published in: International Conference on Image Processing (ICIP) 2017

OPTIMISTIC AND PESSIMISTIC NEURAL NETWORKS FOR OBJECT RECOGNITION

Rene Grzeszick *Sebastian Sudholt* *Gernot A. Fink*

TU Dortmund University, Germany
email: {rene.grzeszick,sebastian,sudholt,gernot.fink}@tu-dortmund.de

ABSTRACT

In this paper the application of uncertainty modeling to convolutional neural networks is evaluated. A novel method for adjusting the network's predictions based on uncertainty information is introduced. This allows the network to be either optimistic or pessimistic in it's prediction scores. The proposed method builds on the idea of applying dropout at test time and sampling a predictive mean and variance from the network's output. Besides the methodological aspects, implementation details allowing for a fast evaluation are presented. In the evaluation on the ILSVRC2014 and VOC2011 datasets it will be shown that modeling uncertainty allows for improving the performance of a given model purely at test time without any further training steps.

Index Terms— Deep Neural Networks, Dropout, Output Modeling

1. INTRODUCTION

Convolutional neural networks (CNNs) show state-of-the-art results in many computer vision applications. Their usage includes scene [1] and object classification [2, 3], object detection [4], scene parsing [5], face recognition [6], medical imaging [7] and many more. Lately, their applications even extended to non-vision tasks like the recognition of acoustic scenes [8] or activity recognition [9].

While CNNs work well in practice, it has been shown that they are typically over-confident in their predictions [10]. It is possible to generate arbitrary images that are nevertheless classified with high scores after the softmax computation at the network's output layer. There is no measure of uncertainty associated with their output. Compared to many traditional pattern recognition approaches, this is a major shortcoming [11, 12]. The most prominent example are Bayesian models [11]. A typical approach is, for example, model averaging where predictions are made by a set of plausible models and their predictions are integrated into a single representative one [11, 13]. This allows for computing uncertainty based on a predictive mean and variance. Uncertainty is often similarly modeled in regression tasks. Bayesian approaches and also Support Vector Regressors allow for computing confidence bands based on a variance estimation [12, 14] which results in an interval around the regressed values.

The issue of uncertainty modeling for Deep Neural Networks has recently been addressed in [15, 16]. In [16] it is argued that applying dropout resembles the training of a set of different network models. Dropout is a mechanism that is frequently used at training time in order to avoid overfitting [17]. A random set of neurons is dropped from the network so that multiple paths through the network are learned. This can also be thought of as learning an ensemble of different classifiers that are based on different combinations of neurons. The number of classifiers in the ensemble is exponential in the number of neurons and all networks make use of heavy parameter sharing [17]. In [16] it is further argued that a forward pass through the network in which the weights are divided by two is an approximation of model averaging for multilayer perceptrons. This approach is then combined with maxout neurons. A maxout neuron simply returns the maximum activation from a set of inputs. A maxout network learns to have roughly the same output regardless which inputs are dropped and, therefore, interacts well with the dropout paradigm.

In [15] a similar approach has been introduced. A connection has been drawn between deep neural networks and Gaussian processes [18]. Without changing the neural network, it can be shown that applying dropout to each layer can theoretically be interpreted as a Bayesian approximation of Gaussian processes. Thus, a predictive mean and variance can be computed from the results of multiple forward passes through the network where different neurons are dropped which then allows for associating uncertainty with the network's output. The computed predictive mean is basically another approach to model averaging [11] and improves the performance compared to the implicit model averaging that is performed by computing a single forward pass.

The contribution of this paper is an extension of the theoretical model provided in [15]. Based on this theoretical model, optimistic or pessimistic network behavior is defined. Implementation details, providing a fast evaluation of neural networks under the given model are shown. An evaluation on two object recognition tasks, namely the ILSVRC2014 and the VOC2011 dataset, is provided.

