Bayesian Neural Networks with Soft Evidence

Edward Yu [1]

Abstract

Bayes's rule deals with hard evidence, that is, we can calculate the probability of event A occurring given that event B has occurred. Soft evidence, on the other hand, involves a degree of uncertainty about whether event B has actually occurred or not. Jeffrey's rule of conditioning provides a way to update beliefs in the case of soft evidence. We provide a framework to learn a probability distribution on the weights of a neural network trained using soft evidence by way of two simple algorithms for approximating Jeffrey conditionalization. We propose an experimental protocol for benchmarking these algorithms on empirical datasets and find that Jeffrey based methods are competitive or better in terms of accuracy yet show improvements in calibration metrics upwards of 20% in some cases, even when the data contains mislabeled points.

1. Introduction

In a traditional probabilistic reasoning framework, we make inferences based on *hard evidence*, or in other words, observation of some event B. Then we may be able to say something about the likelihood of another event A by calculating $P(A|B)$. Take the classic case of Bernoulli trials, in which we are flipping a (possibly biased) coin multiple times. Each flip of the coin constitutes hard evidence, as we are certain whether the coin landed heads or tails, and we can record the event definitively: either $B = 1$ or $B = 0$. Then we may use a traditional Bayesian framework to estimate the probability the next flip will result in heads or tails.

In contrast, *soft evidence* arrives in the form of another probability distribution $R(B)$ as described by Peng et al. (2010), or more simply, as an "unreliable testimony" of B accompanied by a strength of belief in the $[0, 1]$ interval as in Chan & Darwiche (2005) and Jacobs (2019). In the example of Bernoulli trials, imagine that in poor visibility conditions we are not able to ascertain definitively whether the coin landed heads or tails, but we estimate that there is a 80% probability it landed heads, i.e., $R(B = 1) = .8$. Soft evidence appears in many real world contexts, from unreliable witness testimonies in murder trials (Fields, 2013) to game theoretic decision making (Dietrich et al., 2016). One problem setting that we focus on in this paper is that of crowdsourced data, where each data point has multiple annotators, yet there is no "true" label available, only a categorical distribution over possible labels (Rodrigues et al., 2013). Logically it seems there should be a way to make use of soft evidence that takes into account the uncertainty of the label, in fact, it seems almost wasteful and anti-Bayesian to throw away the uncertainty and train with a point estimate (hard label).

However, since Bayes' theorem cannot directly be applied, there is no consensus on how to update one's beliefs in order to calculate $P(A|B)$, or in a generalized machine learning setting, how to train models given soft evidence. One sensible approach is Jeffrey's rule of conditioning (Jeffrey, 1990) (Shafer, 1981), which can be thought of as an modification of the posterior probability distribution given a set of constraints. For a Bernoulli constraint, Jeffrey's rule reduces to standard probabilistic reasoning in the case where we observe B with probability $R(B) = 1$, and is otherwise a convex combination of $P(A|B = 1)$ and $P(A|B = 0)$. Although it seems natural to interpret soft evidence in this way, it is of philosophical debate (beyond the scope of this paper) whether it is "correct". Suffice it to say that there are both many extensions of Jeffrey's original rule (Ichihashi & Tanaka, 1989) (Benferhat et al., 2010) (Benferhat et al., 2011) (Smets, 1993), and alternative approaches based on the broader field of Dempster-Shafer theory (Shafer, 1992) (Denoeux, 2019).

There are several related areas of study when it comes to applying soft evidence to practical machine learning domains. Pearl's method is another method of dealing with soft evidence, whereby a Bayesian network is extended with an additional node, and soft evidence is recast as hard evidence of a virtual event. In fact, Chan & Darwiche (2005) show that one may convert between Pearl's method and Jeffrey's method by specifying likelihood ratios. Peng et al. (2010) extend this work from theory to application by providing

[1]Genesis Global Trading, New York, NY, United States. Correspondence to: Edward Yu <eyu@genesistrading.com>.

Presented at the ICML 2021 Workshop on Uncertainty and Robustness in Deep Learning.,

several algorithms for approximating Jeffrey's method on Bayesian networks. We in turn build upon Peng et al. (2010) and extend the algorithms for usage in neural networks.

