, every $\mathbf{v}_{k,j}$ is an estimation in a higher dimensional space of its projection $\mathbf{u}_j$. Hence, it is reasonable to expect to recover the projection by means of an inverse transformation $B_{k,j} \in \mathbb{R}^{S\times S'}$, such that

$$\hat{\mathbf{u}}_j = B_{k,j}\cdot\mathbf{v}_{k,j}, \quad \text{subject to} \quad B_{k,j}A_{k,j} = \mathbb{I}_S.$$

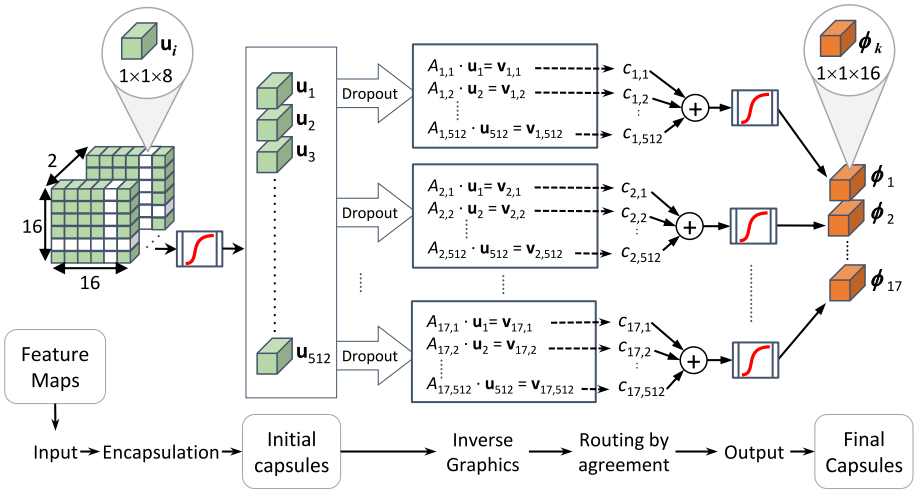


Fig. 1. CapsNet module. Capsules are created reshaping the feature maps incoming from a previous NN. Then each one is mapped K times into a higher dimensional space. The results are combined and squashed into the outgoing capsules. The block with the red line is the squash function. In this Figure $M = N = D = 16$, $J = 512$, $S = 8$, $S' = 16$ and $K = 17$. Dropout is proposed in this paper as part of the Bayesian formulation.

Algorithm 1 Capsule network feed-forward pass in training.

Hyperparameters: J, K and S' are positive integers arbitrarily chosen.

Preconditions: The reminder of $M \cdot N \cdot D$ divided by J must be 0.

Postconditions: $0 < \|\phi_k\|_2 \leq 1$ for $k = 1 \dots K$.

Def: $\texttt{Reshape}_{J\times S} : \mathbb{R}^{M\times N\times D} \to \mathbb{R}^{J\times S}$, with $J = MND/S$.

Def: $\texttt{squash}(\mathbf{u}) = \left(\|\mathbf{u}\|_2^2 \cdot \mathbf{u}\right)/\left(1 + \|\mathbf{u}\|_2^2 \cdot \|\mathbf{u}\|_2\right)$

Steps:

1.*Input:* $\psi \in \mathbb{R}^{M\times N\times D}$.

2.*Encapsulation:* $\mathbf{U} = \{\mathbf{u}_j\}_{j=1\dots J} = \texttt{Squash}\left(\texttt{Reshape}_{J\times S}(\psi)\right)$,
~ with $\mathbf{u}_j \in \mathbb{R}^S$ and $\mathbf{U} \in \mathbb{R}^{J\times S}$.

3.*Inverse graphics:* $\mathbf{v}_{k,j} = A_{k,j} \cdot \mathbf{u}_j$, ~~ with $\mathbf{v}_{k,j} \in \mathbb{R}^{S'}$,
~ $A_{k,j} \in \mathbb{R}^{S'\times S}$,and $\mathbf{V}_k = \{\mathbf{v}_{k,j}\}_{k=1\dots K} \in \mathbb{R}^{K\times S'}$.

4.*Routing:* Compute coefficients $c_{k,j}$ with the
~ *Routing by agreement* algorithm, as in [12],
~ s.t. $\sum_{k=1}^{K} c_{k,j} = 1$; $c_{k,j} \in \mathbb{R}$.

5.*Output:* $\phi_k = \texttt{Squash}\left(\sum_{j=1}^{J} c_{k,j}\mathbf{v}_{k,j}\right)$

Return: $\Phi = \{\phi_k\}_{k=1\dots K}$, where $\phi_k \in \mathbb{R}^{S'}$ and $\Phi \in \mathbb{R}^{K\times S'}$

Clearly, the first candidate for $B_{k,j}$ is $A_{k,j}^{(-1)}$, the pseudo-inverse of $A_{k,j}$. However, this pseudo-inverse has a closed form when $A_{k,j}$ has full rank, which is a very weak constraint.

Instead we propose learning the matrix $B_{k,j}$ at the same time as $A_{k,j}$. This is equivalent to relaxing the constraint on the invertivility of the retro-projection so $\hat{\mathbf{u}}_j = B_{k,j}A_{k,j}\mathbf{u}_j \approx \mathbf{u}_j$.

