Citation   J. Lee and G. AlRegib, Gradients as a Measure of Uncertainty in Neural Networks, 27th
              IEEE International Conference on Image Processing (ICIP), Abu Dhabi, United Arab Emi-
              rates (UAE), 2020.

Review   Date of Publication: October 25, 2020

Bib   @inproceedings{lee2020gradients,
              title={Gradients as a Measure of Uncertainty in Neural Networks},
              author={Lee, Jinsol and AlRegib, Ghassan},
              booktitle={IEEE International Conference on Image Processing (ICIP)},
              year={2020}}

Contact   {jinsol.lee, alregib}@gatech.edu
              https://ghassanalregib.info/

GRADIENTS AS A MEASURE OF UNCERTAINTY IN NEURAL NETWORKS

Jinsol Lee and Ghassan AlRegib

OLIVES at the Center for Signal and Information Processing,
School of Electrical and Computer Engineering,
Georgia Institute of Technology, Atlanta, GA, 30332-0250
{jinsol.lee, alregib}@gatech.edu

ABSTRACT

Despite tremendous success of modern neural networks, they are known to be overconfident even when the model encounters inputs with unfamiliar conditions. Detecting such inputs is vital to preventing models from making naive predictions that may jeopardize real-world applications of neural networks. In this paper, we address the challenging problem of devising a simple yet effective measure of uncertainty in deep neural networks. Specifically, we propose to utilize backpropagated gradients to quantify the uncertainty of trained models. Gradients depict the required amount of change for a model to properly represent given inputs, thus providing a valuable insight into how familiar and certain the model is regarding the inputs. We demonstrate the effectiveness of gradients as a measure of model uncertainty in applications of detecting unfamiliar inputs, including out-of-distribution and corrupted samples. We show that our gradient-based method outperforms state-of-the-art methods by up to 4.8% of AUROC score in out-of-distribution detection and 35.7% in corrupted input detection.

Index Terms— gradients, uncertainty, unfamiliar input detection, out-of-distribution, image corruption/distortion

1. INTRODUCTION

Deep neural networks (DNNs) have achieved significant improvements in a variety of applications, including object detection [1], image classification [2], and natural language processing [3]. Despite these advancements, DNNs are prone to failure when deployed in real-world environments as they often encounter data that diverges from training conditions [4, 5]. Neural networks make predictions regardless of their familiarity with given inputs. Combined with their overconfident nature [6, 7], the difficulty in distinguishing naive predictions about unfamiliar inputs from knowledgeable predictions about familiar inputs hinders practical implementation of DNNs. To ensure a certain level of reliable performance, models must be capable of identifying unfamiliar inputs, which then can be handled via predetermined fallback mechanisms instead of making erroneous conjectures.

At the core of detecting unfamiliar inputs lies the concept of uncertainty and the methodology to measure it. In this work, we mainly consider *model uncertainty*, which refers to the uncertainty in model parameters due to limited data and knowledge [8]. It is directly related to the problem of out-of-distribution (OOD) detection, where the goal is to detect the samples drawn from different data distribution than in-distribution or training samples. Recent studies [9, 10] utilized the prediction probability statistics as an uncertainty measure. Hendrycks & Gimpel [9] introduced a base-
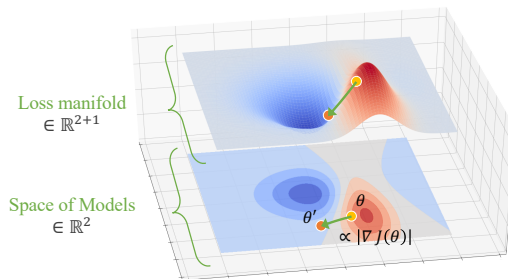

Fig. 1: Illustration of a space of models.

line method of thresholding samples based on the predicted softmax probability. Liang *et al.* [10] improved their method with input and output processing to further distinguish in-distribution and OOD samples. DeVries & Taylor [11] utilized prediction confidence as an uncertainty measure, obtained from an augmented confidence estimation branch on a pre-trained classifier. Lee *et al.* [12] proposed a confidence metric with Mahalanobis distance to characterize OOD samples. These approaches investigate model uncertainty with activation-based measures and typically utilize additional modules for pre- and post-processing as well as quantifying uncertainty that require calibrations.

