Uncertainty Estimation for Classifying Medical Images

Course Project Report for **DH307**: **R & D Project** (Autumn '23)

Saumya Sheth
Department of Mechanical Engg,
210020120@iitb.ac.in

***Abstract*—This research and development project, supervised by Amit Sethi, addresses a crucial challenge in deploying Artificial Intelligence (AI) models for medical image classification. The goal is to enhance AI systems' adaptability by enabling them to identify instances of uncertainty in real-world clinical settings, where personnel and data may differ from the initial training environment. Insights from two pivotal research papers guide our exploration. The first investigates distributional uncertainty in salient object detection models, adapting class-aware distribution gap exploration techniques. The second employs evidential deep learning, utilizing subjective logic and Dirichlet distributions to explicitly quantify classification uncertainty, achieving success in detecting out-of-distribution queries and improving resilience against adversarial perturbations. The practical implementation focuses on the second paper, demonstrating the application of evidential deep learning to quantify uncertainty in medical image classification. This work contributes valuable insights to healthcare analytics and AI/ML, specifically in developing deployable cautious models for medical image classification.**

I. Introduction

This report includes the overview and learnings from 2 research papers : 1) Modeling the Distributional Uncertainty for Salient Object Detection Models 2) Evidential Deep Learning to Quantify Classification Uncertainty. I have also tried to implement the second research paper. I have submitted the code, it is there in the zip file that I have submitted.

The intersection of Artificial Intelligence (AI) and healthcare presents an unprecedented opportunity to revolutionize medical image classification, with the promise of improved diagnostic capabilities. However, this transition from controlled training environments to the dynamic realities of clinical settings introduces a critical challenge: navigating uncertainties inherent in scenarios divergent from the model's initial training data. This research and development project, guided by Amit Sethi, embarks on a nuanced exploration of uncertainty estimation in medical image classification. The overarching aim is to equip AI models with the acumen to discern instances of uncertainty, fostering cautious decision-making in deployment scenarios characterized by diverse personnel and data. Drawing inspiration from two seminal research papers, our exploration delves into the intricacies of distributional uncertainty in salient object detection models and the explicit quantification of classification uncertainty through evidential deep learning. This report chronicles our exploration, methodology, and practical implementation, contributing substantive insights at the intersection of healthcare analytics and AI/ML.

II. Modeling the Distributional Uncertainty for Salient Object Detection Models

In this section, we will discuss the paper titled 'Modeling the Distributional Uncertainty for Salient Object Detection Models'. In this paper, we tackle a significant gap in salient object detection (SOD) models—the lack of explicit consideration for the distributional uncertainty between training and testing datasets. Focusing on epistemic uncertainty, specifically distributional uncertainty, we delve into class-aware distribution gap exploration techniques, such as long-tail learning, single-model uncertainty modeling, and test-time strategies, uniquely adapted for our class-agnostic SOD task. Our innovation lies in defining out-of-distribution (OOD) samples in SOD, where dissimilarity to the training dataset is determined by breaking fundamental saliency priors like the center prior, color contrast prior, and compactness prior. Unlike conventional OOD definitions, we treat OOD as a continuous challenge, not limited to closed-world training categories. Extensive experimental results validate the effectiveness of existing distribution gap modeling techniques for SOD. Our key findings suggest that both train-time single-model uncertainty estimation techniques and weight regularization solutions, preventing excessive model activation drift, hold promise for effectively modeling distributional uncertainty in the context of SOD, shedding light on an essential yet often overlooked aspect of saliency detection.

Background : Suppose the training set is D = $\{x_i, y_i\}_{i=1}^{N}$ of size N is sampled from a joint data distribution p(x,y), where i indexes the samples and is omitted when it's clear. The conventional classifier is trained to maximize the conditional log-likelihood $\log p_\theta(y|x)$, where θ represents model parameters. When deploying the trained model in real-world, its performance depends on whether the test sample x^* is from the same joint data distribution p(x,y). For x^* from p(x,y) (indicating x^* is in-distribution sample), its performance is impressive. However, when it's from a different distribution other than p(x,y) (i.e. x^* is out-of-distribution sample), the resulting p($y|x^*$) often yield incorrect predictions with high confidence. The main reason is that

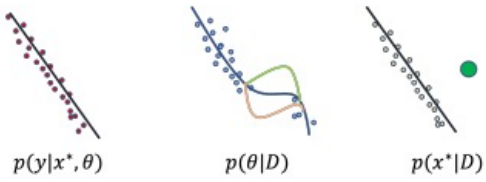

Figure 1. Visualization of different types of uncertainty, where aleatoric uncertainty ($p(y|x^*,\theta)$) is caused by the inherent randomness of the data, model uncertainty ($p(\theta|D)$) happens when there exists low-density region, leading to multiple solutions within this region, and distributional uncertainty ($p(x^*|D)$) occurs when the test sample x^* fails to fit in the model based on the training dataset D.

p($y|x^*$) does not fit a probability distribution over the whole joint data space. To fix the above issue, deep hybrid models (DHMs) can be used to model the joint distribution: p(x,y) = p(y|x)p(x). Although the trained model may still inaccurately assign high confidence p($y|x^*$) for out-of-distribution sample x^*, effective marginal density modeling of p(x^*) can produce low density for it, leading to reliable p(x^*,y).
Given a testing sample x^*, with a deep hybrid model, proposes to factorize the posterior joint distribution p($x^*, y|\theta, D$) via:

$$p(x^*, y|\theta, D) = \underbrace{p(y|x^*,\theta)}_{\text{data}}\ \underbrace{p(x^*|D)}_{\text{distributional}}\ \underbrace{p(\theta|D)}_{\text{model}}, \qquad (1)$$

where the basic assumption is that p($x^*|\theta$,D) = p($x^*|D$), which is true as θ is obtained based on D. p($y|x^*,\theta$) is used to model data uncertainty or aleatoric uncertainty, which is inherent in the data generation process. p($x^*|D$) represents distributional uncertainty, explaining the distribution gap between the test sample and the training dataset. p($\theta|D$) is model uncertainty, representing the uncertainty of model parameters given the current training dataset. The latter two can be combined and termed as “epistemic uncertainty”.

The paper addresses the challenges of aleatoric and distributional uncertainty in saliency detection. Aleatoric uncertainty, inherent and unexplainable with more data, contrasts with model uncertainty, which can be reduced with additional training data. The focus is on ”distributional uncertainty,” indicating the level of ”out-of-distribution” (OOD) difficulty to mitigate. The study defines OOD samples for saliency detection as those deviating from fundamental saliency priors like center, color contrast, and compactness. Unlike discrete OOD, saliency OOD is considered ”continuous.” The paper introduces the exploration of distributional uncertainty for saliency detection, adapting class-aware distribution gap techniques for a class-agnostic task. Key findings from extensive experiments include the superiority of deep ensemble structures over Monte Carlo dropout, challenges with categorical distribution-based longtail solutions for continuous saliency detection, the effectiveness of single-model uncertainty methods, especially at training time, and the potential of test-time testing solutions in generating reliable distributional uncertainty for salient object detection through data augmentation.

