ABSTRACT

Neural networks are powerful function approximators with tremendous potential in learning complex distributions. However, they are prone to overfitting on spurious patterns. Bayesian inference provides a principled way to regularize neural networks and give well-calibrated uncertainty estimates. It allows us to specify prior knowledge on weights. However, specifying domain knowledge via distributions over weights is infeasible. Furthermore, it is unable to correct models when they focus on spurious or irrelevant features. New methods within explainable artificial intelligence allow us to regularize explanations in the form of feature importance to add domain knowledge and correct the models’ focus. Nevertheless, they are incompatible with Bayesian neural networks, as they require us to modify the loss function. We propose a new explanation regularization method that is compatible with Bayesian inference. Consequently, we can quantify uncertainty and, at the same time, have correct explanations. We test our method using four different datasets. The results show that our method improves predictive performance when models overfit on spurious features or are uncertain of which features to focus on. Moreover, our method performs better than augmenting training data with samples where spurious features are removed through masking. We provide code, data, trained weights, and hyperparameters.

Keywords Explainable Artificial Intelligence · Deep Learning · Bayesian Neural Networks

1 Introduction

Neural networks (NNs) have in recent years shown high performance and been successful in many applications [Goodfellow et al., 2016, Silver et al., 2018, Esteva et al., 2019, Kiran et al., 2022]. However, they can overfit on spurious features in training datasets and lose the ability to generalize [Szegedy et al., 2014, Lapuschkin et al., 2019]. To prevent overfitting, we use regularization techniques like weight regularization, dropout, early stopping, and explanation regularization [Ross et al., 2017]. A probabilistic approach to regularizing NNs is to leverage Bayesian inference [Blundell et al., 2015, Jospin et al., 2022]. In Bayesian NNs, we find the posterior distribution on weights rather than point estimates. To find the posterior distribution, we define a prior distribution on weights that moves them towards our preferred choices. As the amount of data increases, the prior weighs less [Blundell et al., 2015, Prince, 2023]. Although Bayesian inference gives us well-calibrated uncertainty estimates, this principled way to regularize NNs is incompatible with newer methods that regularize explanations. Explanation regularization came as a response to the need of explainable NNs [Ross et al., 2017, Teso and Kersting, 2019, Rieger et al., 2020]. In explanation regularization, we have annotated masks that we refer to as explanation feedback. They indicate areas in the input space irrelevant for predictions, which is seen in Fig. 1. Furthermore, Bayesian inference regularizes the model via prior on weights. It is unable to say anything regarding the input space. In contrast, explanation regularization enables...
Figure 1: Method Overview. a) During training, a NN gets an input sample $X_i \in \mathbb{R}^{(w \times h \times c)}$ from the training dataset and tries to match the prediction $\hat{y}_i$ with the ground truth label $y_i$. Our method provides the NN with additional evidence in the form of explanation feedback $E_i \in \{0, 1\}^{(w \times h)}$. A value of 1 in $E_i$ indicates a region in the input space as irrelevant to the prediction, while 0 indicates that we do not have any concern. The explanation feedback is used to regularize the model’s focus to give correct explanation and add domain knowledge. b) A new input sample $X_j$ from the test dataset is fed to the model and an explanation is generated. Without explanation regularization, the NN uses the patch to make the prediction. With our method, the NN no longer looks at the patch in the image. The skin images are from the International Skin Imaging Collaboration (ISIC) dataset [Codella et al., 2019 | Tschandl et al., 2018 | Rieger et al., 2020].
2 Background

We introduce the background on Bayesian NN [Prince, 2023; Murphy, 2023] and the local reparameterization trick (LRT) [Kingma et al., 2015] that our method relies on. The loss function introduced in this section will be used in Section 3.

