Augmented Dirichlet Prior Networks for uncertainty estimation and out of distribution detection

Advait Savant as14634
Shreya Sinha ss14468
Saksham Garg sg6482

Abstract

Deterministic deep neural nets, generally trained on a maximum likelihood objective, have been efficient in modelling versatile distributions and have emerged as an established approach in various modeling tasks in the last decade. However, deep neural nets tend to have inaccurate confidence estimates and can make overconfident, uninformed predictions. This is especially the case when there can be high uncertainty associated with the concerned data. Establishing an estimation of uncertainty in standard deep neural networks has utility, from adversarial attacks to AI safety to out of distribution detection. A dirichlet prior network gives us a useful framework to quantify predictive uncertainty distinctly in terms of data uncertainty, distributional uncertainty and model uncertainty [8] [11]. The concentration parameters and the subsequent distribution over the simplex can help us separate uncertainty in prediction arising from intrinsically noisy/ edge case data or out of distribution data. [3] establishes a method of using a pre trained classifier network to develop dirichlet concentration parameters from classifier outputs. Improving on the results of [3] , we propose a method to augment a classifier network into a dirichlet prior network. We also propose a new loss for dirichlet prior networks, in consideration of the resulting classifier as an energy based model and the factorization characteristics of data in view of the likelihood ratio test [10]. Looking at the utility of incorporating image transformations in out of distribution detection, self supervised feature learning [1][2], we propose a feature learning approach. We establish a setting for, perform and evaluate experiments on uncertainty estimation and out of distribution detection with CIFAR-10 as in domain data and SVHN as out domain data.

1 Introduction

Consider a data generating process represented by a joint distribution $p(x, y)$ for an image classification task and the subsequent sampled dataset (x_i, y_i) where x_i is an input image and output $y_i \epsilon 1, .., k$ represents the corresponding class of the image. A neural network discriminatively models the conditional distribution of y given x based on a parameterization θ.

Contingent on the sampled dataset, for a given x*, the distribution over the outputs is given by,

$$p(y/x*, D) = \int p(y/x*; \theta) p(\theta/D) d\theta$$

The posterior distribution of over parameters computationally intractable. So is the subsequent integral over the parameters. In practice, we obtain a point estimate of the parameters using the maximum likelihood estimate,

$$\theta^* \simeq \max_{\theta} p(D/\theta)$$

The maximum likelihood formulation does not represent a degree of belief over the estimated parameters. This results in the inablity of the model to express a measure of confidence in its predictions, potentially leading to erroneous confidence estimates [11].

We analyze the posterior over the outputs in the first equation again, the posterior distribution over the parameters given the dataset can be substituted by a variational approximation q over theta, for computational tractability.

$$p(y/x*, D) = \int p(y/x*; \theta) q(\theta) d\theta$$

Monte carlo methods are used as follows. The integral is an expectation with respect to q, a distribution over θ. We approximate the expectation by sampling M arbitrary parameter values from q, then find the posterior over y given x and these parameters, and represent the posterior of y given x and D, which is aggregated over the parameters,as the mean of the posteriors given parameters.

$$p(y/x*, D) = \sum \frac{1}{M} p(y/x*; \tilde{\theta})$$

$$\tilde{\theta} \simeq q(\theta)$$

Consider the distribution $p(y_i/x_i, \theta_i)$ representing posterior probabilities. We look at each distribution $p(y_i/x_i, \theta_i)$, based on θ_i as a sample coming from μ, a distribution over the posterior distributions of y. This distribution over the posterior probabilities will directly depend on the distribution q over theta, since q generates theta samples, each of which correspond to a posterior distribution over y. We can see that a $q(\theta)$ induces an equivalent and implicit $u(\omega/x)$ where ω is $p(y/x, \theta)$.

If the samples theta coming from q are such that, for every sample, the resulting posteriors over y is similar and the ensemble of posteriors cluster around a certain cornet of the K class simplex governing our classifcation task, this would represent low predictive uncertainty. Similarly, if the samples theta coming from q are such that, the resulting posteriors over y as dissimilar and the ensemble of the posteriors scatter around the simplex, this can be viewed as high uncertainty.

