Open World, Uncertainty, Robustness - Open Set Classification

Antonio Carta

University of Pisa

Introduction

Outline

OOD Detection:

- Softmax thresholding
- ODIN
- Ensembles

Open World:

- Learning in an open world
- Explicit models for background classes
- Open Set Recognition: Mahalanobis distance and OpenMax

Motivation

- In practical problems, we don't always have a training data that covers the entire test distribution
- We assume the model is unreliable for out-of-distribution (OOD) samples
- We would like the model to be able to estimate the uncertainty (of the data and its predictions) correctly
- We may need to identify and reject unknown inputs

Motivation

- Uncertainty quantification is a fundamental issue for many applications, such as safety-critical domains
  - A self-driving car should (safely) stop in uncertain conditions and give control to the passenger
  - Medical systems should give uncertainty estimate so that an expert can determine the correct course of action
- Open issue even with SotA models. ChatGPT is often extremely confident, regardless of the actual correctness of the output

Open Set Recognition

- **closed world**: every ML problem you know up to now
- **open world**: when unknown data is available at test time (anomalies, background, novel classes)

OWL Challenges

- What happens when the model encounters unseen classes during testing?
- What happens if the model doesn't even know how the unseen classes look like?

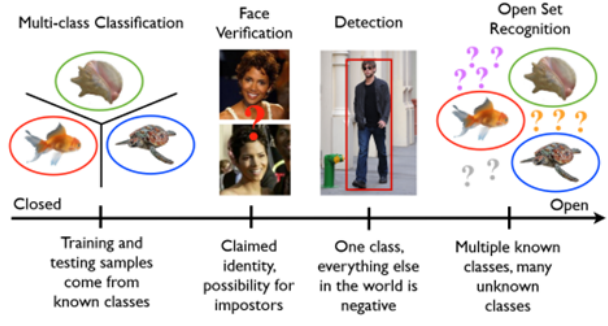


Fig. 1. Vision problems arranged in order of "openness". For some problems, we do not have knowledge of the entire set of possible classes during training, and must account for unknowns during testing. In this article, we develop a deeper understanding of those open cases.

Figure 1: W. J. Scheirer et al. "Towards Open Set Recognition." TPAMI, 2012.

Open World Learning

Knowing in an Open World

- **Known**: in-distribution samples and predicted correctly with high confidence (correct redictions)
- **Known Unknowns**: low-confidence samples, such as (low confidence) successfull recognized anomalies
- **Unknown Unknowns**: out-of-distribution samples with highly confident predictions (the model shouldn't have high confidence here)

⚠ Unknown Unknowns

- If the test distribution drifts, the model may see something completely new.
- The model doesn't know what it doesn't know.
- How do we identify unknown unknowns?

Open World Assumption

- **Closed world assumption**: the model «knows what it needs to know» such as which classes are present in the data
- **Open world assumption**: the model may encounter new data at test time

i Open Set Recognition

- **Open World**: at test time the model may see data from novel classes that were unknown at training time
- We don't expect the model to generalize to unseen classes
- It is difficult to train the model to recognize novel classes as unknown samples

OWA in Formal Logic

An example in formal logic systems in classic AI:

- Closed-world means that any true statement is known to be true
- Open-world means that a statement may be true but yet unknown
- The logic needs to be adapted to deal with unknown truth values
- Probably the first example of open world assumption in AI and something you may have seen in previous courses (AIF)

OWL Approaches

- **Anomalies**: detect out-of-distribution samples via OOD detection methods
- **Prior Knowledge**: train the model to recognize a «background» class
- **Open Set Recognition**: model the space of «things you know» and never predict outside of it

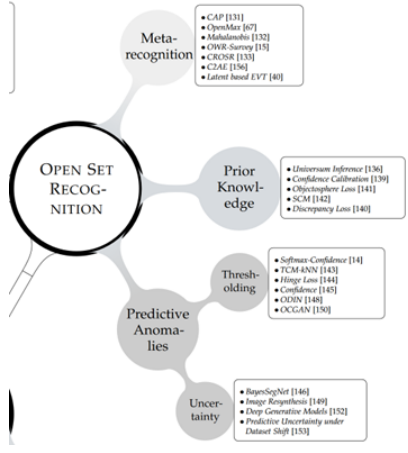


Figure 2: M. Mundt et al. "A Wholistic View of Continual Learning with Deep Neural Networks: Forgotten Lessons and the Bridge to Active and Open World Learning."

Statistical Dispersion

We want to model expected and unexptected outputs

- In statistics, dispersion is a property of a distribution measuring how «stretched» it is
- Some measures: variance, standard deviation, IQR
- Example: in a unimodal distribution we can use the mean and variance to identify uncertain samples