2.1 Bayesian Neural Network

In NNs, we learn the weights $w$ via maximum likelihood estimation. Given a dataset $D = \{(x_i, y_i)\}_{i=0}^N$ with $N$ samples, we optimize the objective defined by $\arg \max_w \sum_{i=0}^N \log P_r(y_i|x_i, w)$ assuming that the samples are independent and identically distributed. There are several choices of regularization, one is to use the maximum a posteriori estimation defined by $\arg \max_w \sum_{i=0}^N \log P_r(y_i|x_i, w) + \log P_r(w)$, where $P_r(w)$ moves the weights towards the choices we prefer to prevent overfitting. $P_r(w)$ is referred to as the prior, and reflects our prior belief of what the weights should be before seeing the data. The prior imposes L1 or L2 regularization depending on if it is Laplace or Gaussian respectively.

Both maximum likelihood estimation and maximum a posterior estimation focus on finding point estimates of the weights. In Bayesian NNs, we represent weights as probability distributions and not as point estimates. To compute the likelihood term will weigh more.

The objective maximizes the log likelihood of the data like in the maximum likelihood estimation. It is important to note that we are maximizing with respect to the distribution parameters $\theta$ and not the weights themselves like in maximum likelihood estimation where we treated them as point estimates. The objective additionally minimizes the KL divergence between the variational distribution and the prior distribution, moving the probability mass towards our choice of weights. The objective has to trade off between these two quantities, but as the amount of data increases, the likelihood term will weigh more. To optimize Eq. (1), we use stochastic gradient descent with the reparameterization trick [Kingma and Welling, 2014; Blundell et al., 2015]. We model the variational distribution with a fully factorized Gaussian distribution defined by $q_\theta(w) = \prod_{i=0}^N \mathcal{N}(w_i|\mu_i, \sigma^2)$ using the mean field approximation. To sample weights, we first sample noise $\epsilon \sim \mathcal{N}(\epsilon|0, 1)$, thereafter compute $w_i = \mu_i + \sigma_i \epsilon$ for $i \in \{1, 2, \ldots, n\}$ independently. By using the reparameterization trick, we can update the parameters backpropagation. The loss function we optimize in a Bayesian CNN using minibatches is defined by

$$L = D_{KL}(q_\theta(w)\|P_r(w)) - \frac{N}{M} \sum_{i=1}^M \sum_{j=1}^J \log P_r(y_i|x_i, w_j = \mu + \sigma \epsilon),$$

(2)

where $M$ is the number of minibatches, $N$ is the number of samples in our dataset and $J$ the number of Monte Carlo samples. We use fully factorized Gaussians for both the variational distribution and the prior distribution so that the KL divergence term can be solved in closed form [Kingma and Welling, 2014].

2.2 Local Reparameterization Trick

To reduce variance of Eq. (2), [Kingma et al., 2015] propose the local reparameterization trick (LRT). Instead of sampling weights as in Eq. (2), LRT samples activations. Thus, the uncertainty is moved from weights that affect all samples to activations that is local and sample dependent. The LRT loss function is defined by

$$L^{LRT} = D_{KL}(q_\theta(w)\|P_r(w)) - \frac{N}{M} \sum_{i=1}^M \sum_{j=1}^J \log P_r(y_i|x_i, a_{i,j}),$$

(3)

where we sample activations $a$ rather than weights. We omit $x_i$ in the condition as no extra information is added given that we know the activations. Do note that we do need to know $x_i$ in the first place to compute the activations. We
show how these activations are sampled in fully connected layers and in convolutional layers [Kingma et al., 2015, Molchanov et al., 2017].

**Fully Connected Layer.** Assume that the input to a layer is \( b \in \mathbb{R}^m \), to compute the activation, we compute the mean and variance of the activation defined by \( \delta = \sum_{i=1}^{m} b_i w_{i,j} \) and \( \gamma^2 = \sum_{i=1}^{m} b_i^2 w_{i,j}^2 \). Thus, the distribution on the activation is \( \mathcal{N}(\alpha \delta, \gamma^2) \) and can be sampled as shown in Section 2.1.