2. REVIEW OF TEST DROPOUT

In the following section the idea presented in [15] is briefly reviewed. It has been shown that a deep neural network with

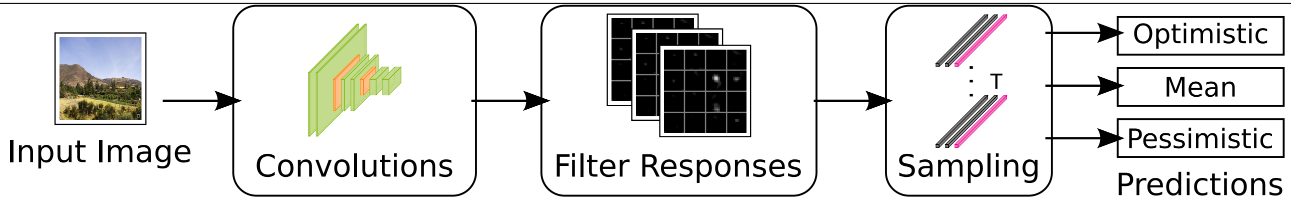

Fig. 1. Overview of the proposed architecture. The convolution layers of the CNN are evaluated in a single forward pass, computing the input features for the fully connected part. Multiple passes through the fully connected part using dropout are then used for computing a robust prediction.

dropout applied before every weight layer is mathematically equivalent to an approximation of a deep Gaussian process. This model allows for the computation of a predictive mean and variance.

Let W_i^j be a binary variable for the i^{th} node in layer j. Each binary variable takes the value 1 with probability p_j for the network's layer j. It is dropped out if the binary variable is set to 0. Given T repetitions and J layers, a random set of variables $\omega_t = \{\boldsymbol{W}\}_i^J$ is randomly drawn for each repetition t. For the input value x and the prediction $\hat{y}_t(x, \omega_t)$ at repetition t, the predictive mean

$$E(y) \approx \tau^{-1}\boldsymbol{I}_C + \frac{1}{T}\sum_{t=1}^{T}\hat{y}_t(x,\omega_t) \tag{1}$$

and predictive variance

$$Var(y) \approx \tau^{-1}\boldsymbol{I}_C + \frac{1}{T}\sum_{t=1}^{T}\hat{y}_t(x,\omega_t)^T\hat{y}_t(x,\omega_t) - E(y)^TE(y) \tag{2}$$

can be computed with $\boldsymbol{I}_C$ denoting a vector of ones of length C, which equals the number of classes at the output layer of the network. Furthermore, the standard deviation is denoted by $\sigma(y) = \sqrt{Var(y)}$. In this formulation τ refers to the model precision:

$$\tau = \frac{pl^2}{2N\lambda} \tag{3}$$

with λ being the weight-decay parameter, l being the prior length scale, N being the number of the input samples and p being the inverse of the dropout probability of the network (c.f. [15]). This reasoning can be applied to every neural network using standard dropout.

3. OPTIMISTIC AND PESSIMISTIC NETWORKS

Following this theoretical model, a novel modeling of network behavior is proposed. Using confidence intervals, a neural network can either be optimistic or pessimistic in its predictions, as illustrated in Fig. 1.

3.1. Confidence Intervals

Under the assumption that the output of a deep Gaussian process and, therefore, the output of the neural network is normal distributed, it is possible to compute a confidence interval for the predictions based on the mean and standard deviation:

$$[E(y) - z_{1-\frac{\alpha}{2}}\frac{\sigma(y)}{\sqrt{T}}\,;\,E(y) + z_{1-\frac{\alpha}{2}}\frac{\sigma(y)}{\sqrt{T}}] \tag{4}$$

with $z_{1-\frac{\alpha}{2}}$ being the $1-\alpha/2$ quantile of the normal distribution. Choosing an appropriate value for α, a confidence interval can be derived (i.e. for 99% certainty, $\alpha = 0.01$).

Note that given the mean and variance, i.e. from a validation set, a guaranteed confidence interval can be computed based on the number of runs T:

$$T = \left(z_{(1-\frac{\alpha}{2})}\frac{\sigma(y)}{E(y)}\right)^2 \quad . \tag{5}$$

Adjusting the parameter T, it can be said that the performance of the model will be within the respective confidence interval with a probability of α.

3.2. Definition of Optimistic and Pessimistic Behavior

The result that is obtained from computing the confidence interval provides an upper and lower bound for the prediction which is dependent on the confidence level. Hence, the true mean is between these bounds with probability $1-\alpha/2$. The goal is to use this information in order to improve the results of a convolutional neural network.

