Quantifying the Classification Uncertainty of Neural Networks through a Bayesian Perspective

Bedirhan Çaldır, Atılberk Çelebi

CMPE Department, Boğaziçi University

Motivations

- Developing a statistical approach to detect uncertainty of the Neural Network outputs in order to answer how certain a MLE classification is
- Utilizing a Dirichlet distribution to model the uncertainty as the Dempster–Shafer Theory of Evidence suggests
- Increasing the robustness of a Neural Network by identifying the out-of-distribution queries and making it more secure against potential modern tricks such as adversarial attacks

Introduction

The conventional deterministic neural networks have become very popular since they enable us to train top-notch predictors on a wide range of machine learning problems. However, they are trained in order to minimize a prediction loss given a dataset but the trained model is unable to give the level of confidence of its prediction. That means, sometimes, the network is not confident about its decision and simply *pretends* having an opinion just for the sake of providing an answer to the asked question.

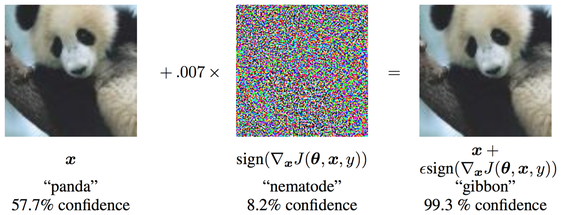

Figure: Adversarial attack example; a panda picture is *attacked* with a special matrix to fool the network [2]

Sensoy et al. [1], try to tackle the uncertainty estimation problem of neural networks with an approach from a **Theory of Evidence** perspective. They propose EDL (Evidential Deep Learning) where they interpret *softmax*, the standard output of a classifier network, as the parameter set of a categorical distribution. Hence, instead of the point estimate of a softmax output, the predictions of the learner is represented as a distribution over possible softmax outputs simply by replacing the parameter set with the parameters of a Dirichlet density. In other words, the resultant model is a Dirichlet distribution on class probabilities with a novel loss function subject to the network weights using standard backpropagation. The predictive distribution is fit to data by minimizing the Bayes risk with respect to the L2-Norm loss which is regularized by an information-theoretic complexity term, Kullback-Leibler (KL) divergence. Any misleading evidence is penalized by incorporating this KL divergence whose effect is gradually increased (annealed) throughout the training in order to force the evidences support the *I do not know* state. The experimental results suggest that the method improves the state of the art in two uncertainty modelling benchmarks: detection of out-of-distribution queries and endurance against adversarial perturbations.

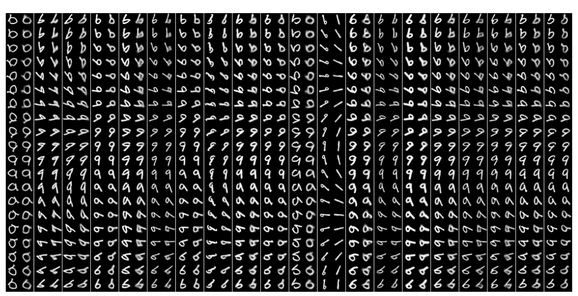

Figure: Out-of-distribution example; rotated versions of the **training** samples are classified improperly [3]

Theory of Evidence Perspective

The idea is replacing the softmax layer of a DNN with a ReLU (or any other activation function with non-negative output) and changing the loss function in a way that it is consisting of both the output loss and a regularization term of a KL divergence to learn the uncertainty representation. Such an approach allows us to keep the regular usage of NNs (prediction) as well as modeling the uncertainty by feeding the output of the last layer (interpreted as the belief mass of the classes) to a Dirichlet as input so that the output could be evaluated to see how likely such an output is. This approach is adopted from *Dempster-Shafer Theory of Evidence* [4] [5].

Suppose that there are K outputs of an NN. Then we can write the following equality

$$u + \sum_{k=1}^{K} b_k = 1$$

where b_k corresponds to k^{th} ReLU output which will be interpreted as the **belief mass** of the k^{th} class and u is the **uncertainty mass** of the particular outputs.