**Convolutional Layer.** Assume that the input to a layer is \( B \in \mathbb{R}^{w' \times h' \times m} \) and the weights \( W \) is also a matrix. We assume only a single feature map to simplify the calculations. The mean and variance are defined by \( \delta = B \ast W \) and \( \gamma^2 = B^2 \ast W^2 \) where * is the convolution operator and \( (\cdot)^2 \) is applied element-wise. The distribution on activations \( A \) is then \( \mathcal{N}(\text{vec}(A)) \text{vec}((\delta, \gamma^2)) \) and the reparameterization trick can be used to sample activations.

## 3 Related Work

Explainable artificial intelligence (XAI) aims to assist humans understand artificial intelligence systems, their strength and weaknesses, provide understanding of how they will perform in unknown situations [Gunning and Aha, 2019]. Methods to understand machine learning models are often divided into interpretable models and post hoc explainability [Lipton, 2018; Arrieta et al., 2020; Murphy, 2023]. Our method goes under post hoc explainability methods that are applied to models after training. The method we propose is related to a line of work that corrects or prevents models to look at spurious features. As far as we know, [Ross et al., 2017] introduced the first method to correct and prevent models to look at spurious features in the context of XAI. To prevent models from learning spurious features, Ross et al. [2017] regularizes the input gradient in an area specified by an explanation feedback. That is, they minimize the \( \ell_2 \) norm of the input gradient in the region that is specified to be irrelevant by the user. Liu and Avc [2019] use a similar approach to Ross et al. [2017] in text classification to make a model focus less on certain words. Similarly, working on text, Du et al., 2019 encourages sparse importance values on irrelevant features and that the models should be uncertain when important features are removed. Rieger et al., 2020 regularize explanations leveraging the method contextual decomposition explanation penalization. This allows them to penalize both feature importance and interaction. Shao et al., 2021 regularize explanations using influence functions and show that it is better than using input gradients. Erion et al., 2021 regularizes explanations by specifying domain knowledge regarding how explanations should be before training. For example, the total variation between feature importance values for pixels in image data should be low. Like the abovementioned methods, Selvaraju et al., 2019 propose a new loss function to align human feedback on important features and where models look. Common to all these approaches is that they modify the loss function by augmenting it with additional terms. This, however, makes it impossible to minimize ELBO as the loss function is modified and augmented with new terms. We instead introduce a simple approach leveraging LRT to add explanation feedback to prevent models to look at irrelevant features and add domain knowledge.

Differently from previously mentioned methods, Schramowski et al., 2020, Teso and Kersting, 2019 propose a model agnostic approach to regularize explanations by augmenting the training dataset with counterexamples. These counterexamples are the same as the samples in the training dataset, but where spurious features have been modified. These modifications can be replacing spurious features with random noise or use feature values from other samples without spurious features. We show in the experiments that it is less effective than our approach since location dependent spurious features cannot be removed. Furthermore, sometimes background information can be a positive influence, but this method does not allow partial use of features by models. Lastly, creating counterexamples introduces out-of-distribution samples into the training dataset that can negatively affect training.

## 4 Method

We detail our method by first setting up the model and dataset assumptions. Afterward, we detail how to regularize explanations in Bayesian CNN using our method. We assume that we have a Bayesian CNN represented by \( Pr(y|X, w) \). Furthermore, we assume access to a dataset \( D = \{(X_i, y_i, E_i)\}_{i=1}^N \) where \( X_i \in \mathbb{R}^{(w \times h \times c)} \) is an input image and \( y_i \in Y \) is a target label. \( Y \) is the set of real numbers if it is a regression task or a set of class labels for classification. \( E_i \in \{0, 1\}^{(w \times h)} \) is an explanation feedback. A value of 1 in \( E_i \) indicates an area of \( X_i \) where the NN should not focus on when predicting \( y_i \). A value of 0 points at an area where no feedback is given, that is, it does not matter what the model does in that region.

