How Does Predictive Uncertainty Quantification Correlate with the Plausibility of Counterfactual Explanations

Dimitar Nikolov
Supervisor(s): Patrick Altmeyer, Cynthia Liem

EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 23, 2024

Name of the student: Dimitar Nikolov
Final project course: CSE3000 Research Project
Thesis committee: Cynthia Liem, Patrick Altmeyer, Bernd Dudzik

Abstract

Counterfactual explanations can be applied to algorithmic recourse, which is concerned with helping individuals in the real world overturn undesirable algorithmic decisions. They aim to provide explanations to opaque machine learning models. Not all generated points are equally faithful to the model, nor equally plausible. On the other hand, predictive uncertainty quantification is used to measure the degree of certainty a model has in its predictions. Previously, it has been shown that it is possible to generate more plausible counterfactual explanations utilising predictive uncertainty. This work investigates this further by using multiple models innately supporting uncertainty quantification and comparing the produced counterfactual explanations to those produced by the models' ordinary counter-part. Predictive uncertainty tends to enhance the plausibility of the counterfactuals on visual datasets. Furthermore, we are positive that predictive uncertainty correlates proportionally with plausibility. This correlation has important implications for both research and real-world applications, as it suggests that integrating uncertainty quantification in model development can improve the quality and trustworthiness of algorithmic explanations.

1 Introduction

In the recent years, deep-learning models have achieved significant results in terms of predictive accuracy and performance [1]. However, this is by no means sufficient to gain trust in the decisions models take in critical applications concerning human lives like autonomous driving or medical diagnosis [2]. Predictive uncertainty quantification and explainability through counterfactuals are two procedures that are capable of improving the trust-worthiness of a model.

Predictive uncertainty aims to estimate the degree of certainty a model has in its predictions. There is a wide variety of different approaches to quantify uncertainty in a model [3]. The current paper focuses on some of the fundamental approaches for quantifying uncertainty. The Bayesian approach produces probability distributions instead of point estimates. In Bayesian deep learning, the weights in the network are treated as random variables in themselves and the goal is typically to estimate how they are distributed. However, the posterior distribution is intractable to solve analytically [4]. Some methods aim to approximate this distribution like Laplace Approximation [5] and Variational Inference [6] while others produce samples from it like Monte Carlo Dropout [7], Monte Carlo DropConnect [8], and Monte Carlo Markov Chain with Stochastic Gradient Langevin Dynamics [9]. Even though Deep Ensembles [10] are sometimes considered as competing approaches to the Bayesian, they can also be considered as "approximate Bayesian marginalisation" [11, p. 1].

However, uncertainty quantification is just a way to express to what extent the predictions of a model can be deemed as confidently made. While this approach helps in understanding the confidence level of model predictions, it does not directly address the need for interpretability. To gain more trust in model predictions, another crucial concept to consider is counterfactual explanations. Counterfactual explanations do not aim to open the black box of deep models, which are extremely complex in terms of the number of parameters [12]. Instead, they provide a means for algorithmic recourse, helping individuals in the real world overturn undesirable algorithmic decisions by perturbing the feature vector [13]. These perturbed results are not equally appropriate to explain the model. Altmeyer et al. [14] show that fidelity, which refers to how well explanations match the predictions of a black-box model, is not a sufficient measure to ensure faithfulness of the counterfactual explanations. They provide a definition of faithfulness that can lead to more suitable metrics to evaluate the quality of a counterfactual.

The main question in this work is **how these two ways to improve trust-worthiness correlate with one another**. Zhuang and Huang [15] adopt counterfactual explanations as a means to measure how well their BayesCNN can discriminate between simulated images and real images and thus preventing

the model from overconfident predictions. Singla et al. [16] use counterfactual generation as a way to augment their initial dataset and hence improve the robustness of their model. Schut et al. [17] suggest that predictive uncertainty can lead to more interpretable counterfactual explanations but they apply the uncertainty quantification technique in combination with adversarial training. However, the correlation between faithfulness of counterfactual explanations and just predictive uncertainty still remains unclear. Specifically, the gaps addressed in this work are the lack of structural comparison between deep models providing uncertainty quantification and their point-estimate counterparts. Additionally, there have not been evaluations considering different types and modalities of data.

The contributions in this work are as follows.

- Comparison of deep models providing uncertainty quantification with their point-estimate counterparts;
- Measuring correlation over datasets from different categories – synthetic, tabular and visual;
- Under the proposed framework, predictive uncertainty models produce more plausible counterfactual explanations for the visual dataset, and less plausible for the tabular datasets;

The rest of the paper proceeds as follows. In Section 2 background on counterfactual explanations and the approaches to estimate predictive uncertainty is provided. This is followed by a detailed explanation on the methodology used to measure the correlation in Section 3. The experimental setup is described in Section 4 and it is directly followed by the results from these experiments in Section 5. Limitations of the work and directions for future research can be found in Section 6 followed by the measures taken to make the research responsible in Section 7 and a conclusion in Section 8.

2 Background

Let $\mathcal{D} = (\mathcal{X}, y)$ be a multiclass labeled data where $\mathcal{X} = \{x\}_{i=1}^{N}$ is a set of N data points of dimension D and y are their labels. θ denotes the set of all the parameters of a deep model including its structure and $f_\theta(x)$ is the output of a model under the parameters θ. An artificial neural network, parameterised by θ, aims to find a point setting of the estimate θ_{ML}, satisfying the maximum likelihood criterion [18]:

$$\theta_{ML} = \arg\max_{\theta} p(\mathcal{D}|\theta) = \arg\max_{\theta} \log p(\mathcal{D}|\theta) \tag{1}$$

2.1 Counterfactual Explanations

Counterfactual explanations are not just a means to explain black-box models, but they can also be applied to algorithmic recourse in an understandable manner [14]. Let $x \in \mathcal{X}$ be an arbitrary data point, let $y = f_\theta(x)$ be the outcome of classifying the data point, and L be the loss function of the model. A counterfactual explanation is a point x' which is a perturbation of the original point x such that $f_\theta(x') = y'$ where y' is called the target class and $y' \neq y$. Altmeyer et al. [14] state that most counterfactual generators are gradient-based and rely on the following objective:

$$\min_{\mathbf{Z}' \in \mathcal{Z}^L} \{L(f_\theta(g(\mathbf{Z}')), y') + \lambda cost(g(\mathbf{Z}'))\} \tag{2}$$

In Equation 2 $g(\cdot)$ denotes a function that maps from a counterfactual state, that can potentially be a latent space, to the feature space and $cost(\cdot)$ is a penalty function that aims to regularise the produced counterfactuals.

This work sticks to the definitions of plausibility and faithfulness described by Altmeyer et al. [14] which are the following.

Definition 1. *Let $\mathcal{X}|y' = p(x|y')$ denote the true conditional distribution of samples in the target class y'. Then x' is considered a plausible counterfactual if $x' \sim \mathcal{X}|y'$.*

Definition 2. *Let $\mathcal{X}_\theta|y' = p_\theta(x|y')$ denote the conditional distribution of x in the target class y', where θ denotes the parameters of a model M_θ. Then x' is considered a faithful counterfactual if $x' \sim \mathcal{X}_\theta|y'$.*

Sticking to these definitions, faithfulness shifts from what plausibility defines as adhering to the data, to adhering to what the model learnt about the data. Intuitively, predictions with lower predictive uncertainty should be more plausible given the fact that the model has learnt the underlying data well [19].

Altmeyer et al.[14] argue that faithfulness of counterfactuals as desideratum is more important than plausibility, and for that reason they propose a gradient-based generator which uses the Energy Constraint Conformal objective (ECCo) that utilises two penalties – over the closeness and over the free energy of the model. This work lies on the assumption that ECCo generates more faithful counterfactuals than the generic gradient-based generator. We stick to the following formula to compute implausibility:

$$impl(x', X_{y'}) = \frac{1}{|X_{y'}|} \sum_{x \in X_{y'}} dist(x', x) \tag{3}$$

where $X_{y'}$ is a subsample of points in the class y' [14].

2.2 Predictive Uncertainty Quantification

The Bayesian approach treats the model as a distribution, thus allowing us to express our prior beliefs $p(\theta)$ about the parameters. When data is observed these beliefs are updated according to the Bayes' rule [18]:

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})} \tag{4}$$

Therefore, predictions for a new data point x^* are [18]:

$$p(y^*|x^*, \mathcal{D}) = \int p(y^*|x^*, \theta)p(\theta|\mathcal{D})\, d\theta \tag{5}$$

Bayesian neural networks (BNNs) have inherent support for capturing model uncertainties since their output is a distribution in itself. However, non-linearities in artificial neural networks (ANNs) make the integral in Equation 5 intractable, and thus it requires approximations. Deep neural networks often remain underspecified by the available data and multiple samples from the likelihood in Equation 5 can lead to "compelling explanations for the data" [11].

