sciendo

JAISCR, 2022, Vol. 12, No. 1, pp. 61– 73
10.2478/jaiscr-2022-0005

EVALUATING DROPOUT PLACEMENTS IN BAYESIAN REGRESSION RESNET

Lei Shi*, Cosmin Copot, Steve Vanlanduit

InViLab, Falcuty of Applied Engineering, University of Antwerp
Groenenborgerlaan 171, 2020 Antwerp, Belgium

**E-mail: lei.shi@uantwerpen.be*

Submitted: 4th December 2020; Accepted: 2nd July 2021

Abstract

Deep Neural Networks (DNNs) have shown great success in many fields. Various network architectures have been developed for different applications. Regardless of the complexities of the networks, DNNs do not provide model uncertainty. Bayesian Neural Networks (BNNs), on the other hand, is able to make probabilistic inference. Among various types of BNNs, *Dropout as a Bayesian Approximation* converts a Neural Network (NN) to a BNN by adding a dropout layer after each weight layer in the NN. This technique provides a simple transformation from a NN to a BNN. However, for DNNs, adding a dropout layer to each weight layer would lead to a strong regularization due to the deep architecture. Previous researches [1, 2, 3] have shown that adding a dropout layer after each weight layer in a DNN is unnecessary. However, how to place dropout layers in a ResNet for regression tasks are less explored. In this work, we perform an empirical study on how different dropout placements would affect the performance of a Bayesian DNN. We use a regression model modified from ResNet as the DNN and place the dropout layers at different places in the regression ResNet. Our experimental results show that it is not necessary to add a dropout layer after every weight layer in the Regression ResNet to let it be able to make Bayesian Inference. Placing Dropout layers between the stacked blocks i.e. Dense+Identity+Identity blocks has the best performance in Predictive Interval Coverage Probability (PICP). Placing a dropout layer after each stacked block has the best performance in Root Mean Square Error (RMSE).

Keywords: Regression, Bayesian Neural Network, MC Dropout

1 Introduction

Deep learning has shown great success in the field of computer vision. Various networks have shown high capabilities in image recognition [4, 5, 6, 7]. Based on these networks, more work have been applied for tasks such as object detection [8, 9, 10], semantic segmentation [11, 12, 13], pose estimation [14, 15, 16], gaze estimation [17, 18, 19] and so on. In general, these networks use convolutional neural networks (CNN) to extract features and apply different functions for different types of tasks. For classification tasks such as image classification, SoftMax + Fully Connected (FC) layers are usually used to get class probabilities. For regression tasks such as pose estimation, FC layers are normally applied to obtain output vector(s). ResNet [7] uses shortcut connections to overcome the vanishing gradient problem when the layers going deeper. In addition to its use classification tasks, ResNet is used a lot in regression tasks [20, 21, 22, 23]. An empirical study [24] is performed to analyze the VGG-16 [5] and ResNet-50 networks for regression task. The inputs of these works are images hence CNNs are used to extract features.

The regression task in computer vision takes images as the input to the network. In other applications besides computer vision, the input data are often in the form of vectors. Common machine learning based regression techniques include Decision Tree Regression (DTR) [25], Support Vector Regression (SVR)[26] and Neural Network (NN). DNNs are also used for regression tasks when the input data is a vector. In these cases, the DNNs used in computer vision need to be modified. One common solution is replacing the convolutional layers with FC layers [27, 28, 29]. In [28], a modified ResNet, we refer it as Regression ResNet (RRN), is used for the regression. The convolutional layers and pooling layers in ResNet are replaced with FC layers. The authors of [28] show that RRN has better performance than the DTR, SVR and NN. IRNet and SRNet [27] are similar to [28], the main difference is the strategy of residual connections and the number of hidden units. The authors compare IRNet with 10 other machine learning algorithms (including Random Forest, DTR, SVR and so on). Their results show that the IRNet outperforms the 10 machine learning algorithms. In [29], Multi-Layer Perceptron (MLP) is adopted in the Encoder-Decoder architecture. Residual shortcut connections are added between Encoders and Decoders. Adding shortcut connections can increase the performance of regression. The proposed model also has better performance than MLP.

Despite the huge success of DNN, it does not provide weight uncertainty of the model thus it is not able to make probabilistic inference. Bayesian Neural Network (BNN) can capture model uncertainty [30]. A modern technique used in BNNs is Variational Inference (VI), which learns approximate posterior distributions [31]. VI based BNN models include Bayes by Backprop [32], Dropout as a Bayesian Approximation [33] and Bayesian Hypernetwork [34, 35].

Dropout as a Bayesian Approximation adds Dropout [36] layers to the weight layers. The authors of [33] have shown that it is equivalent to approximating Bayesian inference in deep Gaussian processes. It has the advantage of simple implementation compared to other BNNs. Several works have adopted this technique into existing DNNs. In [1], Dropout as a Bayesian Approximation is applied on PoseNet [15] to make Bayesian inference on camera pose, it uses GoogLeNet [6] as the backbone. In [2], SegNet [12] also uses Dropout as a Bayesian Approximation for semantic segmentation. Segnet uses a Encoder-Decoder frame and VGG-16 as the encoder and the decoder is in the reverse order. In [3], the Bayesian QuickNAT is used for whole-brain segmentation on Magnetic Resonance Imaging (MRI) images. The QuickNAT is based on Fully Convolutional Network (FCN) [11]. Although the Dropout as A Bayesian Approximation is easy to implement, adding a Dropout layer after each convolutional layer will make the regularization too strong [2] in DNNs. It will lead to a long training process. In all the works [1, 2, 3] mentioned above, a reduced number of dropout layers are added at different places in the networks to avoid the strong regularization. All the results show that not adding a dropout layer after each convolutional layer is a better option. Notwithstanding there are numerous DNN architectures proposed in various fields of researches, there is no analytical solution showing the optimal locations to put dropout layers in a certain type of network architecture. Moreover, placing dropout in DNNs for regression and in ResNet is less explored.

