The Uncertainty of Counterfactuals in Deep Learning

Katherine E. Brown
kebrown46@tntech.edu
Department of Computer Science
Tennessee Technological University
1 William Jones Drive
Cookeville, TN 38505

Steve Talbert
steven.talbert@ucf.edu
College of Nursing
University of Central Florida
12201 Reserach Parkway #300
Orlando, FA 32826

Douglas A. Talbert
dtalbert@tntech.edu
Department of Computer Science
Tennessee Technological University
1 William Jones Drive
Cookeville, TN 38505

Abstract

Counterfactuals have become a useful tool for explainable Artificial Intelligence (XAI). Counterfactuals provide various perturbations to a data instance to yield an alternate classification from a machine learning model. Several algorithms have been designed to generate counterfactuals using deep neural networks; however, despite their growing use in many mission-critical fields, there has been no investigation to date as to the epistemic uncertainty of generated counterfactuals. This could result in the use of risk-prone explanations in these fields. In this work, we use several data sets to compare the epistemic uncertainty of original instances to that of counterfactuals generated from those instances. As part of our analysis, we also measure the extent to which counterfactuals can be considered anomalies in those data sets. We find that counterfactual uncertainty is higher in three of the four datasets tested. Moreover, our experiments suggest a possible connection between reconstruction error using a deep autoencoder and the difference in epistemic uncertainty between training data and counterfactuals generated from that training data for a deep neural network.

Introduction

Deep learning has achieved state-of-the-art performance in many tasks (Krizhevsky and others 2012; Akkus and others 2017); however, their black-box nature has limited its inclusion in critical-decision fields such as medicine (Begoli, Bhattacharya, and Kusnezov 2019). Techniques in *uncertainty quantification* and *explainable artificial intelligence* have been researched and developed to improve the understandability of these state-of-the-art techniques.

Uncertainty quantification in deep learning measures the reliability of a neural network's decision based on its variance when the network is subject to minor perturbations (Gal and Ghahramani 2016). Previous work (Brown and Talbert 2019; Leibig and others 2017) has successfully applied uncertainty quantification in a clinical decision support setting.

Explainable artificial intelligence (XAI) encompasses techniques that attempt to provide human-readable reasoning behind a machine learning model's decision (Montavon, Samek, and Müller 2018). Counterfactuals are one such XAI technique (Wachter, Mittelstadt, and Russell 2017; Karimi, Barthe, and others 2020; Mothilal, Sharma, and Tan 2020; White and Garcez 2019). Counterfactuals explain a machine learning decision by generating data instances that are similar to a source data point but that would result in an alternate classification from the machine learning model.

Despite their development and advancement, counterfactuals have not been applied extensively to critical fields due to possible safety concerns (Prosperi and others 2020). We hypothesize that counterfactuals are more uncertain than their original data. As counterfactuals are increasingly used to influence behaviors and decisions, uncertainty quantification may be a useful tool to detect risky, and possibly harmful, explanations.

Motivation

Counterfactual explanations and other XAI techniques are gaining popularity, due in part to the General Data Protection Regulation (GDPR) legislation in the European Union (Goodman and Flaxman 2017); however, counterfactuals are not a flawless method of explainability. First, counterfactuals have been noted to be contradictory, leading to difficulty in evaluation (Mothilal, Sharma, and Tan 2020). Moreover, counterfactuals have been criticized for their potential security risk and possible leakage of confidential information (Sokol and Flach 2019). Finally, (Prosperi and others 2020) emphasized the need for safeguards when using counterfactuals generated from predictions of a machine learning model in critical fields such as medicine.

These are valid concerns if counterfactuals are used to inform changes in behavior to yield a desired classification. If, for example, counterfactuals are used by somebody to adjust their situation in order to receive a loan (Mothilal, Sharma, and Tan 2020; Wachter, Mittelstadt, and Russell 2017), the suggestions that the potential loanee follows should, with high probability, result in the desired outcome. If, however, the counterfactual results in high uncertainty from the model, then the probability that the suggestion results in the desired outcome is reduced. This incorrectness of the counterfactual could be the result of re-training the model under random initial weights, the use of a proxy model to maintain trade secrets, etc. The risk in the loan-defaulting example is magnified in other fields, such as medicine, where patients' health can be affected by an algorithm's decision and an associated counterfactual.