Lemma. *Let A and B two matrices such that B is left-compatible with A. If $\|BA - \mathbb{I}\|_F^2 \approx 0$ then B is a consistent left pseudo-inverse matrix of A in the sense of the Frobenius norm.*

Proof. Consider the vector $\mathbf{u}$, its projection $\mathbf{v} = A\mathbf{u}$ due to the linear transformation A and its estimated retro-projection $\hat{\mathbf{u}} = B\mathbf{v}$ due to the linear transformation B. Assuming that $\|BA - \mathbb{I}\|_F^2 \approx 0$, we have that

$$0 \leq \|\hat{\mathbf{u}} - \mathbf{u}\|_F^2 = \|BA\mathbf{u} - \mathbf{u}\|_F^2 \leq \|(BA - \mathbb{I})\|_F^2 \cdot \|\mathbf{u}\|_F^2 \approx 0.$$

Let $\hat{\mathbf{v}} = A\hat{\mathbf{u}}$ be the projection of the estimated retro-projection $\hat{\mathbf{u}}$. Hence, $0 \leq \|\hat{\mathbf{v}} - \mathbf{v}\|_F^2 = \|A\hat{\mathbf{u}} - A\mathbf{u}\|_F^2 \leq \|A\|_F^2 \cdot \|\hat{\mathbf{u}} - \mathbf{u}\|_F^2$. Therefore, if $\|\hat{\mathbf{u}} - \mathbf{u}\|_F^2 \approx 0$, then $\|\hat{\mathbf{v}} - \mathbf{v}\|_F^2 \approx 0$ too. □

Thus we are allowing a certain amount of uncertainty in the only step of the Capsnet that is learned during back-propagation. Moreover, we can control it by assuming a prior jointly over $A_{k,j}$ and $B_{k,j}$,

$$p(A_{k,j}, B_{k,j}) = \frac{1}{\sqrt{2\pi}\sigma_b} \exp\left(-\frac{\|B_{k,j}A_{k,j} - \mathbb{I}_S\|_F^2}{2\sigma_b^2}\right), \tag{15}$$

where σ_b represents the uncertainty in the elements of matrix B.

We then can incorporate this prior to (1), assuming the same likelihood as in (3), and assuming that matrices $A_{k,j}$, $B_{k,j}$ are independent from parameters $\mathbf{W}$, to obtain the following optimization problem, similar to (13),

$$\{\mathbf{W}^*, s^*, s_b^*\} = \arg\min_{\mathbf{W},s,s_b} \left\{e^{-s}\mathcal{L}_{\text{MSE}} + s + e^{-s_b}\mathcal{L}_{\text{b}} + s_b\right\}, \tag{16}$$

$$\text{where} \quad \mathcal{L}_{\text{b}} = \frac{1}{JK}\sum_{j=1}^{J}\sum_{k=1}^{K} \|B_{k,j}A_{k,j} - \mathbb{I}_S\|_F^2 \tag{17}$$

and $s_b = \log\sigma_b$. In other words, there are two new terms due to the uncertainty in the elements of matrix B. Remember that, in order to be bayesian, the above expression imposes doing dropout in all the layers of the full NN. With regards to a Capsnet module within such a NN, this is only possible after the Encapsulation and before the Inverse Graphics. Thus we propose a dropout that randomly sets to zero some capsule components before going into the Inverse graphics step. □

4. Capsnet architecture for the human 3.6 challenge

In order to try the bayesian formulation on a real problem and show its performance, in this paper we attempt to infer the 3D human pose from a single 2D RGB image; where the pose is given by the 3D coordinates of 17 joints in a human skeleton.

Since there are infinite different projections from a 3D space into 2D, the 3D pose reconstruction using 2D data is an ill-posed optimization problem which needs to be regularized. This can be done in a variational framework as in [33]. Another approach is to cast it as a regression problem to be solved with a DNN that estimates every joint. Dealing in this way with such an ill-posed problem requires prior information. For instance in [21] a biological probabilistic model is proposed. The other option is to do Deep Regression Bayesian Networks. To the best of our knowledge the

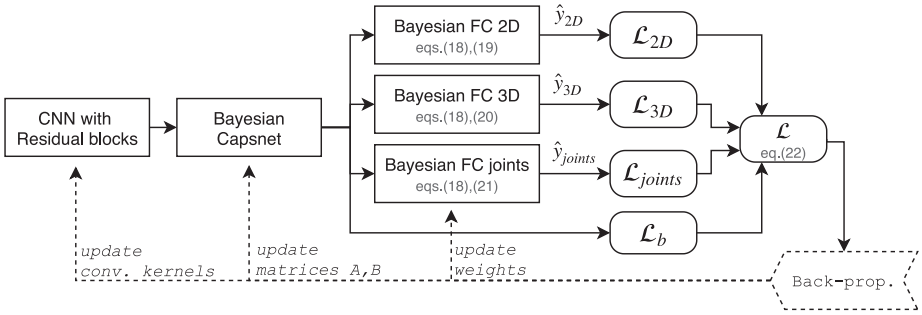


Fig. 2. Global structure of the proposed CapsNet-based Bayesian Neural Network for the Human3.6M challenge. The input image is first analyzed through a CNN with Residual blocks. The feature maps are sent into a Bayesian CapsNet as defined in this paper. The CapsNet output provides encoded vectorial features that are decoded in three different versions of the same concept: 2D and 3D coordinates together with Joint heat maps. The loss is a self-balanced combination of the loss from each task plus the one due to the prior on the Capsnet.

work that best fits in this line is [34]. These works cited are commented later in Section 5.

In this paper we solve the regression problem in a model-free NN, end-to-end, architecture that incorporates Bayesian Capsnets as defined above. The whole system attempts to minimize a multi-task objective to improve the accuracy of the main target (3D coordinates) avoiding the use of other techniques such as transfer-learning, using other datasets or taking into account the previous images if the image belongs to a video-sequence. The data set used is known as *Human 3.6*, described in detail in Section 5.

The architecture proposed is depicted in Fig. 2 and consists of 3 modules, each being a NN of a different kind. The input image is firstly processed by a CNN with residual blocks. Then a Bayesian Capsnet transforms the feature maps into capsules and finally three different Bayesian Fully Connected (FC) NN produce the estimation of the 3D pose together with a reconstruction of the 2D pose and heat maps of each joint. Each one is described in detail below.

4.1. CNN With residual blocks

The input image is first processed by a CNN consisting of 4 convolutional layers with residual units, kernels of size 9×9, *stride*$= 2$ and *padding* "valid", followed by a ReLU activation function. The combination of stride and padding produces a reduction of the input size. Besides, the layers produce 32, 64, 128 and 16 feature maps respectively. Thus, with this configuration the output tensor of this module has size $16 \times 16 \times 16$.