Each b_k and u is defined as follows

$$b_k = \frac{e_k}{S} \quad and \quad u = \frac{K}{S} \tag{1}$$

where e_k is the evidence of the k^{th} class and S is the strength of the Dirichlet we'll use and defined as

$$S = \sum_{k=1}^{K} (e_k + 1)$$

Replacing $e_k + 1$ with α_k

$$\alpha_k = e_k + 1$$

and using the resultant simplex vector a in a Dirichlet as the density

$$D(\boldsymbol{p}|\boldsymbol{\alpha}) = \begin{cases} \frac{1}{B(\boldsymbol{\alpha})} \prod_{i=1}^{K} p_i^{\alpha_i - 1} & \text{for } \boldsymbol{p} \in \mathcal{S}_K \\ 0 & \text{otherwise} \end{cases}$$

As a result, we can define $\mathcal{S}_K$ as

$$\mathcal{S}_K = \{\boldsymbol{p} | \sum_{i=1}^{K} p_i = 1 \text{ and } 0 \leq p_1, ..., p_K \leq 1\}$$

and the probability of k^{th} can still be calculated as

$$\hat{p}_k = \frac{\alpha_k}{S} \tag{2}$$

Proposed Loss Functions

1 - Integrating out class probabilities of the posterior when $D(\boldsymbol{p}|\boldsymbol{\alpha})$ is treated as prior on the likelihood $Mult(\boldsymbol{y_i}|\boldsymbol{p_i})$ and taking negative log

$$\mathcal{L}_i(\Theta) = -log(\int \prod_{j=1}^{K} p_{ij}^{y_{ij}} \frac{1}{B(\alpha_i)} \prod_{j=1}^{K} p_{ij}^{\alpha_{ij}-1} d\boldsymbol{p}_i) = \sum_{j=1}^{K} y_{ij}(log(S_i) - log(\alpha_{ij})) \tag{3}$$

2 - Using cross-entropy loss

$$\mathcal{L}_i(\Theta) = \int [\sum_{j=1}^{K} -y_{ij} log(p_{ij})] \frac{1}{B(\alpha_i)} \prod_{j=1}^{K} p_{ij}^{\alpha_{ij}-1} d\boldsymbol{p}_i = \sum_{j=1}^{K} y_{ij}(\psi(S_i) - \psi(\alpha_{ij})) \tag{4}$$

3 - Using sum of squares loss

$$\mathcal{L}_i(\Theta) = \int ||\boldsymbol{y}_i - \boldsymbol{p}_i||_2^2 \frac{1}{B(\alpha_i)} \prod_{j=1}^{K} p_{ij}^{\alpha_{ij}-1} d\boldsymbol{p}_i = \sum_{j=1}^{K} \mathbb{E}[(y_{ij} - p_{ij})^2] \tag{5}$$

$$= \sum_{j=1}^{K} (y_{ij} - \frac{\alpha_{ij}}{S_i})^2 + \frac{\alpha_{ij}(S_i - \alpha_{ij})}{S_i^2 (S_i + 1)} = \sum_{j=1}^{K} (y_{ij} - \hat{p}_{ij})^2 + \frac{\hat{p}_{ij}(1 - \hat{p}_{ij})}{(S_i + 1)}$$

Regularization with KL Divergence

$$\mathcal{L}(\Theta) = \sum_{i=1}^{N} \mathcal{L}_i(\Theta) + \lambda_t \sum_{i=1}^{N} KL[D(\boldsymbol{p}_i|\widetilde{\boldsymbol{\alpha}}_i)||D(\boldsymbol{p}_i|\langle 1, ..., 1\rangle)]$$

$$KL[D(\boldsymbol{p}_i|\widetilde{\boldsymbol{\alpha}}_i)||D(\boldsymbol{p}_i|\langle 1, ..., 1\rangle)] = log(\frac{\Gamma(\sum_{k=1}^{K} \widetilde{\alpha}_{ik})}{\Gamma(K) \prod_{k=1}^{K} \Gamma(\widetilde{\alpha}_{ik})}) + \sum_{k=1}^{K} (\widetilde{\alpha}_{ik} - 1)[\psi(\widetilde{\alpha}_{ik}) - \psi(\sum_{k=1}^{K} \widetilde{\alpha}_{ik})]$$

where Γ is the Gamma function, ψ is the Digamma function and λ_t is the annealing coefficient depending on the epoch.