In this work, we propose a simple yet effective method to measure model uncertainty of deep neural networks. Specifically, *we utilize backpropagated gradients to probe the knowledge of fully trained networks and determine the model's familiarity with given inputs*. Our intuition is that gradients correspond to the amount of changes it requires to properly represent a given sample. In comparison with aforementioned activation-based methods [9, 10, 11, 12], our method in estimating model uncertainty takes into account both activation and loss associated with given inputs as well as the relationship between the two, preserving more information about model uncertainty. In addition, our method does not require any calibration or extra processing. We empirically show that our gradient-based method outperforms activation-based methods across multiple tasks and datasets. The contributions of this paper are three-fold.

- We provide an interpretation of gradients in the space of models from a perspective of model uncertainty.
- We introduce an efficient framework to generate gradients for uncertainty characterization.
- We apply our gradient-based approach to applications of out-of-distribution detection and corrupted input detection to achieve the state-of-the-art performance.

2. BACKGROUND

Given any single network architecture, the set of models is defined by all possible configurations of its parameters. In the immense space of these models, we manage to find a seemingly endless number of parameterizations that perform well via various initialization and optimization techniques. In particular, stochastic gradient-based optimization has been fundamental to the success of modern neural networks. It is built on the assumption that the model is defined on a relatively smooth space where the first derivatives exist [13]. Each update of gradient-based optimization, as shown in Eq. (1), aims to minimize an objective function $J(\theta)$ with respect to model parameters $\theta \in \mathbb{R}^d$ by updating the parameters in the opposite direction of the gradients of the objective function $\nabla_\theta J(\theta)$.

$$\theta' = \theta - \eta \cdot \nabla_\theta J(\theta) \tag{1}$$

The direction of the gradients determines the *direction* of the update, and the magnitude of gradients scaled by learning rate η governs the *size* of the update. It is a process of maneuvering the space of models from one parameterization to another. In Fig. 1, we visualize a space of models for an architecture of two parameters and its arbitrary loss manifold. The model space is two-dimensional as defined by the number of parameters. The loss manifold is depicted with one extra dimension since, for each parameterization, a loss value can be computed with respect to a given input. In training, each iteration involves finding a model parameterization by optimizing on the loss manifold. On the space of models, a model that performs better than the current parameterization lies in the opposite direction of the gradients at a distance of some proportion of the gradient magnitude.

3. GRADIENTS AS A MEASURE OF UNCERTAINTY

Gradient-based optimization involves larger updates when there is a larger gap between predictions and correct labels for given inputs. It implies that the model requires more significant adjustments to its parameters, as it has not learned enough features to represent the inputs or relationships between learned features and classes for correct prediction. We propose to utilize gradients to quantify uncertainty of fully trained neural networks. Our intuition is that the magnitude of the gradients is directly proportional to the amount of each update in training. In the context of the space of models, the size of each update represents the distance between two distinct model parameterizations: one before the update (θ) and the other afterward (θ'), as

shown in Fig. 1. We can exploit this idea of distance in the space of models, represented by the magnitude of gradients, as a measure of uncertainty. *During inference*, gradients can be generated in a similar manner as in training process. *The obtained gradients are not pertinent to model updates; however, they contain more reliable information about the familiarity of the model with given inputs, as the model in question is a converged solution and the required model update should not be drastic for inputs that are similar to training data.* In the following sections, we describe the process of generating and utilizing such gradients in detail.