The residual units were proposed in [35] for allowing features to skip convolutional layers, thus acting as an identity function, which is known to be hard to "mimic" by a convolutional network.

In this module Dropout is not used. The reason is that convolutional neurons are equivalent to fully connected neurons with many of their inputs disconnected, thus having dropout implicitly implemented.

Finally, we stress that the activation of the last layer in [12] is the squash function, not ReLU, as in ours. Thus, we force the CNN to produce representations in the positive $(16 \times 16 \times 16)$ - dimension *cuadrant*.

4.2. Bayesian capsnet

Our Bayesian Capsnet follows the five steps described in Section 3.

Input. It is a tensor of size $M \times N \times D = 16 \times 16 \times 16$. As in [12], it is modified with the squash function so its norm is upper bounded to 1.

Encapsulation. According to Algorithm 1, we have to choose J and S, the number of capsules that we initialize and their size such that the precondition is satisfied. Given the input tensor, our choice is $J = 512$ capsules of size $S = 8$, satisfying that $512 \times 8 = 16 \times 16 \times 16$. Each capsule is cloned $K = 17$ times. This choice is arbitrary so we set 17 because there are 17 joints to predict. Thus we will have one final capsule for every joint. In this step, we introduce dropout by setting to zero a 30% of the $512 \times 8 \times 7$ components that we have, considering all the capsules and all the clones. The result is grouped in blocks as depicted in Fig. 1.

Inverse graphics. The procedure is the one described in Algorithm 1. Specifically, there are 17 blocks and 512×17 matrices $A_{k,j}$, each one producing a respective capsule $\mathbf{v}_{k,j}$ of size $S' = 16$.

Routing by agreement. Capsules outgoing from each block are averaged according to coefficients $c_{k,j}$. These coefficients are computed using the *Routing by agreement* algorithm, described in detail in [12], where authors defined it as a version of the Expectation-Maximization procedure. This algorithm runs twice in the feed-forward step of training and never in the back-propagation step.

Output. The capsules resulting from the Routing are finally squashed. Therefore this Capsnet module produces 17 capsules of 16 components, each capsule with norm upper bounded to one.

4.3. Estimation and reconstruction

In our architecture, capsules encode joints of the skeleton sketch. Our goal is to estimate the coordinates of each joint in 3D, their projections in 2D and a heat map of each joint at the same time. The hypothesis is that these three tasks are complementary, so they help each other.

To this end we use three Fully Connected (FC) Neural Networks, one for each output, $\hat{y}_{3D}$, $\hat{y}_{2D}$ and $\hat{y}_{joints}$. Let us define the following layers:

$\texttt{DenseReLU}_{[x]}$ a Dense layer of x neurons with ReLU activation,
$\texttt{DenseSigm}_{[x]}$ a Dense layer of x neurons with Sigmoid activation,
$\texttt{DenseTanh}_{[x]}$ a Dense layer of x neurons with Tanh activation,
$\texttt{Drop}_{[x]}$ a x% dropout layer,

then the following expressions describe each FC:

$$z_1 = z_2 = z_3 = \texttt{Drop}_{[15]}\Big(\texttt{DenseReLU}_{[2048]}\Big(\texttt{DenseReLU}_{[1024]}(\Phi)\Big)\Big), \tag{18}$$

$$\hat{y}_{2D} = \texttt{DenseSigm}_{[34=17\times 2]}(z_1), \tag{19}$$

$$\hat{y}_{3D} = \texttt{DenseTanh}_{[51=17\times 3]}(z_2), \tag{20}$$

$$\hat{y}_{joint} = \texttt{Reshape}_{256\times 256\times 16}\Big(\texttt{DenseReLU}_{[699632]}(z_3)\Big). \tag{21}$$

Table 1
State-of-the-Art works that have used the Human3.6M dataset with a summary of their qualities (E2E and R.I.) features (P. and IM.) and aids used (T., T.L., M. and O.D.).

Works	E2E	R.I.	P.	IM.	T.	T.L.	M.	O.D.
Ours	Y	Y	Y*	Y				
Tome [21]				Y		Y	Y	Y
Tekin [37]				Y		Y	Y	Y
Zhou [38]				Y	Y		Y	Y
Katircioglu [39]			Y*	Y		Y	Y	Y
Bogo [40]			Y*	Y			Y	Y
Sanzari [41]			Y*	Y			Y	Y
Rogez [26]				Y				Y
Liu [28]				Y	Y			Y
Wang [29]				Y	Y			
Tian [27]				Y			Y	

Y: yes, **Y***: yes in Section 5.4; **E2E.** End-to-end approach, **R.I.** Rotation invariant, **P.** Procrustes transformation, **IM.** Image as input, **T.** Temporal information, **T.L.** Transfer Learning, **M.** Biometric Model, and **O.D.** Other datasets.

Notice that they do not share layers but the structure is identical and the header is different. The use of the sigmoid and the tanh activation function in the headers of the 2D and 3D reconstruction respectively is doubly motivated. On the one hand, the range of these outcomes coincides with that of their activations; that is, $\hat{y}_{2D} \in (0,1)$ and $\hat{y}_{3D} \in (-1,1)$. On the other hand, they are only used in the header to minimize the possible vanishing gradient effect. Once the error is back-propagated through this layer, it will find ReLU activations, which are more appropriate for preventing this problem.

The loss function for this problem is the expansion of (16) to our set of tasks $\mathcal{T} = \{2D, 3D, joints\}$ so

$$\mathcal{L} = e^{-s_b}\mathcal{L}_b + s_b + \sum_{\tau\in\mathcal{T}} \left(e^{-s_\tau}\mathcal{L}_\tau + s_\tau\right). \tag{22}$$

5. Experiments and discussion

In this section we first describe the *Human3.6M* data set [36] together with works related to the 3D human pose estimation from a single 2D image. Then we compare our proposal with the works that attained the best performances in the CVPR'17, in IJCV, Jan.'18 and the state-of-the-art in 2019 so far.