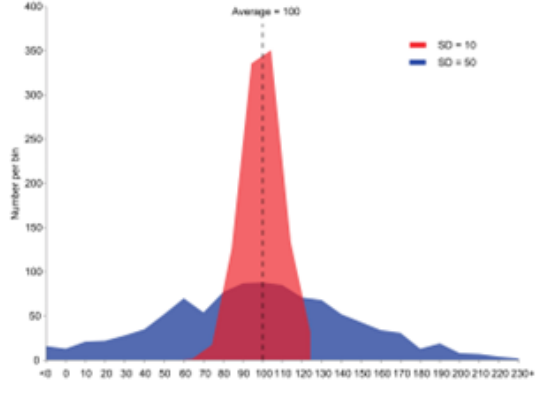


Figure 3: image from https://en.wikipedia.org/wiki/Statistical_dispersion

Aleatoric vs Epistemic Uncertainty

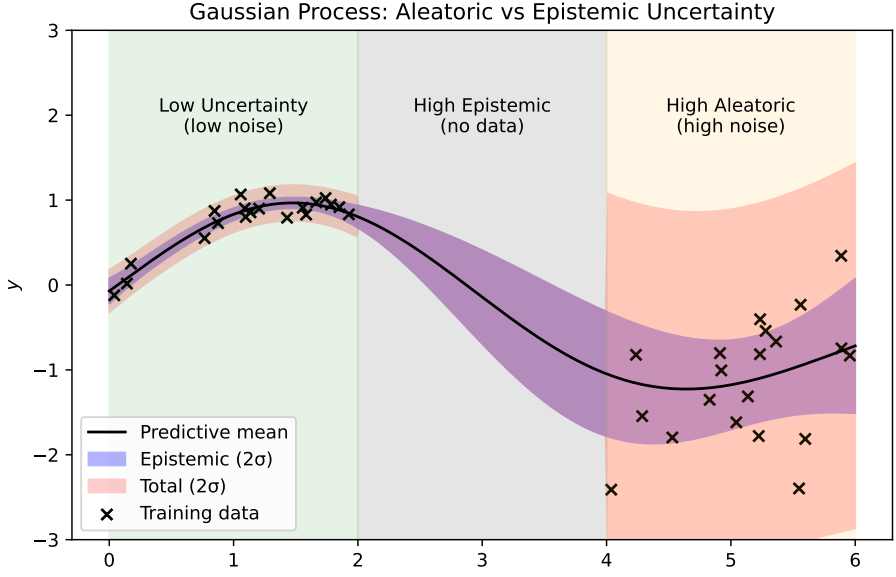
Aleatoric Uncertainty (Data)

- Variance of the *true data generating process* $p(y \mid x, \theta)$
- Noise in the measurements or variance of the data
- Irreducible

Epistemic Uncertainty (Model)

- Variance of the model predictive distribution $p(y \mid x, D)$
  - Notice that in a bayesian framework this quantity depends on the model via the integral $p(y \mid x, D) = \int p(y \mid x, \theta) p(\theta \mid D) d\theta$
- Measures uncertainty in the model outputs
- We can reduce it with more data

Example


Gaussian Process: Aleatoric vs Epistemic Uncertainty
Low Uncertainty
(low noise)
High Epistemic
(no data)
High Aleatoric
(high noise)
Predictive mean
Epistemic (2σ)
Total (2σ)
Training data
y
3
2
1
0
−1
−2
−3
0
1
2
3
4
5
6

Out-of-Distribution Detection

OOD Detection

- We model OOD detection as a binary classification problem
- predict in-distribution vs out-of-distribution
- highly imbalanced
- different types of errors have different cost

Example - Spam Classification

- Highly imbalanced
- putting a normal email in the spam folder is much worse than the opposite error

Imbalanced Binary Classification

- in imbalanced binary classification problems, the accuracy is not a good metric
- if we have 99% positives, a naive "always positive" classifier has a 99% accuracy
- for probabilistic classifiers, classification metrics depend on the threshold that we use on the probability.
  - $\hat{y} = \begin{cases} 1 & \text{if } \hat{p}(y = 1 \mid \mathbf{x}) \geq \tau \\ 0 & \text{otherwise} \end{cases}$
- $\tau = 0.5$ is the obvious choice, but we can tweak it to control the type of errors (false positives vs false negatives)
- We can select the threshold on a separate validation set by maximizing some metric

Precision and Recall

Alternative metrics:

Precision: how may positive classifications are actually correct

$$\text{Precision} = \frac{\text{Relevant retrieved instances}}{\text{All retrieved instances}}$$

Recall: how many true positive are correctly classified

$$\text{Recall} = \frac{\text{Relevant retrieved instances}}{\text{All relevant instances}}$$

F-score: harmonic mean of precision and recall

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

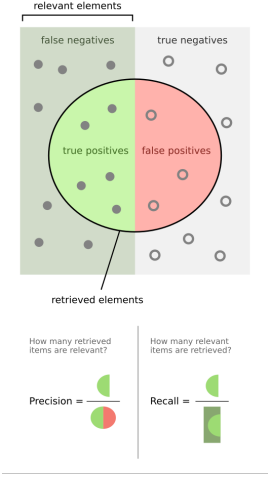


Figure 4: source: wikimedia

Precision-Recall Curve

- We can plot the precision-recall curve of a classifier
- shows the behavior along all the possible thresholds
- the Area Under the Precision-Recall curve (AUPR) is often used as a metric