One technique that can be used to model the potential risk

in deep learning is uncertainty quantification. In the context of deep learning, uncertainty is similar to the variance in the model (Gal and Ghahramani 2016). When the uncertainty of a prediction is high, the model may not have a complete understanding of the decision boundary in proximity to the data instance in question. This could happen because the network has not encountered enough data. Uncertainty has been used to improve the safety and performance of deep learning in a variety of tasks (Leibig and others 2017; Michelmore and others 2018).

Along with uncertainty quantification, we utilize anomaly detection techniques to deduce a possible relationship between generated counterfactuals and the original data with respect to anomaly detection. Anomaly detection techniques focus on modeling "normal" patterns in the data. This model is then used to identify if a datum in question does not conform to this modeled pattern (Chandola, Banerjee, and Kumar 2009).

Our hypothesis is that counterfactuals lead to higher model uncertainty. Such uncertainty could lead to risky behavior should counterfactuals be used to modify behavior. If counterfactuals are to be used and trusted by practitioners in critical fields (e.g., medical professionals) as a tool to better incorporate deep learning and artificial intelligence, the argument can and should be made that counterfactuals should be reliable. We believe this work is the first step in utilizing uncertainty quantification to increase the safety of counterfactuals.

Contributions

To the authors' knowledge, this is the first work investigating the intersection of counterfactual explanations and uncertainty quantification in deep learning; however, it is important to note that uncertainty of local explanations has been explored elsewhere (Zhang and others 2019). In three of the four evaluated datasets, we identify that model uncertainty is higher for counterfactuals generated from the training data than the original training data itself. Moreover, we find an interesting relationship between heightened uncertainty in the generated counterfactuals and the performance of this data in anomaly detection tasks. This suggests a possible condition for which counterfactuals may be considered as anomalous data.

Background

In this section, we present a summary of the background literature necessary for our approach.

Counterfactuals

Deep learning is a black-box. Neural networks learn by adjusting weights to minimize an objective function (Rumelhart, Hinton, and Williams 1985). Although deep learning techniques result in state-of-the-art performance in many critical tasks (Akkus and others 2017), it comes at the cost of not being able to interpret or understand the predictions inherently from the model itself. This can result in other, potentially weaker algorithms, such as decision trees, to be favored over deep learning, due to their inherent interpretability (Cooper and others 1997).

Counterfactuals are one technique to rectify this situation. A counterfactual is a perturbation to a data instance that produces an alternate classification as the original data instance (Wachter, Mittelstadt, and Russell 2017). Interest in counterfactuals has exploded as a result of the General Data Protection Regulation (GDPR) in the Eurpoean Union (Goodman and Flaxman 2017). This legislation has increased concerns regarding the need to provide explanations for decisions made by AI.

A counterfactual example details "how" a data instance can change to result in the desired classification. The common example used is when somebody is denied a loan (Wachter, Mittelstadt, and Russell 2017; Mothilal, Sharma, and Tan 2020). Counterfactuals can be generated that provide possible explanations on how the person should change (i.e., more education, increase in income, change in job type) to receive the desired loan.

Since their inception as a tool for XAI, there have been many techniques developed for generating counterfactuals (Mothilal, Sharma, and Tan 2020; White and Garcez 2019; Mahajan, Tan, and Sharma 2019). Some of the techniques focus on providing more feasible and actionable directions for change (Mothilal, Sharma, and Tan 2020) while other techniques (Mahajan, Tan, and Sharma 2019) focus on the causality aspect of counterfactuals.

The technique utilized in this paper is a variation of Wachter's original formulation of optimization-based counterfactual generation (Wachter, Mittelstadt, and Russell 2017). It uses randomized initialization to produce multiple counterfactual explanations and is implemented in (Mothilal, Sharma, and Tan 2020).

Uncertainty Quantification

Uncertainty quantification is a powerful tool with the potential to increase trust of deep learning and other black-box AI techniques in various critical fields. A recent editorial in *Nature* (Begoli, Bhattacharya, and Kusnezov 2019) expressed the need for uncertainty quantification for deep learning in medical settings.