5.1. Human 3.6 data set and related works

The 3D human pose estimation problem has attracted a lot of interest during the last years both in the computer vision and machine learning communities. It has been tackled with a wide variety of solutions, many of them so intrinsically different that are not comparable with others. For example, the variational approach in [33] uses multiple views for reconstructing the volume rather than the position of the joints; while the Deep Regression Bayesian Network proposed in [34] for reconstructing 2D images (inpainting, block occlusion and face restoration).

Table 1 collects the state-of-the-art works that aim at estimating the 3D coordinates of every joint in the skeleton sketch from 2D images taken from a single view, summarizing the main features of their solutions. The two columns on the left are qualities that our work exhibit, namely being an end-to-end approach (column E2E) and that the pose is invariant to rotations with respect to the vertical axis (column R.I.) The next two columns are features of the method used. Specifically, some works use the Procrustes transformations (column P.) and all of them take an image as the input data to the whole system (column IM.). The four right-most columns refer to aids that are used by the methods listed to improve their quantitative results.

By end-to-end we mean that the 3D estimation does not have to depend on anything other than the 2D image. In that sense the other works use a *pipeline* that firstly focusses on performing very well in an estimation of the 2D joints, an then estimates the third dimension. Additionally, these works use other data sets, not only Human3.6M. (column O.D.) Another non end-to-end approach, that also uses more data sets, consists of learning a richer set of features that, afterwards, by Transfer Learning (column T.L.), are used for training their solutions. Other aids are using previous frames to improve the prediction in the current time (column T) or having a biometric model (column M). Finally, notice that the comparisons cannot mingle methods that use the Procrustes transformations with methods that do not.

The Human 3.6 data set consists of 15 videos that add up to a total of 3.6 million RGB images. Each video shows a person doing a different activity. The videos are sub-sampled at 10 fps, resulting in 311,724 images for training and 110,040 for testing. The person is bounding-boxed in every frame and, together with the images, the data set provides a ground truth given by the 3D coordinates of each joint in the skeleton, scaled to the interval [0,1], as an array of size 17×3 elements. Additionally, from this information we generate *Joint heat maps*, i.e. images with 17 channels, each of them representing a binary mask of the joint locations as depicted in Fig. 3. We also crop every frame by the bounding box and re-scale to 300×300 to obtain the input image.

5.2. Training

We trained our NN following the *Protocol* #1 as described in [36]: subjects S1, S5, S6, S7, S8 are used for training, while S9 and S11 are kept for test. The reported metrics are the averaged errors of the Euclidean Distances of the 17 joints (we do not use others joints to help the training step). We used `AdamOptimizer` starting with a learning rate of 10^{-5} and batch-size of 1, and increasing it by 10 until 20 when the loss reaches a plateau. The experiments were run in an Intel i7-7700, 3.6 GHz CPU with 16 GB of memory and NVIDIA GTX1080TI. The algorithms are based on the Tensorflow 1.13 library, which was the stable version to date.

5.3. Comparison with state-of-the-art at CVPR'17

We first compare our NN with the top-3 less averaged error on the same dataset and with the same protocol as reported in Tome et al.[21] at CVPR'17, namely: Zhou et al.[38], Tekin et al.[37] and Tome et al.[21], who reports two results due to different versions in the same paper. We have removed Sanzari et al [41] from this top-3 because they use the Procrustes transformation. For this reason they are also referenced in the comparison of [39], and we compare against them in the Section 5.4; so in the end they are also included. The best result so far was due to the second proposal of Tome et al; in which the authors use a 6-stage deep architecture to improve belief-maps in each stage, projecting in 2D a proposed 3D pose from a Probabilistic pre-trained 3D Model.

To assess the contribution of the bayesian approach both in the Capsnet and in the FC headers, we try three different versions of our solution. The first version (Ours-I) has dropout only in the Capsnet, but there is no regularization over matrices $A_{k,j}$ nor dropout in the FC headers. The second version (Ours-II) incorporates the regularization over matrices $A_{k,j}$. Finally, the third version (Ours-III) is the full solution as described in Section 4.

The results of the comparison are detailed per activity in the rows of Table 2. Each cell in the table presents the error accumulated by the 17 joints averaged over the whole activity

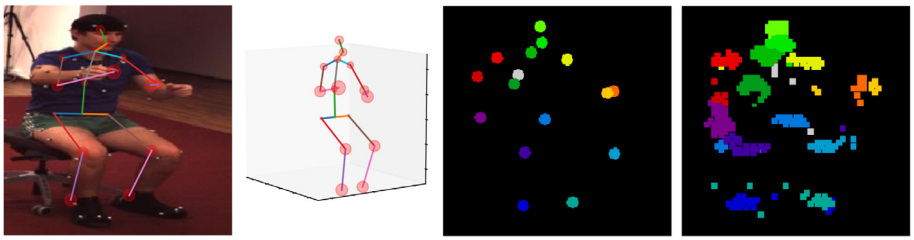


Fig. 3. (a) An input RGB image from the validation set and its 2D predicted coordinates. (b) 3D predicted joints together with the inner epistemic uncertainty (the circle magnitude). (c) Ground truth for the joint heat maps. (d) Joint heat maps estimated.Each joint has a different color.

Table 2
Comparison of the error(mm.) with three versions of the Bayesian Capsule Networks with respect to the top-3 in CVPR'17 Human 3.6 challenge, due to Zhou [38], Tekin [37] and Tome [21]. The background bars are sized with respect to the max. and min. of every row.