We showed in Section 2.2 that training a Bayesian CNN with LRT amounts to minimize Eq. 3. To regularize explanations implies regularizing the input gradients [Ross et al., 2017] or some other quantity [Rieger et al., 2020, Selvaraju et al., 2019]. But to regularize input gradient without changing the objective, we need to know the distribution on input gradients, which we do not know. Instead, we leverage activation outputted from convolutional layers to
Regularizing Explanations in Bayesian Convolutional Neural Networks

A P REPRINT

Figure 2: Finding Activations. Given an explanation feedback $E_i \in \{0, 1\}^{(w \times h)}$ for the sample $X_i \in \mathbb{R}^{(w \times h \times c)}$, we find activations to add the explanation feedback. A value of 1 in $E_i$ indicates irrelevant regions in the input. A value of 0 denotes features that no preference is given. First, $E_i$ is downsized to the size of feature maps of the last convolutional layer using the function $f(\cdot)$. Afterward, since the height and widths are the same, we simply overlay the explanation feedback with the feature maps to find activations to target. Specifically, we inject this information via the likelihood term of Eq. (3). The skin image is from the ISIC dataset [Codella et al., 2019, Tschandl et al., 2018, Rieger et al., 2020].

incorporate the explanation feedback to regularize explanations. To show how our method works, we take the objective in Eq. (3) and show how the likelihood term is computed to incorporate explanation feedback.

We incorporate the explanation feedback via the last convolutional layer in a Bayesian CNN. We downsized the explanation feedback to the size of the activation produced by the last convolutional layer, as seen in Fig. 2 using a function $f(\cdot)$. In practice, the function is implemented using `torch.nn.AdaptiveMaxPool2d` [Ansel et al., 2024]. Then we set the evidence of activation overlapping with 1’s in the explanation feedback to 0. We denote those activations that the explanation feedback indicates are unimportant as $\bar{a}$, while the rest of the activations in the network as $\hat{a}$. When we refer to all activations in the network, we simply write $a$. The log likelihood term with explanation feedback added is defined by

$$
\log P r(y_i, e_i | a_i, x_i) = \log P r(y_i, \bar{a}_i = 0 | x_i) = \log P r(y_i | \bar{a}_i = 0, x_i) + \log P r(\bar{a}_i = 0 | x_i).
$$

Because the size of the explanation feedback is larger than the prediction output, we introduce a hyperparameter $\lambda$ to lower the importance of $\log P r(\bar{a}_i = 0 | \hat{a}_i)$ in Eq. (4) and set it to $\lambda \ll 1$. Note that we still minimize Eq. (3) but add explanation feedback using activations as seen in Fig. 2 via the likelihood term as shown in Eq. (4).

5 Experiments

We first detail our experimental setup, including the datasets used, model architectures, and additional details. Afterward, we show how our model improves the predictive performance while minimizing the models’ focus on spurious features.

5.1 Experimental Setup

To test the performance of our method, we use four different datasets. All of the datasets except for the ISIC skin cancer dataset were downloaded via `torchvision.datasets` [Ansel et al., 2024].

Datasets. We create two versions of Decoy MNIST [Ross et al., 2017] which builds on The MNIST database of handwritten digits [LeCun et al., 2010]. The MNIST dataset consists of black and white images of digits from 0 to 9. The Decoy MNIST dataset adds decoys at the corners and sides of input samples as seen in Fig. 3c. In the first version that we name “color”, the grayscale of decoys in the training has pixel intensity $255 - 25k$, where $k$ is the class label.
Table 1: Predictive performance across four datasets with different variations of Decoy MNIST and ISIC. For datasets with more than two classes, we compute macro-averaged F1 score.