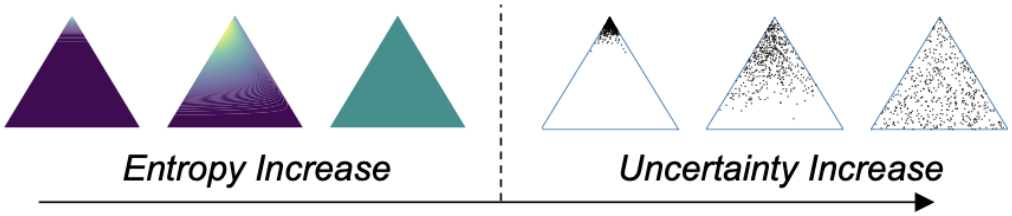

Figure 1: Illustration of Ensembles and induced implicit distribution on the simplex S_k.

Bayesian methods enable the attainment of a distribution over the parameters in some form. They can typically model a prior over the parameters which enables them to model a posterior over the parameters using the likelihood. Or, a distribution over q is obtained by variational approximation etc. This enables them to quantify belief over a set of parameters, have an ensemble of posteriors over y and quantify uncertainty. However, Bayesian networks are computationally expensive, it is difficult to induce and compute ahead from an appropriate prior over parameters, and, as explained in the next sub-section, Bayesian networks and other models are unable to segregate different types of uncertainty.

2 Background

Predictive uncertainty can consist of three aspects, model uncertainty, data uncertainty and distributional uncertainty. Model, or epistemic uncertainty, is the uncertainty in estimating the model parameters from the data. This can be reduced with knowlege. We can use principles to fit the data better, more data can address overfitting etc. Data, or aleatoric uncertainty, arises from the irreducible complexity and inherent randomness of the data generating process. This can be attributed data at the decision boundaries of an ideal model in the arbitrary width/depth limit, to noisy data etc. This can be a known unknown. The model can recognize the concerned data as edge case data while performing predictions. Distributional uncertainty arises from the change in the test and train distributions. This is an unknown unknown. The test data is coming from an out distribution which is different from the in distribution the model is trained on. The model is unfimiliar with the test data. In such a scenario, recognizing a test image as being out of the train distribution concerning the classifier task is especially hard, since, for an out of distribution image, $p_\theta(x)$ from our models' perspective is zero and hence the conditional distribution of a standard classifer is not defined, in practise, it does not have structured outputs.

$$p(y/x) = p(x, y)/p(x)$$

Consider the first equation again,

$$p(y/x*, D) = \int p(y/x*; \theta)p(\theta/D)d\theta$$

For a Bayesian network, $p(y/x*, D)$ is the predictive uncertainty. The $p(\theta/D)$ term represents model uncertainty, $p(y/x*; \theta)$ represents data uncertainty. [8] mentions without getting into detail that a Bayesian network conflates distributional uncertainty with model uncertainty. We interpret this as follows.

A bayesian network samples parameters and aggregates an ensemble to quantify uncertainty. Given some theta coming from $p(\theta/D)$, the uncertainty in prediction of y for some in domain x* stems from data uncertainty. Since the distribution on theta is itself based on the training dataset in concern, given theta, there is difficulty in interpreting $p(y/x*; \theta)$ as being associated with distributional uncertainty. There is not a principled logic to do the same. Any uncertainty as arsing from the distributional mismatch has to be interpreted as influencing, or being associated to, or having influence on, the model parameters based on the dataset, which can be resolved by developing a model with a rich enough dataset.

Following such reasoning, [8], propose to explicitly parameterize a distribution μ over predictive probabilities y as a Prior Network. The utility of doing so is as follows. For a Prior Network, in the case of a correct confident prediction, the distribution over distributions μ will yield a high values on the corner associated with the correct simplex class. In case of a data point with high data uncertainty, μ will yield a relatively sharp distribution at the centre of the simplex.Since this a known unknown. The model can express confidence in its beliefs, in this case, the lack there of, in discriminative prediction. In case of an out of distribution data point, the model will yield a flat distribution over the simplex. This is an unknown unknown.

(a) Confident Prediction (b) High data uncertainty (c) Out-of-distribution

Figure 2: Desired behaviors of a distribution over distributions.