In this work, we conduct an empirical study to analyze the BRRN by Applying Dropout as a Bayesian Approximation to the RRN. We place dropout layers at the different locations in the network architecture and see the impacts of these different placements for regression tasks. We observed that placing a Dropout layer after every FC layer in the RRN is not necessary which is consistent with the findings in [2, 12, 3]. Placing dropout layers between the stacked blocks in the RRN has the best performance in Predictive Interval Coverage Probability (PICP). Placing a dropout layer after each stacked block has the best performance in Root Mean Square Error (RMSE). When designing a BRRN using dropout as a Bayesian Approximation, we demonstrate the schemes of dropout placements that can avoid strong regularization and obtain higher accuracy. Our work provides a reference for deploying BRRN in regression applications.

The remainder of the paper is organized as follows, in Section 2, we show the details of the RRN and the methodology of converting a NN to a BNN. In Section 3, we describe the settings for the experiments. We perform two experiments. First, we test different dropout locations in the residual connections in a residual block. Then we test the dropout locations between the residual blocks of the RRN. The experimental results are demonstrated in Section 4. The conclusion is given in Section 5.

2 Methodology

2.1 Regression ResNet

We use the modified ResNet i.e. the RRN in [28] as the DNN. Figure 1 shows the network architecture. The network consists of stacked Dense blocks and Identity blocks. The convolutional layers and pooling layers in ResNet is replaced by FC layers in the RRN. In one stacked block, one Dense block and two Identity blocks are connected sequentially. The network consists of three stacked blocks. A FC layer is used to readout the output. In Dense blocks, the residual connection is fed to a FC layer and Batch Normalization (BN)[37]. In Identity blocks, the residual connection is directly added to the feedforward output. In both types of blocks, the FC layer is followed by BN and non-linear activation i.e. Rectified Linear Uni (ReLU)[38] except the residual connection in the Dense block.

2.2 MC Dropout

For a Neural Network (NN) with weights $\mathbf{W}$, bias $\mathbf{b}$, adding Dropout layers after each weight layer equivalent to approximate Bayesian inference in deep Gaussian processes [33]. This technique can be referred as MC Dropout [34]. We shortly describe the MC Dropout from [33].

When a dropout operation is applied, some weights are removed and they follow the Bernoulli distribution. For the ith layer in a NN, the output is

$$\hat{\mathbf{y}}_{i+1} = \sigma(\mathbf{x}_i(\mathbf{z}_i\mathbf{W}_i) + \mathbf{b}_i)(\mathbf{z}_{i+1}\mathbf{W}_{i+1}), \qquad (1)$$

where σ is the non-linear activation i.e. ReLU operation and $\mathbf{z}$ is the probability distribution for dropout operation. The cost function $\mathcal{L}$ of a NN is

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} E(\mathbf{y}_i, \hat{\mathbf{y}}_i) + \lambda \sum_{i=1}^{L}(||\mathbf{W}_i||_2^2 + ||\mathbf{b}_i||_2^2), \qquad (2)$$

where $E(\cdot,\cdot)$ is the loss function, $\mathbf{y}$ and $\hat{\mathbf{y}}$ are the ground truth and the prediction of the model respectively, N is the length of the training data. The second term is a L_2 regularization term with decay factor λ. For approximating the Gaussian process model, the variational inference is used and Monte Carlo integration is used for minimizing KL divergence. The cost function is proportional to

$$\mathcal{L}_{GP-MC} \propto \frac{1}{2N}\sum_{i=1}^{N} E(\mathbf{y}_i, \hat{\mathbf{y}}_i) + \lambda \sum_{i=1}^{L}\left(\frac{(1-p_i)l^2}{2\tau N}||\mathbf{W}_i||_2^2 + \frac{l^2}{2\tau N}||\mathbf{b}_i||_2^2\right), \qquad (3)$$

where p_i is the dropout rate in ith layer, τ is model precision and l is prior length scale. The predictive mean and predictive variance can be obtained by iterating the NN forward process T times,

$$\mathbb{E}(\mathbf{y}^*) \approx \frac{1}{T}\sum_{i=1}^{T} \hat{\mathbf{y}}^*(\mathbf{x}^*, \mathbf{W}_1^i, ..., \mathbf{W}_L^i), \qquad (4)$$