In this work, we are concerned with epistemic uncertainty. Epistemic uncertainty results from the model's inability to learn a predictive function from the provided data. This could result from using too simplistic of a model or not having enough data and is typically resolved by correcting one or both of these issues (Gal and Ghahramani 2016). Throughout this work, we will refer to epistemic uncertainty as "model uncertainty" or "uncertainty".

The most common measure of epistemic uncertainty in deep learning is through dropout (Gal and Ghahramani 2016). Dropout (Srivastava and others 2014) is a regularization technique designed to prevent deep neural networks from overfitting by randomly removing weights or neurons from a neural network during training. Gal and Ghahramani demonstrated that keeping dropout *persistent* during training and testing resulted in multiple predictions creating a distribution from which variance (or standard deviation) could be sampled (Gal and Ghahramani 2016). This technique will be referred to as "Bayesian dropout" or "persistent dropout".

Anomaly Detection

Anomaly detection is the practice of detecting non-conforming, or anomalous, data (Chandola, Banerjee, and Kumar 2009). Most machine learning-based anomaly detection techniques are unsupervised. These algorithms are provided with data known, or suspected, to be normal and learns to model this data. The models can then detect anomalies by indicating, usually through some score, when a data point does not align with the learned patterns.

Autoencoders are a type of neural network designed to reconstruct an input vector (Baldi 2012; Hinton and Zemel 1994). Let $X \subseteq \mathbb{R}^n$ denote the training set of the autoencoder, where n is the original length of the data. The encoder function $f : \mathbb{R}^n \rightarrow \mathbb{R}^l$ compresses an input vector to a vector of length l. The decoder function $g : \mathbb{R}^l \rightarrow \mathbb{R}^n$ decompresses the input back to its original length. Then, the output of the autoencoder is given by $x' = g(f(x))$ where $x \in X$. Moreover, reconstruction error is given by $\sum_{i=1}^{m} (x_i - x'_i)^2$ where m is the length of the subset of $\mathbb{R}^n$ of evaluation examples. Autoencoders have been used successfully with anomaly detection in past work (Zhou and Paffenroth 2017). Autoencoders are trained upon the normal data to learn the patterns and structure of the data. This should result in normal instances having a lower reconstruction error than anomalous instances.

One-Class Support Vector Machines (SVMs) are an adaptation of the Support Vector Machine algorithm for use in a semi-supervised, anomaly detection setting (Wang, Wong, and Miner 2004). A one-class SVM is trained upon normal data, and it learns a boundary by which all normal data are enclosed. Data is then classified as normal if it is within the learned boundary or anomalous if not.

Experiments

In this section, we discuss the datasets, experimental methodology, and implementation details of our work.

Datasets

We used a total of four datasets in our experiments. Three of these datasets are from the University of California at Irvine (UCI) Machine Learning Repository (Dua and Graff 2017). The final dataset is a real-world trauma triage dataset. The datasets from UCI were selected due to their use in XAI literature (Wachter, Mittelstadt, and Russell 2017; Montavon, Samek, and Müller 2018) or their biomedical domain. The Trauma Triage dataset is utilized since it is a real-world biomedical dataset.

UCI Data Sets We utilized three datasets from the UCI Machine Learning Repository (Dua and Graff 2017). The first dataset we use is the Adult Income dataset. The goal of this dataset is to predict whether a person has an income above or below $50,000 based on 14 features. We use a modified version of this dataset from (Mothilal, Sharma, and Tan 2020) that consists of only 8 features. Of these 8 features, 2 are continuous while the remaining are categorical.

The second dataset we use is the Wisconsin Breast Cancer dataset. The goal of this dataset is to predict whether the cells in a breast mass are malignant or benign. Each entity contains 32 continuous features that describe the geometrical features of the cells.

Dataset	Hidden Layer Sizes
Income	20, 8
Breast Cancer	512, 128, 32
Diabetes	512, 128, 32
Trauma	64, 8

Table 1: Listing of hidden layer architectures applied to each dataset

The final dataset from UCI Machine Learning Repository is the Pima Diabetes dataset. The classification task in this dataset is determining whether or not a described patient has diabetes given 8 continuous features. These features include medical information such as number of pregancies, blood pressure, insulin usage, body mass index, and so on.