While using dropout and computing the predictive mean should already be more robust than a single pass through the network (cf. [19]), it does not consider the variance that is obtained from scoring the sample multiple times. Given two samples x_1 and x_2 with the same mean but different confidence intervals it can be argued that one should be scored better than the other. This gives rise to optimistic or pessimistic network behavior, which is defined by

$$E_o(y) = E(y) + z_{1-\frac{\alpha}{2}}\frac{\sigma(y)}{\sqrt{T}} \quad \text{and} \tag{6}$$

$$E_p(y) = E(y) - z_{1-\frac{\alpha}{2}}\frac{\sigma(y)}{\sqrt{T}} \tag{7}$$

respectively. By reasoning that dropout models an ensemble, which is generated by multiple paths through the network [17], the two behaviors can be compared to a decision rule of ensemble classifiers such as the maximum rule (c.f. [20]). The optimistic behavior gives more weight to the numerically larger output of the network. This is essentially a bias toward a minority of votes in the ensemble determining neurons in the output layer to be active. The pessimistic formulation favors a minority of votes which are in favor of a neuron in the output layer being inactive and thus to numerically smaller values. In the experiments it will be shown that modifying the network to output a more optimistic or more pessimistic score can improve the results.

3.3. Fast Optimistic and Pessimistic Networks

Theoretically, processing T forward passes would come at a T times higher computation time. Although in practice parallelization using a larger batch size allows for a slightly faster processing, the increased computational cost makes this approach unsuitable for many practical applications.

In typical CNN architectures dropout is only applied during training and most classification networks restrict themselves to dropout in the last weight layers in order to avoid overfitting [2, 3]. Modeling this for the sampling at test time can be used as an advantage. Following the idea of [4] it is assumed that the network consists of two parts: The first part is a fixed set of convolution layers. The activations of these layers are considered as an intermediate feature representation and fixed for a given input image. The second part is the neural network using dropout. The resulting architecture is illustrated in Fig. 1. The convolutions are computed in a single forward pass, then T passes through the fully connected part are computed. The dropout value for each of the j layers of the deep neural network is set to a single value p_{drop}.

4. EVALUATION

The goal of the evaluation is to show the performance improvement that is achieved by evaluating the same models using dropout at test time. A single forward pass without dropout is compared with multiple passes using dropout and computing a predictive mean or the output of the proposed optimistic or pessimistic network behavior. No additional training step is performed. The evaluation is performed on the ILSVRC2014 object classification task and the VOC2011 object prediction task using a VGG16 network architecture.

4.1. ILSVRC 2014

The publicly available VGG16 model [3] has been evaluated as a baseline on the validation set of the ILSVRC2014 classification task [21]. Dropout is applied to the first two fully connected layers (*fc-6* and *fc-7*). In [3] multiple crops which

Network	T	p_{drop}	top-1 error	top-3 error	top-5 error
VGG16 (*) [3]	-	-	28.6%	13.9%	10.1%
Mean	10	0.1	**27.1%**	**12.5%**	**8.8%**
Optimistic	10	0.1	27.3%	12.6%	8.9%
Pessimistic	10	0.1	27.2%	12.9%	9.4%
Mean	100	0.1	27.1%	12.5%	8.9%
Optimistic	100	0.1	27.1%	12.6%	8.9%
Pessimistic	100	0.1	27.1%	12.5%	8.9%
Mean	10	0.5	27.2%	12.8%	9.0%
Optimistic	10	0.5	27.3%	13.1%	9.3%
Pessimistic	10	0.5	27.6%	17.7%	16.5%
Mean	100	0.5	26.9%	**12.4%**	**8.8%**
Optimistic	100	0.5	26.9%	12.5%	8.9%
Pessimistic	100	0.5	**26.8%**	**12.4%**	**8.8%**

Table 1. Error rates for different values of T and p_{drop} on the ILSVRC 2014 classification on the validation set. (*) Results were re-evaluated using the publicly available model.

are extracted at different image scales are evaluated by the CNN. Moreover, the crops are larger than the CNNs input size and then densely evaluated by reshaping the fully connected layers to 7×7 and 1×1 convolution layers. The authors stated that extracting multiple crops at different image scales only slightly improves the performance (150 crops from multiple scales improved the accuracy by 1.3%), but at the cost of a much higher computation time [3]. This evaluation is therefore omitted. Still the dense evaluation at a single scale of 256px (shortest side length) is performed. Instead of using the average, the maximum activation after each of the softmax computations from the dense sampling is used.