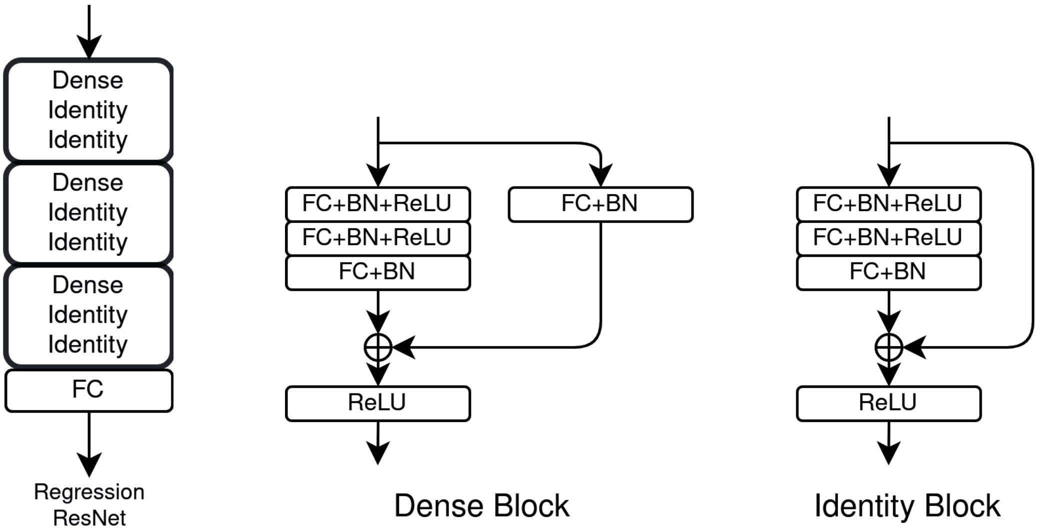
$$Var((\mathbf{y}^{*T})(\mathbf{y})) \approx \frac{1}{T}\sum_{i=1}^{T} \hat{\mathbf{y}}^*(\mathbf{x}^*, \mathbf{W}_1^i, ..., \mathbf{W}_L^i)^T \hat{\mathbf{y}}^*(\mathbf{x}^*, \mathbf{W}_1^i, ..., \mathbf{W}_L^i), \qquad (5)$$

where $\mathbb{E}$ is the predictive mean and Var is predictive variance, $\mathbf{x}^*$ and $\hat{\mathbf{y}}^*$ is the new entry data for inference and the new prediction. The weight decay λ is calculated by

$$\lambda = \frac{pl^2}{2N\tau}, \qquad (6)$$

2.3 Bayesian Regression ResNet

Combining the RRN and MC Dropout, the algorithm for the inference of BRRN is shown below, where $\mathbf{f}$ is the BRRN network and $\mathbf{P}$ is the predictions of the BRRN. Give the dropout rate p, the model precision tau and the length scale l, the weight decay λ can be calculated from Equation 6. To make Bayesian inference for a trained BRRN, simply put a new data entry to the model for T times. The mean prediction and the variance of the



(a) Network architecture of Regression ResNet.

(b) Dense block and Identity block.

Figure 1. Architecture of the Regression ResNet and the Dense block and the Identity block.

predication can be calculated from T predictions.

Algorithm 1: Bayesian Regression ResNet Inference

```
Input: $\lambda$, p, T
Output: $\mathbb{E}, Var$
Data: Test dataset $\mathbf{x}^*$
Enable dropout;
$\mathbf{P} = 0$;
$\mathbf{f}(\lambda, p)$;
for $i \leftarrow 0$ to $T$ do
    $\mathbf{y}^* = \mathbf{f}(\mathbf{x}^*)$;
    $\mathbf{P} = \mathbf{P} \bigcup \mathbf{y}^*$;
end
$\mathbb{E}(\mathbf{P})$; $Var(\mathbf{P})$;
```

3 Experiment Settings

We perform two experiments, *Dropout Placements in Residual Blocks* and *Dropout Placements in Bayesian Regression ResNet*. In *Dropout Placements in Bayesian Regression ResNet*, we evaluate the performance of placing a dropout layer after each weight layer. However, the residual connection will lead to a few dropout placement variants within the Dense block and the Identity block. We test and select the best dropout placement variant in this experiment. In *Dropout Placements in Bayesian Regression ResNet*, Wee use the result from the previous experiment as the baseline i.e. placing a dropout layer after each weight layer. We evaluate different dropout locations in the stacked blocks and compare the performances with the performance of placing a dropout layer after each weight layer.

3.1 Dataset

We use 10 datasets for our experiments. The details of the datasets are indicated in Table 1. The datasets are from various fields. The SafeVS dataset is from [39], the rest datasets are the same as in [33]. The SafeVS dataset collects the hand positions and robot TCP positions and predicts repulsive TCP positions for hand collision avoidance. All the data is not pre-processed except for the normalization. We apply Z Normalization to all datasets. The Z-Normalization is defined as,

$$\mathbf{X}' = \frac{\mathbf{X} - \mu_x}{\sigma_x}, \tag{7}$$

where μ_x and σ_x are the mean and standard deviation of the training data $\mathbf{X}$.

3.2 Evaluation Metrics

The evaluation metrics are RMSE and PICP [40]. The PICP is calculated as follows,

$$P(\hat{y}_{L_i} \leq y_i \leq \hat{y}_{U_i}) > 0.95, \tag{8}$$

Table 1. Datasets details.

	No. of Sample	Input Dimension	Output Dimension
SafeVS	500000	6	3
Kin8nm	8192	8	1
Boston	506	13	1
Concrete	1030	8	1
Energy	768	8	1
Naval	11934	16	1
Power	9568	4	1
Protein	45730	9	1
Wine	1599	11	1
Yacht	308	6	1

where $\hat{y}_{L_i}$ and $\hat{y}_{U_i}$ is the lower bound and the upper bound of the prediction interval. We select the confidence as 95%. A boolean vector **b** summarizes if y_i is captured by the prediction interval,

$$b_i = \begin{cases} 1 & \text{if } \hat{y}_{L_i} \leq y_i \leq \hat{y}_{U_i} \\ 0 & \text{else ,} \end{cases} \tag{9}$$

The PICP is calculated as,

$$PICP = \frac{1}{n}\sum_{i=1}^{n} b_i. \tag{10}$$

We also use the mean rank to compare the performances of the models in datasets. The mean rank R is calculated as,

$$R = \frac{1}{K}\sum_{i=1}^{K} rank_i, \tag{11}$$

where K is the number of datasets. We calculate rank R_{RMSE} in RMSE and the rank R_{PICP} in PICP as well as R_{MEAN} which is the mean of R_{RMSE} and R_{PICP}.