III. Distributional Uncertainty Modeling

As shown in Eq 1, distributional uncertainty indicates the gap between the test sample x^* and the training dataset D. We mainly explore three types of distribution gap modeling techniques, including long-tail learning techniques, single model uncertainty estimation and test-time strategies.

A. Long-tail learning to overcome distribution bias

Long-tail learning methods are developed to address class imbalance in training data, where the ”distribution gap” arises from the disparity between long-tailed training and uniform testing. Existing solutions can be categorized into data resampling-based methods and loss reweighting strategies. Data resampling involves over-sampling tail classes and under-sampling head classes to achieve balance, while loss reweighting regularizes model parameters, emphasizing tail classes through a class-balanced loss function.
Model rebalance techniques : Model-rebalance techniques aim to directly debias models resulting from long-tailed training. An example involves the design of a diverse expert learning network that models different distributions and employs test-time self-supervised learning to determine aggregation weights for diverse experts. The network features a shared extractor and three category prediction heads simulating long-tailed, uniform, and inverse long-tailed distributions. The final loss function is defined based on the outputs from these prediction heads. Additionally, test-time self-supervised learning ensures prediction consistency and the learning of aggregation weights $\{\alpha_l\}_{l=1}^{3}$ to generate the final prediction.
Loss reweighting techniques : Loss reweighting methods aim to balance contributions from head and tail classes. A two-stage learning method is introduced, wherein the initial stage learns feature representation via weight decay, and the second stage combines class-balanced loss function L_{CB}, weight decay, and MaxNorm to learn classification layer weights. The class-balanced loss function is defined to address the imbalances, offering a comprehensive approach to mitigating the effects of long-tailed training on model bias. The class-balanced loss function can be written as:

$$\mathcal{L}_{CB}(x) = \frac{1-\beta}{1-\beta^{N_y}}\log(\hat{y}), \qquad (2)$$

where $\{N_c\}_{c=1}^{C}$ is the number of class c in training dataset, y is the ground truth class label, $\hat{y}$ is the predicted class probability, and β is a hyperparameter.
Post-hoc techniques : Instead of training the model with model/loss rebalance strategies, long-tail learning can also be achieved via post-hoc techniques, i.e. (NorCal) optimizes the general Softmax process by counting the number of each class $\{N_c\}_{c=1}^{C}$ and calculating the class weights. The probability that the image x is predicted to be class c is then defined as:

$$p(c|x) = \frac{\exp(\phi_c(x))/\alpha_c}{\sum_{c'=1}^{C} \exp(\phi'_c(x))/\alpha'_c + \exp(\phi_{c+1}(x))}, \quad (3)$$

where α_c is monotonically increasing with respect to N_c, ϕ_c is the predicted logit of class c and c + 1 is the background class for object detection task. In this way, the scores for head classes will be suppressed.

Long-tail techniques for saliency detection: Applying existing long-tail learning methods to saliency detection encounters a challenge due to the unique nature of saliency being "attribute"-based and continuous, making the identification of tail classes complex. To address this, a "continuous version" of long-tail learning for saliency detection is introduced.

1) Model-Rebalance-Based Saliency Distributional Uncertainty Modeling: To adapt long-tail learning for saliency detection, a multi-head dense prediction network is constructed with a shared encoder f_θ^b and three decoders $\{f_{\theta_l}^d\}_{l=1}^3$ simulating different distributions. The model is trained with a loss function combining the three distribution-head losses. After training, test-time training is performed to learn aggregation weights for each test sample, utilizing a loss function based on cosine similarity. As a binary dense prediction task, the outputs $\hat{y}_l \in \mathbb{R}^{C\times H\times W}$ of these three heads are pixel-wise predictions after Softmax with C = 2. After the first stage training of the multi-head model, the test-time training is performed to learn the aggregation weights $\{\alpha_l\}_{l=1}^3$ for each test sample x_t with loss function defined as:

$$\mathcal{L}_\alpha(x_t) = -\text{sim}(\hat{y}(x_t), \hat{y}'(x'_t)), \quad (4)$$

where $\hat{y}(x_t) = \sum_{l=1}^3 \alpha_l \cdot \hat{y}_l(x_t)$ is the final prediction of x_t, sim(,) is the cosine similarity, x'_t represents the augmented image, where we use the horizontal flip and $\hat{y}'$ represents the inverse augmentation operation on the output.

2) Loss-Reweighting-Based Saliency Detection: Employing a two-stage training framework, the first stage focuses on feature representation learning using conventional models with weight decay. In the second stage, a class-weighted loss function is employed to balance the class distribution, addressing class imbalances.

$$\mathcal{L}_{CB}(x_u) = \alpha_u(y_u \log(\hat{y}_u) + (1 - y_u)\log(1 - \hat{y}_u)), \quad (5)$$

where $\alpha_u = (1-\beta)/(1-\beta^{N_y})$ is a pixel-wise classbalanced weight, and y_u is the ground truth class label of pixel u. In practice, for the second stage training, we freeze all parameters from the first stage training except the last prediction convolutional layer and use weight decay, MaxNorm and pixel-wise class-balanced loss functions simultaneously to optimize the predictions

3) Post-hoc Long-Tail Learning for Saliency Detection: A post-hoc technique is introduced, utilizing the class weighting function to recalibrate output logits and obtain predicted probabilities for saliency detection. The recalibration considers the probability of a pixel being predicted as foreground, incorporating monotonically increasing weights with respect to the number of foreground and background pixels. The probability that pixel x_u is predicted to be foreground is:

$$p_f(x_u) = \frac{\exp(\phi_f(x_u))/\alpha_f}{\exp(\phi_f(x_u))/\alpha_f + \exp(\phi_b(x_u))/\alpha_b}, \quad (6)$$

where α_f and α_b monotonically increase with respect to the number of foreground pixels N_f and background pixels N_b respectively. ϕ_f, ϕ_b are foreground and background logits respectively.