Our expression for predictive uncertainty now becomes,

$$p(y/x*, D) = \int p(y/\mu)p(\mu/x*;\theta)p(\theta/D)d\theta$$

Here, $p(y/\mu)$ can represent the data uncertainty, $p(\mu/x*;\theta)$ can represent the distributional uncertainty and $p(\theta/D)$ can represent model uncertainty [8]. We get a principled, predictive framework which quantifies and delineates predictive uncertainty factors.

We can model such a distribution over the simplex using a dirichlet distribution.

$$Dir(\mu|\alpha) = \frac{\mathcal{T}(\alpha_0)}{\prod_{c=1}^{K}\mathcal{T}(\alpha_c)}\prod_{c=1}^{K}\mu_c^{\alpha_c-1}, \alpha_c > 0; \alpha_0 = \sum_{c=1}^{K}\alpha_c$$

For a given value of the α parameters, we will get a dirichlet distribution which takes in the values of y, which are probability values of the posterior distribution, and assigns to them a probability mass based on the dirichlet distribution expression, thus representing a distribution over the posterior distributions.

The concentration parameters α can be parameterized by a neural network and we make an make a network predict the alphas.[8]

$$p(\mu|x^*;\Theta) = Dir(\mu|\alpha)$$

$$\alpha = f(x*;\theta)$$

A flat dirichlet is used as a target distribution for out of distribution examples. A sharp dirichlet as the correct class is used as the target distribution for in distribution examples. The model is trained to minimize the KL divergence between the predicted distribution induced by predicted alphas and the target distribution as described.

3 Augmented Dirichlet Prior Network

Consider a dirichlet prior network. The posterior over the classes is given by.

$$p(y/x*, \mu) = \int p(y/\mu)p(\mu/x*;\theta)d\mu$$

From properties of the dirichlet distribution, this comes out to be, For a particular class c, the probability values y_c come out to be.

$$y_c = \frac{\alpha_c}{\alpha_0}$$

where,

$$\alpha_0 = \sum_{j=1}^{K} \alpha_j$$

Consider the predicted output of a standard classifier network,

$$y_c = \frac{\exp(z_c(x_c))}{\sum_{j=1}^{K} \exp(z_c(x_j))}$$

where z_c is the last layer output of the neural network. Looking at these similarities, [3][8] note that the standard deep neural network can be viewed as predicting a categorical distribution with dirichlet priors. From [3] , if we were to use a maximum likelihood cross entropy loss training for a dirichlet prior network, the expression is in form similar to that for a standard neural network. If we interpret the output of a standard neural network as a dirichlet prior network, the caveat is that, now the predicted mean y_c is insensitive to arbitrary scaling of α_c. Here, the precision parameter α_0 which controls the sharpness of the distribution is degenerate. We will consider this later in our loss. [pert 2] interprets the exponential output of the classfier network as Dirichlet parameters for a degenerate prior network with a scale constant.

Building on the degenerate dirichlet prior network, [3], interprets the output of a classifier network as dirichlet parameters. In order to enhance robustness and performance on uncertainty measures, it is then proposed to perturb the dirichlet parameters based on a parameterized perturbation function $\epsilon(\alpha; \beta)$ which is given by a linear transformation $W\alpha$ parameterized by W. The subsequent predicted concentration parameters are given by $\alpha+\epsilon(\alpha; \beta)$. The loss function then parameterized by W consists of minimizing the entropy of the dirichlet distribution for in domain data points and maximizing the entropy for dirichlet distribution for out domain data points. While [8] trains an dirichlet prior network from end to end, [3] builds up on a pre-trained classifier network in order to simulate a dirichlet prior network.

The utility of such an approach lies in the fact that, akin to training an effective classifier network, a high computational capacity is required to train an effective dirichlet prior network from scratch. There are many more trained classifier networks on a certain industrial task as are dirichlet prior networks. There is utility in enabling uncertainty estimation building over these networks. Such an approach falls under a transfer learning setting, where representations derived from training a model on one task, are utilized for another downstream task under certain considerations on the alignment of tasks.