4 Results

We demonstrate the experimental results of dropout Placements in Residual Blocks and dropout Placements in Bayesian Regression ResNet in this Section.

4.1 Dropout Placements in Residual Blocks

4.1.1 Baseline

In this experiment, we evaluate the dropout placements in the Dense block and the Identity block. The architecture of the baseline model is shown in Figure 2. The number (N) of the stacked block Dense+Identity+Identity is one. The dropout layer is added after the residual connection and nonlinear activation. The experiment is performed on Boston Housing, Concrete Compressive Strength, Energy Efficiency, Naval Propulsion Plants, Combined Cycle Power Plant, Wine Quality and Yacht Hydrodynamics. The loss function is Mean Square Error (MSE) loss and the optimizer is Adam. After every training epoch, a validation is performed. The final model is the one with the least validation loss. The hyper-parameters for training the baseline model are shown in Table 2.

4.1.2 Dropout Placement Variants

We evaluate three dropout placement variants in the Dense block and two variants in the Identity block. Figure 3 shows the detailed configuration of the variants. Since the model consists of the stacked block combination Dense+Identity+Identity. We name the variant as Di-Ij where i is the index in the Dense block configuration and j is the index in the Identity block configuration. For instance, D1-I1 represents that the Dense block in the network uses the Dropout-Dense-1 (Figure 3) configuration and the Identity blocks use the Dropout-Iden-1 (Figure 3) configuration. The baseline variant is D3-I0.

Table 3 shows the results of Dropout placement variants in the Dense block and the Identity block.

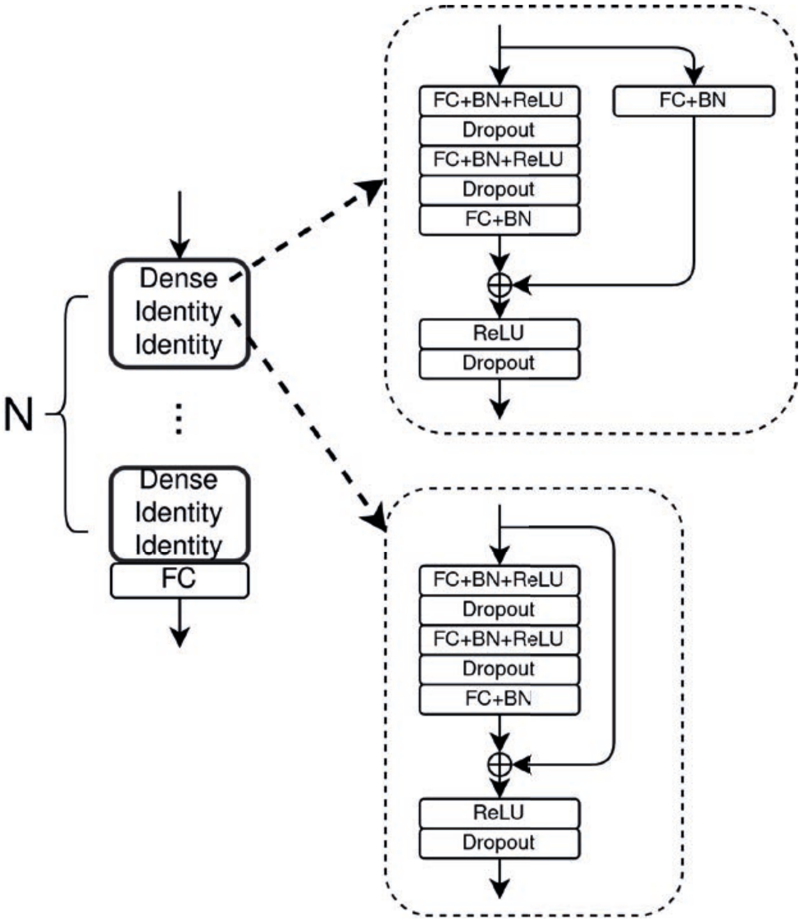


Figure 2. The baseline of the dropout placements. A dropout layer is placed after each weight layer.

Table 2. Hyper parameters of the baseline model for the experiment *Dropout Placements in Residual Blocks*.

	Boston	**Concrete**	**Energy**	**Naval**	**Power**	**Wine**	**Yacht**
Epoch	130	200	200	150	200	200	200
Hidden Unit	32	16	16	32	16	16	16
Batch Size	32	8	64	128	128	32	16
Learning Rate	0.001	1e-4	0.001	1e-4	0.001	1e-4	1e-4
Tau	5e-4	0.03	0.15	0.5	0.05	2	0.01
Dropout Rate	0.1	0.005	0.05	0.005	0.005	0.1	0.005

The baseline model i.e. D3-I0 performs best on the Boston Housing and Naval Propulsion datasets, the ranks are within two. D0-I0 performs best on Energy Efficiency and Yacht Hydrodynamics datasets. The performance on Naval Propulsion is slightly lower. For D1-I1, the performances on Concrete Compressive Strength and Power Plant are the best. D3-I1 also performs well on Boston Housing. Regarding to the ranks of the model variants, we use $\mathbf{R_{RMSE}}$, $\mathbf{R_{PICP}}$ and $\mathbf{R_{MEAN}}$ for the evaluation. Considering the RMSE rank, D2-I1 has the best RMSE mean rank score. D1-I0 has the best PICP mean rank. Considering both RMSE and PICP, D1-I0 and D0-I0 are the only two variants with the rand score below 4 and D1-I0 is around 0.3 better than D0-I0. Overall, D1-I0 has the best PICP mean rank score and the second best RMSE mean rank score.