| Dataset                  | No Regularization | Our Method | Data Augmentation |
|--------------------------|-------------------|------------|-------------------|
|                          | Balanced Accuracy | F1 ↑       | Balanced Accuracy | F1 ↑       | Balanced Accuracy | F1 ↑       |
| Decoy MNIST Color        | 0.966 ± 0.003     | 0.966 ± 0.003 | 0.977 ± 0.001     | 0.977 ± 0.001 | 0.978 ± 0.001     | 0.978 ± 0.001 |
| Decoy MNIST Position     | 0.523 ± 0.025     | 0.524 ± 0.025 | 0.808 ± 0.012     | 0.808 ± 0.012 | 0.594 ± 0.028     | 0.594 ± 0.028  |
| ISIC                     | 0.836 ± 0.002     | 0.846 ± 0.026 | 0.494 ± 0.014     | 0.494 ± 0.014 | 0.408 ± 0.004     | 0.408 ± 0.004  |
| ISIC (No Patch Data)     | 0.744 ± 0.012     | 0.486 ± 0.022 | 0.752 ± 0.006     | 0.752 ± 0.006 | 0.728 ± 0.016     | 0.728 ± 0.016  |
| Oxford-IIIT-Pet          | 0.582 ± 0.002     | 0.579 ± 0.003 | 0.583 ± 0.007     | 0.583 ± 0.007 | 0.545 ± 0.006     | 0.545 ± 0.006  |
| SBD                      | 0.600 ± 0.007     | 0.558 ± 0.023 | 0.687 ± 0.009     | 0.687 ± 0.009 | 0.557 ± 0.019     | 0.557 ± 0.019  |

In the test dataset, $k_i$ is randomly sampled from the set of class labels. The location of the decoy is randomly placed both in the training and test dataset. In the other version called “location”, the location of the decoys follows the class label in the training dataset but is random in the testing dataset. The grayscale intensity is randomly drawn both for the training and testing datasets. The ISIC dataset is a dataset for skin cancer diagnosis [Codella et al., 2019, Tschandl et al., 2018]. We utilize only two classes, benign and malignant. We increase the importance of the malignant class in the loss because the dataset is heavily imbalanced. The version of ISIC dataset we use is curated by using code from Rieger et al. [2020]. The explanation feedback we use is also from Rieger et al. [2020]. Oxford-IIIT-Pet [Parkhi et al., 2012] consists of cat and dog images with 37 different classes of different cat and dog breeds. The semantic boundaries dataset (SBD) [Hariharan et al., 2011] dataset consists of images from the PASCAL VOC 2011 dataset [Everingham et al., 2011]. For the SBD, we use a subset of classes: bird, bus, cat, dog, horse by following Schramowski et al. [2020]. We only use samples where one and only one of these classes appears.

**Models.** We use the LeNet-5 [LeCun et al., 1998] model for the decoy MNIST datasets and AlexNet [Krizhevsky et al., 2012] for the other datasets. We load pretrained weights from PyTorch for AlexNet.

**Software and Hardware.** We used PyTorch Lightning to do the experiments [Falcon and team, 2024]. The experiments ran on a MacBook Pro 2023 with Apple M2 Max chip and 64 GB RAM. We used the MPS backend for GPU accelerated training. The metrics we compute are calculated using scikit-learn [Pedregosa et al., 2011]. The saliency maps are created using Captum [Kokhlikyan et al., 2020].

### 5.2 Predictive Performance

We compare the predictive performance of Bayesian CNNs without any feedback, using data argumentation with counterexamples [Schramowski et al., 2020, Teso and Kersting, 2019], and the method outlined above. The data augmentation approach is, as far as we know, the only approach compatible with Bayesian CNNs because it is model agnostic. For this approach, we first replace a region specified to be irrelevant by the explanation feedback with noise sampled from a uniform distribution on the interval $[0, 1)$ and afterward, we preprocess the images with standardization. We only use 70% of the explanation feedback available, since regularizing all training samples negatively impacts our method in some instances.

We observed during the experiments that there are no performance gains when we apply our method to models that are not focusing on spurious features or when models are not uncertain. That is, if we initialized weights with small variance we could not see performance gain in the datasets without spurious features because pretrained AlexNet weights from PyTorch are already near optimal for the model architecture. Instead, we want to demonstrate our method under the conditions that there are spurious features or when the models are uncertain by initializing with larger variance and compare it to the data augmentation method. The loss because the dataset is heavily imbalanced. The version of ISIC dataset we use is curated by using code from Rieger et al. [2020]. The explanation feedback we use is also from Rieger et al. [2020]. Oxford-IIIT-Pet [Parkhi et al., 2012] consists of cat and dog images with 37 different classes of different cat and dog breeds. The semantic boundaries dataset (SBD) [Hariharan et al., 2011] dataset consists of images from the PASCAL VOC 2011 dataset [Everingham et al., 2011]. For the SBD, we use a subset of classes: bird, bus, cat, dog, horse by following Schramowski et al. [2020]. We only use samples where one and only one of these classes appears.