Trauma Triage Data We also use a trauma triage dataset extracted from the trauma registry of a Level 1 Trauma Center. This dataset attempts to predict whether a patient is severely injured, defined as having an Injury Severity Score greater than or equal to 15 (Sasser and others 2012). The 32 features include physiological parameters (e.g., systolic blood pressure, respiratory rate, and Glasgow Coma Scale score), anatomical criteria, mechanism of injury, age, and multiple computed injury scores (e.g., Revised Trauma Score and the Air Medical Prehospital Transport score).

Experimental Methodology

We will now describe the experimental setup for this work.

Data Preprocessing Each dataset is subject to the following preprocessing steps. First, continuous features are normalized such that their values are between 0 and 1. Categorical variables are one-hot encoded by their possible values. These preprocessing steps align with the data preprocessing described in (Mothilal, Sharma, and Tan 2020) that improves counterfactual generation.

Neural Networks We train a neural network on each dataset. Each of these networks consist of an input layer, multiple hidden layers, and an output layer. Table 1 shows the hidden layer architecture for each dataset. Each hidden layer uses ReLU as its activation function. After each hidden layer, a dropout layer is placed with dropout probability of 0.3 which is active only during training. When measuring uncertainty, this dropout layer is replaced with a persistent dropout that applies dropout during both the training and testing stages. Each network has 2 nodes in the output layer that use softmax as their activation function. Deep learning is implemented using Python 3 and the Keras deep learning API with Tensorflow API (Chollet and others 2015; Abadi and others 2015).

Experiment 1: Uncertainty of Counterfactuals For the first experiment, we use holdout to randomly split the data as follows: 80% training, 10% validation, 10% testing. Each network trains for 1000 epochs. We repeat each experiment 5 times per dataset. After each network is trained, we generate 100 counterfactuals from the training data. We use

the first 25 instances and generate 4 counterfactuals from this data. We use the implementation of (Mahajan, Tan, and Sharma 2019) included in the DICE-ML package (Mothilal, Sharma, and Tan 2020) with default parameters.

We then apply Bayesian dropout to the training set, training counterfactual set, testing set, and testing counterfactual set. Note that weights are transferred from the trained, non-Bayesian model to the Bayesian model to reflect the model from which we are collecting uncertainty information. We use 50 forward passes in the persistent dropout neural network to measure uncertainty values and average the values for each data subset. We report average uncertainty (i.e., variance in softmax value) for each data subset (training, training counterfactuals, testing, and testing counterfactuals) for each dataset.

Experiment 2: One-Class Support Vector Machine Based upon preliminary results, we designed two additional experiments to ascertain the nature of counterfactuals in relation to the original dataset. For the first of these experiments, we generate 100 counterfactuals by generating four counterfactuals for the first 25 instances in the training set. The first experiment uses an one-class SVM to determine if counterfactuals can be identified as anomalous data through using an anomaly detection technique.

The original data is split into a 90% training set and a 10% holdout set. The holdout set is combined with the counterfactuals to form the evaluation set. Fifty percent of the evaluation set is used as a validation set with the other half being used as test set. We then train a one-class support vector machine using the training set and use the validation set to select the appropriate parameter for ν based on F1-score of the model on the validation data. We then evaluate the model on the test set. For this experiment, we report accuracy, precision, recall, F1-Score, and area under the receiver operating characteristic (AUC) curve for 30 executions of the algorithm. The one-class SVM is implemented using the Scikit-Learn library (Pedregosa and others 2011).

Experiment 3: Reconstruction using Deep Autoencoder The final experiment we perform uses a deep autoencoder trained on the original training data and measures reconstruction error. We split the original training data into a 80% training set, a 10% validation set and a 10% testing set. Further, we use the same counterfactuals generated for Experiment 2. The architecture for the auotencoder is given in Table 2. The autoencoder is optimized with the Adam algorithm to minimize mean squared error (Kingma and Ba 2014). The autoencoder is trained for 1000 epochs or until validation loss does not decrease for 50 epochs. We then calculate reconstruction error on the testing data and the counterfactuals for each dataset. Along with reconstruction error, we also report the Pearson correlation coefficient between average reconstruction error for the training and counterfactuals with percent change in uncertainty between training and training counterfactuals for each dataset for 30 executions of the algorithm. The autoencoder is implemented using Python 3 and the Keras deep learning API with Tensorflow API (Chollet and others 2015; Abadi and others 2015).