A related field is that of learning noisy labels, where data is given in the form $\{x_i, y_i \sim R(\gamma)\}$, however unlike the soft evidence case where $R(\gamma_i)$ is given, in the noisy label case only y_i is observed. Rolnick et al. (2017) posit that under certain types of noise, the noisiness of the labels can be ignored and a neural network is trained as if the noise did not exist. Karimi et al. (2020) give a survey of methods for dealing with label noise, from label preprocessing to weighted loss functions. Since soft evidence can be converted to a noisy label by taking $y_i = \arg\max_\gamma R(\gamma)$, we use methods from learning noisy labels as baselines.

The main contributions of this paper are:

1. To our knowledge, we are the first to extend Jeffrey conditionalization to the training of neural networks. We provide end-to-end algorithms, including two approximation methods for Jeffrey's method: one simple ensemble method and one method based upon the *Bayes by Backprop* method (Blundell et al., 2015).
2. We benchmark against methods in related domains, including baselines such as ensembles which ignore the soft evidence and training methods for noisy labels. We show empirical success in reducing calibration error, while maintaining equal or better levels of accuracy.

The rest of the paper is organized as follows: Section 2 defines Jeffrey's rule and gives an example. Section 3 gives two approximation algorithms for learning neural networks using Jeffrey's rule. Section 4 defines the experimental protocol, including evaluation criteria, methodology for generating corrupted data, baseline models, and results. Section 5 concludes.

2. Jeffrey's Rule of Conditioning

In this section we explore the preliminaries behind Jeffrey's rule. The following two definitions are due to Chan & Darwiche (2005). Consider a set of mutually exclusive and exhaustive events $\gamma_1, ..., \gamma_n$. We wish to update a probability distribution $P(\alpha)$, given a set of soft evidence constraints of the form $R(\gamma_i) = q_i$ for $i = 1, 2, ..., n$.

Definition 2.1 (probability kinematics)**.** Suppose there are two distinct probability distributions P and P', that is, it is the case that $P(\gamma_i) \neq P'(\gamma_i)$ for at least one i. P' is obtained from P by probability kinematics on $\gamma_1, ..., \gamma_n$ iff for every event α in the probability space

$$P(\alpha|\gamma_i) = P'(\alpha|\gamma_i) \text{ for } i = 1, 2, ..., n$$

Definition 2.2 (Jeffrey's Rule)**.** Jeffrey's rule produces the sole distribution J that is obtained from P by probability kinematics on $\gamma_1, ..., \gamma_n$:

$$\begin{aligned} J(\alpha) &= \sum_{i=1}^{n} R(\gamma_i) \frac{P(\alpha, \gamma_i)}{P(\gamma_i)} \\ &= \sum_{i=1}^{n} P(\alpha|\gamma_i) R(\gamma_i) \end{aligned} \tag{1}$$

Rephrased, Jeffrey's rule finds the closest distribution (minimum KL-divergence (Peng et al., 2010)) to P that satisfies the constraints R. If there is no uncertainty, i.e., $R(\gamma_1) = 1$, then $J(\alpha) = P(\alpha)$, and Jeffrey conditionalization does not update $P(\alpha)$. Otherwise, the posterior is a mixture of all possibilities of γ.

3. Jeffrey's and Neural Networks

Now consider a Bayesian neural network with weights w and data $D_k = \{(x_i, y_i \sim R(\gamma))\}$. The posterior predictive distribution is

$$p(y|x, D_k) = \int p(y|x, w) p(w|D_k) dw \tag{2}$$

This neural network is dependent on one particular instantiation of the data. In a standard setting with fixed (x, y), this is the only way to proceed. In contrast, a Jeffrey neural network would depend on all possible instantiations of the data:

$$J(y|x) = \sum_k p(y|x, D_k) R(D_k) \tag{3}$$

Unfortunately this is computationally intractable as this would require enumerating through all instantiations of D_k, and training a separate neural network for each possibility of D_k. We give the following two algorithms as practical alternatives.

3.1. Sparse K Approximation

We limit ourselves to training K neural networks as computational resources permit. This is a Monte Carlo direct sampling method. It is apparent that $\lim_{k\to\infty} \bar{J}(y|x) = J(y|x)$, as every possible instantiation of D_k will be sampled a proportional number of times according to $R(D_k)$. In the case

Algorithm 1 Sparse K Approximation

for $k = 1$ **to** K **do**
  Draw sample $D_k \sim R(D)$
  Train Bayesian neural network $p(y|x, D_k)$
end for
Ensemble is then $\bar{J}(y|x) = \sum_k p(y|x, D_k)$

where k is finite, this can be viewed as a generalization of *bagging* (Breiman, 1996): we create bootstrap samples of data to train on, and average the predictors. Whereas in the bagging procedure we sample uniformly from the empirical distribution $\widehat{R}(D)$, here we sample from the true distribution $R(D)$.