We observed during the experiments that there are no performance gains when we apply our method to models that are not focusing on spurious features or when models are not uncertain. That is, if we initialized weights with small variance we could not see performance gain in the datasets without spurious features because pretrained AlexNet weights from PyTorch are already near optimal for the model architecture. Instead, we want to demonstrate our method under the conditions that there are spurious features or when the models are uncertain by initializing with larger variance and compare it to the data augmentation method. Tables[1] and[2] indicate that our method can improve model performance when models have overfitted to spurious features or the model is uncertain. The sample standard deviations shown in Tables[1] and[2] are computed by training three models using a 3-fold cross-validation and testing the three models on an independent test dataset.

For the data augmentation method, we see that the method can affect results negatively when the models are not overfitting to spurious features but still uncertain. This indicates that background information can be useful, but since the information is removed entirely, the models cannot take advantage of it. While our method tries to tell the models where to not look, we do not remove the information entirely and can use the hyperparameter $\lambda$ to balance this aspect.
Regularizing Explanations in Bayesian Convolutional Neural Networks

Figure 3: Examples of saliency maps on samples randomly drawn from the test dataset. More examples can be found in the link given on the first page.

Table 2: For dataset with more than two classes, we compute one-vs-rest to get the AUC scores. To compute overlap, we use input gradient for Decoy MNIST and Grad-CAM for the rest of the datasets. Some entries are missing standard deviation, since it is less than 0.001.

| Dataset             | No Regularization AUC ↑ | Overlap ↓ | Our Method AUC ↑ | Overlap ↓ | Data Augmentation AUC ↑ | Overlap ↓ |
|---------------------|-------------------------|-----------|------------------|-----------|-------------------------|-----------|
| Decoy MNIST Color   | 0.999                   | 0.028 ± 0.004 | 1.000           | 0.009     | 1.000                   | 0.007     |
| Decoy MNIST Position| 0.880 ± 0.011           | 0.033 ± 0.001 | 0.977 ± 0.003   | 0.015     | 0.887 ± 0.009           | 0.050 ± 0.002 |
| ISIC                | 0.921 ± 0.001           | 0.229 ± 0.008 | 0.920 ± 0.001   | 0.002     | 0.919 ± 0.002           | 0.252 ± 0.013 |
| ISIC (No Patch Data)| 0.849 ± 0.003           | n/a        | 0.850 ± 0.002   | n/a       | 0.850 ± 0.002           | n/a       |
| Oxford-IIIT-Pet     | 0.968 ± 0.001           | 0.121 ± 0.002 | 0.967 ± 0.001   | 0.176 ± 0.004 | 0.962 ± 0.009           | 0.127 ± 0.003 |
| SBD                 | 0.866 ± 0.004           | 0.538 ± 0.021 | 0.892 ± 0.004   | 0.515 ± 0.017 | 0.827 ± 0.008           | 0.541 ± 0.008 |

5.3 Model Focus

Table 2 demonstrate that our method is good at removing the models’ focus on spurious features. The overlap is computed by calculating how much importance is on the area the explanation feedbacks indicate as unimportant divided by the total amount of importance across the entire image. To do the overlap calculation, we use input gradient [Simonyan et al., 2014] for the MNIST dataset and we used Grad-CAM [Selvaraju et al., 2017] for the rest of the datasets. Figs. 3a to 3c show that our method can guide models away from spurious features and focus on what is important. For ISIC, data augmentation replace irrelevant regions with random noise but seems to be unable to make the models not look at patches. This indicates that when the location of features matter and not only their appearance, then counterexamples are unable to change model focus.