2.2.1 Variational Inference

Hinton and van Camp [20] and Graves [21] suggest that the integral in Equation 5 can be approximated by variational learning. Blundell et al. [6] further develop this idea by constructing their algorithm Bayes by Backpropagation (BBB). In essence, variational inference aims to find the parameters ϕ of a simpler and easier to sample distribution $q_\phi(\theta)$ on the parameters of the deep learning model that minimises the Kullback-Leibler divergence with the true posterior distribution:

$$\begin{aligned} \phi^* &= \arg\min_\phi \mathrm{KL}\left[q_\phi(\theta) \,||\, p(\theta|\mathcal{D})\right] = \\ &= \arg\min_\phi \mathrm{KL}\left[q_\phi(\theta) \,||\, p(\theta)\right] - \mathbb{E}_{q_\phi(\theta)}\left[\log p(\mathcal{D}|\theta)\right] \end{aligned} \tag{6}$$

Equation 6 is known as the expected lower bound (ELBO) or variational free energy [6]. Note that optimising this objective will produce a distribution q_ϕ that provides good explanation of the data while being still close to the prior [22]. The first line of the objective demonstrates the aim to minimise the divergence between the candidate distribution q and the true posterior, whereas the second line demonstrates that this is equivalent to maximising the expected log-likelihood while minimising the divergence of q from the prior. Therefore, optimising this loss function and sampling from the distribution yields approximate samples from the posterior. In this work the prior of the weights θ considered is a multivariate Gaussian.

2.2.2 Dropout and DropConnect

Dropout is a regularisation technique to prevent a model from over-fitting [23]. In an artificial neural network the feedforward transition from level l to level $l+1$ can be described as [24, p. 163]:

$$y^{(l+1)} = \sigma(W^{(l+1)}y^{(l)} + b^{(l+1)}) \tag{7}$$

, where y^l is the output vector of layer l, $W^{(l+1)}$ is the weight matrix between layer l and $l+1$, and b^{l+1} is the bias vector at layer $l+1$. Dropout introduces a variable $z^{(l)}$ for each layer l such that $z_i^{(l)} \sim Bernoulli(p)$, where p is the probability of dropout. Now the feedforward transition from level l to level $l+1$ becomes:

$$y^{(l+1)} = \sigma(W^{(l+1)}(y^{(l)} \circ z^{(l)}) + b^{(l+1)}) \tag{8}$$

, where $\circ$ denotes the piecewise mulitplication of the two vectors. In this way, dropout effectively eliminates certain perceptrons in the propagation [23].

Gal and Ghahramani [7] in their work show that a neural network applying non-linearities with arbitrary depth and dropout applied as a regularisation technique during training and prediction time is equivalent from a mathematical point of view to an approximation of a Bayesian neural network.

This idea is further developed by Mobiny et al. [8] who do not limit the idea of switching parts of the network off to whole perceptrons but generalise it to switching weights off. Let $Z^{(l+1)}$ be a matrix with the same size as $W^{(l+1)}$ and let each of the weights be distributed as $Z_{i,j}^{(l+1)} \sim Bernoulli(p)$. Then the feedforward transition in DropConnect becomes:

$$y^{(l+1)} = \sigma((W^{(l+1)} \circ Z^{(l+1)})y^{(l)} + b^{(l+1)}) \tag{9}$$

where $\circ$ denotes the piecewise multiplication of the two matrices. This work considers only dropout as an approximation technique since it lies on the assumption that they will perform very similarly in the context of the research.

2.2.3 Laplace Approximation

Laplace approximation constructs a Gaussian posterior following $q(\theta) = \mathcal{N}(\theta; \theta_{MAP}, \Sigma)$, where θ_{MAP} denotes the maximum a posteriori solution, i.e., $\theta_{MAP} = \arg\max_{\theta} \log p(\mathcal{D}|\theta) + \log p(\theta)$, and Σ is the inversion of the Hessian of the negative log posterior with respect to parameters, i.e., $\Sigma^{-1} = -\nabla^2_{\theta\theta}(\log p(\mathcal{D}|\theta) + \log p(\theta))|_{\theta=\theta_{MAP}}$ [25]. This matrix is computationally intractable in the general case due to the large number of parameters in the deep models, but using second-order optimisation techniques Botev et al.[26] show that blocks of this matrix can be efficiently computed. Ritter et al. [5] take this insight even further by modelling the posterior over the weights as a Gaussian, using Laplace approximation [27] with Kronecker factored covariance matrices.

2.2.4 Stochastic Gradient Descent with Langevin Dynamics

Robbins and Monro [28] define a class of methods called stochastic optimisation that uses a subset of the original data and uses it to approximate the true gradient over the whole dataset in the following manner. Let $X_t = \{x_{t1}, ...x_{tn}\}$ be a subset with size n of the original dataset taken at each iteration t. The update equation for the parameters is given by:

$$\Delta\theta_t = \frac{\epsilon_t}{2}(\nabla \log p(\theta_t) + \frac{N}{n}\sum_{i=1}^{n} \nabla \log p(x_{ti}|\theta_t)) \tag{10}$$

where ϵ_t is a sequence of step sizes [9]. Constraints of significant importance for the step size are:

$$\sum_{t=1}^{\infty} \epsilon_t = \infty \qquad \sum_{t=1}^{\infty} \epsilon_t^2 < \infty \tag{11}$$

The first constraint guarantees that the parameters will reach high-probability regions regardless of their initial values, while the second constraint ensures that the parameters will converge to the mode instead of fluctuating around it [9]. The step size $\epsilon_t = a(b+t)^{-\gamma}$ is usually determined with polynomial decay over time where $\gamma \in (0.5, 1]$ [9]. Welling and Teh [9] utilise a class of Monte Carlo Markov chain (MCMC) techniques called Langevin dynamics which uses Robbins-Monro [28] optimisation with additional amount of Gaussian noise, normalised by the step size. This can be expressed in the following equation:

$$\Delta\theta_t = \frac{\epsilon_t}{2}(\nabla \log p(\theta_t) + \frac{N}{n}\sum_{i=1}^{n} \nabla \log p(x_{ti}|\theta_t)) + \eta_t \qquad \eta_t \sim \mathcal{N}(0, \epsilon_t) \tag{12}$$

They show that for large t the updates cease to be gradient updates and approach Langevin dynamics, which converges to the posterior distribution.

2.2.5 Deep Ensembles

Deep ensembles are a stack of M models which are trained independently and they all get a vote at prediction time [24, p.249]. Therefore, an ensemble can be represented as a set of parameters $\{\theta_m\}_{m=1}^{M}$. Lakshminarayanan et al. [10] show that the different predictions from the models participating in the ensemble can be used to estimate the uncertainty of the model. For simplicity, in this paper the ensemble will consist only of models with one and the same architecture and will be trained through one and the same procedure [10].

3 Methodology

In this section we will describe the methodology that has been used in order to quantify the degree of correlation between predictive uncertainty quantification and plausibility of counterfactuals. In Subsection 3.1 the framework for making models comparable is described followed by the metrics used to compute predictive uncertainty in Subsection 3.2.

3.1 Making Models Comparable

In order to make models comparable, this paper adheres to the following framework. For each dataset that is being evaluated, a base artificial neural network with a training procedure is defined. This base network is used as a template for the considered approaches. From now on we refer to the structure of the different models as skeletons (e.g. the deep ensemble skeleton for MNIST dataset) and individual concrete realisations of those skeletons as instances.

When it comes to deep ensembles, the skeleton for each dataset is defined as using a stack of M models with identical architecture and training procedure to the base artificial neural network of the respective dataset. The dropout skeleton also has an identical architecture and training procedure as the base ANN but has additional dropout layers with probabilities depending on the dataset. It is important to note that those dropout probabilities will not be switched off during prediction time and predictive uncertainty will be evaluated over multiple non-deterministic predictions following the work of Gal and Ghahramani [7]. The variational inference and stochastic gradient descent with Langevin dynamics skeletons are chains of models structurally identical to the artificial neural network base model. These chains of model realisations are treated as an ensemble of models when predictions are made and predictive uncertainty is computed. The Laplace approximation skeleton utilises a trained base ANN and fits a Laplace approximation to that.

3.2 Measure for Predictive Uncertainty

Gal [29, p. 51] in their work compare three different approaches to quantify uncertainty – variation ratios [30], predictive entropy [31], and mutual information [31]. This work will opt for using

predicitive entropy as a means to estimate predictive uncertainty because of its simplicity and intuitiveness. The formula according to which it is being computed is the following:

$$\mathbb{H}\left[y^*|x^*, \mathcal{D}\right] := -\sum_c p(y^* = c|x, \mathcal{D}) \log p(y^* = c|x, \mathcal{D}) \tag{13}$$

This formula is suitable to gain a measure for uncertainty in all of the above-mentioned models.