4.2 Dropout Placements in Bayesian Regression ResNet

4.2.1 Baseline

We evaluate the dropout placement in the BRR. The baseline model for this experiment is similar to the one in Figure 2. The N is set to three, which is the same as the RR. A dropout layer is placed after each weight layer. In the case of residual connection, we adapt the D1-I0 variant configuration in each of the Dense+Identity+Identity stacked blocks. We refer the baseline model as *DO-Dense*. We applied the same baseline model to Kin8nm dataset, Protein dataset and SafeVS dataset. The loss function and the optimizer is the same as in the previous experiment. The hyper-parameters for the baseline model in the datasets are shown in Table 4.

Table 4. Hyper parameters of the baseline model for the experiment *Dropout Placements in Bayesian Regression ResNet*.

	Kin8nm	**SafeVS**	**Protein**
Epoch	200	200	250
Hidden Unit	16	32	16
Batch Size	16	5000	512
Learning Rate	0.005	0.001	0.001
Tau	2	1e-5	0.5
Dropout Rate	5e-4	0.1	0.005

4.2.2 Dropout Placement Variants

We evaluate five Dropout placement variants, DO-Inter, DO-InterD, DO-Before, DO-After and DO-SE in the Regression ResNet. The details of the variants are indicated in Figure 4. In DO-Inter, all three stacked Dense+Identity+Identity blocks are separated by Dropout layers. In DO-InterD, one Dropout layer is added after each Dense block and Identity block. In DO-Before and DO-After configurations, the one Dropout layer is added before and after each stacked block. The DO-SE only places Dropout layers at the start and the end of the stacked blocks.

Table 5 shows the RMSEs and PICPs of the dropout placement variants on SafeVS dataset. DO-After obtains the best RMSE among all variant configurations. DO-Before, DO-After and DO-SE generally perform well than DO-Dense, DO-Inter and DO-InterD in RMSE. The differences are clearly observable. For PICP (1), all other variants perform better than the baseline model. For PICP (2) and PICP (3), DO-Before has comparable results with the baseline model, the rest variant configurations have obvious better results than the baseline. Considering the RMSE and PICPs together, all the tested dropout placement variants outperform the baseline i.e. DO-Dense. DO-InterD has extremely well performance on PICP (2) and PICP (3) but it only performs better than the baseline on PICP (1). DO-Inter and DO-SE have similar results on PICP (2) and PICP (3), however DO-SE has higher PICP (1) than DO-Inter. In addition, The RMSE of DO-SE is lower than DO-Inter. The difference is around 6. In total, DO-SE has the best performance in PICP among all dropout Placement Variants and DO-After has the least RMSE.

Table 6 shows the RMSEs and PICPs of the dropout placement variants on Kin8nm and Protein dataset. The RMSEs of all variants on Kin8nm dataset are comparable. DO-After has the least RMSE. Except for DO-SE, the RMSEs of all other variants are around 0.25. The PICP of DO-Inter is the highest among all variants on Kin8nm. For the Protein dataset, DO-After also has the least RMSE and highest PICP. For the baseline model, the performance in PICP is only better than DO-SE for Kin8nm and DO-Before for Protein. Although the RMSE for Kin8nm is the second best, the score is not obviously better than other Dropout variants.