B. Single-model uncertainty

Uncertainty estimation is a crucial aspect aimed at assessing the uncertainty inherent in the data or the model. As illustrated in Figure 1, aleatoric uncertainty is considered inherent and cannot be mitigated with additional data. Our specific focus lies on epistemic uncertainty, particularly concerning the modeling of out-of-distribution (OOD) samples, aligning with the concept of distributional uncertainty estimation. The conventional approach to addressing OOD detection involves designing an OOD detector g(x) based on a confidence score function h(x). This detector categorizes input samples as either in-distribution (ID) or out-of-distribution (OOD) using a threshold τ. The OOD threshold is selected to ensure accurate classification of a high fraction of ID data. The central concern in OOD detection is defining a reliable score h(x), which can be achieved through various approaches, including post-hoc techniques such as gradient-based methods and energy-score-based methods, as well as training techniques involving loss function regularization, data reprocessing, and feature regularization. The formula representing the OOD detector is given by:

$$g(x) = \begin{cases} \text{ID}, & \text{if} \quad h(x) \geq \tau, \\ \text{OOD}, & \text{if} \quad h(x) < \tau, \end{cases} \quad (7)$$

1) Post-hoc techniques: Post-hoc techniques for uncertainty estimation include methods that either directly compute confidence from the model output, such as Maximum Class Probability (MCP), or reinterpret the Softmax output through an energy-based model formulation. Energy-based models either renormalize the logit space based on training dataset statistics or compute the Kullback–Leibler divergence of the output distribution from the uniform distribution, assuming OOD samples have a uniform predictive distribution across categories.

The MCP method assumes that correctly classified samples tend to have higher maximum softmax probabilities than misclassified or OOD samples. Specifically, for pixel u, the MCP (confidence) is obtained as $h_u = \max\{p_f, p_b\}$, where

p_f and p_b are the probabilities of pixel u being predicted as foreground/background in a binary segmentation task.
The energy-based model, introduced as the Energy Score, defines free energy $E_\theta(x)$ as $-\log \int \exp(-E_\theta(x, y'))\, dy'$. Connecting this with the Softmax function, the confidence score is defined as $h = -E_\theta(x)$.
SML calculates mean (m_f, m_b) and variance (v_f, v_b) of foreground and background logits on the training set, standardizing max logits. The standardized max logit for pixel u is $\phi_s = (\phi_m - m_f)/v_f$ if the label of pixel u is foreground, followed by boundary suppression and smoothing operations to enhance the confidence map.
Gradient-based Confidence (GC) assumes a uniform predictive distribution for OOD samples. An extension, ExGrad, calculates pixel confidence $h(u) = 1 - 2.p_f.p_b$, considering the probabilities of pixel u being in different classes in a classification setting.
2) Training techniques: Training a single-model uncertainty estimation network involves various techniques such as loss function regularization, data re-processing, and weight regularization. One approach, proposed by, introduces an additional CondifNet to learn the true class probability (TCP), reflecting prediction confidence. This method is adapted for saliency detection by adding a decoder to predict the confidence map $h \in \mathbb{R}^{H \times W}$. The output is constrained by the Mean Squared Error (MSE) loss with the true class probability value, defined as $\mathcal{L}^{tcp}(x_u) = (h_u - p_u^\star)^2$, where $p_u^\star$ is the ground truth class probability of pixel u, and uncertainty is defined as u = 1 - h.

Alternatively, data-renormalization, achieves single-model uncertainty estimation by training on a dataset shifted by random constant biases. This method alters the conventional model, allowing it to generate multiple predictions for an input x by setting a random anchor z. The model is trained to map the input x to label y by considering the pair (z,xz). In the experiment,a random image is selected from the mini-batch as a random anchor z, concatenate it with the residual between the input image and the anchor xz, and feed this to the model f_θ. The loss function is defined as $\mathcal{L}^{dc}(x) = \mathcal{L}(f_\theta(z, x-z), y)$

Another technique, introduced by (ReAct), involves rectified activation on the features r(x) of the model through a set threshold α. The threshold is determined based on the p-th percentile of activations estimated on the in-distribution data. ReAct is performed on the features of the trained base model's penultimate layer, statistically obtaining a threshold α. After several counts, the threshold is set to 5, and the features after ReAct are fed into the last layer to obtain predictions, following the process described as $r'(x) = \min(r(x), \alpha)$ and $\hat{y} = f_l(r'(x))$

C. Test-time Strategies

Test-time strategies play a crucial role in refining model predictions during deployment. Two prominent approaches, test-time training (TTT) and test-time augmentation (TTA), enhance the adaptability of models to unseen data.
Test-time Training (TTT) : TTT involves adapting the model parameters for each test sample by optimizing specific parts. The loss function focuses on the consistency of predictions before and after augmentation t, and the model parameters are updated accordingly:

$$\mathcal{L}^{TTT}(x) = \mathcal{L}(f_\theta(x), f_\theta(t(x))),$$
$$\theta \leftarrow \theta - r\nabla_\theta \mathcal{L}^{TTT}(x),$$

where r is the learning rate.
Test-time Augmentation (TTA) : TTA enhances network performance through data augmentation during testing without adding additional parameters. Multiple augmentations T are selected, and their predictions are integrated to obtain the final prediction result:

$$y_{TTA}(x) = \sum_{t \in \mathcal{T}} \alpha_t \cdot f_\theta(t(x)),$$

where α_t represents the integration weight for transformation t.
Test-time Strategies for Saliency Detection: For saliency detection, specific strategies are employed:
Test-time Training with Self-Supervised Learning:
A teacher-student network is adopted, where the student network is trained on the original image, and the teacher network is trained on the augmented image. Self-supervised learning optimizes parameters based on the consistency loss between teacher and student predictions.

$$\mathcal{L}(x) = \mathcal{L}(f_{\theta_s}(x), f_{\theta_t}(t(x))),$$
$$\theta_s \leftarrow \theta_s - r\nabla_{\theta_s}\mathcal{L}(x),$$
$$\theta_t \leftarrow \alpha\theta_t + (1-\alpha)\theta_s.$$

Cycling Test-time Augmentations for Stability:
Augmentations are selected based on ranking loss predictions during training. Augmentations with the smallest loss are iteratively applied to stabilize the predictions.

$$\mathcal{L}_{\text{loss}}(x) = \mathcal{L}_{\text{rank}}(f_l(x), \mathcal{L}(f_\theta(x), y)),$$
$$t = \min_{t \in \mathcal{T}} f_l(t(x)), \quad x \leftarrow t(x).$$

This process can be repeated to obtain multiple predictions, with entropy values used for weighted integration and uncertainty focus.
Transformation Candidates for Saliency Detection:
Transformations such as adding random Gaussian noise, horizontal flipping, and size scaling are applied to saliency detection tasks, providing diverse transformation candidates for TTA.
These strategies collectively contribute to refining model predictions and ensuring adaptability during deployment.

IV. Experiments

Training/testing dataset: In line with conventional practices, models undergo training on the DUTS training dataset, containing 10,553 samples. Subsequently, the models are assessed on three benchmark testing datasets: DUTS testing

dataset, ECSSD, and DUT dataset.
Evaluation metrics: Model performance is evaluated using three standard metrics—maximum F-measure, IoU, and Accuracy. The F-measure, denoted as F_β, incorporates precision and recall based on binarized saliency predictions and ground truth maps. IoU (Intersection over Union) score is calculated using binary predicted masks with adaptive thresholds and ground truth. Accuracy assesses the proportion of correctly assigned pixels in binarized saliency predictions. To gauge the distributional uncertainty modeling capability, additional metrics are reported for out-of-distribution evaluation: area under the receiver operating characteristics (AUROC) and false positive rate at a true positive rate of 95% (FPR95).