As for [3], we observe that care must be taken regarding the factors of variation in domain outputs and the extent to which they differ from out domain outputs. We know that convolutional neural networks have useful high level feature learning capabilities for image data such as representations having covariance to transformations of the data. It would be, to some extent, reasonable to expect the in domain output characteristics of a classifier to be significantly different from an out domain data output characteristics, given that the classifier itself is a baseline [7] for out of distribution tasks. In practise, this is true to a only to a certain extent, for reasons based on our discussions regarding confidence estimates of classifier networks. In view of [3], for the task of estimation of concentration parameters, we would consider the classifier output as a signal, albeit not a strong one. One criticism of [3] would be, for example, if certain in domain and out domain data points give very similar classifier outputs. The concentration parameter training would proceed in a way that two very similar inputs are mapped to two different outputs from the same parameterization. On one hand, this could enhance the separability, but on the other hand, this could also be a matter of sub-optimal training procedure.

We would hypothesize that the task of training for concentration parameters based on just classifier outputs, while having with empirical backing, can be difficult and can be enhanced by virtue of attempting to use certain data characteristics based on the classifier. We propose the incorporation of data descriptors as features derived from the classifier network as a supplementary signal to the

classifier outputs for the task of estimation of concentration parameters. As for the aligning nature of the downstream task, we observe that the target dirichlet parameters for a certain class would have a high magnitude at that class and flat elsewhere, the target classifier output would be one in the concerned class and zero elsewhere. Akin to [8][3], we notice similarity in form. For a trained classifier network, the classification task is linear in the feature space represented by the penultimate layer outputs of a classifier. A neural network can be considered as developing a feature space for the classification task. We evaluate the utility of such features for our task and see them as representing the input data. We note that using linear classifier probes could be useful. We also attempt to use self supervised learning models designed for feature representation and downstream tasks [2]. We propose a feature representation method based in consideration of out domain data. We replicate the results of [3] and we show improvement over them.

4 Proposed Loss

[5] work on re interpreting the output of a classifier network to represent join probabilities without changing the function.

$$p(y_i/x) = \frac{\exp(z_i(x))}{\sum_{j=1}^{K} \exp(z_j(x))}$$

For a classifier network, $z_i(x)$ is the corresponding logit.

$$p(x, y_i) = \frac{\exp\left(z_i(x)\right)}{Z(\theta)}$$

$Z(\theta)$ is the normalizing constant.

Marginalizing over y, they get,

$$p(x) = \sum_{i=1}^{K} p(x, y_i) = \frac{\sum_{i=1}^{K} \exp(z_i(x))}{Z(\theta)}$$

For a dirichlet prior network,

$$p(y_i/x) = \frac{f_\alpha(x; y_i)}{\sum_{j=1}^{K} f_\alpha(x; y_j)}$$

$$p(y_i/x) = \frac{f_\alpha(x; y_i)}{\alpha_0(x)}$$

Consider interpreting the next equation as a joint distribution, we can see the marginalization will be wrt all x and all y to obtain $Z(\theta)$

$$p(y_i, x) = \frac{f_\alpha(x; y_i)}{Z(\theta)}$$

Now,

$$p(y_i/x) = p(y, x)/p(x) = \frac{p(y, x)}{\sum_{j=1}^{K} p(x, y_j)}$$

In our system, the denominator will yield $\alpha_0(x)$ and the subsequent predictive distribution as a result of this interpretation will be the correct predictive distribution. Thus, akin to similarity in form with [5], we can consider our interpretation to be valid as it is functionally appropriate and yields the correct predictive. This is analogous to the step taken to go from the first to the second equation in this section. For our dirichlet prior network,

$$p_\theta(x) = \frac{\sum_{j=1}^{K} \exp(z_j(x))}{Z(\theta)}$$

[10] develop a factorization for the distribution of a model over image data, justify it empirically and establish a likelihood ratio test. In a classification task, every input data image can be considered to have two components. A semantic component X_S characterized by the distinct features of the in domain data and a background component x_B consisting of background statistics. The factorization can be done as follows,

$$p_\theta(x) = p_\theta(x_B)p_\theta(x_S)$$

For an in domain data, the semantic likelihood will be high, the semantic likelihood would be low for out domain data. The background likelihood can be considered equivalent. With respect to an auxilary model for out of distribution data, use the log of the likelihood ratios as a measure of determination for in domain characteristics of a certain data point. Assume for now that there exists such a model, in practise, the auxilary model can be trained by perturbing the inputs like fgsm [4]. For our approach, as we elaborate, we could work in terms of the model parameters itself. For a given data point and a given auxilary model, the model increases $p_\theta(x)$ relative to the auxilary model. This can be incorporated as an auxilary loss function.