Stage	Layer	Size	Activation
Encoder	Input Layer	X	N/A
	Hidden Layer	64	ReLU
	Hidden Layer	32	ReLU
	Hidden Layer	16	ReLU
Decoder	Hidden Layer	32	ReLU
	Hidden Layer	64	ReLU
	Output Layer	X	Sigmoid

Table 2: Architecture of Autoencoder where X is the input size of the data.

Results

In this section, we present results of the experiments described in the Experimental Methodology section.

Experiment 1: Uncertainty of Counterfactuals

Table 3 presents the average uncertainty values for the training data and counterfactuals generated from the training data.

Dataset	Training Data	Training CFs	Percent Change
Breast Cancer	0.036	0.0481	32.258
Diabetes	0.086	0.081	-5.796
Income	0.067	0.130	96.004
Trauma	0.069	0.168	143.750

Table 3: Average uncertainty for training data and training counterfactuals along with percent change. CFs denote counterfactuals

Table 3 quantifies the percent change between the uncertainty resulting from the training counterfactuals and the uncertainty from the training data itself. We can see that the Diabetes dataset resulted in a negative change whereas the Breast Cancer, Income, and Trauma datasets resulted in positive change with Breast Cancer having the smallest positive change and Trauma having the largest positive change.

Experiment 2: One-Class Support Vector Machines

Dataset	Accuracy	F1-Score	AUC
Breast Cancer	0.902	0.947	0.963
Diabetes	0.637	0.609	0.674
Income	0.913	0.854	0.969
Trauma	0.902	0.947	0.999

Table 4: Performance metrics for the One-Class SVM

Table 4 presents the accuracy, precision, recall, F1 score, and area under the ROC curve (AUC). We can see that the One-Class SVM achieves superb results on the Breast Cancer, Income, and Trauma datasets but has subpar performance on the Diabetic dataset.

Experiment 3: Reconstruction using Deep Autoencoders

Table 5 lists the reconstruction error for the autoencoder on the training data (normal data) and the counterfactuals gen-

Dataset	Training Data	Training CFs	Ratio
Breast Cancer	0.037	0.851	23.288
Diabetes	0.002	0.005	2.729
Income	0.118	3.645	30.793
Trauma	0.109	19.247	177.196

Table 5: Average Reconstruction Error of autoencoder per dataset. CFs denote counterfactuals

erated from the training data as well as the ratio of the reconstruction error of the counterfactuals to the reconstruction error of the normal data.

For all datasets, the reconstruction error of the counterfactual data subset is larger than the reconstruction of the held out normal data; however, the magnitude of this difference is larger for the datasets that had a higher uncertainty for the training counterfactuals than for the original training instances.

Table 6 lists the Pearson correlation coefficient between the percent change in uncertainty from Table 3 and each of the following: reconstruction error from normal training data, reconstruction error from counterfactuals generated from training data, and the ratio of counterfactual reconstruction error to normal reconstruction error. Each of these coefficients are very close to 1 indicating the potential of high correlation between each of the tested results; however, we stress that the uncertainty results are collected from only five executions, and further conclusions should only be made with more executions.

Values	Correlation Coefficient
Normal Reconstruction Error	0.933
Counterfactual Reconstruction Error	0.871
Ratio of Counterfactual to Normal Error	0.852

Table 6: Listing of correlation coeffcients between % Change in uncertainty and data listed in the *Values* column

Discussion

In this section, we discuss our results and their implications.

Uncertainty of Counterfactuals

From our results, we see that for three of the four datasets, uncertainty in the training data is less than the uncertainty of the counterfactuals generated from the training data. We find that our hypothesis does not hold across all datasets, however, since this does not occur in the Diabetes dataset.