3.1. Gradient generation framework

To use gradients as a measure of uncertainty for a fully trained network, we first address the process of obtaining gradients. It requires backpropagation of arbitrary loss between predictions that the model made for given inputs and their ground truth labels; however, the labels are not available at inference. While we may be able to analyze gradients for every class that the model is trained for, the number of loss and gradient computations increases linearly with the number of possible classes for the model. For a more efficient gradient generation process, we construct confounding labels. We define a *confounding label* as a label that is different from ordinary labels on which a model is trained. In an image classification setting, an ordinary label consists of a single class (i.e. one-hot vector), whereas a confounding label may include multiple classes or none—a vector of length C with n number of 1's where C is the number of classes in training and $n \in \{0, \ldots, C\} \setminus \{1\}$. We describe the effect of a confounding label in Fig. 2(a). Consider a trained classifier that is trained with images and labels of two classes: "dog" and "horse". With an ordinary label (i.e. "dog" or "horse"), when a model is familiar with an input image, the model has features learned during training to represent the input. In addition, the model is able to associate the already-learned features with one of the trained classes provided by the given ordinary label. On the other hand, a confounding label is specifically designed to be unfamiliar to the model. With a confounding label (i.e. "car") and a familiar input, the model does not need to learn features again. The only necessary adjustments to model parameters would be to learn the relationship between the already-learned features and the confounding label. If the model is unfamiliar with the given input, however, the amount of updates required to 1) learn new features to properly represent the new input, and 2) associate the new features with the confounding label, would
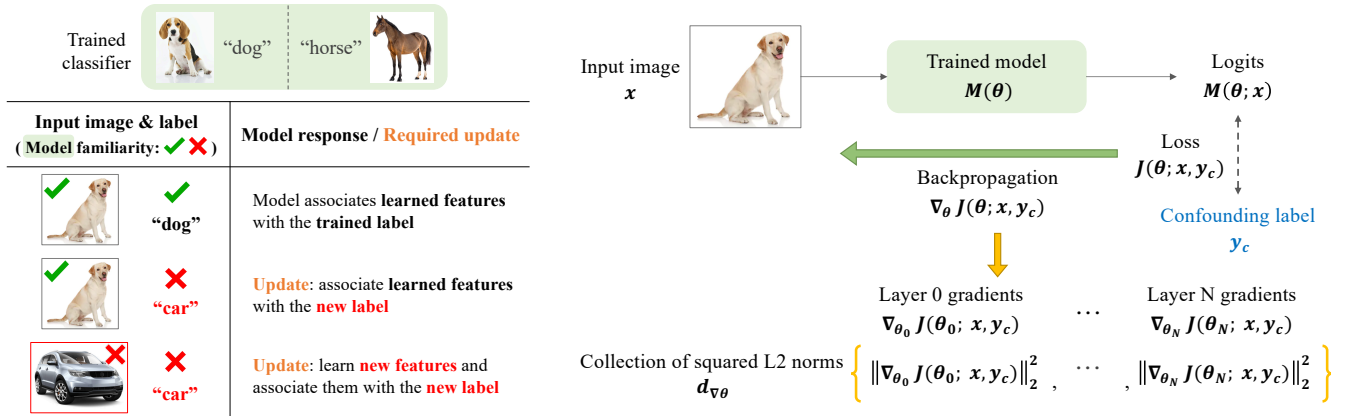
(a) The effect of a confounding label.

(b) Gradient generation framework with confounding label.

Fig. 2: Quantifying uncertainty of neural networks with gradients. Confounding labels are utilized for effective gradient generation.

