Empirical study on label smoothing in neural networks

Mauro Mezzini
Roma Tre University
Via Castro Pretorio, 20
00182, Rome, Italy
mauro.mezzini@uniroma3.it

ABSTRACT

Neural networks are now day routinely employed in the classification of sets of objects, which consists in predicting the class label of an object. The softmax function is a popular choice of the output function in neural networks. It is a probability distribution of the class labels and the label with maximum probability represents the prediction of the neural network, given the object being classified. The softmax function is also used to compute the loss function, which evaluates the error made by the network in the classification task. In this paper we consider a simple modification to the loss function, called label smoothing. We experimented this modification by training a neural network using 12 data sets, all containing a total of about 1.5×10^6 images. We show that this modification allow a neural network to achieve a better accuracy in the classification task.

Keywords

neural networks; label smoothing; regularization; softmax; visual domain decathlon challenge;

1 INTRODUCTION

One of the most important and studied problem in artificial intelligence and computer vision is the *object classification problem* [1]. In this problem we have a set of objects, which can be images, speech, sounds and so on and we may suppose that the numerical representation of an object is a n-dimensional vector $x \in \mathbb{R}^n$. There exists a function $f : \mathbb{R}^n \to \{1,2,\ldots,K\}$, that associate to each object x a *class* $f(x)$, where $K \in \mathbb{N}$, is the number of different classes (e.g. x is an image and $f(x)$ is the subject of the image). The solution to the classification problem consists in determining a function equivalent to f. Neural networks (NN) recently achieved a very high accuracy in the classification tasks of images [2, 3, 4, 5, 6, 7]. The classification problem can also be related to other tasks such as object detection [8], image segmentation [9].

The output of the NN, is a highly complex non-linear function $z(x) \in \mathbb{R}^K$, which, in turn, is dependent on the parameters of the NN. This function is used to obtain a probability distribution of the class j given the object x, called the *softmax function*. If $z_j(x)$ is the j-th component of $z(x)$, $j = 1,\ldots,K$ then the *softmax* function is given by

$$p(j|x) = \frac{e^{z_j(x)}}{\sum_{i=1}^{K} e^{z_i(x)}} \tag{1}$$

The goal is then to obtain a softmax function such that $f(x) = \text{argmax}_j p(j|x)$. In order to do this a NN is trained using a *training set D*. Each element of D is a couple $(x_i, f(x_i))$, $x_i \in \mathbb{R}^n$, $i = 1,2,\ldots,N$ and N is the size of the training set. In order to determine the accuracy of the network, a *loss function* is employed, that assigns to each object x a quantitative measure of the error the NN made in classifying x. Often, the loss function takes the negative logarithm of the softmax function as follows

$$L(x) = -\log p(f(x)|x) \tag{2}$$

and, in order to improve the NN, the gradients (with respect to the parameters of the NN, hidden in the definition of z) of the mean of the loss function over all training set

$$L = -1/N \sum_{i=1}^{N} \log p(f(x_i)|x_i) \tag{3}$$

are back propagated along all the layers of the NN. The negative logarithm of the softmax function can be interpreted as the cross-entropy between the probability distribution given in 1 and the true probability $q(j|x)$ of the class j given the object x.

$$\mathbb{H}(q,p) = -\sum_{i=1}^{K} q(i|x) \log p(i|x) \tag{4}$$

https://doi.org/10.24132/CSRN.2018.2802.25

ISBN 978-80-86943-41-1

In most of the literature regarding the cross-entropy loss function in NN, the probability distribution q is taken as follows

$$q(j|x) = \begin{cases} 1 & \text{if } f(x) = j \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

and having chose q as in (5), and substituting it in (4), one obtains (2) which is commonly referred to as the *categorical cross-entropy*. More in general the loss function can be interpreted as the Kullback-Leibler divergence [10] between the true probability q and p, denoted $D(q||p)$, that is

$$D(q||p) = \sum_{i=1}^{K} q(i|x) \log \frac{q(i|x)}{p(i|x)} \tag{6}$$

Note that the function given in (6) will be equivalent to (4) when we choose q as in (5).