3.2. Jeffrey's By Backprop

Blundell et al. (2015) introduced the now widely adopted *Bayes by Backprop* algorithm to train Bayesian neural networks. The idea is to use variational inference to find a distribution on weights that approximate the true Bayesian posterior. Using only a minor modification to the loss function, we introduce a similar *Jeffrey's by Backprop* method whose output will be a probability distribution over weights. We introduce a distribution over weights $q(w|\theta)$ parameterized by θ. The objective of the *Bayes by Backprop* method is to learn the optimal parameters θ that minimize the Kullback-Leibler divergence between $q(w|\theta)$ and $P(w|D)$.

$$\theta^* = \arg\min_\theta \mathrm{KL}[q(w|\theta)||P(w|D)] \tag{4}$$

KL divergence is defined as:

$$\mathrm{KL}[q||p] = \int q(x) \log\left(\frac{q(x)}{p(x)}\right) \, dx \tag{5}$$

In the spirit of Jeffrey's rule, we modify the objective to minimize the sum of KL divergences across all possible $P(w|D_k)$.

$$\theta^* = \arg\min_\theta \sum_k \mathrm{KL}[q(w|\theta)||P(w|D_k)]R(D_k) \tag{6}$$

$$= \arg\min_\theta \left[\sum_k \int q(w|\theta) \log \frac{q(w|\theta)}{P(w)P(D_k|w)} dw\right] R(D_k) \tag{7}$$

$$= \arg\min_\theta \sum_k \Big(\mathrm{KL}[q(w|\theta)||P(w)] \tag{8}$$

$$- E_{q(w|\theta)}[\log P(D_k|w)]\Big) R(D_k) \tag{9}$$

$$= \arg\min_\theta \mathrm{KL}\left[q(w|\theta)||P(w)\right] \tag{10}$$

$$- \sum_k E_{q(w|\theta)}[\log P(D_k|w)]R(D_k) \tag{11}$$

This leads to the loss function

$$F(D,\theta) = \mathrm{KL}[q(w|\theta)||P(w)] \tag{12}$$

$$- \sum_k E_{q(w|\theta)}[\log P(D_k|w)]R(D_k) \tag{13}$$

and the approximate loss

$$F(D,\theta) \approx \sum_{i=1}^{n} \Big[\log q(w^{(i)}|\theta) - \log P(w^{(i)}) \tag{14}$$

$$- \log P(D^{(i)}|w^{(i)})\Big] \tag{15}$$

where $w^{(i)}$ is a Monte Carlo sample of weights drawn from the variational posterior $q(w|\theta)$, and $D^{(i)}$ is sampled from $R(D)$. With the modified loss, we may proceed in the exact same manner as outlined in (Blundell et al., 2015), and train by backpropagation.

4. Experiments

To ensure that results are reproducible, code and data for experiments are available on our Github at `https://github.com/edwardyu/soft-evidence-bnn`.

4.1. Evaluation Criteria

Our goal is not to beat state-of-the-art benchmarks on the datasets, but rather to show an improvement in both/either accuracy or calibration metrics over the baseline models. A well calibrated model is one that is correct on high confidence predictions and incorrect on low confidence predictions. We use two *proper scoring rules* to measure calibration, the negative log-likelihood score (NLL) and the Brier score (Guo et al., 2017) (Lakshminarayanan et al., 2016). The negative log likelihood is defined as

$$\mathcal{L} = -\log J(y|x) \tag{16}$$

And the Brier score is

$$\mathcal{L} = \frac{1}{C}\sum_{c=1}^{C}(R(y=c) - J(y=c|x))^2 \tag{17}$$

All errors reported in parentheses are 1 standard deviation.

4.2. Models and Baselines

For all methods, we set a common base learner: a neural network with a fixed architecture trained via the *Bayes by Backprop* algorithm (Blundell et al., 2015). The specific implementation is based on an open source library created by Esposito (2020). We compare the following methods:

1. Sparse K approximation to Jeffrey's (**SparseK**): see algorithm 1.
2. Jeffrey's by Backprop (**JNN**): see section 3.2.
3. Noisy labels (**NL**): train on $D = \{(x_i, \arg\max_\gamma R(\gamma_i))\}$. In any hard label dataset this is the default assumption.