Results

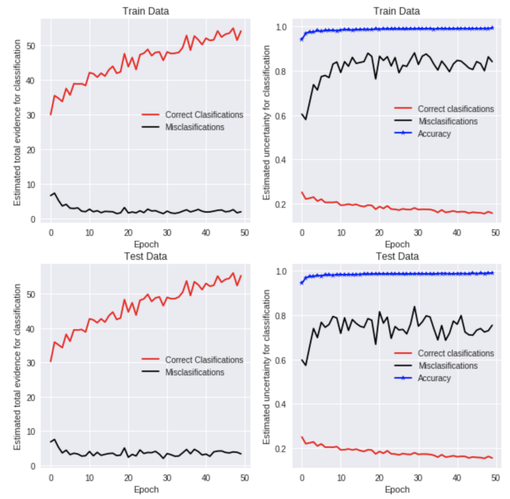

Figure: Average total evidence and prediction uncertainty in addition to accuracy for the training&test sets

In this figure, on the left-hand side, learning process of the estimated total evidence can be seen as the NN is trained through epochs. Note that it's increasing for the correctly classified samples while decreasing for the misclassified samples. On the right-hand side, it can be seen how NN learns to represent uncertainty as well as the accuracy. Note that it's increasing for the misclassified samples while decreasing for the correctly classified samples.

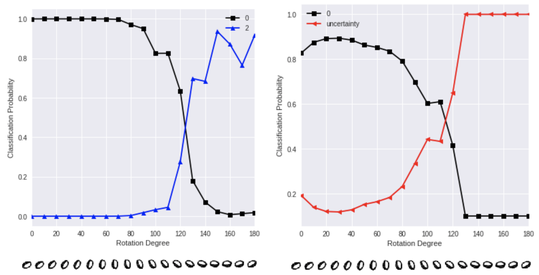

Figure: Rotation of 0 on LeNet

Figure: Rotation of 0 on LeNet-EDL Eqn.5

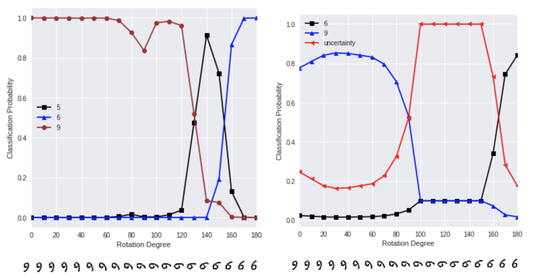

Figure: Rotation of 9 on LeNet

Figure: Rotation of 9 on LeNet-EDL Eqn.5

Here, rotation of two selected digits and its effect on the regular and EDL LeNet architectures can be seen. Note that EDL captures the uncertainty and blocks misclassification for the digit 0. Digit 9 is represented as a special case since its 180 degree rotation corresponds to digit 6. Beside this special situation, it prevents to classify it as digit 5 between degrees of 120 and 160.

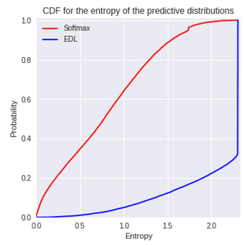

Figure: Softmax vs EDL - NotMNIST

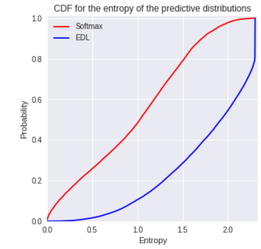

Figure: Softmax vs EDL - CIFAR5

As we can see from the CDFs, the EDL model produces higher expected entropy for out-of-distribution data.

Conclusion

We've worked on a mechanism to quantify the uncertainty of the NN outputs as regular softmax layer incapable of assign certainty to its outputs. By using Dempster-Shafer belief model, we introduced such an ability to LeNet so that it is taught to say *I don't know* to the out-of-distribution samples like rotated MNIST, NotMNIST and unseen part of CIFAR10 datasets. It's shown that Eqn.5 is the best loss function resulting in the highest expression of uncertainty. The results show that it works as expected in terms of expressing the uncertainty and still preserves the ability to classify unseen samples.

Future Work

- Comparing the results against other related works such as Gaussian Processes and Bayesian Neural Networks with Variational Bayes
- Measuring performance against Adversarial attacks
- Measuring performance on more complex architectures
- Improving loss function
- Annealing regularization term differently

References

[1] Sensoy, Murat, Lance Kaplan, and Melih Kandemir. "Evidential deep learning to quantify classification uncertainty." Advances in Neural Information Processing Systems. 2018.

[2] Uesato, J., O'Donoghue, B., Oord, A. V. D., Kohli, P. (2018). Adversarial risk and the dangers of evaluating against weak attacks. arXiv preprint arXiv:1802.05666.

[3] https://blog.singularitynet.io/can-deep-networks-learn-invariants-1e06a5052555

[4] A.P.Dempster. A generalization of Bayesian inference. In *Classic works of the Dempster-Shafer theory of belief functions*, pages 73–104. Springer, 2008

[5] A. Josang. Subjective Logic: A Formalism for Reasoning Under Uncertainty. Springer, 2016

View Article

View Code