The reason that the probability q is taken as in (5) is motivated by the obvious fact that the training set D is prepared before the actual training and who prepares the training set knows "for sure" (a priori) that the object x has a class $f(x)$. However, even who prepares the training set is subject of error and there is a degree of uncertainty for some images. For example, the CIFAR-10 [11] training set is composed by 50000 images of resolution $32 \times 32 \times 3$ and there are 10 different classes. Some images appears as a confuse blob of green and brown color. A person that looks at one of these images barely recognizes it as a frog and the classification of the image is based more on excluding that the image could not be an airplane, a car and no one of the other classes, rather than based on a certainty that the image is a frog.

One of the main problem in training a neural network is the over-fitting. The over-fitting of a neural network, is the problem that prevent the network to generalize and obtain accurate prediction on samples not contained in the training set.

Moreover and worse, one finds that, sometimes, the training process spends a lot of time to reduce the loss (3) without even achieve better fitting on the training set. With a close inspection in fact one can observe that a considerable amount of the time is spent, by the training process, on to make better an already good and accurate prediction. Ad example, suppose there are $K = 10$ different classes and take the softmax function computed by a network for an object α as follows: $p(f(\alpha)|\alpha) = 0.19$ and $p(i|\alpha) = 0.09$ for $i \neq f(\alpha)$. With this value the network accurately predicts the class $f(\alpha)$ and the value of the loss function ((3)) for the sample α is $-\log p(f(\alpha)|\alpha) = 1.6607$. However if we assign to the softmax function a different value, for example $p(f(\alpha)|\alpha) = 0.91$ we obtain a loss equal to 0.0943 which is considered better to the training process with respect to the previous value. But this new value does not improve neither the accuracy of the network on the training set nor the accuracy of the network on the test set.

So based on this observation we propose a simple modification to the cross-entropy called label smoothing. The rest of the paper is organized as follows. In Section 2 we present and briefly discuss the related works. In Section 3 we present the modification to the loss function. In Section 4 we present the results of the experiments. In Section 5 we conclude the paper.

2 RELATED WORKS

In literature numerous strategies are used to prevent the over-fitting. Data augmentation is one of the best practices employed [12, 13, 14]. In this case, each sample of the training set is modified. For example an image is manipulated by shifting it, rotating it or by changing the level of its brightness. The manipulated images are added and used in the training set. Dropout technique [15] also has been proved effective in reducing the over-fitting. In this case a random sample of neurons of a layer of the network are dropped out and the forward and backward propagation is made only on the thinned network. Batch normalization [16] is found to be a form of regularization of the network.

In [17] it has been proposed to disturb the loss layer by randomly changing the label of each sample according to a multinulli distribution. In this way the label of a sample can be different from the true label. The results of the experiments made on five different data sets show that the method effectively prevent the network in over-fitting. However the authors used in their experiment a LeNet-like [18] or AlexNet [2] models which have a shallow architecture (only 5 layers deep) and they are somewhat "older" models. In our experiments we make use of the recent ResNet model [3, 19] and a much deeper architecture on 12 different data sets.

In [20] label smoothing methods are proposed that modified the loss function by using its own prediction distribution.

The concept of label smoothing regularizations (LSR) has been investigated in [21]. They established the ground-truth probability distribution as

$$q(j|x) = (1-\varepsilon)\delta_{j,f(x)} + \varepsilon u(j)$$

where $\delta_{j,f(x)} = 1$ if $j = f(x)$ and $\delta_{j,f(x)} = 0$ otherwise, and $u(j)$ is a fixed distribution. By setting $\varepsilon = 0.1$ and $u(j) = 1/K$ they reported a gain of 0.2% in the accuracy of the Inception model on the ImageNet dataset [22].

In [23] it is proposed a regularization technique which consist in the following. If the network is over-fitting this means that the entropy of the softmax, given by

$$\mathbb{H}(p) = -\sum_{i=1}^{K} p(i|x) \log p(i|x))$$

is low. Therefore the idea is to penalize the loss function by adding a negative entropy to the loss function as follows

$$L'(x) = -\log p(f(x)|x) - \beta \mathbb{H}(p);$$

where the parameter β control the strength of the penalization.

All the above mentioned works experimented the proposed methods either on one or two benchmark data sets or are carried on using somewhat "older" NN models. the contribution of this paper is to extend the experimentation on wide range of data sets and using the latest state-of-the-art models.