It is known that epistemic uncertainty increases when the model is tasked with classifying data outside the explored decision space (Gal and Ghahramani 2016). This might suggest that, for datasets for which our hypothesis holds, the counterfactuals generated lie outside the explored decision space.

Anomaly Detection Techniques

Using this intuition of epistemic uncertainty, we implemented a deep autoencoder and one-class support vector machine to determine if the generated counterfactuals are different enough to be detected by traditional anomaly detection methods. For datasets where our hypothesis holds, we hypothesize that these anomaly detection techniques will be successful in identifying the counterfactuals from the original dataset.

The one-class support vector machine (SVM) has accuracy and F1-score above 90% for the Breast Cancer, Income, and Trauma datasets. Thus, when uncertainty has a positive change from training data to training counterfactuals, the one-class SVM successfully detects the counterfacutals from the original data. When uncertainty decreases, as in the Diabetes dataset, the one-class SVM has accuracy and F1-score less than 65%. This supports the intuition that elevated counterfactual uncertainty indicates that the counterfactuals are dissimilar to the true dataset.

We also consider the reconstruction error when each of these datasets are trained on and reconstructed by a deep autoencoder. We train the deep autoencoder on 90% of the original training data and evaluate the reconstruction error on the remaining 10% of the original training data and the counterfactuals. Through compressing the data into a latent space and decompressing it, a deep autoencoder can capture the patterns and structure of normal data (Zhou and Paffenroth 2017). Anomalies can then be detected by their reconstruction error in comparison to the reconstruction error of the normal data. From our results (Table 5), we can clearly see that for the Breast Cancer, Income, and Trauma datasets, reconstruction error of the counterfactual examples is greater than the reconstruction error of the held-out normal examples. Although this also applies to the Diabetes dataset, it is evident from the ratios that the effect is not as pronounced as with the other datasets.

Based on both the one-class SVM and the deep autoencoder results, we can see that counterfactuals that result in higher uncertainty are more anomalous with respect to the original dataset. Moreover, this leads us to the belief that highly uncertain counterfactuals are likely to be generated in unexplored space relative to the original data. If this is the case, practitioners should be aware that although counterfactuals represent changes in the original data to produce an alternate prediction, the actual classification of the counterfactual could be wrong and should only serve as a reference point for the model's classification.

Finally, we wish to present an interesting result in Table 6. There is an overwhelmingly positive correlation between the percent change in uncertainty and each of the normal reconstruction error, counterfactual reconstruction error, and ratio of counterfactual reconstruction error to normal reconstruction error. Based on our discussion in the previous paragraph, this would suggest that there is a relationship between difficulty of reconstruction and change in uncertainty. If this correlation indeed exists, it supports the claim that the counterfactuals lie outside the explored decision space. We wish to note more data on the change in uncertainty should be collected before making statistical claims.

Conclusion

In this work, we investigated the effects of counterfactuals on epistemic uncertainty of a neural network. We found that, for some datasets, counterfactuals generate a higher uncertainty than the original data. This result holds for three of the four datasets evaluated. Moreover, when uncertainty increased between counterfactuals and original data, we observed that counterfactuals appear to be more anomalous and can be determined from normal data by anomaly detection methods. Further, we discovered a possible correlation between reconstruction error and change in uncertainty from counterfactuals to normal data. These findings indicate that counterfactual explanations may pose risk in safety-critical settings.

For future work, we would like to investigate this correlation further with more rigorous studies. We also plan to evaluate this method with other datasets and counterfactual generating algorithms.

References

Abadi, M., et al. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Software available from tensorflow.org.

Akkus, Z., et al. 2017. Deep Learning for Brain MRI Segmentation: State of the Art and Future Directions. *Journal of Digital Imaging* 30(4):449–459.

Baldi, P. 2012. Autoencoders, Unsupervised Learning, and Deep Architectures. In *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, 37–49.

Begoli, E.; Bhattacharya, T.; and Kusnezov, D. 2019. The Need for Uncertainty Quantification in Machine-Assisted Medical Decision Making. *Nature Machine Intelligence* 1(1):20–23.

Brown, K., and Talbert, D. 2019. Estimating Uncertainty in Deep Image Classification. In *Proceedings of the American Medical Informatics Association Annual Symposium*.