6 Conclusion and Discussion

We have introduced a new explanation regularization methods that is compatible with the Bayesian formalism. Our focus has been to introduce a method that can be used with Bayesian CNNs and not compete with methods trying to
improve model focus on regular NNs. Beyond this, we provide the opportunity to add domain knowledge in the input space. The experiments across four datasets show that our method can improve predictive performance of Bayesian CNNs when they overfit to spurious features or are uncertain where to focus. Moreover, we can remove focus on spurious features, no matter if it is because of appearance or their location.

While our method is simple, it has limitations. Like other explanation regularization methods, our method requires human labor to specify explanation feedback that can be labor-intensive. In the future, intelligent ways to obtain explanation feedback should be considered. We regularize across all channels in a region in the convolutional layers, which can potentially be undesirable. We should for future work investigate adaptive methods to intelligently select specific filters to regularize.

Acknowledgements

The underlying template for this paper is made by [Kour, 2020], licensed under the MIT License (https://github.com/kourgeorge/axriv-style/blob/master/LICENSE.txt). The mathematical notation we use follows [Goodfellow et al., 2016] closely.

References

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep Learning, MIT Press, 2016. URL http://www.deeplearningbook.org

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. Science, 2018.

Andre Esteva, Alexandre Robicquet, Bharath Ramsundar, Volodymyr Kuleshov, Mark DePristo, Katherine Chou, Claire Cui, Greg Corrado, Sebastian Thrun, and Jeff Dean. A guide to deep learning in healthcare. Nature Medicine, 2019.

B. Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A. Al Sallab, Senthil Kumar Yogamani, and Patrick Pérez. Deep Reinforcement Learning for Autonomous Driving: A Survey. IEEE Trans. Intell. Transp. Syst., 2022.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In Proc. of ICLR, 2014.

Sebastian Lapuschkin, Stephan Wäldchen, Alexander Binder, Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Unmasking Clever Hans predictors and assessing what machines really learn. Nature Communications, 2019.

Alejandro Barredo Arrieta, Natalia Díaz Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. Inf. Fusion, 2020.

Andrew Slavin Ross, Michael C. Hughes, and Finale Doshi-Velez. Right for the Right Reasons: Training Differentiable Models by Constraining their Explanations. In Proc. of IJCAI, 2017.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight Uncertainty in Neural Network. In Proc. of ICM, 2015.

Laurent Valentin Jospin, Hamid Laga, Farid Boussaïd, Wray L. Buntine, and Mohammed Bennamoun. Hands-On Bayesian Neural Networks - A Tutorial for Deep Learning Users. IEEE Comput. Intell. Mag., 2022.

Simon J.D. Prince. Understanding Deep Learning. The MIT Press, 2023.

Stefano Teso and Kristian Kersting. Explanatory Interactive Machine Learning. In Proc. of AIES, 2019.

Laura Rieger, Chandan Singh, W. James Murdoch, and Bin Yu. Interpretations are Useful: Penalizing Explanations to Align Neural Networks with Prior Knowledge. In Proc. of ICML, 2020.

Noel C. F. Codella, Veronica Rotemberg, Philipp Tschantl, M. Emre Celebi, Stephen W. Dusza, David A. Gutman, Brian Helba, Aadi Kalloo, Konstantinos Liopyris, Michael A. Marchetti, Harald Kittler, and Allan Halpern. Skin Lesion Analysis Toward Melanoma Detection 2018: A Challenge Hosted by the International Skin Imaging Collaboration (ISIC). CoRR, 2019.

Philipp Tschantl, Cliff Rosendahl, and Harald Kittler. The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. Scientific Data, 2018.

Kevin P. Murphy. Probabilistic Machine Learning: Advanced Topics. MIT Press, 2023.
Diederik P. Kingma, Tim Salimans, and Max Welling. Variational Dropout and the Local Reparameterization Trick. In Proc. of NIPS, 2015.

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational Inference: A Review for Statisticians. Journal of the American Statistical Association, 2017.

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In Proc. of ICLR, 2014.