3 THE LABEL SMOOTHING

The softmax function assign to each object x and to each possible class c a value $0 \leq p(c|x) \leq 1$. In predicting the class of the object x, using the NN, we simply take $\hat{c} = \text{argmax}_c p(c|x)$; we do not care which is the value of $p(\hat{c}|x)$, we care only that $\hat{c}$ is the class such that $p(\hat{c}|x) > p(c|x)$ for all $c \neq \hat{c}$; furthermore, in measuring the accuracy of the prediction we do not care how high is the $p(\hat{c}|x)$. For example, suppose there are 10 different classes, that is $K = 10$, and suppose that $p(\hat{c}|x) = 0.2$ and $p(c|x) = 0.8/(K-1)$ for $c \neq \hat{c}$. If $\hat{c} = f(x)$ then we consider the network very accurate in predicting the class of x, even tough 0.2 is very far from 1.0. This is sometimes similar in what a human do in recognize an image. Sometimes we have not the certainty that the class of an image is a deer but we can exclude that it is a dog and it is a horse, so we classify it as a deer.

So, in order to apply this idea we use, as a loss function, the cross-entropy between the softmax function (1) and a probability distribution other than (5). If $0 \leq \gamma \leq 1$ then we may choose q as

$$q(j|x) = \begin{cases} \gamma & \text{if } f(x) = j \\ (1-\gamma)/(K-1) & \text{otherwise} \end{cases} \qquad (7)$$

with the constraint that $\gamma > 1/K$. The above methods can be implemented in existing software almost effortless. We made extensive experiments using state of the art NN for classification on several data sets. We report the results of the experiments in the following section. We found that, in many settings, there is a value of γ that improve the accuracy of the net with respect to the categorical cross-entropy.

4 THE EXPERIMENTS

We used the ten datasets of the Visual Domain Decathlon challenge presented in [24], the MNIST dataset of handwritten digit recognition [18] and the CIFAR-10 dataset. The detailed description of all these datasets is provided in [25], see also [11, 26, 27, 28, 29, 30, 31, 32]. We used Tensorflow framework [33] with Keras high-end library [34] on two Tesla P100-SXM2 GPUs. We trained each dataset with the model ResNet v2 [19] or with the model ResNet v1 [3]. In order to optimize the limited resource of GPU time and memory we choose to implement the ResNet v2 model with 83 layers and the ResNet v1 model with 20 layers. We used the last model only for the dataset ImageNet and MNIST. The optimizer method utilized for training the network is Adam [35], with initial learning rate of 0.001 which is reduced to of a factor 10^{-1} after 80, 120 and 160 epochs and of a factor of 0.5×10^{-3} after 180 epochs, for a total of 200 epochs. Each data set, with the exception of ImageNet data set, has been normalized by subtracting the mean over all the training sample.

Due to limited time [1], the training of ImageNet data set has been stopped after 100 epochs. Furthermore the model ResNeXt having 83 levels it has been trained only for $\gamma = 1$ ad with $\gamma = 0.9$. Other experiments on ImageNet were done by training a 20 layer ResNet v1 for the value of $\gamma \in \{1.0, 0.9, 0.8, 0.5\}$. In addition the minibach size for the imagenet training has been put at 64 training sample. To each image of the data set has been applied a simple data augmentation manipulation consisting in randomly shift the image, horizontally and vertically, up to 10% of the original width and height respectively, followed by a random horizontal flip.

Since the goal of the study is to asses the differences of accuracy of the network for different values of the parameter γ, we did not intend to compare our results to the best state-of-the-art models.

For each data set we tested the value of γ from 1 to a value greater than $1/K + 0.1$ where K is the number of different classes of the data set.

In Table 1 and 2 we report the results of the experiments made on the ten data sets of the Visual Domain Decathlon challenge, the MNIST and CIFAR-10 data sets. The value of γ ranged from 1.0 to $1/K + 0.1$ with step value of 0.1. For the value of γ and for each data set, is reported the best accuracy attained on the validation set. We can see that there are sometimes dramatic improvement in the accuracy in some of the data set of the experiment as reported in Table 3 in which it is compared the value of the accuracy with $\gamma = 1.0$ and the best accuracy obtained among all different values of γ. We can observe that one limitation of our approach is that the value of γ, for which the best accuracy is attained, is not the same on the various data sets making difficult to adapt this method to other cases.