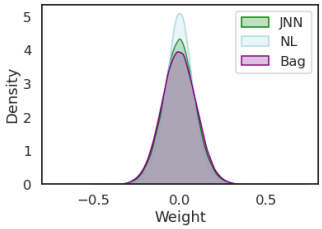

Figure 1. A histogram of weights from models trained on the CIFAR-10H dataset. JNN and Bag models have increased uncertainty in the weights due to training on soft labels. The NL method first converts the soft label to a hard label.

4. Noisy labels ensemble (**NLE**): train on $D = \{(x_i, \arg\max_\gamma R(\gamma_i))\}$ using K base learners, and output the majority vote class.

5. Bagging (**Bag**): Sample data from empirical distribution $\widehat{R}(D)$, train on sampled data.

All methods are trained for 100 epochs, and we set $K = 3$ for ensemble methods.

4.3. CIFAR-10H

The CIFAR-10H dataset consists of 10,000 color images belonging to 10 possible classes (Peterson et al., 2019). It is a subset of the popular CIFAR-10 dataset; the difference is that CIFAR-10H is soft labeled, with each image having multiple annotations. In total the authors collected 511,400 human labels, but removed annotators who scored low on attention-checks. On average, the final dataset is not very noisy: the mean vote share for the most popular label is 95.44%. We train using a Bayesian version of the LeNet architecture (LeCun et al., 1998).

Table 1 shows results. The relatively low accuracies across the board highlights the difference in calibration. When comparing to hard label method **NL**, all other methods significantly reduce the NLL and Brier scores. However, it should be noted that **JNN** is a single network method, and thus uses much less compute resources than the ensemble methods. **SparseK** achieved the highest accuracy and lowest calibration error.

Table 1. Results on CIFAR-10H

Model	Accuracy	NLL $\times 10$	Brier $\times 10^3$
SparseK	$\mathbf{47.47(\pm 0.68)}$	$\mathbf{1.91(\pm 0.02)}$	$\mathbf{7.47(\pm 0.08)}$
JNN	$46.98(\pm 0.37)$	$2.63(\pm 0.29)$	$8.78(\pm 0.43)$
NL	$47.02(\pm 0.65)$	$3.93(\pm 0.14)$	$9.85(\pm 0.03)$
NLE	$45.93(\pm 1.83)$	$2.03(\pm 0.06)$	$7.76(\pm 0.25)$
Bag	$46.13(\pm 0.47)$	$1.97(\pm 0.04)$	$7.64(\pm 0.14)$

4.4. LabelMe

The LabelMe dataset is an image classification dataset that appears in many forms, here we use a subset obtained from Rodrigues & Pereira (2018). The dataset consists of 2688 images, each corresponding to one of the following categories: highway, inside city, tall building, street, forest, coast, mountain or open country. This is a soft evidence dataset: each image was labeled by multiple annotators on the Amazon Mechanical Turk platform. On average each image has 2.547 labels and the annotators have a mean accuracy of 69.2%. When training on this dataset, we used the very deep convolutional network architecture (VGG-11) as proposed by Simonyan & Zisserman (2014).

Table 2 shows results. The labels in this dataset are noisier than CIFAR-10H, and Jeffrey-based methods show an increase in accuracy compared to their Bayesian counterparts. As before, any ensemble method or **JNN** reduces the calibration error when compared with the default hard label method **NL**.

Table 2. Results on LabelMe

Model	Accuracy	NLL $\times 10$	Brier $\times 10^3$
SparseK	$70.73(\pm 1.60)$	$1.35(\pm 0.03)$	$8.16(\pm 0.18)$
JNN	$\mathbf{70.76(\pm 0.97)}$	$\mathbf{1.19(\pm 0.05)}$	$\mathbf{6.70(\pm 0.21)}$
NL	$64.11(\pm 1.94)$	$2.22(\pm 0.32)$	$8.87(\pm 0.29)$
NLE	$66.89(\pm 0.56)$	$1.33(\pm 0.06)$	$7.82(\pm 0.39)$
Bag	$67.99(\pm 2.52)$	$1.37(\pm 0.07)$	$8.03(\pm 0.37)$