Dmitry Molchanov, Arsenii Ashukha, and Dmitry P. Vetrov. Variational Dropout Sparsifies Deep Neural Networks. In Proc. of ICML, 2017.

David Gunning and David W. Aha. DARPA's Explainable Artificial Intelligence (XAI) Program. AI Mag., 2019.

Zachary C. Lipton. The mythos of model interpretability. Commun. ACM, 2018.

Frederick Liu and Besim Avci. Incorporating Priors with Feature Attribution on Text Classification. In Proc. of ACL, 2019.

Mengnan Du, Ninghao Liu, Fan Yang, and Xia Hu. Learning Credible Deep Neural Networks with Rationale Regularization. In Proc. of ICDM, 2019.

Xiaoting Shao, Arseny Skryagin, Wolfgang Stammer, Patrick Schramowski, and Kristian Kersting. Right for Better Reasons: Training Differentiable Models by Constraining their Influence Functions. In Proc. of AAAI, 2021.

Gabriel G. Erion, Joseph D. Janizek, Pascal Sturmfels, Scott M. Lundberg, and Su-In Lee. Improving performance of deep learning models with axiomatic attribution priors and expected gradients. Nat. Mach. Intell., 2021.

Ramprasaath Ramasamy Selvaraju, Stefan Lee, Yilin Shen, Hongxia Jin, Shalini Ghosh, Larry P. Heck, Dhruv Batra, and Devi Parikh. Taking a HINT: Leveraging Explanations to Make Vision and Language Models More Grounded. In Proc. of ICCV, 2019.

Patrick Schramowski, Wolfgang Stammer, Stefano Teso, Anna Brugger, Franziska Herbert, Xiaoting Shao, Hans-Georg Luigs, Anne-Katrin Mahlein, and Kristian Kersting. Making deep neural networks right for the right scientific reasons by interacting with their explanations. Nat. Mach. Intell., 2020.

Jason Ansel, Edward Z. Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Vozenensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, Geeta Chauhan, Anjali Chourdia, Will Constable, Alban Desmaison, Zachary DeVito, Elias Ellison, Will Feng, Jiong Gong, Michael Gschwind, Brian Hirsh, Sherlock Huang, Kshiteej Kalambarkar, Laurent Kirsch, Michael Lazos, Mario Lezzcano, Yanbo Liang, Jason Liang, Yinghai Lu, C. K. Luk, Bert Maher, Yunjie Pan, Christian Puhrsch, Matthias Reso, Mark Saroultim, Marcos Yukio Siraichi, Helen Suk, Shunting Zhang, Michael Suo, Phil Tillet, Xu Zhao, Eikan Wang, Keren Zhou, Richard Zou, Xiaodong Wang, Ajit Mathews, William Wen, Gregory Chanan, Peng Wu, and Soumith Chintala. PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation. In Proc. of ASPLOS, 2024.

Yann LeCun, Corinna Cortes, and CJ Burges. MNIST handwritten digit database. AT&T Labs, 2010. URL http://yann.lecun.com/exdb/mnist.

Omkar M. Parkhi, Pablo Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In Proc. of CVPR, 2012.

Bharath Hariharan, Pablo Arbeláez, Lubomir D. Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In Proc. of ICCV, 2011.

M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2011 (VOC2011) Results, 2011. URL http://host.robots.ox.ac.uk/pascal/VOC/.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. Proc. IEEE, 1998.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In Proc. of NIPS, 2012.

William Falcon and The PyTorch Lightning team. PyTorch Lightning, August 2024.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake VanderPlas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. Scikit-learn: Machine Learning in Python. J. Mach. Learn. Res., 2011.

Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kluishkina, Carlos Araya, Siqi Yan, and Orion Reblitz-Richardson. Captum: A unified and generic model interpretability library for PyTorch. CoRR, 2020.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. In Proc. of ICLR Workshop Track Proceedings, 2014.
Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. In Proc. of ICCV, 2017.

George Kour. A Latex style and template for paper preprints (based on NIPS style), 2020. URL https://github.com/kourgeorge/arcxiv-style MIT License, Copyright (c) 2020 George Kour.