[1] On two Tesla P100-SXM2, each epoch require more than 8000 seconds to terminate

	γ				
Dataset	**0.10**	**0.20**	**0.30**	**0.40**	**0.50**
aircraft	24.82	36.43	39.08	40.82	41.03
cifar100	59.25	66.76	69.06	71.20	71.39
daimlerp.	-	-	-	-	-
dtd	28.21	27.78	25.92	26.98	25.49
gtsrb	99.68	99.78	99.81	99.86	99.83
omniglot	82.87	83.50	84.22	84.75	85.21
svhn	-	92.41	93.05	93.64	93.62
ucf101	64.79	72.78	74.37	75.91	74.88
vgg-flow.	51.03	52.80	50.93	49.66	48.09
ImageN.(a)	-	-	-	-	-
ImageN.(b)	-	-	-	-	35.59
cifar10	-	86.98	88.77	90.33	90.76
mnist	-	99.41	99.46	99.61	99.54

Table 1: For the value of $\gamma \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$ and for each dataset is reported the best accuracy attained on the validation set. The data sets daimlerpedcls, $K = 2$ classes while svhn, cifar10, and mnist have $K = 10$ classes. Therefore there are no data for the value of $\gamma \leq 1/K$ on these data sets. The model (a) used for ImageNet is ResNet v2 with 83 layers trained for 100 epochs, while the model (b) is ResNet v1 with 20 layers trained for 200 epoch. The model used for mnist is ResNet v1 with 20 layers.

	γ				
Data set	**0.60**	**0.70**	**0.80**	**0.90**	**1.00**
aircraft	38.06	41.09	36.10	38.63	34.75
cifar100	71.80	71.88	71.73	71.34	70.83
daimlerp.	99.80	99.83	99.97	99.91	99.98
dtd	25.39	25.23	23.42	23.68	25.28
gtsrb	99.86	99.85	99.82	99.87	99.94
omniglot	85.21	84.61	83.28	82.01	77.95
svhn	93.79	93.95	94.07	94.22	93.82
ucf101	74.37	74.17	74.17	72.17	70.12
vgg-flow.	46.32	48.48	45.83	44.26	42.30
ImageN.(a)	-	-	-	45.93	42.87
ImageN.(b)	-	-	43.00	43.95	44.76
cifar10	91.60	91.95	92.20	92.54	93.31
mnist	99.53	99.55	99.55	99.61	99.56

Table 2: For the value of $\gamma \in \{0.6, 0.7, 0.8, 0.9, 1.0\}$ and for each data set is reported the best accuracy attained on the validation set. The model (a) used for ImageNet is ResNet v2 with 83 layers trained for 100 epochs, while the model (b) is ResNet v1 with 20 layers trained for 200 epoch. The model used for mnist is ResNet v1 with 20 layers.

Data set	**Best**	**Acc. $\gamma = 1$**	**diff %**
aircraft	41.09	34.75	15.41
cifar100	71.88	70.83	1.46
daimlerp.	99.98	99.98	0.00
dtd	28.21	25.28	10.38
gtsrb	99.94	99.94	0.00
omniglot	85.21	77.95	8.52
svhn	94.22	93.82	0.42
ucf101	75.91	70.12	7.63
vgg-flow.	52.80	42.30	19.89
ImageN.(a)	45.93	42.87	7.12
ImageN.(b)	44.76	44.76	0.00
cifar10	93.31	93.31	0.00
mnist	99.61	99.56	0.05

Table 3: The best accuracy achieved compared to the accurcay of the model trained with $\gamma = 1$.

5 CONCLUSIONS

We proposed a simple and easy to implement method of regularization of the model, called label smoothing, and we made extensive experiments on several data sets using state-of-the art very deep models. We showed that this method may be very effective in regularize the model and mitigate over-fitting. Future researches may investigate to mix the method proposed in this paper with the ones proposed in [17, 23], and at the same time, extending the experiments to different models.

6 REFERENCES

[1] Y. LeCun, Y. Bengio, G. E. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444.

[2] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: F. Pereira, C. J. C. Burges, L. Bottou, K. Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 25, Curran Associates, Inc., 2012, pp. 1097–1105.

[3] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778. doi:10.1109/CVPR.2016.90.

[4] C. Szegedy, S. Ioffe, V. Vanhoucke, A. A. Alemi, Inception-v4, inception-resnet and the impact of residual connections on learning, in: S. P. Singh, S. Markovitch (Eds.), Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA., AAAI Press, 2017, pp. 4278–4284.