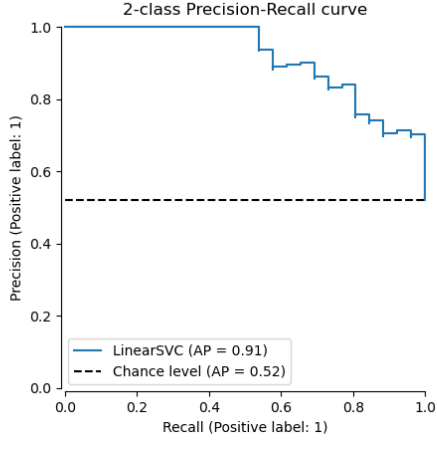


Figure 5: source: scikit-learn docs

How to Select a Classification Threshold

- on a separate validation set
- Option 1: optimize the AUPR
- Option 2: set desired minimal recall/precision values and find the best threshold that satisties it
- Option 3: if you have a cost for positive/negative errors, you can find the threshold that minimizes the cost (a.k.a. decision theory)

OOD Evaluation

- false positive (fp): a negative predicted as positive
- true positive (tp): a positive predicted as positive
- true positive rate (tpr): tpr = tp / (tp + fn)
- Area Under Precision-Recall Curve (AUPR)

OOD Detection Methods

- softmax thresholding
- ODIN

Softmax Thresholding

Observation

Correctly classified examples tend to have greater maximum softmax probabilities than erroneously classified and out-of-distribution examples, allowing for their detection (Hendrycks, 2017)[1]

Maximum Softmax Score:

- softmax probabilities: $P(y = k \mid x) = \frac{\exp(f_k(x))}{\sum_{j=1}^{K} \exp(f_j(x))}$
- max softmax score: $S(x) = \max_{k \in \{1,\ldots,K\}} P(y = k \mid x)$
- In-Distribution $(x) = \begin{cases} 1, & \text{if } S(x) \geq \tau \\ 0, & \text{if } S(x) < \tau \end{cases}$

[1] A. R. Dhamija et al. 2018. "Reducing Network Agnostophobia." NIPS

Softmax Thresholding - Finding τ

- τ is chosen using a validation set
- maximize AUPR or binary classification metrics (e.g. FPR@95%TPR)

How to compute it:

- order data by max softmax probability $S(x)$
- for each τ split the ordered data into ID/OOD and compute the recall

ODIN

Observation

temperature scaling and small perturbations to the input can separate the softmax score distributions between in- and out-of-distribution images. Improves (Hendrycks, 2017)[2]

- If the input is in a "flat region" of the input space a small perturbation shouldn't decrease the probability too much
- We expect ID samples to be in a flat region because small perturbations should not change the prediction (i.e. a model should be robust to augmentations for ID data)
- We expect OOD samples to be more sensitive to small perturbations (i.e. a model may be less robust to augmentations for OOD data)
- ADVANTAGES: simple post-training method and it can be added to any pretrained supervised model

[2] A. R. Dhamija et al. 2018. "Reducing Network Agnostophobia." NIPS

ODIN - Input Perturbation

Input perturbation:

$$\tilde{\mathbf{x}} = \mathbf{x} - \varepsilon \operatorname{sign}\left(-\nabla_{\mathbf{x}} \log S_{\hat{y}}(\mathbf{x}; T)\right)$$

- Make a small step away from the high probability inputs
- note the gradient is w.r.t. the input
- the perturbed image has a slightly lower likelihood because we are moving in a the direction that maximally decreases it (i.e. the gradient direction)
- we are assuming that OOD inputs to have larger gradients

ODIN - Detector

- **Temperature scaling**: $S_i(\mathbf{x}; T) = \frac{\exp(f_i(x)/T)}{\sum_{j=1}^{N} \exp(f_j(x)/T)}$
  - Make the distribution more or less peaked to control the True Positive Rate
  - Temperature scales changes the softmax output distribution towards more uniform (high T) or more peaked (low T) distributions
- **Softmax score**: $S_{\hat{y}}(x; T) = \max_i S_i(x; T)$
  - Select the max probability score
- **Detector**: $g(x; \delta, T, \varepsilon) = \begin{cases} 1 & \text{if } S_{\hat{y}}(\tilde{x}; T) \leq \delta \\ 0 & \text{if } S_{\hat{y}}(\tilde{x}; T) > \delta \end{cases}$
- Use a threshold on the score to discriminate ID/OOD samples

ODIN - Full Method

- Input perturbation: $\tilde{x} = x - \varepsilon \operatorname{sign}\left(-\nabla_x \log S_{\hat{y}}(x;T)\right)$
- Temperature scaling: $S_i(x;T) = \frac{\exp(f_i(x)/T)}{\sum_{j=1}^{N} \exp(f_j(x)/T)}$
- Softmax score: $S_{\hat{y}}(x;T) = \max_i S_i(x;T)$
- Detector: $g(x;\delta,T,\varepsilon) = \begin{cases} 1 & \text{if } S_{\hat{y}}(\tilde{x};T) \leq \delta \\ 0 & \text{if } S_{\hat{y}}(\tilde{x};T) > \delta \end{cases}$
- T, δ selected via model selected to achieve a desired true positive rate