Evaluate Sample Difficulty:

Due to the inherent "continuous" nature of saliency, identifying out-of-distribution samples based solely on categories or category distribution is challenging. Instead, sample-difficulty indicator is adopted to roughly determine the difficulty of each test sample. Sample difficulty through gradient variance during training is measured. By setting multiple checkpoints during training, gradient maps for each sample are obtained. The variance of these gradient maps (VoG) is used as a difficulty score. Large VoG scores indicate significant gradient changes during training, identifying difficult samples, which are considered out-of-distribution in our context. Unless specified otherwise, samples for visual comparisons are selected from the hard sample pool based on the VoG score.

Uncertainty Computation:

Except for methods directly predicting confidence/uncertainty, we determine uncertainty by assessing the entropy of the predicted probability. While variance is commonly employed for uncertainty generation in prediction ensemble-based regression models, for our binary segmentation task, entropy of the mean prediction serves as a more suitable confidence measure.
Entropy, a measure of disorder or unpredictability, is utilized in our context. For in-distribution (ID) samples, our goal is to generate focused predictions on a specific category, resulting in low entropy. Conversely, for out-of-distribution (OOD) samples, the model should produce a uniform distribution across categories, leading to high entropy. Consequently, the entropy score proves valuable for distinguishing in-distribution and out-of-distribution samples.

Performance of Distribution Gap Modeling Techniques for Saliency Detection

Conventional Solutions Analysis:
Two extensively studied conventional epistemic uncertainty modeling strategies are Monte Carlo (MC) dropout and Deep ensemble. MC dropout approximates a Bayesian neural network by applying dropout after the convolutional layer. This is further relaxed where it's proven that adding dropout in the decoder is sufficient for Bayesian approximation. Deep ensemble involves multiple mapping functions from the input space to the output space, proving to be the most effective model uncertainty modeling technique, albeit computationally more expensive compared to the free-lunch MC-dropout technique. Quantitative evaluation results of traditional uncertainty modeling methods (dropout and deep ensemble). Compared with the base model and Dropout strategy, the deep ensemble method integrates information from multiple decoders, enhancing accuracy in distributional uncertainty modeling.
Long-tail Learning-based Methods:
Performance of long-tail methods for SOD indicates consistently inferior performance of each solution compared to the base model. This is primarily because long-tail learning models rely on the size of each category as class-balance weight, assuming no transition across categories. However, as saliency is a continuous attribute, foreground and background can be transferred flexibly, making the continuous adaptation of discrete long-tail methods less effective for our task.
Single-model Uncertainty Techniques:
Multiple post-hoc and training regularization methods are employed for single-model uncertainty modeling. Prediction accuracy is hardly affected. Post-hoc techniques fail to improve uncertainty quality. In contrast, training techniques, especially TCP and ReAct, are effective in generating reliable uncertainty. Visualizations of generated uncertainty maps from TCP and ReAct clearly demonstrate their advantages for both accurate prediction and reliable uncertainty generation. Although DC is efficient for single-model uncertainty generation, its perturbation-based approach significantly affects accuracy for pixel-level prediction.
Test-time Solutions:
Test-time strategies aim to enhance performance through self-supervised adaptive learning on the test set (CoTTA) or aggregation of multiple augmentations of the test data (CTTA). The test-time training method may lead to the model focusing on mis-classified pixels in self-supervised learning, resulting in performance degradation. In such cases, a carefully designed strategy to prevent excessive model drift is necessary. Conversely, the test-time testing method has the potential to generate reliable uncertainty if proper augmentation policies are learned. Qualitative results illustrate that the use of data augmentation methods can help explore low-density areas and calibrate mis-classified regions to learn correct distributional uncertainty. However, for dense prediction tasks like SOD, augmentations affecting pixel information, such as interception on test images, are challenging due to the requirement of pixel-by-pixel correspondence. Augmentation degree limitations and restrictions on pixel value changes in dense prediction tasks hinder flexibility compared to image classification tasks.

Analysis:

MC Dropout and Deep Ensemble:
MC dropout achieved by randomly dropping connections between neurons during both training and testing, and Deep ensemble, designed to achieve multiple mapping functions, were explored. Experiments revealed that MC dropout, despite being a ”free lunch” method, fails to generate reliable uncertainty without proper control of the dropout mask. Long-tail solutions, effective for class-independent classification tasks, were found to be less effective in modeling distributional uncertainty for the class-dependent continuous segmentation task.
Single-Model Uncertainty Techniques:
Single-model uncertainty techniques were promising for directly addressing the distribution gap. Post-hoc methods, relying too much or being based on biased assumptions (e.g., gradient-based confidence estimation methods assuming uniform distribution for out-of-distribution samples), were scrutinized. Training regularization methods, involving loss regularization, feature activation, or data regularization, proved suitable for the task, avoiding class-independent assumptions.
Test-time Strategies:
Test-time strategies especially test-time training, were considered straightforward and promising. While current experiments may not have generated reliable uncertainty maps, it is believed that using suitable regularization to control the drift degree of test-time training can be promising for generating reliable distributional uncertainty. The distribution of the AUROC metric on the DUTS testing dataset illustrated the effectiveness of deep ensemble and TCP in generating reliable distributional uncertainty.

V. Conclusion

Despite substantial progress in improving SOD model performance on benchmark testing datasets, little attention has been given to the out-of-distribution problem in SOD. This paper represents the first effort to investigate the out-of-distribution discovery issue for SOD, aiming to explain the distribution gap. Extensive experimental results verified the effectiveness of deep ensemble and the single-model uncertainty estimation technique, TCP, in generating reliable distributional uncertainty. Long-tail learning solutions, while effective for class-independent classification tasks, failed to generalize well to the class-dependent task. The current implementation of test-time training, though not improving uncertainty quality, shows promise for further exploration with regularization terms to control the model’s drift degree for better uncertainty generation.

Now 2nd research paper:

EVIDENTIAL DEEP LEARNING TO QUANTIFY CLASSIFICATION UNCERTAINTY

Abstract:
While deterministic neural networks have demonstrated proficiency in learning effective predictors across diverse machine learning tasks, a common limitation arises from the inherent ignorance of prediction confidence in standard training approaches, which minimize prediction loss. In contrast to Bayesian neural networks that infer prediction uncertainty indirectly through weight uncertainties, this paper introduces an explicit modeling approach based on the theory of subjective logic. Through the incorporation of a Dirichlet distribution on class probabilities, predictions from a neural network are treated as subjective opinions. The function collecting evidence leading to these opinions is learned by a deterministic neural network from the available data. The resulting predictor for a multi-class classification problem is represented by another Dirichlet distribution, with parameters determined by the continuous output of a neural network.
A preliminary analysis delves into how the unique properties of the proposed loss function contribute to enhanced uncertainty estimation. The method demonstrates remarkable success in detecting out-of-distribution queries and exhibits robustness against adversarial perturbations. The novel perspective offered by this approach sheds light on the explicit modeling of prediction confidence through subjective logic, providing insights into improved uncertainty characterization in neural network predictions.