Another metric taken into consideration is the Expected Calibration Error of a model which is another means to asses the predictive uncertainty of models. Calibration refers to the degree to which a model's predicted probabilities of outcomes reflect the true likelihood of those outcomes. A well-calibrated model will have predicted probabilities that match the actual probabilities of the outcomes. This metric is global and is being computed per instance of a model, while predictive entropy is a local measure and is being computed per predicted sample. This measure can be expressed by the following formula

$$ECE = \sum_{t=1}^{T} \frac{|B_t|}{n} |acc(B_t) - conf(B_t)| \tag{14}$$

where T is the number of bins and B_t is the set of predictions in the t^{th} bin [32]. The accuracy of the predictions in each bin – $acc(.)$, and the confidence of the predictions in each bin – $conf(.)$, are evaluated as shown in Equation 15 and Equation 16 respectively.

$$acc(B_t) = \frac{1}{|B_t|} \sum_{i \in B_t} 1(\hat{y}_i = y_i) \tag{15}$$

$$conf(B_t) = \frac{1}{|B_t|} \sum_{i \in B_t} \hat{p}_i \tag{16}$$

The intuition behind these computations in ECE is to group predictions with similar confidence levels, allowing for a comparison between the model's predicted probabilities and the actual outcomes within each group.

4 Experimental Setup

The purpose of this section is to elaborate on the way experiments have been conducted, datasets involved in the experiments together with the structure of the deep models, and the packages used for the training procedure, and metrics used for evaluation of the correlation.

4.1 Base Models and Datasets

Our work considers more simplistic deep models because of performance reasons. More complex models require more time for the experiments which are conducted. Therefore, complex architectures for some of the datasets like LeNet-5 [33] are not considered. We evaluate our findings over 4 different datasets capturing different properties of data. These diverse qualities of the data are important to verify how they influence the results. For interpretability and intuitiveness of the results, we first consider a synthetic dataset of linearly separable points from TAIJADATA.JL [34]. We also performed experiments over two datasets containing tabular data – California Housing [35] and German Credit [36]. Finally, we consider MNIST [33] as an image dataset. Due to its large data size we will consider just a subset consisting of the first 10,000 samples from it for training.

All the base ANN models for different datasets contain a single hidden layer of fully connected perceptrons. The activation function for the hidden and the output layers are respectively ReLU (rectified linear unit) [37] and softmax while the loss function is cross-entropy. The choice of output

Dataset	Input	Hidden	Output	Epochs	Train Batch	Accuracy	Samples
Linearly Separable	2	5	2	4	16	99.60%	100
California Housing	8	12	2	10	16	84.78%	200
German Credit	20	40	2	10	16	78.90%	200
MNIST	784	32	10	10	32	95.20%	400

Table 1: Structure, training parameters and accuracy of the fully connected ANNs used for different datasets. Samples refer to the size of the batch, sampled to perform experiments on.

layer activation function and the loss function are based on the fact that we consider classification in our research and the output of the deep model should be a valid probability vector. Further information on the training procedure and structure of the ANN can be found in Table 1. When it comes to implementation we use FLUX.JL [38, 39] for the Artificial Neural Networks.

4.2 Uncertainty Quantification Models

As mentioned in Section 3 the template for the models supporting uncertainty quantification will be the same as the base ANN model. In this work we stick to default parameters described later on for the predictive uncertainty skeletons since the focus is not performance in terms of predictive accuracy. The aim is to make different skeletons as comparable as possible and for that reason the procedures are tuned to be as close as possible to the base ANN model.

We will use 5 models stacked in an ensemble as Lakshminarayanan et al. [10] suggest in their work. The code base can be found in WHAT-MAKES-MODELS-EXPLAINABLE.JL [40] package.

When it comes to variational inference, using multi-variate Guassian as a prior over the weights, we produce 5 samples from the posterior and treat them as an ensemble to produce multiple predictions. The parameters of the Gaussian prior are mean 0 and standard deviation $\frac{1}{\alpha}$ where $\alpha = 0.09$ [41]. In this work we utilise the TURING.JL [41] package to perform variational inference, and more concretely the state-of-the-art No-U-Turn-Sampler of TURING.JL with acceptance probability 0.65 was used to sample from the posterior. Since this package's main focus is not Bayesian Neural Networks, it turns out to be computationally prohibitive to perform experiments on the higher dimensional datasets like German Credit and MNIST.

The implementation of the Langevin dynamics BNN follows closely the work of Welling and Teh [9] and was implemented by Callh [42] in his blog post. This work generalises the approach to multiclass classification. We use 1,000 steps to converge and pick 15 of these steps to form 15 models that are treated as an ensemble to make predictions and measure uncertainty.

Dropout uncertainty estimation is quite straightforward since it is implemented using the Dropout layer from the FLUX.JL [38, 39]. This layer is applied after the hidden layer in each model with probability 0.1 and for higher dimensional data (German Credit and MNIST) also applied after the input layer again with probability 0.1. To estimate uncertainty we make predictions 10 times with the Dropout layers enabled.

Finally, for the Laplace approximation we use the LAPLACEREDUX.JL [43] which follows closely the work of Daxberger et al. [44].

4.3 Counterfactual Explanations

This work utilises the COUNTERFACTUALEXPLANATIONS.JL [45] package to generate counterfactual explanations. The procedure of evaluating correlation consists of the following steps. First, for each dataset we sample 5 random batches of the points. These batches will later be used for counterfactual generation. Using the whole datasets will be computationally expensive, while using a single sample can lead to over-optimistic or over-pessimistic results. The size of those batches depends on the size of

Model	Generator	Linearly Separable	California Housing	German Credit	MNIST
Neural Network	Generic	0.83	0.02	−0.75	0.98
	ECCo	0.79	−0.33	−0.90	0.94
Dropout	Generic	0.46	−0.36	−0.54	0.59
	ECCo	0.48	−0.15	−0.76	−0.23
Ensemble	Generic	0.60	0.87	−0.23	0.51
	ECCo	0.90	0.80	−0.53	−0.08
Langevin Dynamics	Generic	0.86	0.10	0.66	–
	ECCo	0.77	0.29	0.73	–
Laplace	Generic	0.84	0.03	–	–
	ECCo	0.84	0.84	–	–
Variational Inference	Generic	0.38	−0.04	–	–
	ECCo	0.62	0.89	–	–

Table 2: Correlation coefficient of 10 instances of a skeleton of a model between the average plausibility and average entropy of the counterfactuals produced by the model instance. Dash(–) means that the computation is prohibitive by the setup.

the dataset. Concrete sample sizes are shown in Table 1. Second, for each dataset two counterfactual generators are defined – generic generator with a closeness penalty of 0.0 and ECCo generator with closeness penalty 0.0 and energy penalty of 0.5. The closeness penalty would inevitably influence the plausibility term since it is distance based. This is why it is important that the two generators have the same closeness penalty. Third, for each model skeleton 10 instances are trained according to the procedure of the skeleton defined. This practice is used again to avoid over-optimistic or over-pessimistic results by avoiding initialisation pits.

For each of the instances of a model the expected calibration error (ECE) is evaluated over the whole dataset. Then for each of the instances of a model one counterfactual is generated for each factual in each of the samples with a random label. After that, for each counterfactual the predictive entropy and implausibility are computed. The average of these measures per sample are taken. For each instance of a model the mean and the standard deviation of the metrics are computed and then for each skeleton the mean and the standard deviation of the means are taken as a metric how good the model is performing.

To measure correlation, the Pearson correlation coefficient [46] is computed between mean predictive entropy and plausibility and between ECE of an instance and mean plausibility of samples per instance. Another comparison is made by seeing which models produce most plausible counterfactuals on average per skeleton. The results are reported more concretely in Section 5.

5 Results and Discussion

In this section the results from the experiments are presented and discussed. As mentioned, we evaluate the correlation between the plausibility metric and the two predictive uncertainty metrics – predictive entropy and expected calibration error. Additionally, we look at the average plausibility measure of the generated counterfactuals per model skeleton. Intuitively, the expectations are that more plausible counterfactual explanations will have lower predictive uncertainty.