[5] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in: 2015 IEEE International Conference on Computer Vision

(ICCV), 2015, pp. 1026–1034. doi:10.1109/ICCV.2015.123.

[6] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, X. Tang, Residual attention network for image classification, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Vol. 00, 2017, pp. 6450–6458. doi:10.1109/CVPR.2017.683.

[7] J. Hu, L. Shen, G. Sun, Squeeze-and-excitation networks, CoRR abs/1709.01507.

[8] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, S. Belongie, Feature pyramid networks for object detection, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[9] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, in: N. Navab, J. Hornegger, W. M. Wells, A. F. Frangi (Eds.), Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, Springer International Publishing, Cham, 2015, pp. 234–241.

[10] S. Kullback, R. A. Leibler, On information and sufficiency, Ann. Math. Statist. (1) 79–86. doi:10.1214/aoms/1177729694.

[11] A. Krizhevsky, Learning multiple layers of features from tiny images, Tech. rep. (2009).

[12] J. Salamon, J. P. Bello, Deep convolutional neural networks and data augmentation for environmental sound classification, IEEE Signal Processing Letters 24 (3) (2017) 279–283. doi:10.1109/LSP.2017.2657381.

[13] P. Y. Simard, D. Steinkraus, J. Platt, Best practices for convolutional neural networks applied to visual document analysis, Institute of Electrical and Electronics Engineers, Inc., 2003.

[14] L. Perez, J. Wang, The effectiveness of data augmentation in image classification using deep learning, CoRR abs/1712.04621.

[15] N. Srivastava, G. 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.

[16] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in: ICML, 2015.

[17] L. Xie, J. Wang, Z. Wei, M. Wang, Q. Tian, DisturbLabel: Regularizing CNN on the loss layer, Vol. 2016-December, IEEE Computer Society, 2016, pp. 4753–4762. doi:10.1109/CVPR.2016.514.

[18] Y. L. Cun, B. Boser, J. S. Denker, R. E. Howard, W. Habbard, L. D. Jackel, D. Henderson, Advances in neural information processing systems 2, 1990, Ch. Handwritten Digit Recognition with a Back-propagation Network, pp. 396–404.

[19] K. He, X. Zhang, S. Ren, J. Sun, Identity mappings in deep residual networks, in: Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV, 2016, pp. 630–645.

[20] S. E. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, A. Rabinovich, Training deep neural networks on noisy labels with bootstrapping, CoRR abs/1412.6596.

[21] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, CoRR abs/1512.00567.

[22] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248–255. doi:10.1109/CVPR.2009.5206848.

[23] G. Pereyra, G. Tucker, J. Chorowski, L. Kaiser, G. E. Hinton, Regularizing neural networks by penalizing confident output distributions, CoRR abs/1701.06548.

[24] 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017, IEEE Computer Society, 2017.

[25] Visual domain decathlon, 2017.
URL http://www.robots.ox.ac.uk/~vgg/decathlon/

[26] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, A. Vedaldi, Fine-grained visual classification of aircraft, Tech. rep. (2013). arXiv:1306.5151.

[27] S. Munder, D. M. Gavrila, An experimental study on pedestrian classification, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (11) (2006) 1863–1868. doi:10.1109/TPAMI.2006.217.

[28] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, A. Y. Ng, Reading digits in natural images with unsupervised feature learning, 2011.

[29] M. E. Nilsback, A. Zisserman, Automated flower classification over a large number of classes, in: 2008 Sixth Indian Conference on Computer Vision, Graphics Image Processing, 2008, pp. 722–729. doi:10.1109/ICVGIP.2008.47.

[30] Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition, Neural Networks 32 (2012) 323 – 332, selected Papers from IJCNN 2011. doi:https://doi.org/10.1016/j.neunet.2012.02.016.

[31] B. M. Lake, R. Salakhutdinov, J. B. Tenenbaum, Human-level concept learning through probabilistic program induction 350 (6266) (2015) 1332–1338. doi:10.1126/science.aab3050.

[32] K. Soomro, A. R. Zamir, M. Shah, Ucf101: A dataset of 101 human actions classes from videos in the wild, CoRR abs/1212.0402.

[33] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, X. Zheng, Tensorflow: A system for large-scale machine learning, 2016.

[34] F. Chollet, et al., Keras, https://github.com/keras-team/keras (2015).

[35] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization., CoRR.