I. INTRODUCTION

The beginning of this decade has witnessed the pervasive influence of the deep learning paradigm in the field of machine learning. Modern deep neural network architectures, complemented by recent innovations like dropout, batch normalization, and skip connections, have achieved remarkable success across diverse machine learning applications. These advancements have led to unprecedented prediction accuracies, surpassing human-level performance in certain cases. While the surge in interest and investment in deep learning research is evident, it has also brought about a pressing need for addressing challenges related to robustness, sample efficiency, security, and interpretability.
In scenarios with ample labeled data, achieving high accuracy by adhering to a set of established guidelines has become commonplace. Consequently, the focus of the upcoming era is expected to shift from merely improving test set accuracy to addressing other critical questions. Key considerations include the network's ability to identify data points from unrelated distributions, its capability to express uncertainty by acknowledging when it does not have the necessary knowledge (e.g., recognizing a cat picture after being trained on handwritten digits), and, crucially, its resilience against adversarial attacks. These challenges have prompted research into Bayesian Neural Networks (BNNs), which aim to estimate prediction uncertainty by approximating the moments of the posterior predictive distribution.
This paper specifically concentrates on the uncertainty estimation problem, approaching it from the perspective of the Theory of Evidence. The standard softmax output of a classification network is interpreted as the parameter set of a categorical distribution. By replacing this parameter set with the parameters of a Dirichlet density, the predictions of the learner are represented as a distribution over potential softmax outputs, rather than a singular estimate. The resulting model incorporates a specific loss function, minimized through standard backpropagation subject to neural network weights.
Through a series of experiments, the paper demonstrates the superiority of this technique over state-of-the-art BNNs, particularly in applications where high-quality uncertainty modeling is crucial. Notably, the predictive distribution of the proposed model approaches the maximum entropy setting more closely than BNNs when confronted with inputs from distributions distinct from the training samples. The model's sensible response to input digit rotation is illustrated, showcasing a reduction in classification probabilities and an increase in prediction uncertainty, unlike the standard softmax that maintains high confidence for incorrect classes under significant rotations. Furthermore, the model exhibits enhanced robustness to adversarial attacks across two benchmark datasets.

II. DEFICIENCIES OF MODELING CLASS PROBABILITIES WITH SOFTMAX

The conventional approach in deep neural networks involves employing the softmax operator to convert the continuous activations of the output layer into class probabilities. This yields a model that can be interpreted as a multinomial distribution, where discrete class probabilities are determined by the outputs of the neural network. For a K-class classification problem, the likelihood function for an observed tuple (x, y) is expressed as: $P(y|x,\theta) = \text{Mult}(y|\sigma(f_1(x,\theta)),\ldots,\sigma(f_K(x,\theta)))$ Here, Mult(.....) represents the multinomial mass function, $f_j(x,\theta)$ denotes the jth output channel of an arbitrary neural network f(.) parameterized by θ, and $\sigma(u_j) = \frac{e^{u_j}}{\sum_{i=1}^{K} e^{u_i}}$ is the softmax function. The continuous neural network adjusts the ratio of class probabilities, and softmax compresses these ratios into a simplex. The final likelihood is maximized with respect to the neural network parameters θ, often framed as minimizing the negative log-likelihood or cross-entropy loss for computational convenience:
$-\log p(y|x,\theta) = -\log \sigma(f_y(x,\theta))$ This loss function is recognized as Maximum Likelihood Estimation (MLE), a frequentist technique that lacks the capacity to infer the variance of the predictive distribution. Softmax, by design, tends to inflate the probability of the predicted class due to the exponentiation applied to neural network outputs. Consequently, uncertainty estimations based on softmax can be unreliable, as the absolute distance of the predicted label for a new observation provides limited information beyond its relative comparison with other classes.

Drawing inspiration from prior works, the inadequacies of softmax are illustrated in Figure 1 (left panel) using the example of LeNet attempting to classify a rotated digit 1 from the MNIST dataset. As the digit undergoes continuous rotation, LeNet, employing softmax, struggles to maintain accurate classification. For moderate degrees of rotation, correct classification is achieved, but as rotation increases, misclassifications occur (e.g., as 2 or 5). The softmax-based probabilities remain high even for misclassified samples. In contrast, the proposed approach in this paper accurately quantifies the uncertainty of predictions (Figure 1, right panel).

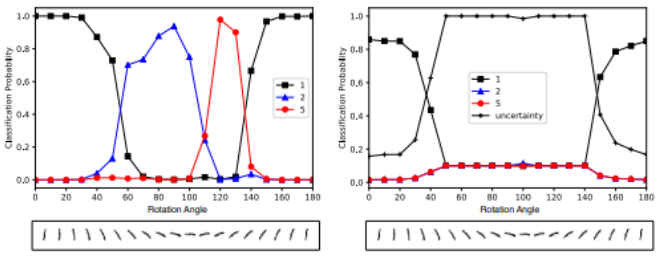

Figure 1: Classification of the rotated digit 1 (at bottom) at different angles between 0 and 180 degrees. **Left:** The classification probability is calculated using the *softmax* function. **Right:** The classification probability and uncertainty are calculated using the proposed method.

III. Uncertainty and the Theory of Evidence

The Dempster–Shafer Theory of Evidence (DST) serves as a generalization of the Bayesian theory, extending it to incorporate subjective probabilities. Within DST, belief masses are assigned to subsets within a frame of discernment, which outlines the possible exclusive states, such as various class labels for a given sample.

In this context, a belief mass is a measure assigned to any subset, including the entire frame, indicating the belief regarding the possible states. Subjective Logic (SL) operationalizes DST's concept of belief assignments through the utilization of a Dirichlet Distribution. This approach provides a structured theoretical framework for the quantification of belief masses and associated uncertainty.

Subjective Logic Formalism

SL introduces a frame with K mutually exclusive singletons (representing class labels) and assigns belief masses b_k to each singleton (where k = 1,...,K). Additionally, an overall uncertainty mass u is considered. These values adhere to the constraint $u + \sum_{k=1}^{K} b_k = 1$, highlighting that belief masses and uncertainty together account for the entire possibility space.

Belief mass b_k for a specific singleton is determined by the evidence e_k related to that singleton. The calculations are expressed as:

$b_k = e_k/S$

Where $S = \sum_{i=1}^{K}(e_i + 1)$. Notably, the uncertainty u is inversely proportional to the total evidence.