Looking at the correlation coefficients between average plausibility per instance of a model and average entropy of the various instances of the model in Table 2, a general trend of positive correlation

Model	Generator	Linearly Separable	California Housing	German Credit	MNIST
Neural Network	Generic	0.10	0.28	−0.56	−0.84
	ECCo	0.04	0.25	−0.60	−0.92
Dropout	Generic	−0.27	0.17	−0.12	0.31
	ECCo	−0.15	0.18	−0.23	0.26
Ensemble	Generic	−0.81	0.80	0.08	−0.17
	ECCo	−0.72	0.77	−0.25	0.06
Langevin Dynamics	Generic	−0.71	−0.61	−0.87	–
	ECCo	−0.63	−0.41	−0.24	–
Laplace	Generic	−0.42	0.07	–	–
	ECCo	−0.26	0.40	–	–
Variational Inference	Generic	−0.16	0.03	–	–
	ECCo	0.09	−0.22	–	–

Table 3: Correlation coefficient between 10 instances of a skeleton of a model between the average plausibility of the counterfactuals produced by the model instance and the expected calibration error of the model instance evaluated over the dataset. Dash(–) means that the computation is prohibitive by the setup.

Model	Generator	Linearly Separable	California Housing	German Credit	MNIST
Neural Network	Generic	$\mathbf{3.72 \pm 0.81}$	$\mathbf{3.24 \pm 0.03}$	6.19 ± 0.05	19.45 ± 0.21
	ECCo	4.02 ± 1.22	3.33 ± 0.08	6.75 ± 0.28	19.46 ± 0.20
Dropout	Generic	3.99 ± 0.69	3.27 ± 0.03	6.21 ± 0.02	18.90 ± 0.04
	ECCo	3.88 ± 0.86	3.38 ± 0.05	6.90 ± 0.10	18.91 ± 0.04
Ensemble	Generic	5.06 ± 0.75	3.38 ± 0.03	$\mathbf{6.18 \pm 0.02}$	$\mathbf{18.89 \pm 0.03}$
	ECCo	5.83 ± 1.04	3.64 ± 0.20	7.21 ± 0.08	18.93 ± 0.05
Langevin Dynamics	Generic	4.34 ± 1.33	3.49 ± 0.14	6.30 ± 0.07	–
	ECCo	4.42 ± 1.49	4.27 ± 0.60	6.60 ± 0.13	–
Laplace	Generic	3.97 ± 0.85	3.32 ± 0.02	–	–
	ECCo	4.11 ± 1.33	3.59 ± 0.23	–	–
Variational Inference	Generic	3.78 ± 0.84	3.36 ± 0.04	–	–
	ECCo	3.92 ± 1.12	3.38 ± 0.07	–	–

Table 4: Average plausibility per model skeleton and dataset for different generators. Dash (–) means that the computation is prohibitive by the setup. Bolded results are the best average plausibility scored per dataset.

can be observed. For most of the model skeletons like the Synthetic dataset and the California housing dataset, higher entropy means less plausible counterfactuals. This is intuitive in the sense that models that are more confident about their prediction will be more capable to draw an accurate decision boundary and, therefore, produce more reliable counterfactual explanations. The only dataset that makes an exception in this case is the German Credit. The explanation provided here is that this dataset is more complicated than the others in the following directions. It is a tabular dataset and unlike California Housing it has a significantly higher dimension. Moreover, it consists of only 1000 samples which is significantly less compared to California Housing's 5000 samples. Apart from that, there is a significant imbalance between the classes. Finally, the base artificial neural network used for predictions has the lowest average predictive accuracy in comparison to the other datasets as shown in Table 1. Therefore, due to its innate complexity the German Credit dataset makes an exception in the results it achieves in comparison to the other datasets.

The results from the comparison of average plausibility of the instances per skeleton and the evaluation of the expected calibration error are inconclusive. Table 3 shows that the simple linearly separable dataset experiences a negative trend in almost all of the examined skeletons – less calibrated models produce more plausible counterfactual explanations, contrary to the expectations. Moreover, it can be observed that the models experiencing harshest negative correlation per dataset also produce least plausible counterfactuals on average when comparing the results to Table 4. Namely, the ensemble of the Linearly separable dataset has the lowest correlation coefficients – -0.81 and -0.72, and also produces the least plausible counterfactuals on average with mean distance 5.06 and 5.83. The case is the same for the other datasets.

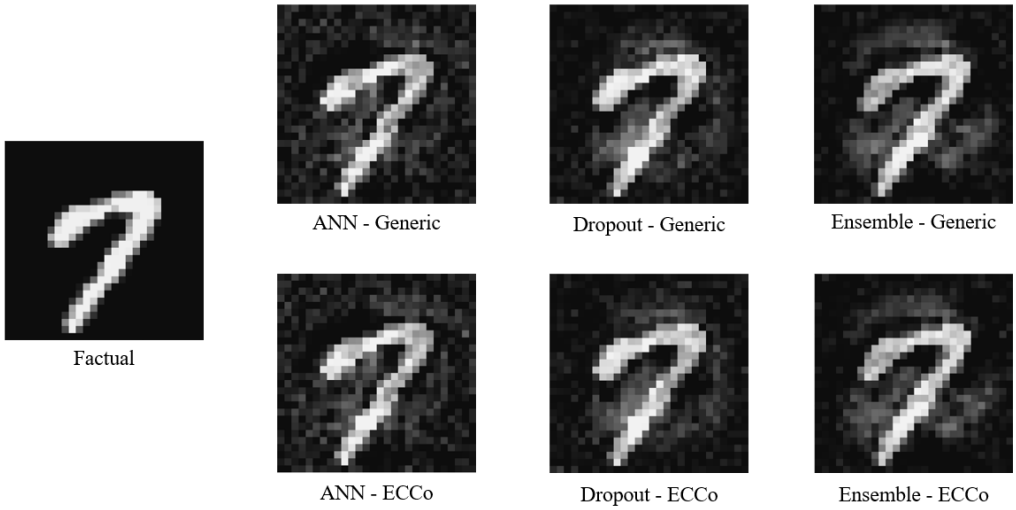

Figure 1: Generated counterfactuals on MNIST dataset with different instances of models. The factual class is '7' and the target class is '2'

Finally, in Table 4 we can observe that the average plausibility of models supporting predictive uncertainty quantification produce less plausible counterfactual explanations for the simpler datasets, and more plausible for the more complex ones. It is worth to mention that these results do not show that support for uncertainty quantification implies more plausible counterfactual explanations in all the cases. There are certain places where uncertainty quantification models perform significantly worse than the basic models, e.g. the ensemble for the linearly separable dataset is two standard deviations away from what the basic model achieves. On the other hand, MNIST uncertainty quantification models perform significantly better than the plain model. This can be explained by the fact that the predictive uncertainty models become more hesitant in their predictions overall when trained on tabular and linearly separable datasets [47].

Looking qualitatively at the results in Figure 1, it can be observed that the generated counterfactuals by the artificial neural network on the MNIST dataset look more like adversarial examples. On the contrary, it can be easily spotted that the ensemble has generated a silhouette of the target class in the background. This shows that models with support for predictive uncertainty do have a better understanding of the features of the underlying data and for that reason they are capable of producing more plausible counterfactuals. It is possible that properties of the visual data make it more suitable for deep learning models, since this is the only dataset that shows significant quantitative and qualitative improvement in comparison to the base artificial neural network.

6 Limitations and Future Work

Even though different approaches were taken to analyse how predictive uncertainty quantification and counterfactual explanations, limitations can easily be identified.

Firstly, the plausibility measure still suffers from the fact that a distance measure may not be the most accurate means to evaluate plausibility over different kinds of data [14]. Tabular data is not guaranteed to be in clusters for example, while for the linearly separable dataset used this metric makes more sense. Furthermore, based on the qualitative results on the MNIST dataset it can be seen that the counterfactuals generated will not be close to the original dataset distance-wise but they are significantly more plausible from a human point of view.

Secondly, packages in Julia are not well-suited for Bayesian deep learning in the context of variational inference. Most of them are not intuitive to use and support just a single output layer [48, 49]. The most suitable package used was TURING.JL [41] but it had its limitations in the sampling efficiency. This meant that experiments over larger models like the ones for the German Credit dataset and the MNIST dataset ware computationally infeasible. It is our firm recommendation that further research into how to optimise the process of variational inference for deep models such that experiments can be conducted.

Thirdly, we have identified that the only dataset that consistently performs better with the predictive uncertainty models is the visual dataset. It is possible that deep learning models capture better the features of visual datasets and predictive uncertainty helps with the more accurate learning of the decision boundary but further research is required into this topic.

Last but not least, there is no suitable implementation of DropConnect [8] compatible with FLUX.JL. Implementation of such a layer will benefit regularisation layers further and will enable the conduct of experiments for correlation with this approach.

7 Responsible Research

Research in the field of machine learning is mainly concerned with privacy issues. The datasets used in this research are a part of the package TAIJADATA.JL [34]. These datasets are under the MIT free license and are completely anonymised, thus posing no privacy issues. Reproducibility of the results is ensured through saving the models used to produce them in files. The training procedure behind each model and the structure of each model are explicitly specified in Section 4. Exact reproduction may not be possible due to stochasticity in the processes, but through large samples and averaging over the outcomes we ensure lack of bias. Examples of such assurance is that the models are trained 10 times to avoid optimistically good or pessimistically bad initialisations and metrics are evaluated on multiple random samples. This work is concrete about how the experiments were conducted and which parts of the larger datasets were omitted in the result report, as specified by the Netherlands Code of Conduct for Research Integrity [50, p. 17]. Furthermore, the whole code base is present in the package WHAT-MAKES-MODELS-EXPLAINABLE.JL [40]. Last but not least, every work contributing to this paper is explicitly stated through a reference.