Also, for a single given model, upto a scale constant, following our $p_\theta(x)$ interpretation, we can maximize the in domain probability relative to the out domain probability. Each obtained by a sum over the logits. Notice that this takes care of, attempts to solve, the degraded precision problem in degenerate prior networks since the sum over the logits is the precision parameter.

We denote our model as $p_\theta(x)$ and and auxilary model which just captures background statistics as $p_{\theta^0}(x)$.

We see how we can factorize the input image data x as a background component x_B and a specific component x_S. We see how we can interpret the output of a classifier neural network in terms of joint probabilities, and subsequently model the probability of an input image. We denote our model as $p_\theta(x)$ and and auxilary model which just captures background statistics as $p_{\theta^0}(x)$.

We see how we can factorize the input image data x as a background component x_B and a specific component x_S.

For our in domain image, if we take the ratios of the likelihood,

$$LLR(x^{in}) = \log \frac{p_\theta(x^{in})}{p_{\theta_0}(x^{in})}$$

$$LLR(x^{in}) = \log \frac{p_\theta(x_B^{in}) * p_\theta(x_S^{in})}{p_{\theta_0}(x_B^{in}) * p_{\theta_0}(x_S^{in})}$$

$$LLR(x^{in}) = \log p_\theta(x_S^{in}) - \log p_{\theta_0}(x_S^{in})$$

We see that relative to the auxilary background model, the value of the specific component of our image in our model should be high. $LLR(x^{in})$ should be high.

For our out domain image, if we take the ratios of the likelihood,

$$LLR(x^{out}) = \log \frac{p_\theta(x^{out})}{p_{\theta_0}(x^{out})}$$

$$LLR(x^{out}) = \log \frac{p_\theta(x_B^{out}) * p_\theta(x_S^{out})}{p_{\theta_0}(x_B^{out}) * p_{\theta_0}(x_S^{out})}$$

$$LLR(x^{out}) = \log p_\theta(x_S^{out}) - \log p_{\theta_0}(x_S^{out})$$

The specific component of the out domain images in our model shouldn't be any higher than the value of the background model. $LLR(x^{out})$ should be low.

If we were to subtract, the difference ought to be high,

$$LLR_{difference} = LLR(x^{in}) - LLR(x^{out})$$

As an auxilary cost, we would want to maximize this quantity in our set up,

$$LLR_{difference} = [\log p_\theta(x_S^{in}) - \log p_{\theta_0}(x_S^{in})] - [p_\theta(x_S^{out}) - \log p_{\theta_0}(x_S^{out})]$$

Rearranging,

$$LLR_{difference} = [\log p_\theta(x_S^{in}) - \log p_\theta(x_S^{out})] - [p_{\theta_0}(x_S^{in}) - \log p_{\theta_0}(x_S^{out})]$$

If we focus on the first term, we see that it is terms of our model parameters theta. The second term is not dependent on our model parameters. We can choose to maximize the first term in our model. Therefore, our auxiliary loss becomes,

$$LLR_{loss} = \log p_\theta(x_S^{out}) - \log p_\theta(x_S^{in})$$

As a caveat, we do not want to increase or decrease the logits in an unrestrained manner, so we use the sigmoid of the sum. We are motivated in this loss by [9] who has a well defined loss function, developed from KL divergence considerations, which consists of maximizing the sum of the logits for in domain examples and minimizing the sum of logits for out domain examples aimed at improving performance.

$$\mathcal{L}_{dirichlet} \;=\; \mathbb{E}_{p_{in}(x)}\left[KL[Dir(\mu|\hat{\alpha}) \parallel\; p(\mu|x;\theta)]\right] \;+\; \mathbb{E}_{p_{out}(x)}\left[KL[Dir(\mu|\hat{\alpha}) \parallel\; p(\mu|x;\theta)]\right]$$

$$\mathcal{L}_{auxilary} \;=\; \mathbb{E}_{p_{out}(x)}\left[\sum_{i=1}^{K}\sigma(z_i(x))\right] \;-\; \mathbb{E}_{p_{in}(x)}\left[\sum_{i=1}^{K}\sigma(z_i(x))\right]$$

$$\mathcal{L}_r \;=\; \mathcal{L}_{dirichlet} \;+\; \lambda * \mathcal{L}_{auxilary}$$