Dirichlet Distribution Representation

The Dirichlet distribution is characterized by parameters $\alpha = [\alpha_1, ..., \alpha_K]$. Its probability density function (pdf) is given by:

$D(p|\alpha) = \frac{1}{B(\alpha)} \prod_{i=1}^{K} p_i^{\alpha_i - 1}$ for $p \in S^K$

Where S^K represents the K-dimensional unit simplex.

Practical Illustration

Consider a belief mass assignment b = [0.8,0,...,0] for a 10-class classification problem. In this scenario, the prior distribution becomes a uniform Dirichlet distribution, reflecting total uncertainty. Following training, if belief masses become b=[0.8,0,...,0], the Dirichlet strength is computed as S=50, and the corresponding Dirichlet distribution is $D(p|\alpha = [41, 1, ..., 1])$.

Neural Network Opinion Formation

The proposition here is that a neural network has the capacity to formulate opinions for classification tasks in the form of Dirichlet distributions. If $\alpha_i = [\alpha_{i1}, ..., \alpha iK]$ represents the parameters of a Dirichlet distribution for the classification of a sample, then $(\alpha_{ij} - 1)$ signifies the total evidence estimated by the network for the assignment of the sample to the j-th class.

Moreover, the epistemic uncertainty of the classification can be readily computed using Equation before.

$\hat{p}^k = \frac{\alpha_k}{S}$

This interpretation provides a nuanced understanding of how a neural network can articulate opinions using Dirichlet distributions and quantifies the uncertainty through a systematic framework.

IV. Learning to Form Opinions

In our approach, we address the limitation of the softmax function, which provides a singular estimate for class probabilities without capturing associated uncertainties. Instead, we propose using multinomial opinions, modeled as Dirichlet distributions, to represent probability distributions for class assignments. Our strategy involves designing and training neural networks to formulate multinomial opinions for the classification of a given sample i, represented by a Dirichlet distribution $D(p_i|\alpha_i)$, where p_i is a simplex denoting class assignment probabilities.

Our neural network architecture for classification closely resembles traditional models, with a key modification: we replace the softmax layer with an activation layer (e.g., ReLU) to ensure non-negative output. This output serves as the evidence vector for the predicted Dirichlet distribution. For a given sample i, denoting the evidence vector predicted by the network as $f(x_i|\Theta)$, where Θ represents the network parameters, the corresponding Dirichlet distribution parameters are set as $\alpha_i = f(x_i|\Theta) + 1$. Once these parameters are determined, the mean of the Dirichlet distribution, i.e., α_i/S_i, serves as an estimate for the class probabilities.

Let y_i be a one-hot vector encoding the ground-truth class of observation x_i with y_{ij} = 1 and y_{ik} = 0 for all $k \neq j$, and α_i be the parameters of the Dirichlet density on the predictors. First,

we can treat $D(p_i|\alpha_i)$ as a prior on the likelihood $Mult(y_i|p_i)$ and obtain the negated logarithm of the marginal likelihood by integrating out the class probabilities

$$\mathcal{L}_i(\Theta) = -\log\left(\int \prod_{j=1}^{K} p_{ij}^{y_{ij}} \frac{1}{B(\boldsymbol{\alpha}_i)} \prod_{j=1}^{K} p_{ij}^{\alpha_{ij}-1} d\mathbf{p}_i\right) = \sum_{j=1}^{K} y_{ij}\Big(\log(S_i) - \log(\alpha_{ij})\Big) \quad (3)$$

and minimize with respect to the α_i parameters. This technique is well-known as the Type II Maximum Likelihood.

Alternatively, we can define a loss function and compute its Bayes risk with respect to the class predictor. Note that while the above loss in Equation 3 corresponds to the Bayes classifier in the PAC-learning nomenclature, ones we will present below are Gibbs classifiers. For the cross-entropy loss, the Bayes risk will read

$$\mathcal{L}_i(\Theta) = \int \left[\sum_{j=1}^{K} -y_{ij}\log(p_{ij})\right] \frac{1}{B(\boldsymbol{\alpha}_i)} \prod_{j=1}^{K} p_{ij}^{\alpha_{ij}-1} d\mathbf{p}_i = \sum_{j=1}^{K} y_{ij}\Big(\psi(S_i) - \psi(\alpha_{ij})\Big), \quad (4)$$

where $\psi()$ is the digamma function. The same approach can be applied also to the sum of squares loss $||y_i p_i||_2^2$, resulting in

$$\begin{aligned}\mathcal{L}_i(\Theta) &= \int ||\mathbf{y}_i - \mathbf{p}_i||_2^2 \frac{1}{B(\boldsymbol{\alpha}_i)} \prod_{i=1}^{K} p_{ij}^{\alpha_{ij}-1} d\mathbf{p}_i \\ &= \sum_{j=1}^{K} \mathbb{E}\Big[y_{ij}^2 - 2y_{ij}p_{ij} + p_{ij}^2\Big] = \sum_{j=1}^{K} \Big(y_{ij}^2 - 2y_{ij}\mathbb{E}[p_{ij}] + \mathbb{E}[p_{ij}^2]\Big).\end{aligned}$$

Among the three options presented above, we choose the last based on our empirical findings. We have observed the losses in Equations 3 and 4 to generate excessively high belief masses for classes and exhibit relatively less stable performance than Equation 5. We leave theoretical investigation of the disadvantages of these alternative options to future work, and instead, highlight some advantageous theoretical properties of the preferred loss below. The first advantage of the loss in Equation 5 is that using the identity

$$\mathbb{E}[p_{ij}^2] = \mathbb{E}[p_{ij}]^2 + \mathrm{Var}(p_{ij}),$$

we get the following easily interpretable form

$$\begin{aligned}\mathcal{L}_i(\Theta) &= \sum_{j=1}^{K} (y_{ij} - \mathbb{E}[p_{ij}])^2 + \mathrm{Var}(p_{ij}) \\ &= \sum_{j=1}^{K} \underbrace{(y_{ij} - \alpha_{ij}/S_i)^2}_{\mathcal{L}_{ij}^{err}} + \underbrace{\frac{\alpha_{ij}(S_i - \alpha_{ij})}{S_i^2(S_i+1)}}_{\mathcal{L}_{ij}^{var}} \\ &= \sum_{j=1}^{K} (y_{ij} - \hat{p}_{ij})^2 + \frac{\hat{p}_{ij}(1-\hat{p}_{ij})}{(S_i+1)}.\end{aligned}$$

By decomposing the first and second moments, the loss aims to achieve the joint goals of minimizing the prediction error and the variance of the Dirichlet experiment generated by the neural net specifically for each sample in the training set. While doing so, it prioritizes data fit over variance estimation, as ensured by the proposition below.

Proposition 1.

For any $\alpha_{ij} \geq 1$, the inequality $\mathcal{L}_{ij}^{var} < \mathcal{L}_{ij}^{err}$ is satisfied.