(a) Squared L_2 norms of gradients at different parameter sets: $\|\nabla J(\theta)\|_2^2$.
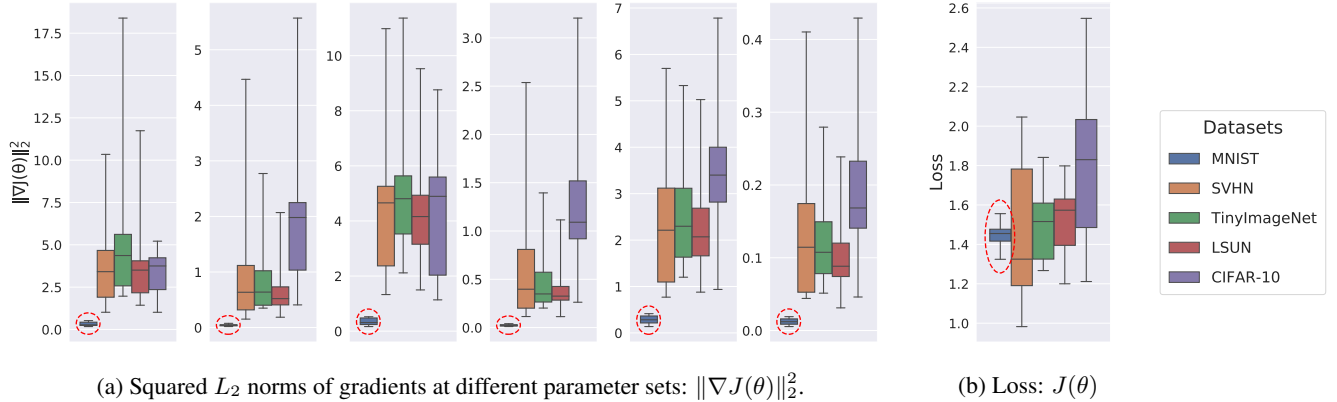
(b) Loss: $J(\theta)$

Fig. 3: Comparison between gradients and loss in distinguishing in-distribution and out-of-distribution datasets. Each plot shows distributions of per-class average values of considered attributes ($\|\nabla J(\theta)\|_2^2$ or $J(\theta)$) for all datasets.

be larger.

We discuss the framework to collect gradients with confounding labels from a fully trained network in Fig. 2(b). Given an input image, the model yields logits. We utilize binary cross entropy loss between the logits and a confounding label as shown in Eq. (2), where $\hat{y}_i$ is the predicted probability for class i and y_i is the true probability represented by the confounding label. Through backpropagation of the loss, gradients are generated for each set of model parameters (i.e. weight and bias parameters of network layers). While any form of conserving the magnitude would be valid, we measure the squared L_2 distance of the gradients for each parameter set and concatenate them to represent the given input.

$$J(\theta) = \frac{1}{C} \sum_{i=0}^{C} \left(y_i \cdot \log\left(\hat{y}_i\right) + (1 - y_i) \cdot \log\left(1 - \hat{y}_i\right) \right) \quad (2)$$

3.2. Demonstration of the effectiveness of gradients

To demonstrate the effectiveness of gradients as a measure of model uncertainty, we present preliminary experiments in out-of-distribution detection setups. First, we train a ResNet-18 [2] model with the training set of MNIST [14]. The trained model is then used to generate gradients as described in Section 3.1 on the test sets of MNIST, CIFAR-10 [15], TinyImageNet [16], LSUN [17], and SVHN [18]. MNIST is chosen to be an in-distribution dataset as it is the simplest among all considered datasets. We employ a network architecture that is large enough to learn classification for all datasets. With the magnitude of gradients, we show the disparity between the learned features in training and the necessary features to represent test images.

In Fig. 3(a), we visualize the distributions of per-class average magnitude of gradients, captured in different parameter sets of the model. Note that the distribution of MNIST is highlighted in red circle in each plot for clarity. The separation between in-distribution and OOD datasets is more evident in some parts of the network than others because each layer captures information about different aspects of given inputs. But overall, we observe a sharp distinction based on the familiarity of the model, with smaller gradient magnitude for in-distribution/familiar datasets and significantly larger for OOD/unfamiliar datasets. In addition, we show the distribution of loss values in Fig. 3(b) for further analysis. Gradients and loss are intertwined in the process of backpropagation, but the loss is limited to a single value per iteration. We observe apparent overlaps of loss values across different datasets, demonstrating that loss

alone is inadequate for detecting unfamiliar inputs. On the contrary, the gradients are of the same dimensions as their corresponding parameter sets, preserving more information about the current state of the model and the necessary adjustment for better representation of given inputs.