8 Conclusion

This work looks deeper into the question how predictive uncertainty quantification correlates with plausibility of counterfactual explanations. It provides a framework to make models produced by different approximations to Bayesian neural networks comparable with deep ensembles and artificial neural networks. The correlation between plausibility as defined in this paper and local predictive uncertainty measured through predictive entropy is positive in the general case. Furthermore, within this framework our proposed models show that predictive uncertainty quantification rarely produces better counterfactuals on tabular datasets. However, the visual dataset selected shows quantitative and qualitative improvement when the more advanced procedures are applied. In conclusion, we can state that predictive uncertainty does have a positive correlation factor with plausibility, but the effect of the model on the plausibility objective depends on the underlying data and the model skeleton used to evaluate it.

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[2] M. Hasan, A. Khosravi, I. Hossain, A. Rahman, and S. Nahavandi, "Controlled Dropout for Uncertainty Estimation," May 2022. arXiv:2205.03109 [cs].

[3] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya, V. Makarenkov, and S. Nahavandi, "A Review of Uncertainty Quantification in Deep Learning: Techniques, Applications and Challenges," *Information Fusion*, vol. 76, pp. 243–297, Dec. 2021. arXiv:2011.06225 [cs].

[4] V. Mullachery, A. Khera, and A. Husain, "Bayesian Neural Networks," 2018.

[5] H. Ritter, A. Botev, and D. Barber, "A SCALABLE LAPLACE APPROXIMATION FOR NEURAL NETWORKS," *International Conference on Learning Representations*, 2018.

[6] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight Uncertainty in Neural Networks," May 2015. arXiv:1505.05424 [cs, stat].

[7] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian Approximation: Appendix," May 2016. arXiv:1506.02157 [stat].

[8] A. Mobiny, H. V. Nguyen, S. Moulik, N. Garg, and C. C. Wu, "DropConnect Is Effective in Modeling Uncertainty of Bayesian Deep Networks," June 2019. arXiv:1906.04569 [cs, stat].

[9] M. Welling and Y. W. Teh, "Bayesian Learning via Stochastic Gradient Langevin Dynamics," *International Conference on Machine Learning*, pp. 681–688, 2011.

[10] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles," Nov. 2017. arXiv:1612.01474 [cs, stat].

[11] A. G. Wilson, "The Case for Bayesian Deep Learning," Jan. 2020. arXiv:2001.10995 [cs, stat].

[12] S. Chakraborty, R. Tomsett, R. Raghavendra, D. Harborne, M. Alzantot, F. Cerutti, M. Srivastava, A. Preece, S. Julier, R. M. Rao, T. D. Kelley, D. Braines, M. Sensoy, C. J. Willis, and P. Gurram, "Interpretability of deep learning models: A survey of results," in *2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, (San Francisco, CA), pp. 1–6, IEEE, Aug. 2017.

[13] S. Wachter, B. Mittelstadt, and C. Russell, "Counterfactual Explanations Without Opening the Black Box: Automated Decisions and the GDPR," *SSRN Electronic Journal*, 2017.

[14] P. Altmeyer, M. Farmanbar, A. van Deursen, and C. C. S. Liem, "Faithful Model Explanations through Energy-Constrained Conformal Counterfactuals," Dec. 2023. arXiv:2312.10648 [cs].

[15] Y. Zhuang and Z. Huang, "SAR Image Authentic Assessment with Bayesian Deep Learning and Counterfactual Explanations," in *Pattern Recognition and Computer Vision* (Q. Liu, H. Wang, Z. Ma, W. Zheng, H. Zha, X. Chen, L. Wang, and R. Ji, eds.), vol. 14428, pp. 442–454, Singapore: Springer Nature Singapore, 2024. Series Title: Lecture Notes in Computer Science.

[16] S. Singla, N. Murali, F. Arabshahi, S. Triantafyllou, and K. Batmanghelich, "Augmentation by Counterfactual Explanation -Fixing an Overconfident Classifier," in *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, (Waikoloa, HI, USA), pp. 4709–4719, IEEE, Jan. 2023.

[17] L. Schut, O. Key, R. McGrath, L. Costabello, B. Sacaleanu, M. Corcoran, and Y. Gal, "Generating Interpretable Counterfactual Explanations By Implicit Minimisation of Epistemic and Aleatoric Uncertainties," Mar. 2021. arXiv:2103.08951 [cs, stat].

[18] J. Antorán, *Understanding Uncertainty in Bayesian Neural Networks*. PhD thesis, University of Cambridge, 2019.

[19] J. Antorán, U. Bhatt, T. Adel, A. Weller, and J. M. Hernández-Lobato, "Getting a CLUE: A Method for Explaining Uncertainty Estimates," Mar. 2021. arXiv:2006.06848 [cs, stat].

[20] G. E. Hinton and D. Van Camp, "Keeping the neural networks simple by minimizing the description length of the weights," in *Proceedings of the sixth annual conference on Computational learning theory - COLT '93*, (Santa Cruz, California, United States), pp. 5–13, ACM Press, 1993.

[21] A. Graves, "Practical Variational Inference for Neural Networks," in *Advances in Neural Information Processing Systems*, vol. 24, 2011.

[22] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning," Oct. 2016. arXiv:1506.02142 [cs, stat].

[23] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014.

[24] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. Adaptive computation and machine learning, Cambridge, Mass: The MIT press, 2016.

[25] Z. Deng, F. Zhou, and J. Zhu, "Accelerated Linearized Laplace Approximation for Bayesian Deep Learning," Oct. 2022. arXiv:2210.12642 [cs].

[26] A. Botev, H. Ritter, and D. Barber, "Practical Gauss-Newton Optimisation for Deep Learning," June 2017. arXiv:1706.03662 [stat].

[27] D. J. C. MacKay, "A Practical Bayesian Framework for Backpropagation Networks," *Neural Computation*, vol. 4, pp. 448–472, May 1992.

[28] H. Robbins and S. Monro, "A Stochastic Approximation Method," *The Annals of Mathematical Statistics*, vol. 22, pp. 400–407, Sept. 1951.

[29] Y. Gal, "Uncertainty in Deep Learning," p. 52, 2016.

[30] L. Freeman, *Elementary Applied Statistics: For Students in Behavioral Science*. For Students in Behavioral Science, Wiley, 1965.

[31] C. E. Shannon, "A Mathematical Theory of Communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, July 1948.

[32] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On Calibration of Modern Neural Networks," Aug. 2017. arXiv:1706.04599 [cs].

[33] Y. LeCun, L. Bottou, Y. Bengio, and P. Ha, "Gradient-Based Learning Applied to Document Recognition," 1998.

[34] P. Altmeyer, "TaijaData.jl," Nov. 2023. original-date: 2023-11-10T11:38:33Z.

[35] R. Kelley Pace and R. Barry, "Sparse spatial autoregressions," *Statistics & Probability Letters*, vol. 33, pp. 291–297, May 1997.

[36] H. Hofmann, "Statlog (German Credit Data)," 1994.

[37] A. F. Agarap, "Deep Learning using Rectified Linear Units (ReLU)," Feb. 2019. arXiv:1803.08375 [cs, stat].

[38] M. Innes, "Flux: Elegant Machine Learning with Julia," *Journal of Open Source Software*, 2018.

[39] M. Innes, E. Saba, K. Fischer, D. Gandhi, M. C. Rudilosso, N. M. Joy, T. Karmali, A. Pal, and V. Shah, "Fashionable Modelling with Flux," *CoRR*, vol. abs/1811.01457, 2018. _eprint: 1811.01457.

[40] P. Altmeyer, "JuliaTrustworthyAI/what-makes-models-explainable: Repo for our research paper "What Makes Models Explainable? Evidence from Counterfactuals"," 2024.

[41] H. Ge, K. Xu, and Z. Ghahramani, "Turing: a language for flexible probabilistic inference," in *International Conference on Artificial Intelligence and Statistics, AISTATS 2018, 9-11 April 2018, Playa Blanca, Lanzarote, Canary Islands, Spain*, pp. 1682–1690, 2018.

[42] S. Callh, "Bayesian inference with Stochastic Gradient Langevin Dynamics | Sebastian Callh personal blog."

[43] P. Altmeyer, "LaplaceRedux.jl," Feb. 2022. Version Number: v0.1.0.