Chandola, V.; Banerjee, A.; and Kumar, V. 2009. Anomaly Detection: A Survey. *ACM Computing Surveys (CSUR)* 41(3):1–58.

Chollet, F., et al. 2015. Keras.

Cooper, G. F., et al. 1997. An Evaluation of Machine-Learning Methods for Predicting Pneumonia Mortality. *Artificial Intelligence in Medicine* 9(2):107–138.

Dua, D., and Graff, C. 2017. UCI Machine Learning Repository.

Gal, Y., and Ghahramani, Z. 2016. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proceedings of The 33rd International Conference on Machine Learning*, 1050–1059.

Goodman, B., and Flaxman, S. 2017. European Union Regulations on Algorithmic Decision-Making and a "Right to Explanation". *AI Magazine* 38(3):50–57.

Hinton, G. E., and Zemel, R. S. 1994. Autoencoders, Minimum Description Length and Helmholtz Free Energy. In *Advances in Neural Information Processing Systems*, 3–10.

Karimi, A.-H.; Barthe, G.; et al. 2020. Model-Agnostic Counterfactual Explanations for Consequential Decisions. In *International Conference on Artificial Intelligence and Statistics*, 895–905.

Kingma, D. P., and Ba, J. 2014. Adam: A Method for Stochastic Optimization.

Krizhevsky, A., et al. 2012. Imagenet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, 1097–1105.

Leibig, C., et al. 2017. Leveraging Uncertainty Information from Deep Neural Networks for Disease Detection. *Scientific Reports* 7(1):17816.

Mahajan, D.; Tan, C.; and Sharma, A. 2019. Preserving Causal Constraints in Counterfactual Explanations for Machine Learning Classifiers. *arXiv preprint arXiv:1912.03277*.

Michelmore, R., et al. 2018. Evaluating Uncertainty Quantification in End-to-End Autonomous Driving Control. *arXiv preprint arXiv:1811.06817*.

Montavon, G.; Samek, W.; and Müller, K.-R. 2018. Methods for Interpreting and Understanding Deep Neural Networks. *Digital Signal Processing* 73:1–15.

Mothilal, R. K.; Sharma, A.; and Tan, C. 2020. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, 607–617.

Pedregosa, F., et al. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12:2825–2830.

Prosperi, M., et al. 2020. Causal Inference and Counterfactual Prediction in Machine Learning for Actionable Healthcare. *Nature Machine Intelligence* 2(7):369–375.

Rumelhart, D. E.; Hinton, G. E.; and Williams, R. J. 1985. Learning Internal Representations by Error Propagation. Technical report, California University San Diego La Jolla Institute for Cognitive Science.

Sasser, S. M., et al. 2012. Guidelines for Field Triage of Injured Patients: Recommendations of the National Expert Panel on Field Triage, 2011. *Morbidity and Mortality Weekly Report: Recommendations and Reports* 61(1):1–20.

Sokol, K., and Flach, P. A. 2019. Counterfactual Explanations of Machine Learning Predictions: Opportunities and Challenges for AI Safety. In *SafeAI@ AAAI*.

Srivastava, N., et al. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *The Journal of Machine Learning Research* 15(1):1929–1958.

Wachter, S.; Mittelstadt, B.; and Russell, C. 2017. Counterfactual Explanations Without Opening the Black Box: Automated Decisions and the GDPR. *Harv. JL & Tech.* 31:841.

Wang, Y.; Wong, J.; and Miner, A. 2004. Anomaly Intrusion Detection Using One-Class SVM. In *Proceedings from the Fifth Annual IEEE SMC Information Assurance Workshop, 2004.*, 358–364. IEEE.

White, A., and Garcez, A. d. 2019. Measurable Counterfactual Local Explanations for any Classifier. *arXiv preprint arXiv:1908.03020*.

Zhang, Y., et al. 2019. "Why Should You Trust My Explanation?" Understanding Uncertainty in LIME Explanations. *arXiv preprint arXiv:1904.12991*.

Zhou, C., and Paffenroth, R. C. 2017. Anomaly Detection with Robust Deep Autoencoders. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 665–674.