5 Feature representation and learning

The classifier response to geometric transformations of images can be used for effectively predicting out of distribution data [1]. Methods such as [2] develop feature representations based on learning invariance to geometric transformations for in domain data. Developing from a pre trained simclr, we can add an auxilary loss to the simclr training objective making the model preferentially learn invariance to geometric transformations for in data as compared to learning invariance to geometric transformations for out data. To the extent that it learns invariance to transformations for out data, which is plausible given background characteristics in the image data, the feature representation should be more receptive of knowledge of transformations on in data. We train an augmented simclr as follows.

$$loss = -sim(z_i, z_j) + \lambda log(1 + sim(v_i, v_j) - sim(z_i, z_j))$$

6 Experiment

On the basis of data, we have divided the analysis into two parts. In one, we use CIFAR10 as our in-distribution data and SVHN as our out-distribution data. In the next part, we use CIFAR10 as our in-distribution data and adverserial examples generated by fast gradient sign method (FGSM) [4]. The architecture used by us can be seen in figure[3].

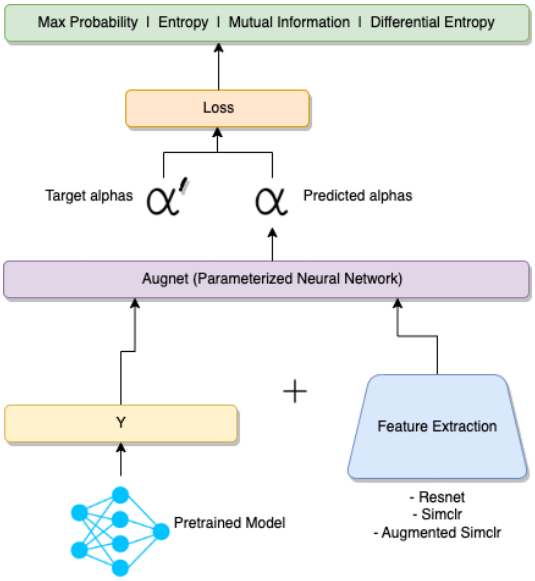

Figure 3: Our Architecture.

We choose our pre-trained model as ResNet18 which we are using to calculate the probabilities of each of the class given by $\mathcal{Y}$. We use three models for feature extraction. 1) We use the features from the penultimate layer of our pre-trained ResNet. 2) We train a Simclr[[2]] model on CIFAR10 for 1200 epochs and use it to extract the features. 3) We train an augmented Simclr which is trained on a combination of in-distribution and out-distribution data with respective temperature scaling. We use these three models to extract the features separately and train them using two fully connected layers, after which we concatenate the predicted outputs from ResNet18 $\mathcal{Y}$ with these extracted features and train them for another few fully connected layers. We call this complete model 'Augnet'. We also try to use temperature scaling of the predicted outputs or $\mathcal{Y}$ using the targets, as an additional parameter to give more weightage to the in-distribution data as compared to out-distribution data during the training of our Augnet. We use the predicted alphas and our novel loss function described above.

To compare with the baseline, we replicate DPN from [8] on ResNet18. We also replicate the work done in [3] with CIIFAR10 and FGSM data.

In practice, for parallel analysis, we analyzed the statistics of differences for the maximum and the second maximum output for Resnet18, for in and out domain data. The in-domain mean difference was greater than the out-domain mean difference, the magnitude of their difference was cogent to some extent but cogent enough to get the high 90s in auroc metrics. We also observed that this magnitude increased a lot for Resnet34, commenting on the role of classifier characteristics for our task.

7 Results and evaluation

We use the same metrics as described in [8] for evaluating our results. The first measure is the probability of the predicted class or max probability which is a measure of confidence in the

prediction. It is given by,

$$\mathcal{P} = max_c P(\omega_c|x^*; \mathcal{D})$$

The second measure is the entropy of the predictive distribution. It behaves similarly to max probability but represents the uncertainty encapsulated in the entire distribution. It is given by,

$$\mathcal{H}[P(y|x; \mathcal{D})] = -\sum_{c=1}^{K} P(\omega_c|x^*; \mathcal{D}) \log P(\omega_c|x^*; \mathcal{D})$$

Max probability and entropy of the expected distribution can be seen as measures of the total uncertainty in the prediction.