Activity	Zhou	Tekin	Tome, I	Ours, I	Tome, II	Ours, II	Ours, III
Directions	87.36	85.03	68.55	79.42	**64.98**	73.15	73.33
Discussion	109.31	108.79	78.27	83.73	**73.47**	84.95	83.45
Eating	87.05	84.38	77.22	84.01	**76.82**	85.87	85.33
Greeting	103.16	98.94	89.05	83.15	86.43	80.12	**79.08**
Phoning	116.18	119.39	91.63	86.42	**86.28**	91.44	89.99
Photo	143.32	**95.65**	110.05	112.38	110.67	109.42	109.95
Posing	106.88	98.49	74.92	81.34	**68.93**	76.4	76.08
Purchases	99.78	93.77	83.71	77.65	74.79	76.72	**73.61**
Sitting	124.52	**73.76**	115.94	105.10	110.19	105.54	104.12
SittingDown	199.23	170.4	185.72	135.55	173.91	**130.15**	136.27
Smoking	107.42	85.08	88.25	88.25	**84.95**	88.07	87.59
Waiting	118.09	116.91	88.73	79.24	85.78	80.25	79.19
WalkDog	114.23	113.72	92.37	87.45	**86.26**	88.75	87.13
Walking	79.39	**62.08**	76.48	67.56	71.36	66.1	66.31
WalkTogether	97.7	94.83	77.95	80.45	**73.14**	76.84	76.88
Avg. by activity	112.91	100.08	93.26	88.78	88.53	87.58	**87.22**
Std. Dev.	27.78	24.21	27.63	16.28	26.21	**15.86**	17.15

Table 3
Comparison of the error(mm.) with three versions of the Bayesian Capsule Networks with respect to the top-3 in IJCV, Jan.'18, due to Sanzari [41], Bogo [40] and Katircioglu [39].The background bars are sized with respect to the max. and min. of every row.

Activity	Sanzari	Bogo	Ours, IV	Katircioglu
Directions	48.82	62	57.55	**43.89**
Discussion	56.31	60.2	61.32	**48.54**
Eating	95.98	67.8	66.48	**46.57**
Greeting	84.78	76.5	64.49	**49.95**
Phoning	96.47	92.1	68	**53.94**
Photo	105.58	77	83.16	**59.29**
Posing	66.3	73	56.05	**43.77**
Purchases	107.41	75.3	54.85	**43.94**
Sitting	116.89	100.3	77.65	**60.2**
SittingDown	129.63	137.3	97.32	**73.64**
Smoking	97.84	83.4	67.31	**51.15**
Waiting	65.94	77.3	59.63	**46.3**
WalkDog	130.46	79.7	64.76	**52.25**
Walking	92.58	86.8	49.96	**39.81**
WalkTogether	102.21	81.7	60.47	**47.18**
Avg. by Activity	93.15	82.03	65.93	**50.69**
Std. Dev.	23.97	17.90	11.74	**8.23**

video-sequence. The last two rows are the average by activity and its standard deviation respectively. Results show that:

1. The best method in [21] is almost matched by Ours-I, and overtaken by Ours-II and Ours-III. We show 8 frames of one test video-sequence together with the 3D estimation and the 3D ground truth in Fig. 5, from different views.
2. The homoscedastic uncertainty σ is much lower with any version of our proposed Bayesian Capsnet than with the methods reported in [21]. This is an expected result, since the loss function includes the homoscedastic uncertainty as an objective. The minimum is attained by Ours-II, i.e. when FC headers have no dropout. Moreover, the improvement due to Ours-III in the average is not as much as the improvement due to Ours-II, suggesting that it could account for an excess of regularization.
3. The epistemic uncertainty σ_w is indirectly estimated as the variance of predictions due to different samples of the Neural Network. In the bayesian formulation, sampling a neural network is approximated as dropout during the evaluation of an instance in all those layers where it is possible. To this end we take the image shown in Fig. 3(a) and produce 50 predictions, each one with a sample from Ours-III. The standard deviation of the 2D and 3D predictions is shown as the magnitude of balls surrounding each joint in Figs. 3(a) and 3(b) respectively. We do not report quantitative results because it would require the process to be repeated for all the images in all the activities.
4. Compared with the rest of solutions in Table 2, our proposal is much more straightforward. Specifically, the proposal in [21] uses other data sets in the stages that precede the estimation of the 3D coordinates, which accounts for a complex system structure.

5.4. Comparison with state-of-the-art at IJCV, January'18

A best performance in 3D human pose estimation from a single image has been recently reported in [39]. However, a key difference with respect to [21] is the use of Procrustes transformation. As we have already mentioned, it is not fair to compare performances with and without this transformation because it induces a different way to measure the errors. For the sake of completeness, we incorporate this transformation to the fully bayesian architecture, referred to as Ours-IV in Table 3, and compare with Sanzari et al. [41], Bogo et al. [40] and Katircioglu et al. [39].

Our proposal ranks second with a significant improvement both in the accuracy and in the homoscedastic uncertainty. However, we remark that [39] is also aided by Transfer learning, Biometric model and the use of other data sets. Specifically, their approach consists of the use of a pre-trained overcomplete 3D pose autoencoder, using the latent space created by each 3D sample as target for a CNN. The final 3D output is given by the pre-trained decoder, which receives an estimation of the latent space from the CNN feeded with a single image. They propose a *ShallowNet-Autoencoder* CNN, which is comparable to our CNN with residual connections, and a *ResNet50-Autoencoder* pre-trained for 2D joint heat-map predictions. With the ShallowNet, they achieve 127.07mm error while the ResNet50 boosts the results to their best (without Procrustes), 67.27 mm; which shows that the improvement is probably due to the deep ResNet50 architecture.

Table 4
Comparison of the error(mm.) with the Bayesian Capsule Networks (Ours-III) with respect to the state-of-the-art in 2019 Human3.6M challenge, due to Wang [29], Rogez [26], Tian [27], Veges [19] and Liu [28]. The background bars are sized with respect to the max. and min. of every row.