The next step towards capturing the behavior of Equation 5 is to investigate whether it has a tendency to fit to the data. We assure this property thanks to our next proposition.

Proposition 2.

For a given sample i with the correct label j, L_i^{err} decreases when new evidence is added to α_{ij} and increases when evidence is removed from α_{ij}. A good data fit can be achieved by generating arbitrarily many evidences for all classes as long as the ground-truth class is assigned the majority of them. However, in order to perform proper uncertainty modeling, the model also needs to learn variances that reflect the nature of the observations. Therefore, it should generate more evidence when it is more sure of the outcome. In return, it should avoid generating evidences at all for observations it cannot explain. Our next proposition provides a guarantee for this preferable behavior pattern, which is known in the uncertainty modeling literature as learned loss attenuation

Proposition 3.

For a given sample i with the correct class label j, L_i^{err} decreases when some evidence is removed from the biggest Dirichlet parameter α_{il} such that $l \neq j$.

Combining these ideas, neural networks using Equation 5 aim to emphasize gathering evidence for the correct class in each sample. This approach helps prevent misclassifications by minimizing confusing evidence. The loss function also works to make predictions more consistent within the training set by boosting evidence when it improves the model's understanding of the data. You can find detailed proofs for these ideas in the appendix.

To calculate the overall loss for a group of training samples, we just add up the individual losses for each sample. As the model learns from the data, it adapts by generating evidence for specific class labels to bring down the overall loss. For instance, if there's a circular pattern in MNIST images associated with the digit zero, the model learns to increase the evidence for recognizing this pattern.

During training, when the model encounters examples that don't match its learned patterns (counterexamples), it adjusts its parameters to reduce evidence for these cases. This adjustment is crucial, especially when there are only a few counterexamples. The challenge here is that reducing evidence for specific patterns may, counterintuitively, increase the overall loss. This situation can lead the model to provide misleading evidence, supporting incorrect labels.

Ideally, if the model can't accurately classify a sample, we'd like the total evidence to shrink to zero. This scenario is represented by a Dirichlet distribution with zero total evidence (S=K), indicating complete uncertainty (u=1). To achieve this, we introduce a Kullback-Leibler (KL) divergence term into our loss function. This term penalizes deviations from the "I do not know" state that don't contribute to improving the model's understanding of the data. The loss with this regularizing term reads

$$\mathcal{L}(\Theta) = \sum_{i=1}^{N} \mathcal{L}_i(\Theta) + \lambda_t \sum_{i=1}^{N} KL[D(\mathbf{p_i}|\tilde{\boldsymbol{\alpha}}_i) \,||\, D(\mathbf{p_i}|\langle 1, \ldots, 1 \rangle)],$$

where $_t = min(1.0, t/10) \in [0, 1]$ is the annealing coefficient, t is the index of the current training epoch, $D(p_i| < 1, ..., 1 >)$ is the uniform Dirichlet distribution, and lastly $\bar{i} = y_i + (1y_i) \odot \alpha_i$ is the Dirichlet parameters after removal of the non-misleading evidence from predicted parameters α_i for sample i. The KL divergence term in the loss can be calculated as

$$KL[D(\mathbf{p_i}|\tilde{\boldsymbol{\alpha}}_i) \,||\, D(\mathbf{p_i}|\mathbf{1})] = \log\left(\frac{\Gamma(\sum_{k=1}^{K}\tilde{\alpha}_{ik})}{\Gamma(K)\prod_{k=1}^{K}\Gamma(\tilde{\alpha}_{ik})}\right) + \sum_{k=1}^{K}(\tilde{\alpha}_{ik}-1)\left[\psi(\tilde{\alpha}_{ik}) - \psi\Big(\sum_{j=1}^{K}\tilde{\alpha}_{ij}\Big)\right],$$

where 1 represents the parameter vector of K ones, $\Gamma(.)$ is the gamma function, and $\psi()$ is the digamma function. By gradually increasing the effect of the KL divergence in the loss through the annealing coefficient, we allow the neural network to explore the parameter space and avoid premature convergence to the uniform distribution for the misclassified samples, which may be correctly classified in the future epochs.

V. Experiments

For the sake of commensurability, we evaluate our method following the experimental setup studied by Louizos et al.. We use the standard LeNet with ReLU non-linearities as the neural network architecture. All experiments are implemented in Tensorflow and the Adam optimizer has been used with default settings for training. In this section, we compared the following approaches: (a) L2 corresponds to the standard deterministic neural nets with softmax output and weight decay, (b) Dropout refers to the uncertainty estimation model, (c) Deep Ensemble (d) FFG refers to the Bayesian neural net with the additive parametrization. (e) MNFG refers to the structured variational inference method, (f) EDL is the method we propose. We tested these approaches in terms of prediction uncertainty on MNIST and CIFAR10 datasets.

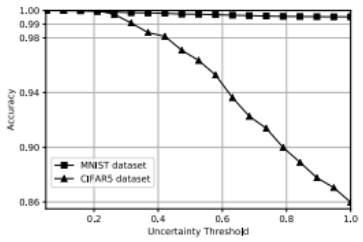

Figure 2: The change of accuracy with respect to the uncertainty threshold for *EDL*.

Method	MNIST	CIFAR 5
L2	99.4	76
Dropout	99.5	84
Deep Ensemble	99.3	79
FFGU	99.1	78
FFLU	99.1	77
MNFG	99.3	84
EDL	99.3	83

Table 1: Test accuracies (%) for MNIST and CIFAR5 datasets.

We also compare their performance using adversarial examples generated using the Fast Gradient Sign method.

Predictive Uncertainty Performance:

We employed the LeNet architecture to train models for MNIST, utilizing 20 and 50 filters of size 5 × 5 in the first and second convolutional layers, along with 500 hidden units in the fully connected layer. Comparable architectures were used for other methods, following the priors and posteriors. The classification performance on the MNIST test set is summarized in Table 1, demonstrating that our approach performs competitively with existing methods. It's important to note a potential misinterpretation in the table for our approach; predictions characterized by total uncertainty (u = 1.0), indicating the model's refusal to predict, are considered failures in overall accuracy calculation.

To address this, Figure 2 illustrates how test accuracy changes by varying the uncertainty threshold. Notably, the accuracy for predictions with uncertainty below a threshold approaches 1.0 as the threshold decreases.

Our approach directly quantifies uncertainty using Equation 1. However, for fairness in evaluation, we adopt the entropy metric, where uncertainty increases with the entropy of predicted probabilities. Equation 2 is used for class probabilities.

In the first set of evaluations, models are trained on the MNIST training split with the LeNet architecture and tested on the notMNIST dataset, containing letters instead of digits. The left panel of Figure 3 displays empirical cumulative distribution functions (CDFs) over the range of possible entropies [0, log(10)] for all models trained with the MNIST dataset. The curves closer to the bottom-right corner, indicating maximum entropy in all predictions, are considered desirable. Our model demonstrates significantly better uncertainty estimates compared to baseline methods.