ODIN - Results

Published as a conference paper at ICLR 2018

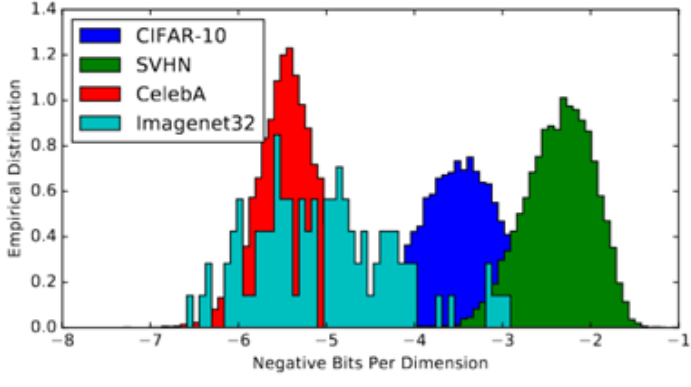
	Out-of-distribution dataset	FPR (95% TPR) ↓	Detection Error ↓	AUROC ↑	AUPR In ↑	AUPR Out ↑
		Baseline (Hendrycks & Gimpel, 2017) / ODIN				
Dense-BC CIFAR-10	TinyImageNet (crop)	34.7/**4.3**	10.0/**4.7**	95.3/**99.1**	96.4/**99.1**	93.8/**99.1**
	TinyImageNet (resize)	40.8/**7.5**	11.5/**6.1**	94.1/**98.5**	95.1/**98.6**	92.4/**98.5**
	LSUN (crop)	39.3/**11.4**	10.2/**7.2**	94.8/**97.9**	96.0/**98.0**	93.1/**97.9**
	LSUN (resize)	33.6/**3.8**	9.8/**4.4**	95.4/**99.2**	96.4/**99.3**	94.0/**99.2**
	Uniform	23.5/**0.0**	5.3/**0.5**	96.5/**99.0**	97.8/**100.0**	93.0/**99.0**
	Gaussian	12.3/**0.0**	4.7/**0.2**	97.5/**100.0**	98.3/**100.0**	95.9/**100.0**
Dense-BC CIFAR-100	TinyImageNet (crop)	67.8/**26.9**	36.4/**12.9**	83.0/**94.5**	85.3/**94.7**	80.8/**94.5**
	TinyImageNet (resize)	82.2/**57.0**	43.6/**22.7**	70.4/**85.5**	71.4/**86.0**	68.6/**84.8**
	LSUN (crop)	69.4/**18.6**	37.2/**9.7**	83.7/**96.6**	86.2/**96.8**	80.9/**96.5**
	LSUN (resize)	83.3/**58.0**	44.1/**22.3**	70.6/**86.0**	72.5/**87.1**	68.0/**84.8**
	Uniform	100.0/**100.0**	35.86/**17.9**	43.1/**99.5**	63.2/**87.5**	41.9/**65.1**
	Gaussian	100.0/**100.0**	41.2/**38.0**	30.6/**40.5**	53.4/**60.5**	37.6/**40.9**

Table 2: Distinguishing in- and out-of-distribution test set data for image classification. All values are percentages. ↑ indicates larger value is better, and ↓ indicates lower value is better. We use $T = 1000$ for all experiments. The noise magnitude ε was selected on a **separate validation dataset**, which is different from the out-of-distribution test sets. On CIFAR-10 pretrained model, we use $\varepsilon = 0.0014$ for all OOD test datasets; and $\varepsilon = 0.002$ for CIFAR-100 pretrained model.

Figure 6: S. Liang et al. 2020. "Enhancing The Reliability of Out-of-Distribution Image Detection in Neural Networks." arXiv:1706.02690

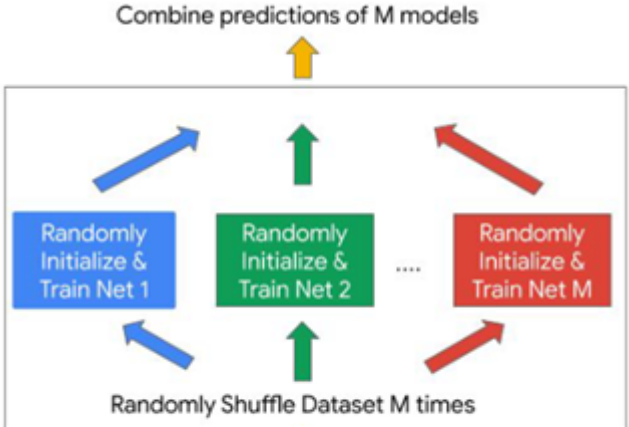
Limits of Uncertainty Estimation

- DNN are overconfident
- Supervised and generative models are overconfident
- Ideally, we would like to use bayesian models to estimate uncertainty
- Often expensive and heavily approximated
- A simpler solution: Ensembles



Deep Ensembles

- use a proper scoring rule as the training criterion,
- use adversarial training to smooth the predictive distributions, and
- train an ensemble



Uncertainty with Ensembles

Empirical observations:

- post-hoc calibration often fails
- marginalize over models (i.e. ensembles!) give surprisingly strong results across a broad spectrum of tasks.