4. EXPERIMENTS

We validate the effectiveness of our proposed method in applications that deal with some form of unfamiliar inputs, including out-of-distribution detection and corrupted input detection. For each application, we distinguish the familiarity of datasets based on the model training process—familiar if the model has been exposed to the dataset in training and unfamiliar otherwise. Specifically, we train ResNet with 18 layers on training set of a dataset and obtain gradient-based representations from test sets of both familiar and unfamiliar datasets. To generate gradients, we utilize a confounding label where all classes are positive (i.e. a vector of all 1's). For each set of familiar and unfamiliar datasets, we use a 40%-40%-20% train-validation-test split of the collected gradient representations and train a simple binary detector of 2 fully-connected layers as an unfamiliar input detector.

4.1. Out-of-distribution detection

For out-of-distribution (OOD) detection experiments, we use various image classification datasets: CIFAR-10 and SVHN as in-distribution, additional TinyImageNet and LSUN as OOD. For fair comparison, we utilize the same experiment setups to replicate results from other state-of-the-art OOD methods [9, 10, 12]. For evaluation, we measure the following metrics: detection accuracy, area under receiver operating characteristic curve (AUROC), and area under precision-recall curve (AUPR). The results of OOD detection are reported in Table 1 and the method with the best result for each metric is highlighted. While both ODIN [10] and Mahalanobis [12] methods involve input and output processing, we report vanilla results (i.e. without any processing) for Mahalanobis method as it shows better performance than ODIN when the additional procedures are allowed. Note that these procedures require delicate calibration of hyperparameters, while our method does not require any. We observe that our method outperforms all other techniques when CIFAR-10 is used as in-distribution and SVHN as OOD, and SVHN as in-distribution and all others as OOD, up to 9% increase

Table 1: The results of distinguishing in- and out-of-distribution test set data on OOD benchmarks. All values are in percentages and the best results are highlighted in bold. For Mahalanobis [12] method, V indicates vanilla results without input pre-processing or feature ensemble, and P+FE includes both.

Dataset Distribution		Detection Accuracy	AUROC	AUPR
In	Out	Baseline [9] / ODIN [10] / Mahalanobis (V) [12] / Mahalanobis (P+FE) [12] / Ours		
CIFAR-10	SVHN	83.36 / 88.81 / 79.39 / 91.95 / **98.04**	88.30 / 94.93 / 85.03 / 97.10 / **99.84**	88.26 / 95.45 / 86.15 / 96.12 / **99.98**
	TinyImageNet	84.01 / 85.21 / 83.60 / **97.45** / 86.17	90.06 / 91.86 / 88.93 / **99.68** / 93.18	89.26 / 91.60 / 88.59 / **99.60** / 92.66
	LSUN	87.34 / 88.42 / 85.02 / **98.60** / 98.37	92.79 / 94.48 / 90.11 / **99.86** / 99.86	92.30 / 94.22 / 89.80 / 99.82 / **99.87**
SVHN	CIFAR-10	79.98 / 80.12 / 74.10 / 88.84 / **97.90**	81.50 / 81.49 / 79.31 / 95.05 / **99.79**	81.01 / 80.95 / 80.83 / 90.25 / **98.11**
	TinyImageNet	81.70 / 81.92 / 79.35 / 96.17 / **97.74**	83.69 / 83.82 / 83.85 / 99.23 / **99.77**	82.54 / 82.60 / 85.50 / **98.17** / 97.93
	LSUN	80.96 / 81.15 / 79.52 / 97.50 / **99.04**	82.85 / 82.98 / 83.02 / 99.54 / **99.93**	81.97 / 82.01 / 84.67 / 98.84 / **99.21**

in performance. Our method is especially effective when there is a larger difference in the complexity of in-distribution and OOD datasets. SVHN comprises color images of house numbers ranging from 0 to 9, while the other datasets include more complex natural images of diverse classes. Our method still shows very comparable result when the model is trained on CIFAR-10 and LSUN is used as OOD. We remark that our gradient-based method outperforms all activation-based methods in most cases.