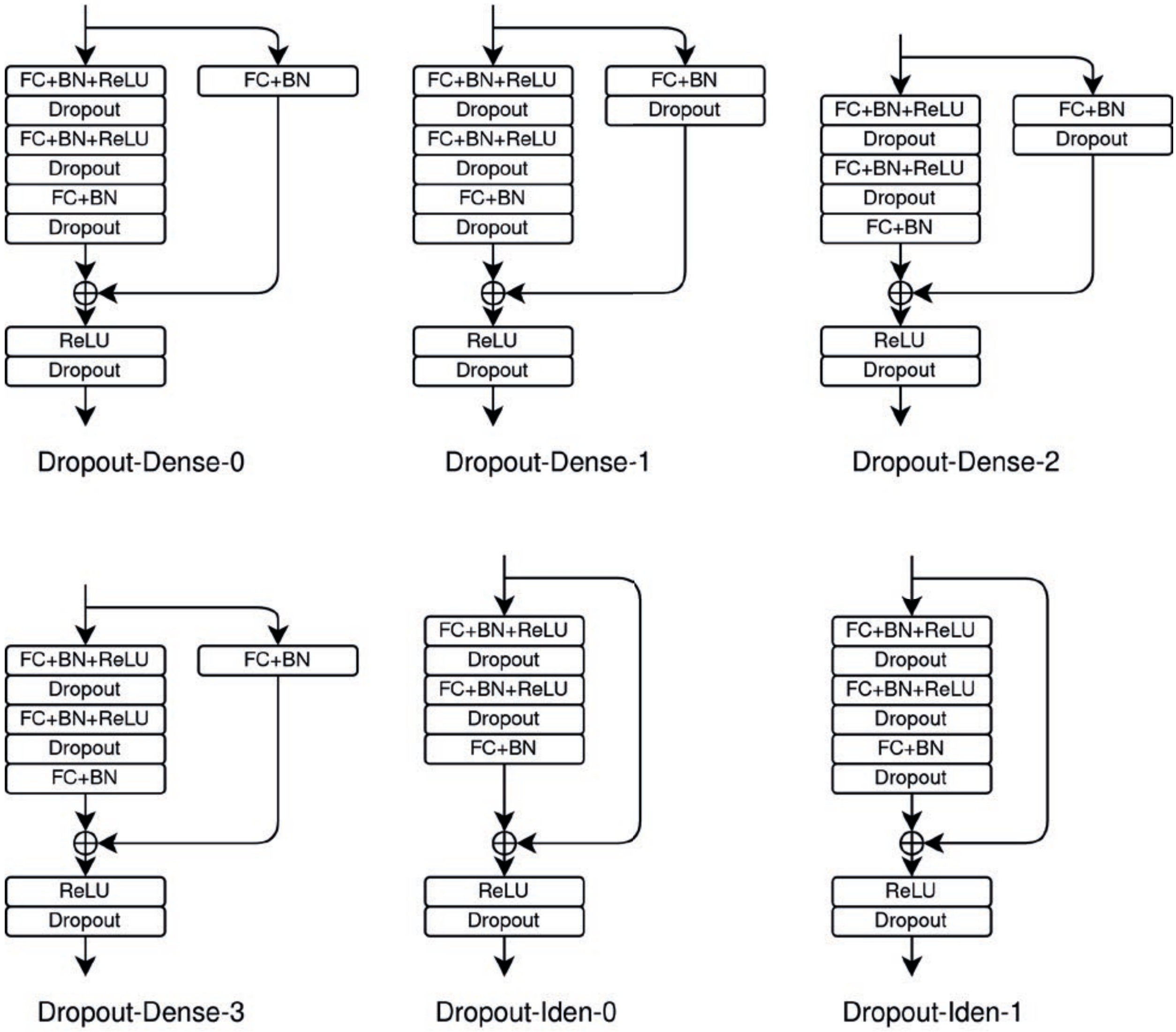


Figure 3. Different Dropout placement variants in the Dense block and the Identity block.

Table 3. Dropout placement variants in the Dense block and the Identity block.

		Baseline(D3-I0)	D0-I0	D1-I0	D2-I0	D0-I1	D1-I1	D2-I1	D3-I1
Boston	**RMSE**	2.4978	2.6249	2.6039	2.5165	3.065	3.318	2.8683	**2.2879**
	PICP	**0.902**	0.8325	0.8039	0.7647	0.7255	0.7451	0.8431	0.8627
Concrete	**RMSE**	18.0635	18.2892	**17.6064**	18.1651	18.0433	17.789	18.0225	18.1268
	PICP	0.5	**0.5577**	0.5288	0.5192	0.5192	**0.5577**	0.5288	0.5481
Energy	**RMSE**	1.7527	**1.631**	1.6541	3.1067	3.5543	2.0553	1.5758	3.6017
	PICP	**0.9539**	0.9231	0.9359	0.8333	0.8718	0.8205	0.8974	0.8846
Naval	**RMSE**	**0.001**	0.0011	0.0029	0.0027	0.003	0.003	**0.001**	0.0028
	PICP	0.8728	**0.8861**	0.788	0.7714	0.7714	0.8362	0.8362	0.8188
Power	**RMSE**	5.0471	5.0156	5.0402	5.0686	5.0524	**4.9525**	5.0687	4.9886
	PICP	0.6415	0.6269	**0.6674**	0.6508	0.6321	0.6601	0.6259	0.6259
Wine	**RMSE**	0.8543	0.8659	0.8481	0.8656	0.8441	0.8656	**0.8437**	0.8482
	PICP	0.8148	0.8395	0.8395	0.8519	**0.8642**	0.8333	0.8272	0.8519
Yacht	**RMSE**	5.8718	**5.4153**	5.8387	5.6126	5.9612	5.8003	5.638	5.4224
	PICP	0.6875	0.7812	**0.8125**	0.6875	0.7188	0.75	0.5625	0.7188
Rank	$\mathbf{R_{RMSE}}$	4.1429	4.2857	3.8571	5.1429	5.8571	4.8571	**3.4286**	4
	$\mathbf{R_{PICP}}$	4.8257	3.1429	**3**	5.5714	5.1429	4.2857	5.2857	4.2857
	$\mathbf{R_{MEAN}}$	4.2143	3.7143	**3.4286**	5.3571	5.5	4.5714	4.3571	4.1429