Activity	Ours, III	Wang	Rogez	Tian	Liu
Directions	73.33	50.03	55.9	**40.3**	50.72
Discussion	83.45	59.96	60	**47.1**	60.04
Eating	85.33	54.66	64.5	57.1	**51.11**
Greeting	79.08	56.55	56.3	**72**	63.65
Phoning	89.99	65.65	67.4	72.1	**59.7**
Photo	109.95	79.63	71.8	**56.5**	69.34
Posing	76.08	52.74	55.1	53	**48.83**
Purchases	73.61	54.81	55.3	54.4	**51.98**
Sitting	104.12	85.85	84.8	80.3	**72.76**
SittingDown	136.27	117.98	**90.7**	116.8	105.31
Smoking	87.59	62.48	67.9	63.3	**58.62**
Waiting	79.19	59.55	57.5	**57.2**	60.98
WalkDog	87.13	65.21	63.3	**48.1**	62.25
Walking	66.31	**41.48**	47.8	**no report**	45.88
WalkTogether	76.88	**48.52**	54.6	**no report**	48.69
Avg. by Activity	87.22	63.67	63.50	62.90	**61.10**
Std. Dev.	17.15	18.21	11.31	18.88	14.14

5.5. Comparison with state-of-the art 2019

Human pose estimation is a field of intense research. The most recent works incorporate to the battery of solutions the following approaches. The architecture proposed in [26] consists of three main components: a generator that suggests candidate poses at different locations in the image, a classifier that scores the different pose proposals; and a regressor that refines them. Besides, this work reports protocols #1 and #3 both with and without the Procrustes transformation. In [28] a novel long short-term dependence-aware (LSTD) module is proposed, which is embedded inside the CNN architecture to boost the intermediate convolutional feature maps for 3D pose estimation. Also in the line of capturing long-range temporal coherency among different human body parts, [29] proposes a model endowed with self-supervised correction mechanism, which involves two dual learning tasks: 2D-to-3D pose transformation and 3D-to-2D pose projection. Their goal is to generate geometrically consistent 3D pose predictions forcing the 2D projections of the generated 3D poses to be identical to the estimated 2D poses. [27] presents a proposal that combines convolutional networks with residual modules and multiscale analysis. They decompose the 3D pose estimation task into two sequential steps: firstly a heatmap of the 2D keypoints (the joints in the simplified skeleton), and secondly a generative method for estimating the 3D pose. Finally, [19] presents a siamese architecture that learns a geometrically interpretable embedding that makes the network robust to new camera views. This architecture is tested quantitatively with protocols #1, #2 and #3 of Human36M and qualitatively on MPII-2D dataset. This paper also presents an ablation study, i.e. a progressive elimination of components until reaching a baseline system that shows a degradation in the performance. However, this work has not been included in the comparison because their inputs are not images but 2D coordinates, unlike all the other works referenced. The architecture proposed in closely related to the so-called hourglass networks, in which a sequence of encoders and decoders successively create a compact representation and a reconstruction from it.

The Table 5 collects the results reported by [26] and [27] together with Ours-III for the Human36M with the protocol #1. It can be observed that resarchers have improved the methods, attaining a better performance and outranking our proposal.

For the sake of completeness, we also have trained and tested the Bayesian Capsule Networks with Procrustes transformation (Ours-IV) following the protocol #3 to compare the quantitative results both with [21] and [26]. The result is shown in Fig. 4. Our results are shown by activity (the markers) and averaged (the line) to be able to compare the performance with the one reported in [26]. It can be seen that our proposal is competitive with the state-of-the-art. Additionally we also report the averaged results in Table 5.

Table 5
Comparison of the Bayesian Capsule Networks (Ours-IV) with respect to those works that report results with Procrustes transformation and in the Human3.6M challenge, protocol #3; namely Tome [21] and Rogez [26].

Work	Error (mm.)	Std. Dev.
Tome [21]	79.6	No report
Ours, IV	71.7	28.07
Rogez [26]	**56.5**	No report

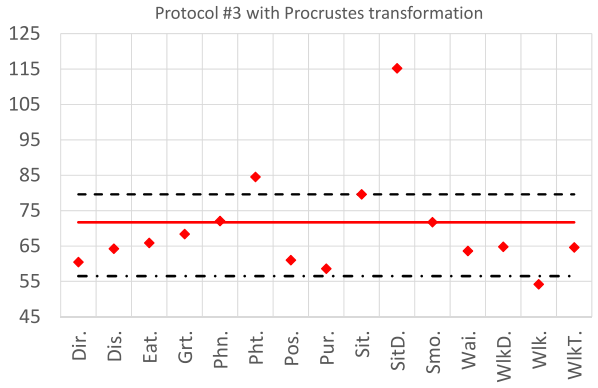


Fig. 4. Comparison of the error (mm.) with the Bayesian Capsule Networks (Ours-IV) and with the protocol #3 with respect to [21] (dashed line) and [26] (dash-dot line) according to the table reported in [26]. The continuous red line is our averaged result and the markers are our results by activity.

5.6. Discussion

The 3D human pose estimation is a challenging problem that has been comprehensively studied. All the papers ranking in the top of performance show qualitative results in different data sets such as SURREAL, DensePose-COCO, Human Pose Evaluator and MPII Human Pose dataset, but when it comes to predicting the exact coordinates of each joint on a skeleton model, Human3.6M is the reference data set used in all the works. This shortage of real ground truth is a severe drawback for a quantitative comparisons, especially in solutions that rely on Deep Learning. This is probably the reason why the works that report the lowest errors on this data set are not end-to-end. Instead, these solutions consist of different subsystems that are trained with other data sets for a specific subtask, such as estimating the coordinates in the 2D projection, and lifting from the 2D coordinates to 3D. On the contrary, the approach of this paper is to estimate the 3D coordinates directly from the image. To this end, we rely on a novel representation (the capsule network) that we extend with a regularization term for enforcing the reversibility of the inner transformations in the capsule networks. This approach has also been explored recently in [29] and also is closely related to the hourglass architecture proposed in [27]. On the other hand, the idea of learning both forwards and backwards has been a recent cornerstone in Generative Adversarial Networks (GAN). It is well known that systems trained with synthetic images perform poorly with real images. This gap has begun to be solved recently by using Cycle GANs. In that sense, the regularization term proposed is an approximation to this behaviour that can be applied to any fully connected layer with linear activations.

Fig. 5. Some results of our proposed Bayesian Capsule Network in a test subject. (Left) The 2D RGB image input, (middle) the 3D predicted Human Pose, and (right)the given ground-truth. 3D skeletons have different viewpoints for the sake of visualization, but all of them have the same orientation.