Y. Ovadia et al. 2019. "Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty Under Dataset Shift."

Results

Expected Calibration Error (ECE) ↓ Measures the correspondence between predicted probabilities and empirical accuracy

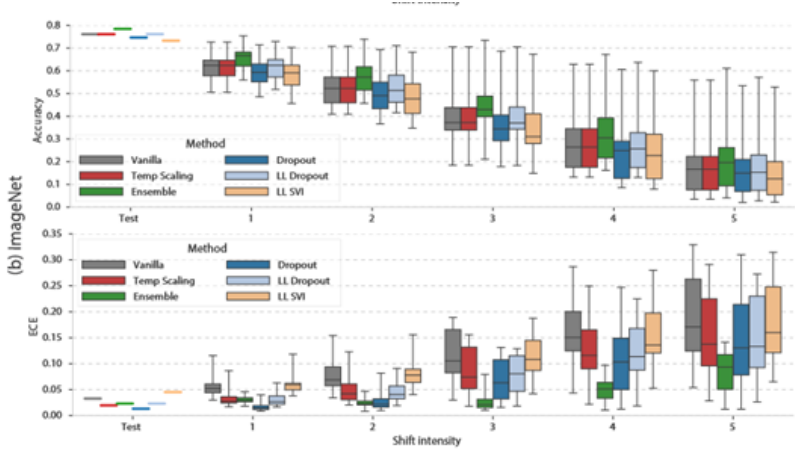


Figure 2: Calibration under distributional shift: a detailed comparison of accuracy and ECE under all types of corruptions on (a) CIFAR-10 and (b) ImageNet. For each method we show the mean on the test set and summarize the results on each intensity of shift with a box plot. Each box shows the

Summing Up

- DNN are overconfident and their confidence estimates cannot be trusted
- Thresholding is a simple and good baseline for OOD detection
- Ensembles show much more consistent results at the price of increased computational cost (both training and inference)

Background Class

Background Class

Sometimes we have access to a large set of «unknown examples»

- Example: let's say we have a supervised problem. We have a subset of classes we are interested in / have already labeled
- The rest are «background classes», and they represent what the model doesn't know
- We can use them for training as an additional explicit «background class»
- Convert an open world problem into a closed world one

Background Class

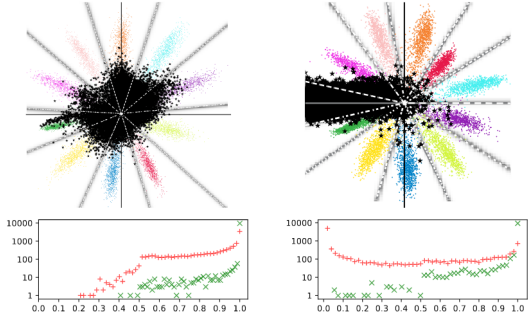
Softmax:

- If we don't train explicitly on the background class, background samples are roughly uniformly distributed
- they have some distinctive features (lower norms) but they cannot be easily separated from ID data

Background:

- when training explicitly on the background class, background samples are separated more easily
- Background samples still overlap with other classes due to them being much more frequent and heterogeneous than all the other classes (imbalanced classification)

Figure: plot of final features in 2D. Black dots are the samples from the background class



(a) Softmax (b) Background

Figure 9: A. R. Dhamija et al. 2018. "Reducing Network Agnostophobia." NIPS

Objectosphere Loss

Objectosphere[3]

- Train with a background class
- OBSERVATION: magnitudes of features for unknown samples in deep feature space are often lower than those of known samples.
- Objectosphere loss explicitly optimize this objective
- known samples should have a magnitude above a specified minimum
- background samples should have magnitude of the features close to zero

[3] A. R. Dhamija et al. 2018. "Reducing Network Agnostophobia." NIPS

Entropic Open-Set Loss

GOAL 1: we want background samples to have uniform probabilities.

Entropic Open-Set Loss:

$$J_E(x) = \begin{cases} -\log S_c(x) & \text{if } x \in \mathcal{D}'_c \text{ is from class } c \\ -\frac{1}{C}\sum_{c=1}^{C} \log S_c(x) & \text{if } x \in \mathcal{D}'_b \end{cases}$$

- $\mathcal{D}'_b$ set of background classes
- for normal classes, it's the usual cross-entropy loss
- for the background class, the loss is minimized when the probabilities are all equal ($S_c(x) = S = \frac{1}{C}$)

Objectosphere Loss

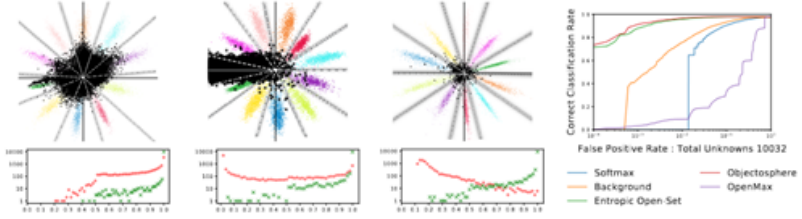
GOAL 2: we want background samples to have small (in norm) feature vectors. Notice that forcing uniform probabilities does not enforce small norm in the feature space.

Objectosphere Loss:

$$J_R = J_E + \lambda \begin{cases} \max(\xi - \|F(x)\|, 0)^2 & \text{if } x \in \mathcal{D}'_c \\ \|F(x)\|^2 & \text{if } x \in \mathcal{D}'_b \end{cases}$$

- $\|F(x)\|^2$ is the norm of the features for a sample x
- background samples should have small norms. Ideally, zero norm.
- To improve the separability of background samples, samples from known classes should have a feature vector that is not too small. At least ξ.