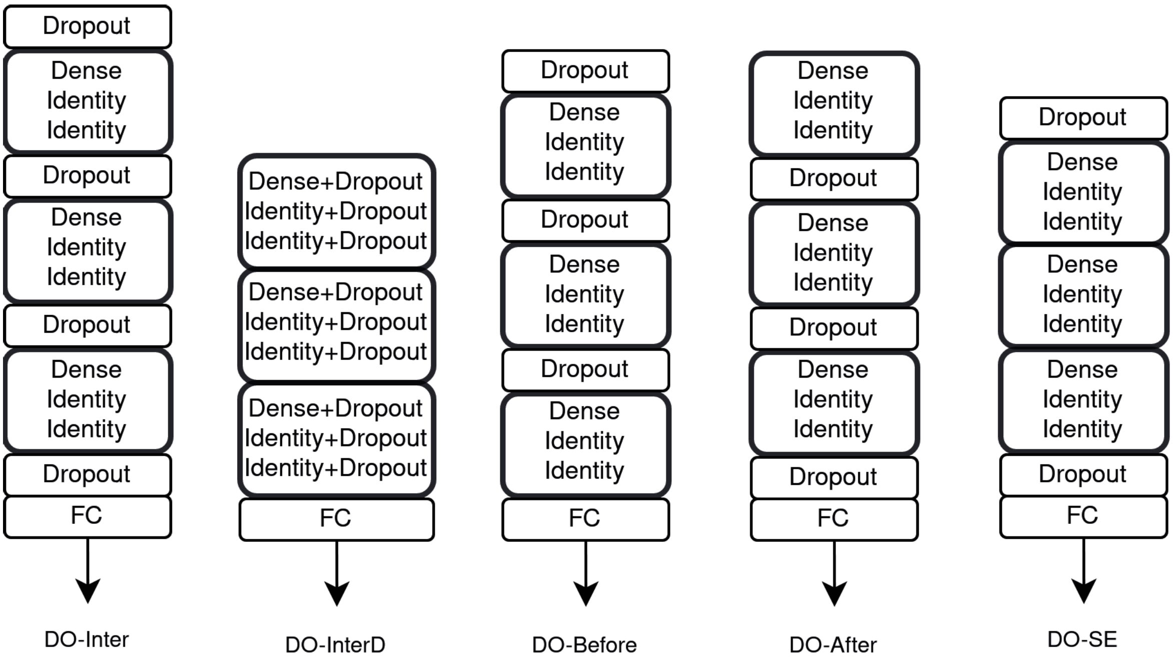


Figure 4. Different dropout placements in the Regression ResNet.

Table 5. Result of Dropout position variants on SafeVS dataset.

	RMSE	**PICP (1)**	**PICP (2)**	**PICP (3)**	**Mean PICP**
Baseline (DO-Dense)	25.0269	0.6418	0.8357	0.8577	0.7784
DO-Inter	24.8448	0.7742	0.9456	0.963	0.8943
DO-InterD	28.2758	0.6544	**0.9969**	**0.9969**	0.8827
DO-Before	19.5191	0.8209	0.8736	0.8466	0.847
DO-After	**15.1608**	0.8158	0.9049	0.9498	0.8902
DO-SE	18.6065	**0.8646**	0.937	0.95	**0.9172**

Table 6. Result of dropout position variants on Kin8nm and Protein dataset.

	Kin8nm		**Protein**	
	RMSE	**PICP**	**RMSE**	**PICP**
Baseline (DO-Dense)	0.2563	0.4872	5.5586	0.5712
DO-Inter	0.2583	**0.6413**	5.7376	**0.6062**
DO-InterD	0.2583	0.6304	5.7965	0.5926
DO-Before	0.2572	0.5	5.4146	0.5634
DO-After	**0.2556**	0.5761	**5.2859**	0.5809
DO-SE	0.2783	0.4239	5.5219	0.575

Overall, the baseline does not outperform the other Dropout variants, placing a Dropout layer after every weight layer in the RRN is unnecessary. It is consistent with the findings in [2, 12, 3]. Considering RMSE as the criterion for model selection, DO-After has the best performance. Considering PICP as the criterion for model selection, DO-Inter has the best performance on Kin8nm dataset and Protein dataset and second best mean PICP in SafeVS dataset. Although DO-SE has the best mean PICP in SafeVS dataset, its performances in Kin8nm and Protein can not compete with DO-Inter.

5 Conclusion

In this paper, we perform an empirical study on evaluating the impact of different dropout placements in a modified ResNet for regression tasks. The MC Dropout technique places a dropout layer after each weight layer in a DNN, which is equivalent to an approximation to deep Gaussian process. Using MC Dropout can easily convert a DNN to a Bayesian DNN, however, the regularization will be too strong due to the deep network architecture. This will lead to a very long training process. Adding fewer dropout layers to a DNN has been shown to be effective to convert the DNN to Bayesian DNN. ResNet was developed for computer vision tasks, it can also be modified to be applied in univariate and multivariate regression tasks. We study the impact of different dropout placements in the Regression ResNet. We find out that placing dropout layers after each FC layer does not outperform the other dropout placement variants. Placing dropout layers after and between the residual blocks have the best RMSE and PICP among all tested variants.

References

[1] Alex Kendall and Roberto Cipolla. Modelling uncertainty in deep learning for camera relocalization. In 2016 IEEE International Conference on Robotics and Automation (ICRA), pages 4762–4769. IEEE, 2016.

[2] Vijay Badrinarayanan Alex Kendall and Roberto Cipolla. Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. In Gabriel Brostow Tae-Kyun Kim, Stefanos Zafeiriou and Krystian Mikolajczyk, editors, em Proceedings of the British Machine Vision Conference (BMVC), pages 57.1–57.12. BMVA Press, September 2017.

[3] Abhijit Guha Roy, Sailesh Conjeti, Nassir Navab, Christian Wachinger, Alzheimer's Disease Neuroimaging Initiative, et al. Bayesian quicknat: model uncertainty in deep whole-brain segmentation for structure-wise quality control. NeuroImage, 195:11–22, 2019.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In Advances in Neural Information Processing Systems, pages 1097–1105, 2012.

[5] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.

[6] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1–9, 2015.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 770–778, 2016.

[8] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 28. Curran Associates, Inc., 2015.

[9] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016.

[10] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In European Conference on Computer Vision, pages 21–37. Springer, 2016.

[11] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 3431–3440, 2015.

[12] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(12):2481–2495, 2017.

[13] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE Transactions on Pattern Analysis and Machine Intelligence, 40(4):834–848, 2017.

[14] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1653–1660, 2014.

[15] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In Proceedings of the IEEE International Conference on Computer Vision, pages 2938–2946, 2015.

[16] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In Proceedings of the IEEE International Conference on Computer Vision, pages 1521–1529, 2017.

[17] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. It's written all over your face: Full-face appearance-based gaze estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, July 2017.

[18] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. Mpiigaze: Real-world dataset and deep appearance-based gaze estimation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 41(1):162–175, 2019.

[19] Seonwook Park, Adrian Spurr, and Otmar Hilliges. Deep pictorial gaze estimation. In Proceedings of the European Conference on Computer Vision, pages 721–738, 2018.