The network has then been evaluated using the mean, optimistic and pessimistic behavior with different parameters for p_{drop} and T. These parameters are chosen empirically for the experiments. The error rates for the top k results are shown in Tab. 1. The top-1 error rate can be improved by 1.8% and the top-5 error rate can be improved by 1.3% compared to a single forward pass. However, there is not much difference between the different network behaviors for this task. A permutation tests (cf. [22]) for top-1, top-3 and top-5 error showed a highly significant difference between all of the pessimistic net's results and the baseline results with a p-value of $p < 0.01$ each.

Not surprisingly the results converge when increasing the number of runs T. Note that with a higher dropout value p_{drop} the number of runs T that are required to improve the result increases. This can be explained by the fact that the variance will be increased the more information is dropped from the model. Overall the results are improved when using a higher dropout ratio. A crucial point is that this improved performance comes at the cost of a higher runtime. Therefore, the computation time for T passes through the network has been evaluated with the results shown in Fig. 2. The fast architecture is compared with a naive approach computing T forward

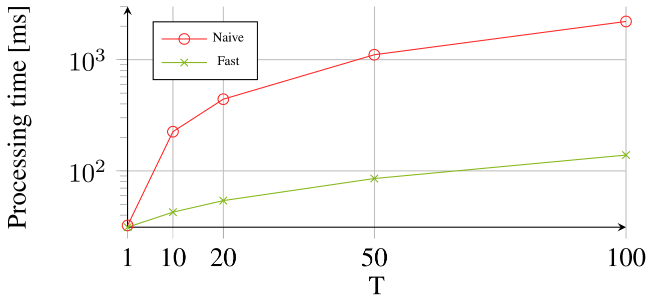

Fig. 2. Computation time in ms on the ILSVRC2014 benchmark on a TitanX using dense sampling with mirroring and a scale size of 256. The runtime in relation to the sampling parameter T is evaluated.

passes through the complete network. Note that the computational overhead created by dropout is neglectable. While the naive approach is almost linear in time, the computation time can be reduced by a large margin when evaluating the fast implementation that pre-computes the filter responses.

4.2. VOC2011 - Multiclass prediction

On the VOC2011 dataset the presence of multiple objects in a scene is predicted [23]. Therefore, a fully convolutional VGG16 multilabel architecture has been evaluated. The fully connected layers of the VGG16 architecture have been replaced by two 3×3 convolution layers, one layer with 512 filters and one layer with exactly one filter per object class. Dropout is applied to the first of these convolution layers. The convolution layers are then followed by a global max pooling. Instead of predicting one label with a softmax layer, a sigmoid layer is used in order to simultaneously derive pseudo-probabilities for all object classes. The training is performed by computing the *cross entropy loss* [24]. The filter functions are pre-trained on ImageNet and the presence prediction is then trained on the VOC2011 training set. 500 000 samples have been created using data augmentation, including Gaussian noise ($\sigma = 0.02$), random translations (up to 5%) and rotations (up to $\leq 10°$). A batch size of 256 has been used for training with 2 000 iterations using a learning rate of 10^{-3}. The learning rate has then been reduced to 10^{-4} computing 2 000 more iterations. The images from the VOC2011 dataset have been re-scaled so that the shortest side has a length of 512px. Note that this is a rather large image size compared to many other tasks [2, 3]. As the network is fully convolutional this image size is beneficial since the objects are larger in terms of the absolute surface area and can be detected more easily by the large receptive field of the network's filters.

The mAP over all classes is computed and the results are shown in Tab. 2. As the predictions are evaluated independently of each other (i.e. two classes can be predicted with a probability of one), the proposed behavior modeling has more influence than in case of a softmax classifier. Also, the mAP is evaluated based on varying thresholds for each of the 20 classes in the VOC dataset. As a result, the influence of the different network behaviors is clearly visible. The mAP can be improved by up to 1% when using an optimistic behavior instead of the predictive mean and by 2.5% compared to the plain evaluation.

Network	T	p_{drop}	mAP
Multiclass baseline	-	-	72.8%
Optimistic	100	0.5	**75.3%**
Mean	100	0.5	74.8%
Pessimistic	100	0.5	74.6%

Table 2. Mean average precision for the presence prediction on the VOC2011 dataset.