Objectosphere



(a) Softmax (b) Background (c) Objectosphere (d) Open-Set Recognition Curve

Figure 1: LeNet++ Responses To Knowns and Unknowns. *The network in (a) was only trained to classify the 10 MNIST classes ($\mathcal{D}'_c$) using softmax, while the networks in (b) and (c) added NIST letters [43] as known unknowns ($\mathcal{D}'_b$) trained with softmax or our novel Objectosphere loss. In the feature representation plots on top, colored dots represent test samples from the ten MNIST classes ($\mathcal{D}_c$), while black dots represent samples from the Devanagari [28] dataset ($\mathcal{D}_a$), and the dashed gray-white lines indicate class borders where softmax scores for neighboring classes are equal. This paper addresses how to improve recognition by reducing the overlap of network features from known samples $\mathcal{D}_c$ with features from unknown samples $\mathcal{D}_u$. The figures in the bottom are histograms of softmax probability values for samples of $\mathcal{D}_c$ and $\mathcal{D}_a$ with a logarithmic vertical axis. For known samples $\mathcal{D}_c$, the probability of the correct class is used, while for samples of $\mathcal{D}_a$ the maximum probability of any known class is displayed. In an application, a score threshold θ should be chosen to optimally separate unknown from known samples. Unfortunately, such a threshold is difficult to find for either (a) or (b), a better separation is achievable with the Objectosphere loss (c). The proposed Open-Set Classification Rate (OSCR) curve in (d) depicts the high accuracy of our approach even at a low false positive rate.*

Figure 10: A. R. Dhamija et al. 2018. "Reducing Network Agnostophobia." NIPS

Limitations

- We can train the model to recognize the unknown
- The open world problem becomes a close world supervised learning problem
- This is possible only because the unknown is actually known (we have the background class)
- What if we don't have background classes?

Open Set Recognition

Open Set Recognition

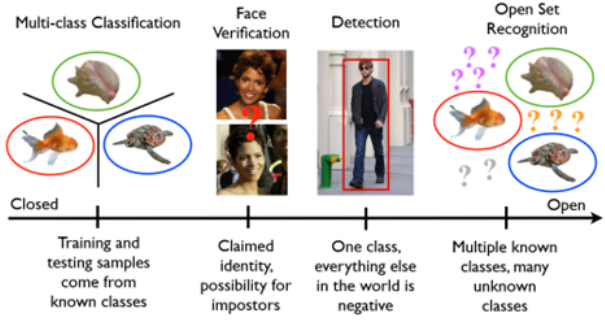


Fig. 1. Vision problems arranged in order of "openness". For some problems, we do not have knowledge of the entire set of possible classes during training, and must account for unknowns during testing. In this article, we develop a deeper understanding of those open cases.

Figure 11: W. J. Scheirer et al. "Towards Open Set Recognition." TPAMI, 2012.

Unbounded Decision Boundaries

Unbounded Decision Boundaries

- Regardless of how far it is from the ID samples, an input will always be classified with a known class
- In an open world setting, we need to allow a reject option

Example: a binary linear model

- Splits the space into two regions
- One side is positive, the other negative
- distance from the classification boundary is ignored

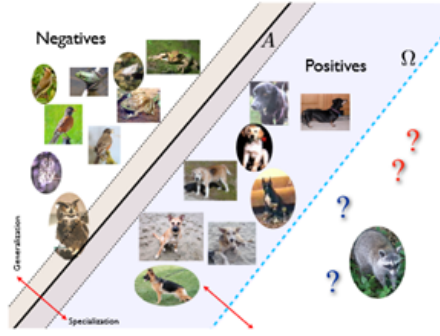
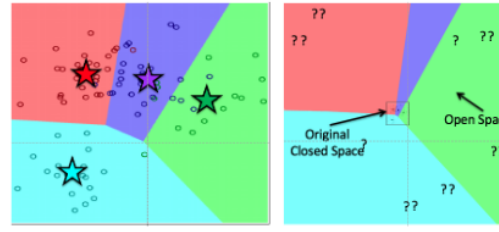


Fig. 2. The Open Set Recognition Problem explicitly assumes not all classes are known *a priori*. Square images are from training, oval images are from testing. The class of interest ("dog") is surrounded by other classes, which can be known ("frog", "birds"), or unknown ("owl","raccoon", "?"). Plane A maximizes the SVM margin making "dog" a half-space – which is mostly open space. The 1-vs-Set machine adds a second plane Ω and defines an optimization to adjust A and Ω to balance empirical and open space risk.

Unbounded Decision Boundaries

- (left) decision boundary close to training examples
- (right) zoomed out, the decision boundaries discriminate OOD examples (often with high confidence!)
- in fact, we could even say that the further we are from the boundary, the more confident the model is in its output!



(a) Example four-class model from closed set point of view. (b) Zooming out to show some open space.

Figure 1: The issue of open space can be seen by zooming out from around the training data. Open space is the region far from training samples. A traditional classifier, e.g., NCM shown here, will label everything including the unknown "?" inputs. Even points infinitely far away are labeled.

Figure 13: T. E. Boult et al. 2019. "Learning and the Unknown: Surveying Steps toward Open World Recognition." AAAI.