[20] Rajeev Ranjan, Vishal M Patel, and Rama Chellappa. Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 41(1):121–135, 2017.

[21] Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun. Cascaded pyramid network for multi-person pose estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 7103–7112, 2018.

[22] Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. In Proceedings of the European Conference on Computer Vision, pages 466–481, 2018.

[23] George Papandreou, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, and Kevin Murphy. Towards accurate multi-person pose estimation in the wild. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 4903–4911, 2017.

[24] Stéphane Lathuilière, Pablo Mesejo, Xavier Alameda-Pineda, and Radu Horaud. A comprehensive analysis of deep regression. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2019.

[25] Wei-Yin Loh. Classification and regression trees. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 1(1):14–23, 2011.

[26] Harris Drucker, Christopher JC Burges, Linda Kaufman, Alex J Smola, and Vladimir Vapnik. Support vector regression machines. In Advances in Neural Information Processing Systems, pages 155–161, 1997.

[27] Dipendra Jha, Logan Ward, Zijiang Yang, Christopher Wolverton, Ian Foster, Wei-keng Liao, Alok

Choudhary, and Ankit Agrawal. Irnet: A general purpose deep residual regression framework for materials discovery. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 2385–2393, 2019.

[28] Dongwei Chen, Fei Hu, Guokui Nian, and Tiantian Yang. Deep residual learning for nonlinear regression. Entropy, 22(2):193, 2020.

[29] Lianfa Li, Ying Fang, Jun Wu, Jinfeng Wang, and Yong Ge. Encoder-decoder full residual deep networks for robust regression and spatiotemporal estimation. IEEE Transactions on Neural Networks and Learning Systems, 2020.

[30] David JC MacKay. A practical bayesian framework for backpropagation networks. Neural Computation, 4(3):448–472, 1992.

[31] Alex Graves. Practical variational inference for neural networks. In Advances in Neural Information Processing Systems, pages 2348–2356, 2011.

[32] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. arXiv preprint arXiv:1505.05424, 2015.

[33] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In Maria Florina Balcan and Kilian Q. Weinberger, editors, Proceedings of The 33rd International Conference on Machine Learning, volume 48 of Proceedings of Machine Learning Research, pages 1050–1059, New York, New York, USA, 20–22 Jun 2016. PMLR.

[34] David Krueger, Chin-Wei Huang, Riashat Islam, Ryan Turner, Alexandre Lacoste, and Aaron Courville. Bayesian hypernetworks. arXiv preprint arXiv:1710.04759, 2017.

[35] Christos Louizos and Max Welling. Multiplicative normalizing flows for variational bayesian neural networks. arXiv preprint arXiv:1703.01961, 2017.

[36] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 15(1):1929–1958, 2014.

[37] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167, 2015.

[38] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In ICML, 2010.

[39] L. Shi, C. Copot, and S. Vanlanduit. A deep regression model for safety control in visual servoing applications. In 2020 Fourth IEEE International Conference on Robotic Computing (IRC), page preprint, 2020.

[40] Tim Pearce, Alexandra Brintrup, Mohamed Zaki, and Andy Neely. High-quality prediction intervals for deep learning: A distribution-free, ensembled approach. In International Conference on Machine Learning, pages 4075–4084, 2018.

Lei Shi received his Bachelor degree in Electrical Engineering and Automation from Shanghai Maritime University, Shanghai, China, in 2006, and Master degree in Mechatronics from Tallinn University of Technology, Tallinn, Estonia, in 2017. He is currently pursuing Ph.D. degree in InViLab, University of Antwerp, Belgium. His research interests include robotics, artificial intelligence, computer vision and eye tracking.

Dr. ir. **Cosmin Copot** received his M.Sc. and M.E. degrees in systems engineering from Technical University of Iasi, Romania, in 2007, and 2008, respectively. He performs his master thesis as Erasmus student at Ghent University (Carpet Wear Classification using Support Vector Machine). In 2011 he received Ph.D. degree from Technical University of Iasi on control techniques for visual servoing systems. In 2012 he started at Ghent University as a post-doctoral researcher within the department of Electrical energy, Systems and Automation. Beginning with 1 December 2015 he is employed at University of Antwerp as doctor assistent within the Electromechanics department. Since 2008, he has published over 100 papers in technical journals and conference proceedings and has served as reviewer for multiple journals and conferences. He has been involved in the organization of several workshops and conferences (the latest is the IFAC conference on PID Control held in Ghent, 9-11 May 2018, www.pid18.ugent.be). He is Associate Editor of International Journal of Advanced Robotic Systems, Frontiers in Control Engineering (Control and Automation Systems section). He is Topic Editor of journal Robotics (MDPI) and he has served as reviewer for multiple international pear review journals and conferences. His research interests include robotics, mechatronic systems, visual servoing systems and control engineering.

Steve Vanlanduit obtained his Ph.D. entitled "High spatial resolution modal analysis" at the Vrije Universiteit Brussel (VUB) in 2001. In 2003 he was appointed as professor in the department of Mechanical Engineering of VUB. In 2014 he became head of the Department of Electromechanical Engineering of the University of Antwerp, and since 2020 he is vice-dean Research of the Faculty of Applied Engineering. The research of prof. Vanlanduit is focused on laser and camera based optical measurement techniques. He has authored over 200 journal publications on the use of optical measurement techniques in different domains (flow measurement, vibration engineering, robotics, quality control, materials inspection, etc.). He is editor of the scientific journals Measurement (Elsevier), Measurement: Sensors (Elsevier), Sensors (MDPI) and Metrology (MDPI). Steve is actively involved in the organization of several international conferences on optical measurement techniques.