4.2. Corrupted input detection

In addition to model uncertainty, we consider *aleatoric uncertainty*, another type of uncertainty that is related to noise in observations [8]. Some examples of the cause of this uncertainty are imperfect data acquisition and environmental factors such as motion blur or weather conditions. Aleatoric uncertainty is considered irreducible even with more data due to the inherent nature of noise, so most approaches focus on designing a robust system that can handle both pristine and noisy inputs. We apply our gradient-based uncertainty measure to directly detect corrupted inputs. We utilize the Mahalanobis method from out-of-distribution detection as a comparison, as corrupted inputs can also be considered as OOD examples and the Mahalanobis method shows the best performance among all compared methods.

For corrupted input detection experiments, we use image classification datasets that are designed for benchmarking robustness of neural networks with realistic challenging conditions. CIFAR-10-C [19] consists of 19 diverse corruption types of 4 categories, including noise, blur, weather and digital, at 5 different severity levels that are applied to validation images of CIFAR-10 dataset. CURE-TSR [20] is a traffic sign recognition dataset that includes real-world and simulated challenging conditions of 11 types and 5 severity levels. For each dataset, a model is trained on training images that are free of corruptions, and the gradients are collected for test sets of both pristine and corrupted images. Among all image corruption types from both datasets, we select common or similar ones to report AUROC scores in Table 2. For CIFAR-10-C dataset, our method outperforms the Mahalanobis method for all corruption types and levels. It shows saturated performance even from severity level 1, while the Mahalanobis method generally shows an increase at higher levels of corruption. Regarding CURE-TSR dataset, our method still outperforms in most cases and show comparable results otherwise, but both methods show more variations in performance across severity levels of corruptions. Note the gap in AUROC scores between the methods at lower levels of corruptions—this implies that our gradient-based method can characterize corruptions effectively, compared to the activation-based method, even at a subtle degree.

Table 2: The results of distinguishing pristine and corrupted images. All values are AUROC scores in percentages and the best results are highlighted in bold.

Dataset	Method	Mahalanobis [12] / Ours				
	Corruption	Level 1	Level 2	Level 3	Level 4	Level 5
CIFAR-10-C	Noise	96.63 / **99.95**	98.73 / **99.97**	99.46 / **99.99**	99.62 / **99.97**	99.71 / **99.99**
	LensBlur	94.22 / **99.95**	97.51 / **99.99**	99.26 / **100.0**	99.78 / **100.0**	99.89 / **100.0**
	GaussianBlur	94.19 / **99.94**	99.28 / **100.0**	99.76 / **100.0**	99.86 / **100.0**	99.80 / **100.0**
	DirtyLens	93.37 / **99.94**	95.31 / **99.93**	95.66 / **99.96**	95.37 / **99.92**	97.43 / **99.96**
	Exposure	91.39 / **99.87**	91.00 / **99.85**	90.71 / **99.88**	90.58 / **99.85**	90.68 / **99.87**
	Snow	93.64 / **99.94**	96.50 / **99.94**	94.44 / **99.95**	94.22 / **99.95**	95.25 / **99.92**
	Haze	95.52 / **99.95**	98.35 / **99.99**	99.28 / **100.0**	99.71 / **99.99**	99.94 / **100.0**
	Decolor	93.51 / **99.96**	93.55 / **99.96**	90.30 / **99.82**	89.86 / **99.75**	90.43 / **99.83**
CURE-TSR	Noise	25.46 / **50.20**	47.54 / **63.87**	47.32 / **81.20**	66.19 / **91.16**	83.14 / **94.81**
	LensBlur	48.06 / **72.63**	71.61 / **87.58**	86.59 / **92.56**	92.19 / **93.90**	94.90 / **95.65**
	GaussianBlur	66.44 / **83.07**	77.67 / **86.94**	93.15 / **94.35**	80.78 / **94.51**	97.36 / 96.53
	DirtyLens	29.78 / **51.21**	29.28 / **59.10**	46.60 / **82.10**	73.36 / **91.87**	98.50 / **98.70**
	Exposure	74.90 / **88.13**	**99.96** / 96.78	**99.99** / 99.26	**100.0** / 99.80	**100.0** / 99.90
	Snow	28.11 / **61.34**	61.28 / **80.52**	89.89 / **91.30**	**99.34** / 96.13	**99.98** / 97.66
	Haze	66.51 / **95.83**	97.86 / **99.50**	**100.0** / 99.95	**100.0** / 99.87	**100.0** / 99.88
	Decolor	48.37 / **62.36**	60.55 / **81.30**	71.73 / **89.93**	87.29 / **95.42**	89.68 / **96.91**