The third measure, Mutual Information (MI) between the categorical label y and the parameters of the model is a measure of the spread of an ensemble $\{P(\omega_c|x^*; \mathcal{D})\}_{i=1}^{M}$ which assess uncertainty in predictions due to model uncertainty. It captures elements of distributional uncertainty. It is expressed as the difference between the total uncertainty, given by the entropy of expected distribution, and the expected data uncertainty, which is given by the expected entropy of each member of the ensemble. The last measure that we use here is differential entropy. This measure is maximized when all categorical distributions are equiprobable, which occurs when the Dirichlet Distribution is flat. Differential entropy is well suited to measuring distributional uncertainty, as it can be low even if the expected categorical under the Dirichlet prior has high entropy, and also captures elements of data uncertainty.

$$\mathcal{H}[P(y|x; \mathcal{D})] = -\int_{S^{K-1}} p(\mu|x^*; \mathcal{D}) \log P(\omega_c|x^*; \mathcal{D}) du$$

We report the AUROC and AUPR for all the above-mentioned metrics for misclassification and out-of-distribution detection.
We get the best results with the features extracted from our augmented SimClr Model with FGSM generated data. It also gives better results for SVHN than the models using general SimClr and Resnet18 features. These results can be seen in figure 7.
For Simclr, our model gives better results for MISC than OOD. Temperature scaling helps both OOD and MISC and improves the result. For FGSM, temperature scaling gives equivalent results as without it. FGSM improves the results when there is no temperature scaling. Comparing Simclr on the loss used by [3], our loss gives comparable results for MISC and slightly better results for OOD. These results can be seen in figures 6 and 5.
When the features are extracted from the penultimate layer of Resnet18, FGSM increases results to some extent as compared to SVHN. We tried different model architectures of Augnet here and found that the increase is completely dependent on the size of the model. Temperature scaling helps with OOD but gives almost equivalent results for MISC. We also compared results on the loss used by [3], our loss gives much better results for OOD and improved results for MISC, as compared to the results for SimClr. The results are reported below:

		out_fgsm			
		Max.P	Ent	MI	D.Ent
miss classification	AUROC	0.82	0.82	0.82	0.82
	AUPR	0.965	0.965	0.965	0.965
Out of distribution	AUROC	0.807	0.807	0.807	0.807
	AUPR	0.85	0.85	0.85	0.85

		out_svhn			
		Max.P	Ent	MI	D.Ent
miss classification	AUROC	0.77	0.77	0.77	0.77
	AUPR	0.959	0.959	0.959	0.959
Out of distribution	AUROC	0.73	0.731	0.731	0.731
	AUPR	0.81	0.81	0.81	0.81

Figure 4: augmented simclr results

		out_fgsm with temperature scaling			
		Max.P	Ent	MI	D.Ent
misc	AUROC	0.75	0.75	0.75	0.75
	AUPR	0.95	0.95	0.95	0.95
Out of distribution	AUROC	0.7	0.7	0.7	0.7
	AUPR	0.79	0.79	0.79	0.79

		out_svhn with temperature scaling			
		Max.P	Ent	MI	D.Ent
misc	AUROC	0.75	0.75	0.75	0.75
	AUPR	0.957	0.957	0.957	0.957
Out of distribution	AUROC	0.689	0.689	0.689	0.689
	AUPR	0.8	0.8	0.8	0.8

Figure 5: simclr results with temperature scaling

		out_fgsm			
		Max.P	Ent	MI	D.Ent
miss classification	AUROC	0.75	0.75	0.75	0.78
	AUPR	0.957	0.957	0.957	0.957
Out of distribution	AUROC	0.67	0.67	0.67	0.67
	AUPR	0.743	0.743	0.743	0.743

		out_svhn			
		Max.P	Ent	MI	D.Ent
miss classification	AUROC	0.7	0.7	0.7	0.7
	AUPR	0.955	0.955	0.955	0.955
Out of distribution	AUROC	0.85	0.68	0.62	0.68
	AUPR	0.75	0.73	0.752	0.8