Finally, regading execution time, the use of this reduced and simplified architecture that we propose provides a reduced time in inference of 0.00863 s./frame.

6. Conclusion

Bayesian Neural Networks are easy to implement and offer two big advantages. By performing dropout and small modifications in the usual loss functions, both in classification and regression, it is possible to: 1) minimize the homoscedastic uncertainty, and 2) use multiple losses for different complementary tasks, self-balancing their contribution to the total loss. In this paper we present, for the first time, a Bayesian Capsule Network. Due to the structure of Capsnets, the bayesian formulation consists of doing dropout on the initial capsules and a regularization term over the linear transformations in the *Inverse Graphics* stage. We test the proposal on an ill-posed problem and show that the results are comparable to the state-of-the-art, but using a straightforward and much simpler approach over the Human3.6M data set.

Declaration of Competing Interest

The authors of this manuscript, whose names are listed above, certify that they have NO affiliations with or involvement in any organization or entity with any financial interest (such as honoraria; educational grants; participation in speakers bureaus; membership, employment, consultancies, stock ownership, or other equity interest; and expert testimony or patent-licensing arrangements), or non-financial interest (such as personal or professional relationships, affiliations, knowledge or beliefs) in the subject matter or materials discussed in this manuscript.

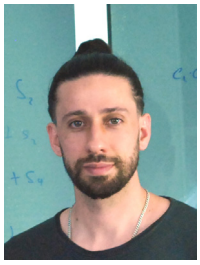
Acknowledgments

This research has been supported by the Spanish Government research fundings TIN-2015-69542-C2-1-R (MINECO/FEDER), RTI2018-098743-B-I00 (MICINN/FEDER) and Y2018/EMT-5062 (Comunidad de Madrid).

References

[1] K. Hao, “We analyzed 16,625 papers to figure out where AI is headed next”, MIT Technol. Rev. (2019) 1–19, https://www.technologyreview.com/s/612768/we-analyzed-16625-papers-to-figure-out-where-ai-is-headed-next.
[2] A. Kleiner, A. Talwalkar, P. Sarkar, M.I. Jordan, A scalable bootstrap for massive data, J. R. Stat. Soc. B 76 (2014) 795816, doi:10.1111/rssb.12050.
[3] J.S. Denker, Y. LeCun, Transforming neural-net output levels to probability distributions, in: Proceedings of the Advances in Neural Information Processing Systems 3, 1990, pp. 853–859.
[4] D.J.C. MacKay, A practical bayesian framework for backpropagation networks, Neural. Comput. 4 (3) (1992) 448–472, doi:10.1162/neco.1992.4.3.448.
[5] C. Blundell, J. Cornebise, K. Kavukcuoglu, D. Wierstra, Weight uncertainty in neural networks, in: Proceedings of the 32nd International Conference on Machine Learning, ICML’15, 2015, pp. 1613–1622.
[6] J.M. Hernández-Lobato, R.P. Adams, Probabilistic backpropagation for scalable learning of bayesian neural networks, in: Proceedings of the 32Nd International Conference on Machine Learning, ICML’15, 2015, pp. 1861–1869.
[7] H. Wang, D. Yeung, Towards bayesian deep learning: a framework and some existing methods, IEEE Trans. Knowl. Data Eng. 28 (12) (2016) 3395–3408, doi:10.1109/TKDE.2016.2606428.
[8] J. Chien, Y. Ku, Bayesian recurrent neural network for language modeling, IEEE Trans. Neural Netw. Learn. Syst. 27 (2) (2016) 361–374, doi:10.1109/TNNLS.2015.2499302.
[9] Y. Zhu, N. Zabaras, Bayesian deep convolutional encoder-decoder networks for surrogate modeling and uncertainty quantification, J. Comput. Phys. 366 (2018) 415–447, doi:10.1016/j.jcp.2018.04.018.
[10] Y. Gal, Z. Ghahramani, Dropout as a bayesian approximation: representing model uncertainty in deep learning, in: Proceedings of the 33rd International Conference on Machine Learning, 48, 2016, pp. 1050–1059.
[11] S. Mandt, M.D. Hoffman, D.M. Blei, Stochastic gradient descent as approximate bayesian inference, J. Mach. Learn. Res. 18 (1) (2017) 4873–4907.
[12] S. Sabour, N. Frosst, G.E. Hinton, Dynamic routing between capsules, in: Proceedings of the Advances in Neural Information Processing Systems, 2017, pp. 3859–3869.
[13] G.E. Hinton, S. Sabour, N. Frosst, Matrix capsules with EM routing, in: Proceedings of the International Conference on Learning Representations, 2018.
[14] Y. LeCun, C. Cortes, MNIST handwritten digit database, 2010 [Online] Available at: http://yann.lecun.com/exdb/mnist/.
[15] J. O'rourke, N.I. Badler, Model-based image analysis of human motion using constraint propagation, IEEE Trans. Pattern Anal. Mach. Intell. 2 (6) (1980) 522–536.
[16] D. Hogg, Model-based vision: a program to see a walking person, Image Vis. Comput. 1 (1) (1983) 5–20.