[44] E. Daxberger, A. Kristiadi, A. Immer, R. Eschenhagen, M. Bauer, and P. Hennig, "Laplace Redux – Effortless Bayesian Deep Learning," Mar. 2022. arXiv:2106.14806 [cs, stat].

[45] P. Altmeyer, A. van Deursen, and C. C. S. Liem, "Explaining Black-Box Models through Counterfactuals," *JuliaCon Proceedings*, vol. 1, p. 130, Aug. 2023.

[46] K. Pearson and G. Francis, "VII. Note on regression and inheritance in the case of two parents," *Proceedings of the Royal Society of London*, vol. 58, pp. 240–242, Dec. 1895.

[47] S. A. Fayaz, M. Zaman, S. Kaul, and M. A. Butt, "Is Deep Learning on Tabular Data Enough? An Assessment," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 4, 2022.

[48] A. A. Wijaya, "andrewwijaya/CS5099-julia-bnn," May 2023. original-date: 2022-08-08T22:19:16Z.

[49] M. Kochenderfer, "sisl/BayesNets.jl," 2024.

[50] KNAW, NFU, NWO, TO2-Federatie, Vereniging Hogescholen, and VSNU, "Nederlandse gedragscode wetenschappelijke integriteit," 2018.

A Hardware Setup

All the experiments were conducted on Predator Triton 500 PT515-51 with Intel(R) Core(TM) i7-9750H CPU. The whole experimental loop for a skeleton takes 45 minutes on average. An exception to that is the Laplace experiment over the California Housing dataset that took 1000 minutes.

B Qualitative Results

Figures 2 to 7 show counterfactuals generated with different generators with different instances of models. The generation is conducted by selecting ten random samples representing each class. For each of those samples, every counterfactual is generated for each target class. The observed behaviour is a transition from counterfactuals looking like adversarial inputs generated with ANN (Figure 2 and Figure 3) to counterfactuals having features from the target class as a silhouette in the background generated with deep ensemble (Figure 6 and Figure 7).

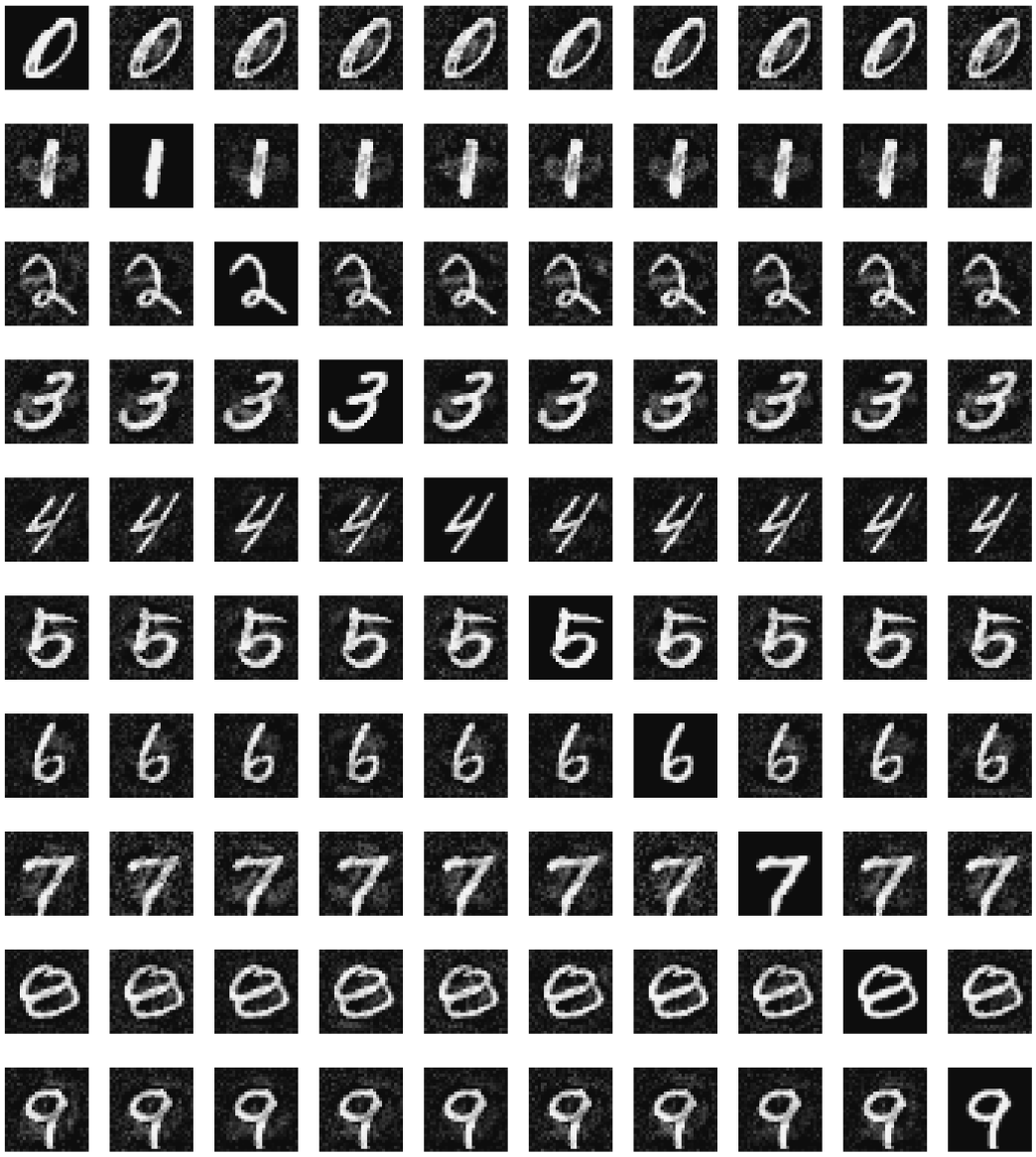

Figure 2: Table with counterfactuals generated with an ordinary artificial neural network and a generic gradient-based generator. The factuals are positioned along the main diagonal of the grid, the number of the row is the original label of the factual, while the number of the column shows the target of the label

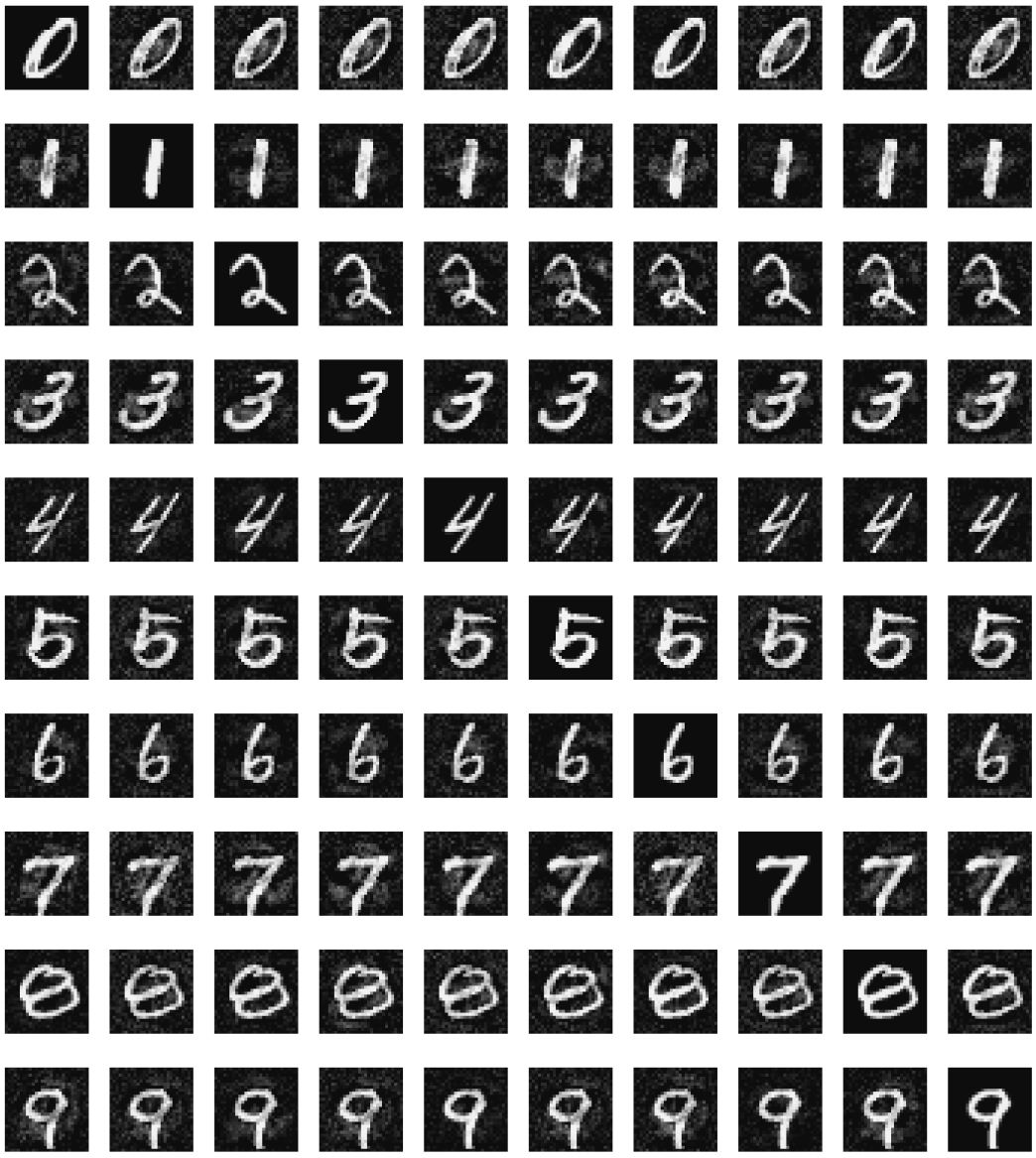

Figure 3: Table with counterfactuals generated with an ordinary artificial neural network and the ECCo generator. The factuals are positioned along the main diagonal of the grid, the number of the row is the original label of the factual, while the number of the column shows the target of the label