5. Conclusions and Future Work

In this paper we showed two methods for training Bayesian neural networks when the dataset contains soft evidence, where labels are uncertain and are only presented as probabilistic inputs. In contrast to previous approaches such as converting to hard labels or bagging, Jeffrey conditionalization is provably optimal in the sense that it follows the principle of probability kinematics. Evaluating on several real-world datasets, we show that Jeffrey based methods are competitive in terms of accuracy, while being better calibrated and showing lower KL-loss. Since our examples involve human labelers, the provided empirical $R(\gamma)$ are likely to differ from the true data distribution. Future work may focus on understanding this scenario better: are there bounds we can place on the loss of our model if $\mathrm{KL}[R(\gamma)||R^{\text{true}}(\gamma)]$ is bounded? Additionally, it may be further worth exploring the relationship between Jeffrey conditionalization and bagging. Does the ample theoretical work on the Jeffrey posterior provide clues as to why bagging works so well in practice?

References

Benferhat, S., Dubois, D., Prade, H., and Annewilliams, M.-A. A framework for iterated belief revision using possibilistic counterparts to jeffrey's rule. *Fundamenta Informaticae*, 99(2):147–168, 2010.

Benferhat, S., Tabia, K., and Sedki, K. Jeffrey's rule of conditioning in a possibilistic framework. *Annals of Mathematics and Artificial Intelligence*, 61(3):185–202, 2011.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*, 2015.

Breiman, L. Bagging predictors. *Machine learning*, 24(2): 123–140, 1996.

Chan, H. and Darwiche, A. On the revision of probabilistic beliefs using uncertain evidence. *Artificial Intelligence*, 163(1):67–90, 2005.

Denoeux, T. Logistic regression, neural networks and dempster–shafer theory: A new perspective. *Knowledge-Based Systems*, 176:54–67, 2019.

Dietrich, F., List, C., and Bradley, R. Belief revision generalized: A joint characterization of bayes' and jeffrey's rules. *Journal of Economic Theory*, 162:352–371, 2016.

Esposito, P. Blitz - bayesian layers in torch zoo (a bayesian deep learning library for torch). https://github.com/piEsposito/blitz-bayesian-deep-learning/, 2020.

Fields, K. L. Toward a bayesian analysis of recanted eyewitness identification testimony. *NYUL Rev.*, 88:1769, 2013.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. *arXiv preprint arXiv:1706.04599*, 2017.

Ichihashi, H. and Tanaka, H. Jeffrey-like rules of conditioning for the dempster-shafer theory of evidence. *International Journal of Approximate Reasoning*, 3(2):143–156, 1989.

Jacobs, B. The mathematics of changing one's mind, via jeffrey's or via pearl's update rule. *Journal of Artificial Intelligence Research*, 65:783–806, 2019.

Jeffrey, R. C. *The logic of decision*. University of Chicago Press, 1990.

Karimi, D., Dou, H., Warfield, S. K., and Gholipour, A. Deep learning with noisy labels: Exploring techniques and remedies in medical image analysis. *Medical Image Analysis*, 65:101759, 2020.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. *arXiv preprint arXiv:1612.01474*, 2016.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Peng, Y., Zhang, S., and Pan, R. Bayesian network reasoning with uncertain evidences. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 18(05):539–564, 2010.

Peterson, J. C., Battleday, R. M., Griffiths, T. L., and Russakovsky, O. Human uncertainty makes classification more robust. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9617–9626, 2019.

Rodrigues, F. and Pereira, F. Deep learning from crowds. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Rodrigues, F., Pereira, F., and Ribeiro, B. Learning from multiple annotators: distinguishing good from random labelers. *Pattern Recognition Letters*, 34(12):1428–1436, 2013.

Rolnick, D., Veit, A., Belongie, S., and Shavit, N. Deep learning is robust to massive label noise. *arXiv preprint arXiv:1705.10694*, 2017.

Shafer, G. Jeffrey's rule of conditioning. *Philosophy of Science*, 48(3):337–362, 1981.

Shafer, G. Dempster-shafer theory. *Encyclopedia of artificial intelligence*, pp. 330–331, 1992.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Smets, P. Jeffrey's rule of conditioning generalized to belief functions. In *Uncertainty in artificial intelligence*, pp. 500–505. Elsevier, 1993.