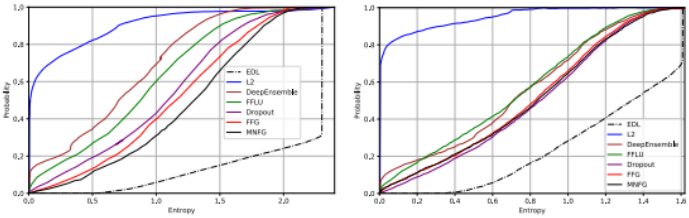

Figure 3: Empirical CDF for the entropy of the predictive distributions on the notMNIST dataset (left) and samples from the last five categories of CIFAR10 dataset (right).

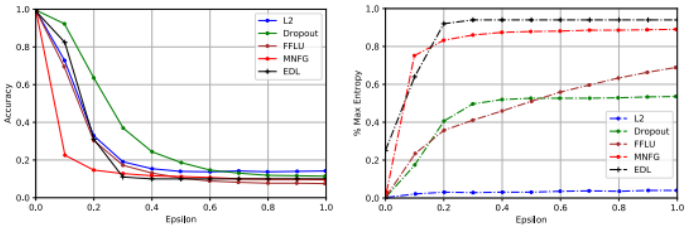

Figure 4: Accuracy and entropy as a function of the adversarial perturbation ϵ on the MNIST dataset.

We also examine the setup, using a subset of CIFAR10 classes for training and the rest for out-of-distribution uncertainty testing. We follow the authors, using the larger LeNet version

with 192 filters in each convolutional layer and 1000 hidden units for fully connected layers. Training involves samples from the first five CIFAR10 categories dog, frog, horse, ship, truck. Table 1 shows the accuracies of trained models on test samples from the same categories. Figure 2 highlights EDL providing more accurate predictions as prediction uncertainty decreases.

To assess prediction uncertainties, models are tested on samples from the last five categories of the CIFAR10 dataset airplane, automobile, bird, cat, deer. These samples are intentionally mispredicted, expecting high uncertainty. Results in the right panel of Figure 3 show that EDL associates significantly more uncertainty with its predictions compared to other methods.

Accuracy and Uncertainty on Adversarial Examples:

We conducted additional evaluations by subjecting our approach to adversarial examples. For each model trained in the previous experiments, we generated adversarial examples using the Fast Gradient Sign method from the Cleverhans adversarial machine learning library, varying the adversarial perturbation coefficient (ϵ). Adversarial examples are crafted using the weights of the models, and correct predictions become progressively challenging as the value of increases. These adversarial examples are then employed to test the trained models. Notably, the Deep Ensemble model is excluded from this set of experiments for fairness, given its training on adversarial examples.

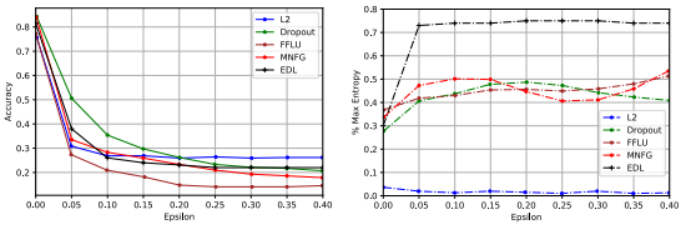

Figure 5: Accuracy and entropy as a function of the adversarial perturbation ϵ on CIFAR5 dataset.

Figure 4 presents the results for models trained on the MNIST dataset, displaying accuracies on the left panel and uncertainty estimations on the right. Uncertainty is quantified as the ratio of prediction entropy to the maximum entropy, denoted as $\%$ max entropy in the figure. It's important to mention that the maximum entropy is log(10) for the MNIST dataset. The figure highlights that Dropout achieves the highest accuracy for adversarial examples, as indicated on the left panel. However, it exhibits overconfidence in all predictions, evident from the right panel, where it places high confidence in incorrect predictions. Conversely, EDL strikes a good balance between prediction uncertainty and accuracy, associating very high uncertainty with incorrect predictions. The same experiment is performed on the CIFAR5 dataset, and Figure 5 illustrates the results. EDL consistently associates higher uncertainty with incorrect predictions, whereas other models tend to be overconfident in their wrong predictions.

VI. Related Work

The evolution of uncertainty-aware predictors aligns with the rise of contemporary Bayesian methodologies in machine learning. Gaussian Processes (GPs) represent a prominent avenue in this domain, exhibiting prowess in accurate predictions and reliable uncertainty quantification. Demonstrating effectiveness in various contexts, such as transfer learning and integration with deep learning, GPs have set benchmarks in active learning by providing closed-form calculations for prediction variances. Given their non-parametric nature, GPs lack deterministic or stochastic model parameters, presenting a significant advantage in universal prediction across diverse datasets.

Another trajectory in uncertainty modeling involves incorporating prior distributions on model parameters, particularly in parametric models. Bayesian Neural Networks (BNNs) fall into this category, introducing a prior on synaptic connection weights to account for parameter uncertainty. However, due to the intractability of calculating the posterior distribution on weights caused by non-linear activations, approximation techniques like Variational Bayes (VB) and Stochastic Gradient Hamiltonian Monte Carlo (SG-HMC) are actively researched for scalable BNN inference. Despite their powerful predictions, BNNs face challenges in obtaining closed-form posterior predictive distributions, relying on Monte Carlo integration and introducing noise to uncertainty estimates. In contrast, our approach sidesteps the inference of uncertainty sources and directly models a Dirichlet posterior, learning its hyperparameters through a deterministic neural network.

VII. Conclusion

This study introduces a predictive distribution for classification by employing a Dirichlet distribution on class probabilities, with neural network outputs assigned to its parameters. Fitting this predictive distribution to data involves minimizing the Bayes risk with respect to L2-Norm loss, regularized by an information-theoretic complexity term. The resulting predictor manifests as a Dirichlet distribution on class probabilities, offering a more nuanced uncertainty model than the standard softmax-output deep networks' point estimates. The interpretation of this predictor is framed within an evidential reasoning perspective, establishing a link from predictions to belief mass and uncertainty decomposition akin to subjective logic. Significantly advancing the state of the art, our predictor excels in two uncertainty modeling benchmarks: i) detection of out-of-distribution queries, and ii) robustness against adversarial perturbations.

References

[1] https://openaccess.thecvf.com/content/CVPR2023/papers/Tian_Modeling_the_Distributional_Uncertainty_for_Salient_Object_Detection_Models_CVPR_2023_paper.pdf
[2] https://arxiv.org/pdf/1806.01768.pdf
[3] https://arxiv.org/pdf/1910.02600.pdf
[4] https://browse.arxiv.org/pdf/2305.16703.pdf