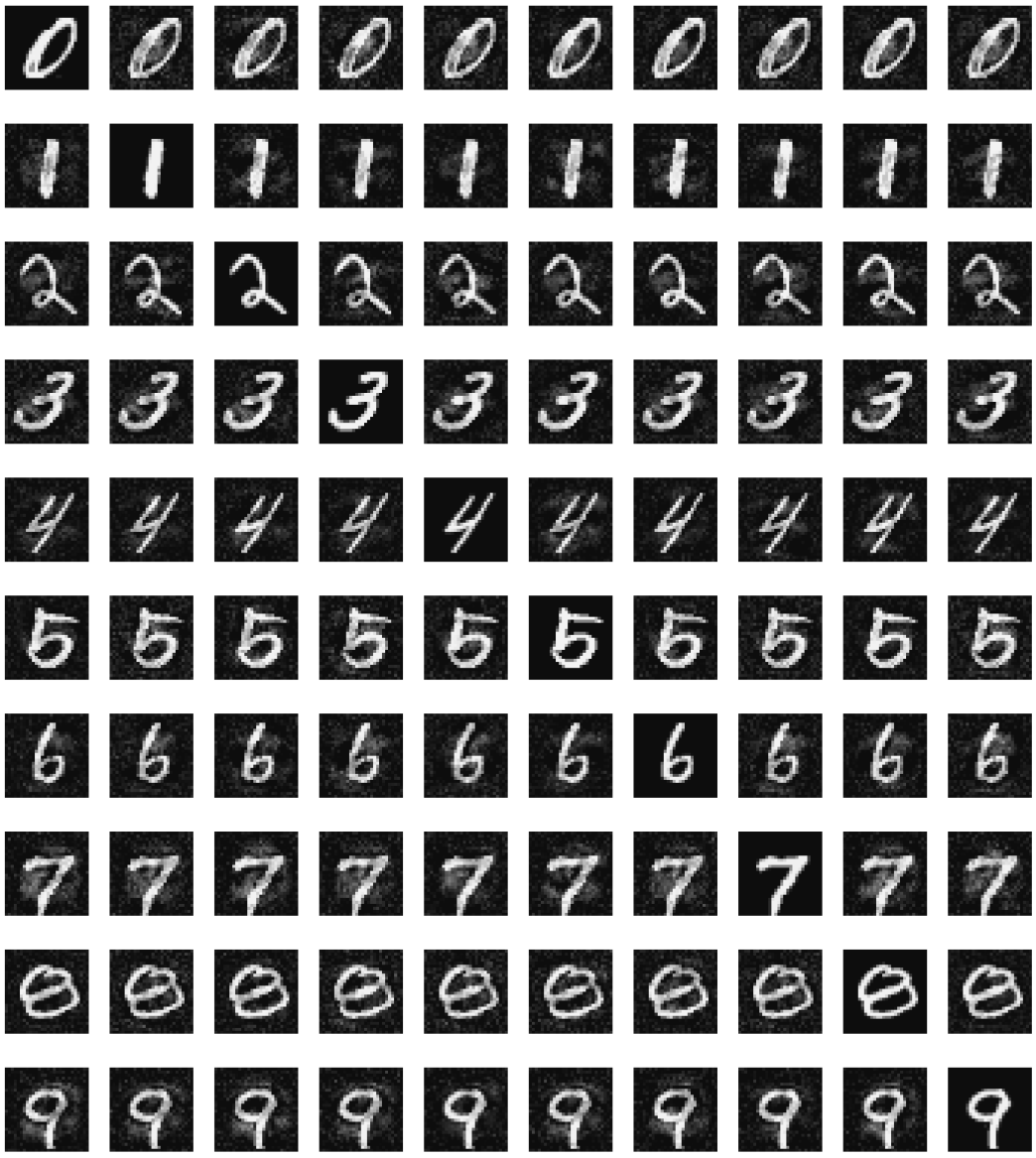

Figure 4: Table with counterfactuals generated with an artificial neural network with dropout applied and a generic gradient-based generator. The factuals are positioned along the main diagonal of the grid, the number of the row is the original label of the factual, while the number of the column shows the target of the label

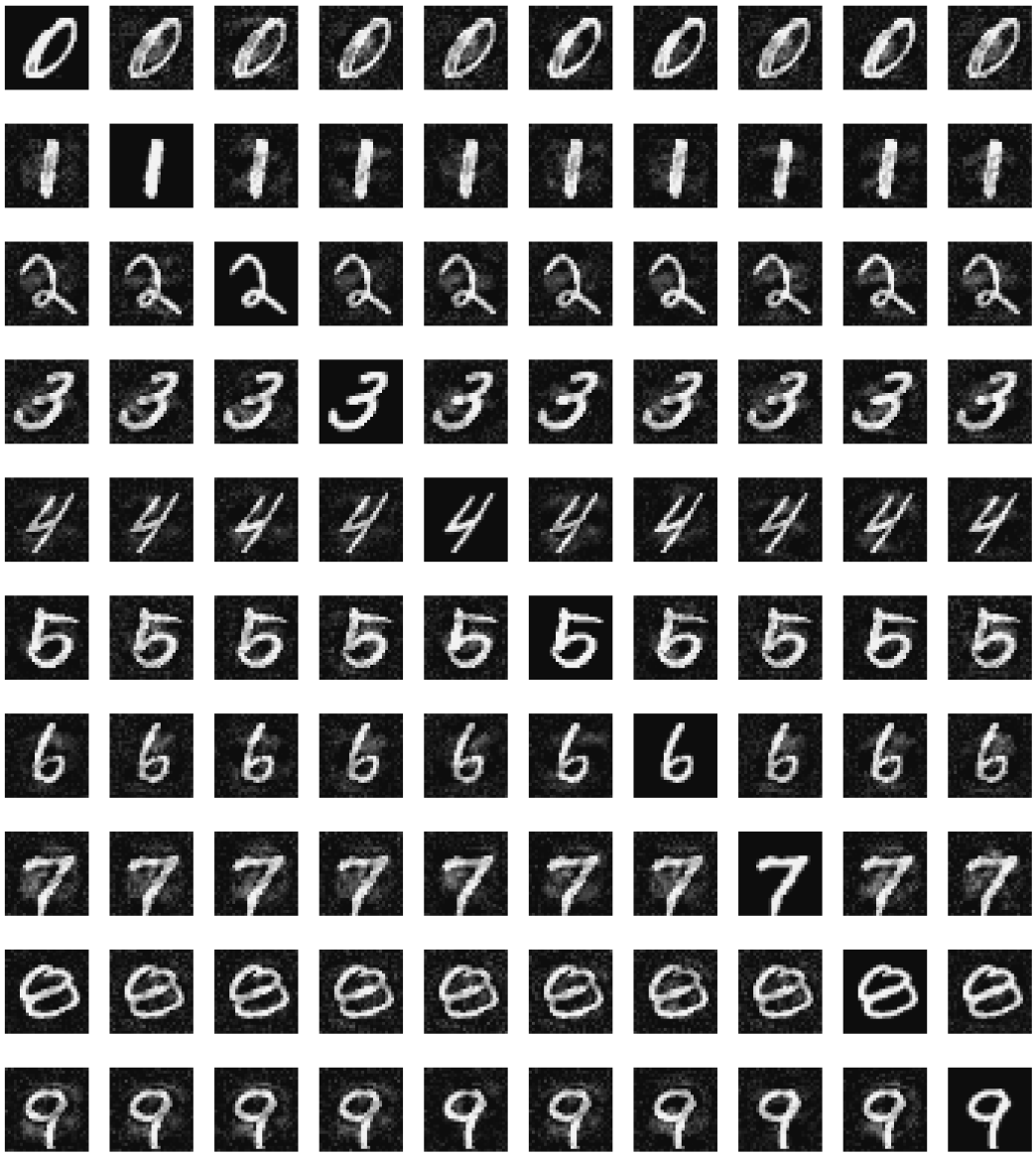

Figure 5: Table with counterfactuals generated with an artificial neural network with dropout applied and the ECCo generator. The factuals are positioned along the main diagonal of the grid, the number of the row is the original label of the factual, while the number of the column shows the target of the label