Figure 6: simclr results without temperature scaling

		out_fgsm with temperature scaling			
		Max.P	Ent	MI	D.Ent
misc	AUROC	0.7	0.7	0.67	0.71
	AUPR	0.89	0.91	0.88	0.89
ood	AUROC	0.77	0.74	0.72	0.68
	AUPR	0.75	0.75	0.71	0.67

		out_svhn with temperature scaling			
		Max.P	Ent	MI	D.Ent
misc	AUROC	0.8	0.77	0.78	0.78
	AUPR	0.91	0.9	0.92	0.91
ood	AUROC	0.87	0.83	0.85	0.77
	AUPR	0.76	0.7	0.73	0.67

Figure 7: resnet results with temperature scaling

		out_fgsm			
		Max.P	Ent	MI	D.Ent
misc	AUROC	0.72	0.7	0.67	0.71
	AUPR	0.92	0.89	0.9	0.9
ood	AUROC	0.65	0.66	0.53	0.66
	AUPR	0.6	0.64	0.67	0.65

		out_svhn			
		Max.P	Ent	MI	D.Ent
misc	AUROC	0.72	0.71	0.73	0.71
	AUPR	0.9	0.9	0.9	0.89
ood	AUROC	0.63	0.66	0.63	0.67
	AUPR	0.61	0.63	0.6	0.64

Figure 8: resnet results without temperature scaling

8 Future Work

The task to obtain robust features- characteristic of in domain data for the task of uncertainty estimation in a framework such as a dirichlet prior network is a potential area of inquiry. Establishing the utility of such features in consideration of their statistics on out data relative to in data could be a pursuit. Dirichlet posterior networks are a new line of research in uncertainty estimation. Effective out of distribution generation in the pipeline for uncertainty estimation, out of distribution detection tasks is a line of inquiry. In this project we attempted to develop a coherent method to enable uncertainty estimation over an image classifier network. We look at the nature of the set up, expanded on prior empirical work in uncertainty estimation with dirichlet prior networks, also expanding on other work, we attempted to develop effective loss, feature formulations and evaluated the system on various metrics.

References

[1] Yuval Bahat and Gregory Shakhnarovich. “Confidence from invariance to image transformations”. In: *arXiv preprint arXiv:1804.00657* (2018).

[2] Ting Chen et al. “A simple framework for contrastive learning of visual representations”. In: *International conference on machine learning*. PMLR. 2020, pp. 1597–1607.

[3] Wenhu Chen et al. “Enhancing the robustness of prior network in out-of-distribution detection”. In: *arXiv: 1811.07308* (2018).

[4] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. “Explaining and harnessing adversarial examples”. In: *arXiv preprint arXiv:1412.6572* (2014).

[5] Will Grathwohl et al. “Your classifier is secretly an energy based model and you should treat it like one”. In: *arXiv preprint arXiv:1912.03263* (2019).

[6] Shiyu Liang, Yixuan Li, and Rayadurgam Srikant. “Enhancing the reliability of out-of-distribution image detection in neural networks”. In: *arXiv preprint arXiv:1706.02690* (2017).

[7] Shiyu Liang, Yixuan Li, and Rayadurgam Srikant. “Enhancing the reliability of out-of-distribution image detection in neural networks”. In: *arXiv preprint arXiv:1706.02690* (2017).

[8] Andrey Malinin and Mark Gales. “Predictive uncertainty estimation via prior networks”. In: *Advances in neural information processing systems* 31 (2018).

[9] Jay Nandy, Wynne Hsu, and Mong Li Lee. “Towards maximizing the representation gap between in-domain & out-of-distribution examples”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 9239–9250.

[10] Jie Ren et al. “Likelihood ratios for out-of-distribution detection”. In: *Advances in neural information processing systems* 32 (2019).

[11] Murat Sensoy, Lance Kaplan, and Melih Kandemir. “Evidential deep learning to quantify classification uncertainty”. In: *Advances in neural information processing systems* 31 (2018).

[12] Murat Sensoy et al. “Uncertainty-aware deep classifiers using generative models”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 04. 2020, pp. 5620–5627.

[13] Apoorv Vyas et al. “Out-of-distribution detection using an ensemble of self supervised leave-out classifiers”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 550–564.

Link to the Github Repository

`https://github.com/sakshamgarg/Augmenting-Dirichlet-Network`