5. CONCLUSION

In this paper, we proposed a simple yet effective measure of uncertainty in deep neural networks with backpropagated gradients. We introduced the concept of the space of models in terms of optimization and motivated the utility of gradients as a distance measure in the space of model that corresponds to the required amount of adjustment to model parameters to properly represent given inputs. We designed a framework for efficient generation of gradients with confounding labels, and utilized the gradients to quantify uncertainty of fully trained networks regarding the given inputs. We validated our approach in the applications of detecting unfamiliar inputs, including out-of-distribution samples and corrupted images. Our gradient-based method outperforms activation-based state-of-the-art methods by up to 4.8% of AUROC score in out-of-distribution detection and 35.7% in corrupted input detection.

6. REFERENCES

[1] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[3] Ian Tenney, Dipanjan Das, and Ellie Pavlick, "Bert rediscovers the classical nlp pipeline," *arXiv preprint arXiv:1905.05950*, 2019.

[4] Dogancan Temel, Jinsol Lee, and Ghassan AlRegib, "Cure-or: Challenging unreal and real environments for object recognition," in *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2018, pp. 137–144.

[5] Dogancan Temel, Jinsol Lee, and Ghassan AlRegib, "Object recognition under multifarious conditions: A reliability analysis and a feature similarity-based performance estimation," in *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2019, pp. 3033–3037.

[6] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[7] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger, "On calibration of modern neural networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 1321–1330.

[8] Alex Kendall and Yarin Gal, "What uncertainties do we need in bayesian deep learning for computer vision?," in *Advances in neural information processing systems*, 2017, pp. 5574–5584.

[9] Dan Hendrycks and Kevin Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," *arXiv preprint arXiv:1610.02136*, 2016.

[10] Shiyu Liang, Yixuan Li, and Rayadurgam Srikant, "Enhancing the reliability of out-of-distribution image detection in neural networks," *arXiv preprint arXiv:1706.02690*, 2017.

[11] Terrance DeVries and Graham W Taylor, "Learning confidence for out-of-distribution detection in neural networks," *arXiv preprint arXiv:1802.04865*, 2018.

[12] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin, "A simple unified framework for detecting out-of-distribution samples and adversarial attacks," in *Advances in Neural Information Processing Systems*, 2018, pp. 7167–7177.

[13] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song, "A convergence theory for deep learning via over-parameterization," *arXiv preprint arXiv:1811.03962*, 2018.

[14] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[15] Alex Krizhevsky, Geoffrey Hinton, et al., "Learning multiple layers of features from tiny images," 2009.

[16] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[17] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao, "Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop," *arXiv preprint arXiv:1506.03365*, 2015.

[18] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng, "Reading digits in natural images with unsupervised feature learning," 2011.

[19] Dan Hendrycks and Thomas Dietterich, "Benchmarking neural network robustness to common corruptions and perturbations," *arXiv preprint arXiv:1903.12261*, 2019.

[20] Dogancan Temel, Gukyeong Kwon, Mohit Prabhushankar, and Ghassan AlRegib, "Cure-tsr: Challenging unreal and real environments for traffic sign recognition," *arXiv preprint arXiv:1712.02463*, 2017.