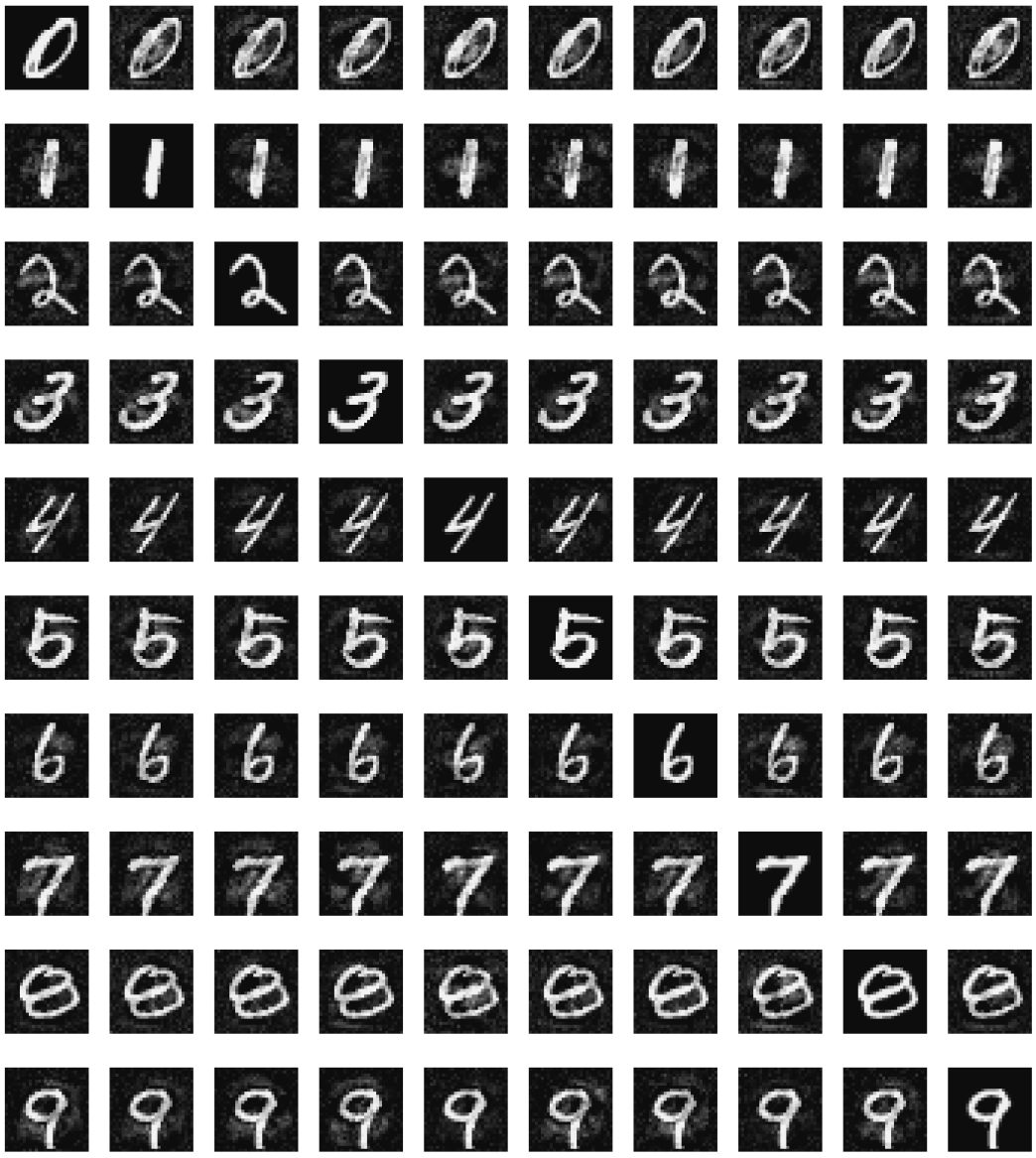

Figure 6: Table with counterfactuals generated with a deep ensemble and a generic gradient-based generator. The factuals are positioned along the main diagonal of the grid, the number of the row is the original label of the factual, while the number of the column shows the target of the label

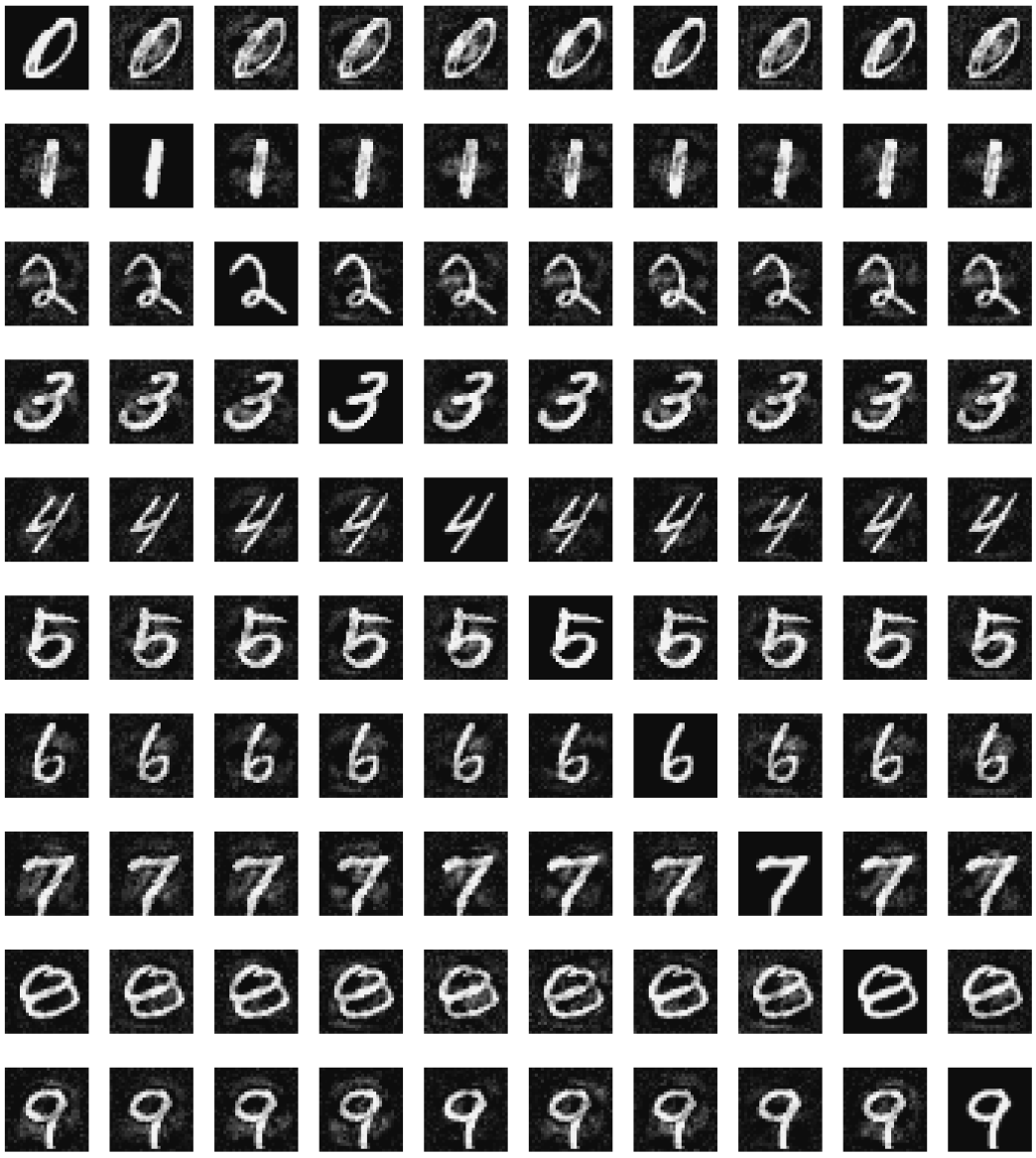

Figure 7: Table with counterfactuals generated with a deep ensemble and the ECCo generator. The factuals are positioned along the main diagonal of the grid, the number of the row is the original label of the factual, while the number of the column shows the target of the label