[17] C. Hong, J. Yu, J. Wan, D. Tao, M. Wang, Multimodal deep autoencoder for human pose recovery, IEEE Trans. Image Process. 24 (12) (2015) 5659–5670.
[18] C. Hong, J. Yu, D. Tao, M. Wang, Image-based three-dimensional human pose recovery by multiview locality-sensitive sparse retrieval, IEEE Trans. Ind. Electron. 62 (6) (2014) 3742–3751.
[19] M. Véges, V. Varga, A. Lorincz, 3D Human pose estimation with siamese equivariant embedding, Neurocomputing 339 (2019) 194–201. doi:j.neucom.2019.02.029
[20] C.-H. Chen, D. Ramanan, 3d human pose estimation= 2D pose estimation+ matching, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 7035–7043.
[21] D. Tome, C. Russell, L. Agapito, Lifting from the deep: convolutional 3D pose estimation from a single image, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
[22] S. Li, A.B. Chan, 3d human pose estimation from monocular images with deep convolutional neural network, in: Proceedings of the Asian Conference on Computer Vision, Springer, 2014, pp. 332–347.
[23] S. Li, W. Zhang, A.B. Chan, Maximum-margin structured learning with deep networks for 3D human pose estimation, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 2848–2856.
[24] J.J. Tompson, A. Jain, Y. LeCun, C. Bregler, Joint training of a convolutional network and a graphical model for human pose estimation, in: Proceedings of the Advances in Neural Information Processing Systems, 2014, pp. 1799–1807.
[25] A. Toshev, C. Szegedy, Deeppose: human pose estimation via deep neural networks, in: Proceedings of the IEEE Conference on Computer Vision And Pattern recognition, 2014, pp. 1653–1660.
[26] G. Rogez, P. Weinzaepfel, C. Schmid, LCR-NET++: multi-person 2D and 3D pose detection in natural images, IEEE Trans. Pattern Anal. Mach. Intell. (2019), doi:10.1109/TPAMI.2019.2892985. Early Access
[27] Y. Tian, W. Hu, H. Jiang, J. Wu, Densely connected attentional pyramid residual network for human pose estimation, Neurocomputing 347 (2019) 13–23.
[28] J. Liu, H. Ding, A. Shahroudy, L. Duan, X. Jiang, G. Wang, A. Kot Chichung, Feature boosting network for 3d pose estimation, IEEE Trans. Pattern Anal. Mach. Intell. (2019), doi:10.1109/TPAMI.2019.2894422. Early Access
[29] K. Wang, L. Lin, C. Jiang, C. Qian, P. Wei, 3D human pose machines with self-supervised learning, IEEE Trans. Pattern Anal. Mach. Intell. (2019), doi:10.1109/TPAMI.2019.2892452.
[30] A. Kendall, Y. Gal, What uncertainties do we need in bayesian deep learning for computer vision? in: Proceedings of the Advances in Neural Information Processing Systems, 2017, pp. 5580–5590.
[31] N. Srivastava, G.E. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958.
[32] A. Kendall, Y. Gal, R. Cipolla, Multi-task learning using uncertainty to weigh losses for scene geometry and semantics, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 7482–7491, doi:10.1109/CVPR.2018.00781.
[33] K. Kolev, M. Klodt, T. Brox, D. Cremers, Continuous global optimization in multiview 3D reconstruction, Int. J. Comput. Vis. 84 (1) (2009) 80–96, doi:10.1007/s11263-009-0233-1.
[34] S. Nie, M. Zheng, Q. Ji, The deep regression bayesian network and its applications: probabilistic deep learning for computer vision, IEEE Signal Process. Mag. 35 (1) (2018) 101–111, doi:10.1109/MSP.2017.2763440.
[35] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[36] C. Ionescu, D. Papava, V. Olaru, C. Sminchisescu, Human3.6m: large scale datasets and predictive methods for 3D human sensing in natural environments, IEEE Trans. Pattern Anal. Mach. Intell. 36 (7) (2014) 1325–1339.
[37] B. Tekin, P. Marquez Neila, M. Salzmann, P. Fua, Learning to fuse 2D and 3D image cues for monocular body pose estimation, in: Proceedings of the International Conference on Computer Vision (ICCV), 2017.
[38] X. Zhou, M. Zhu, S. Leonardos, K.G. Derpanis, K. Daniilidis, Sparseness meets deepness: 3D human pose estimation from monocular video, in: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, 2016, pp. 4966–4975.
[39] I. Katircioglu, B. Tekin, M. Salzmann, V. Lepetit, P. Fua, Learning latent representations of 3D human pose with deep neural networks, International Journal of Computer Vision (2018) 1–16.
[40] F. Bogo, A. Kanazawa, C. Lassner, P.V. Gehler, J. Romero, M.J. Black, Keep it SMPL: automatic estimation of 3D human pose and shape from a single image, in: Proceedings of the European Conference on Computer Vision (ECCV), 2016, pp. 561–578, doi:10.1007/978-3-319-46454-1_34.
[41] M. Sanzari, V. Ntouskos, F. Pirri, Bayesian image based 3D pose estimation, in: Proceedings of the European Conference on Computer Vision (ECCV), Springer, 2016, pp. 566–582.



Iván Ramírez received his M.Sc. degree in telecommunication engineering from Universidad de Sevilla, Seville in 2014 and his Ms.ScCV in Computer Vision from Universidad Rey Juan Carlos, Madrid in 2016. He is currently a Ph.D. candidate at the Department of Computer Science and at the Department of Applied Mathematics at Rey Juan Carlos University. He is a member of CAPO research group and CVIP. His research interests include machine learning for computer vision and variational methods for image processing.

Alfredo Cuesta-Infante is currently Visiting Professor at Univ. Rey Juan Carlos in the Department Computer Science, and member of the High-Performance Computation and Optimization research group in that Univ. He received his M.Sc. degree in Physics (Univ. Complutense de Madrid, UCM) and his Ph.D. in Computer Science (Univ. Nacional de Educación a Distancia, UNED). He has been visiting faculty at Univ. of New Mexico (UNM) and at Massachusetts Institute of Technology (MIT). His research interests include Machine Learning, Computer Vision and Copula theory.

Emanuele Schiavi received his Ph.D. in Applied Mathematics from the University Complutense of Madrid, Spain, in 1997. He is currently a Senior Lecturer in the Department of Applied Mathematics at University Rey Juan Carlos, Madrid, Spain. There he is a member of the Computer Vision and Image Processing Group (CVIP) His current research interests include Partial Differential Equations, Variational Methods, Image Processing and Deep Learning.

Juan J. Pantrigo is currently an associate professor at Universidad Rey Juan Carlos and member of CAPO research group in the Department of Computer Science. He received his M.Sc. degree in fundamental physics from Universidad de Extremadura in 1998 and his Ph.D. from Universidad Rey Juan Carlos in 2005. His research interests include high-dimensional space-state tracking problems, computer vision, heuristic optimization, machine learning and hybrid approaches.