Distance-based Rejection

- Ideally, we would like to have a generative model of the input $p(x; \theta)$ to reject input with low likelihood
- This is very difficult in practice: generative models have poor OOD recognition and will have high lilehood on OOD data

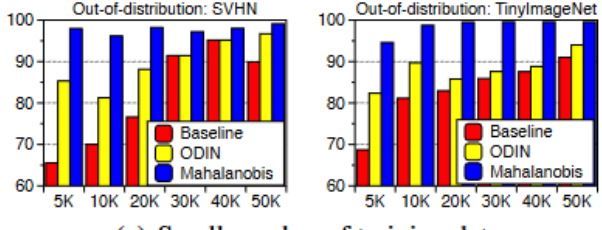
A simpler and more practical solution:

- Reject samples that are too far from the expected inputs, as computed via a simple distance metric
- Instead of the usual softmax classifier, we predict using a distance-based classifier
- We can put a threshold on the maximum distance to reject unknown samples

Mahalanobis Distance

Mahalanobis Distance $M(x)$: distance from the mean, measured in standard deviations

- $M(\mathrm{x}) = \max_c - (f(\mathrm{x}) - \hat{\mu}_c)^\top \hat{\Sigma}^{-1} (f(\mathrm{x}) - \hat{\mu}_c)$
- $\hat{\mu}_c$ class mean
- $\widehat{\Sigma}$ covariance matrix (shared among classes)



(a) Small number of training data

Figure 14: K. Lee et al. A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks." NIPS18

Activation Vectors

Activation Vectors

Activation vectors (e.g. logits) can be used to perform open-set recognition.

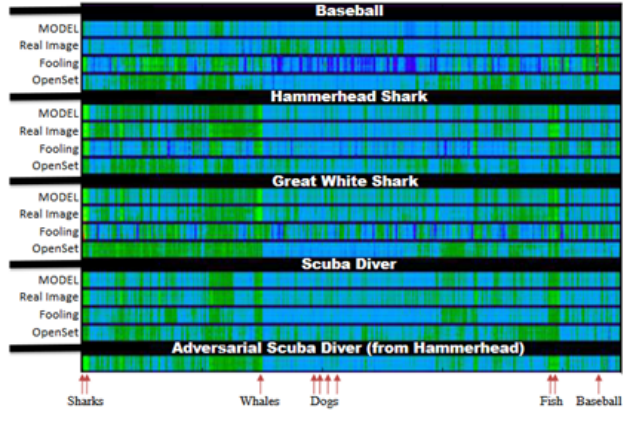


Figure 15: Bendale, Abhijit, and Terrance E. Boult. 2016. "Towards Open Set Deep Networks." In 2016 IEEE CVPR

OpenMax

We now combine several ideas that we have seen in the previous slides with some novel concepts

- modeling tails distribution with Extreme Value Theory and the Weibull distribution
- recognition of unknown instances with distance-based rejection
- softmax thresholding

Extreme Value Theory

- **Extreme Value Theory (EVT)** studies the probability distribution of extreme events (floods, earthquakes, market crashes)
- **open set recognition is an extreme value problem**: what is the likelihood that a novel unseen samples is coming from the train distribution?
- one key results from EVT is that the limiting distribution of extreme values can take a few different forms
  - we will use the **Weibull** distribution
  - we will not explain these result further

Weibull Distribution

pdf:

$$Wei(x \mid \lambda, k) = \frac{k}{\lambda}\left(\frac{x}{\lambda}\right)^{k-1} e^{-(x/\lambda)^k}, \quad x \geq 0$$

cdf: $1 - e^{-(x/\lambda)^k}, \quad x \geq 0$

A useful distribution to model failure times where the failure rate is proportional with time.

- $k < 1$ failure rate decrease with time (e.g. old companies have longer expected lifetimes than young companies)
- $k = 1$ exponential := failure rate constant over time
- $k > 1$ failure rate increases over time

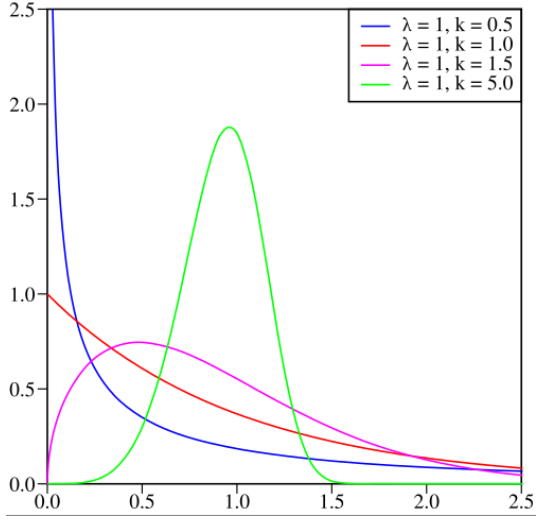


Figure 16: source: wikimedia

OpenMax

- before inference, estimate tails distribution with the Weibull distribution
- at inference time
  - rescale the logits using the rank-scaled Weibull cdf
  - assign removed mass to unknown class
  - classify as unknown if probability confidence is too low or if highest probability is assigned to unknown class

We need some calibration set to select the hyperparameters but we do not use any open set samples otherwise (even when fitting the Weibull).