4.3. Discussion

Both of the networks use a maximum pooling in the final layer. The ILSVRC network pools the maximum response of the dense sampling and the VOC multilabel architecture has a pooling layer that is applied after the convolution layers. It is worth noting that these are important design decisions when applying dropout. This also confirms the findings in [16] where it is argued that maxout layers and dropout in training interact well. A reason is that applying average pooling smooths the differences that are generated by choosing different paths through the network.

5. CONCLUSION

In this paper the application of uncertainty modeling to convolutional neural networks has been evaluated. The proposed method builds on the idea of applying dropout at test time and sampling a predictive mean and variance from the network's output which in turn also allows for adjusting the networks predictions to be more optimistic or more pessimistic. In general the proposed method can be compared to evaluating an ensemble classifier within a single network architecture with the different behaviors representing different decision rules. It could be shown that the results of a given model can be improved for object recognition tasks on the ILSVRC2014 and VOC2011 dataset[1]. On the VOC2011 dataset the results for predicting the presence of multiple objects at a time can be improved by as much as 2.5% using an optimistic evaluation scheme and dropout at test time without any further training.

Acknowledgement

This work has been supported by the German Research Foundation (DFG) within project Fi799/9-1 ('Partially Supervised Learning of Models for Visual Scene Recognition').

[1]An extended version of this paper is available at arXiv: https://arxiv.org/abs/1609.07982

6. REFERENCES

[1] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning Deep Features for Scene Recognition Using Places Database," in *Advances in neural information processing systems*, 2014, pp. 487–495.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[3] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *CoRR*, vol. abs/1409.1, 2014.

[4] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 91–99.

[5] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, "Learning hierarchical features for scene labeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1915–1929, 2013.

[6] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *British Machine Vision Conference*, 2015, pp. 41.1–41.12.

[7] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015, pp. 234–241.

[8] H. Phan, L Hertel, M. Maass, P. Koch, and A. Mertins, "CNN-LTE: a Class of 1-X Pooling Convolutional Neural Networks on Label Tree Embeddings for Audio Scene Recognition," *arXiv preprint arXiv:1607.02303*, 2016.

[9] N. Hammerla, S. Halloran, and T. Ploetz, "Deep, convolutional, and recurrent models for human activity recognition using wearables," in *Proc. Int. Joint Conference on Artificial Intelligence (IJCAI)*, 2016, pp. 1533–1540.

[10] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2015, pp. 427–436.

[11] D. Barber, *Bayesian reasoning and machine learning*, Cambridge University Press, 2012.

[12] K. De Brabanter, P. Karsmakers, J. De Brabanter, J. AK Suykens, and B. De Moor, "Confidence bands for least squares support vector machine classifiers: A regression approach," *Pattern Recognition*, vol. 45, no. 6, pp. 2280–2287, 2012.

[13] I. Park, H. K. Amarchinta, and R. V. Grandhi, "A bayesian approach for quantification of model uncertainty," *Reliability Engineering & System Safety*, vol. 95, no. 7, pp. 777–785, 2010.

[14] C. M. Bishop and S. Q. Cazhaow, "Regression with input-dependent noise: A bayesian treatment," in *Advances in Neural Information Processing Systems 9: Proceedings of the 1996 Conference*. MIT Press, 1997, vol. 9, pp. 347–353.

[15] Yarin Gal and Zoubin Ghahramani, "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning," in *Proc. Int. Conf. on Machine Learning (ICML)*, 2016, pp. 1050–1059.

[16] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. C. Courville, and Y. Bengio, "Maxout Networks," *ICML (3)*, vol. 28, pp. 1319–1327, 2013.

[17] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks From Overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[18] C. E. Rasmussen, *Gaussian processes for machine learning*, Citeseer, 2006.

[19] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Insights and applications," in *Deep Learning Workshop, Int. Conf. on Machine Learning (ICML)*, 2015.

[20] L. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms*, John Wiley & Sons, 2004.

[21] O. Russakovsky, J. Deng, H. Su, Krause. J., S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[22] P. Good, *Permutation Tests - A Practical Guide to Resampling Methods for Testing Hypothesis*, Springer, 2 edition, 2000.

[23] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2011 (VOC2011) Results," Tech. Rep., 2011.

[24] E. M. Hand and R. Chellappa, "Attributes for Improved Attributes: A Multi-Task Network for Attribute Classification," *arXiv*, 2016.