OpenMax - Calibration

hyperparameters: η

- uses logits $v(x)$ as activation vector
- μ_k = mean activation vector (MAV) for a class k
- $d_c = \|v(x) - \mu_c\|$ distance from MAV for instances of class c correctly classified as class c
- fit a per-class Weibull distribution $\rho_j = (\tau_j, \kappa_j, \lambda_j) = \mathrm{FitHigh}\left(\|S_j - \mu_j\|, \eta\right)$
  - k and λ are the Weibull parameters
    - we fit only on the top-η distances.
    - the Weibull distribution is valid for extreme values at the limit. If we fit on all the samples the distribution is not a Weibull and the fit will find a slower decay and a worse ood detection
  - τ_j is a shift parameter. It is the minimum distance among the selected top-η instances.
    - note that we don't need OOD samples to estimate the Weibull parameters

OpenMax - Probability of Unknown

hyperparameters: $

Revise probabilities for top-α classes (others are left as is, they are small anyway...). **Then, assign leftover probability mass to the unknown class**.

- assume we ordered the logits indices by their value such that $s(i)$ is the i-th highest logit

$$\omega_{s(i)}(x) = 1 - \frac{\alpha - i}{\alpha} e^{-\left(\frac{\left\|d_{s_i} - \tau_{s(i)}\right\|}{\lambda_{s(i)}}\right)^{\kappa_{s(i)}}}$$

- NOTE: the Weibull cdf is $1 - e^{-(x/\lambda)^k}, \quad x \geq 0$
- NOTE: we scale the Weibull cdf by the rank $\frac{\alpha - i}{\alpha}$
- $v_i' = v_i \omega_i \left(d_i\right)$ rescale logits v_i according to $\omega_i(d)$
  - only for the top α classes
- $v_{\text{unk}} = \sum_c \left(1 - \omega_c \left(d_c\right)\right) \alpha_c$ assign leftover probability mass to unknown class

OpenMax - Rejection

hyperparameters: ϵ

Classification:

- we have K+1 classes: the original ones plus the unknown class (leftover probability mass)
- normalize logits with a softmax
- if p_unk is the highest probability classify as UNKNOWN
- if class c has the highest probability and $p_c > \epsilon$ classifiy as c
  - NOTE: we need to select a threshold ϵ for the minimum probability.
- if $p_c \leq \epsilon$ classify as UNKNOWN

OpenMax

- before inference, estimate tails distribution with the Weibull distribution
- at inference time
  - rescale the logits using the rank-scaled Weibull cdf
  - assign removed mass to unknown class
  - classify as unknown if probability confidence is too low or if highest probability is assigned to unknown class

We need some calibration set to select the hyperparameters but we do not use any open set samples otherwise (even when fitting the Weibull).

CAP Property (Compact-Abating Probability)

- in OpenMax the probabilities decays to zero as we move towards open space
- open space is identified by the model as anything that is far enough from the MAV of all the classes and has low probability
- the CAP property may seem trivial but not every OOD method satisfy it. For example, softmax thresholding does not guarantee it because examples far in the open space may still have high confidence (incorrectly)

OpenMax - OSR with Deep Networks

source: Bendale, Abhijit, and Terrance E. Boult. 2016. "Towards Open Set Deep Networks." In 2016 IEEE CVPR

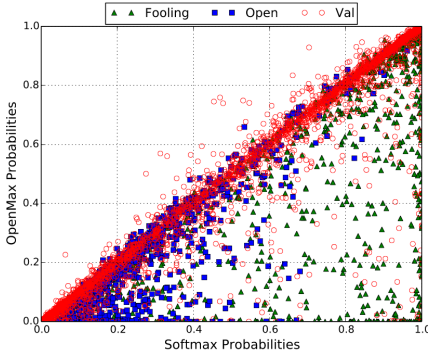


Figure 17: Softmax vs OpenMax probability scores. Softmax is overconfident on open set samples.

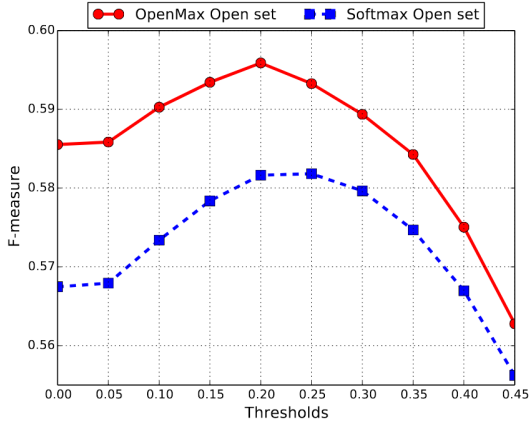


Figure 18: F-measure for different thresholds. Softmax vs OpenMax

Take-Home Messages

- Many real-world problems are much more «open» than the toy examples we study and use in a typical ML class
- Known unknowns can be accounted for during training (background classes, tuning rejection thresholds) to improve the rejection rate of unknown objects
- Unknown unknowns require rethinking the training algorithm to improve the separation between unknown and known samples
- Open world learning and uncertainty quantification are still open challenges

References

- Papers in the footnotes
- Open World Lifelong Learning A Continual Machine Learning Course https://owll-lab.com/teaching/cl_lecture/
  - Recordings are available