Selected Topics in Uncertainty and Distributional Shifts

Qualifying Oral Examination

Stephan Rabanser stephan@cs.toronto.edu

University of Toronto
Department of Computer Science

Vector Institute for
Artificial Intelligence

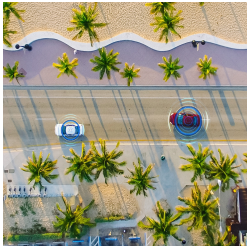
January 17, 2022

About Me

- **Educational Background**:
  - B.Sc. and M.Sc. in Computer Science from Technical University of Munich, Germany.
  - Joined Nicolas' lab as a PhD student in September 2020.
- **Industry/Internship Experience**:
  - Multiple research internships at Amazon / AWS AI Labs.
  - Interned in Zack's lab @ CMU to work on my Master's thesis on distribution shift detection (published at NeurIPS 2019).
- **Research Interests**: Robustness, Safety, Reliability, Uncertainty, Causality, Generative Modeling, Representation Learning, Anomaly Detection, Distribution Shifts, Interpretability, Out-of-Distribution Sample Detection, Healthcare Applications.

Motivation

Machine Learning systems are becoming ubiquitous.



We need a thorough understanding of the robustness properties of ML algorithms to ensure safe deployment, especially in high-stakes decision-making systems.

Image credit: `http://unsplash.com`

Uncertainty Quantifiaction

- Standard supervised learning setup:
  - Dataset $D_p = \{(x_i, y_i)\}_{i=1}^{N}$ where $(x, y) \sim p$ on $\mathcal{D} = \mathcal{X} \times \mathcal{Y}$ with $x \in \mathcal{X}$, $y \in \mathcal{Y}$.
  - Learn a suitable estimation function $h_\theta : \mathcal{X} \to \mathcal{Y}$ producing predictions $\hat{y} = h_\theta(x)$.
- We are interested in both a prediction $\mathbb{E}[y|x]$ and the associated uncertainty $\mathrm{Var}[y|x]$.
- Use the Bayesian inference framework by modeling the likelihood $p(y|x,\theta)$, the prior $p(\theta)$, the posterior $p(\theta|x,y)$, and the predictive distribution $p(y|x)$.

$$p(\theta|x,y) = \frac{p(y|x,\theta)p(\theta)}{\int p(y|x,\theta)p(\theta)d\theta} \qquad p(y|x) = \int p(y|x,\theta)p(\theta|x,y)d\theta$$

- Approximate $p(\theta|x,y)$ via Variational Inference [BCKW15], Markov Chain Monte Carlo [WRV$^+$20], Deep Ensembles [LPB17], and Monte Carlo Dropout.

Weight Uncertainty in Neural Networks [BCKW15]

- Goal: Minimize KL divergence between the Gaussian variational posterior $q(\theta|w)$ and the true posterior $p(\theta|x,y)$:

$$\min_w \mathrm{KL}[q(\theta|w)||p(\theta|x,y)]$$

- Yields variational free energy cost function:

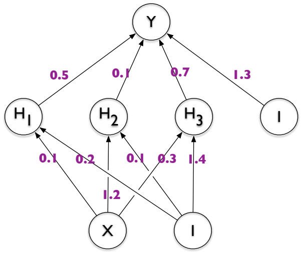
$$\mathcal{F}(\theta,x,y) := \mathrm{KL}[q(\theta|w)||p(\theta)] - \mathbb{E}_{q(\theta|w)}[\log p(y|x,\theta)]$$

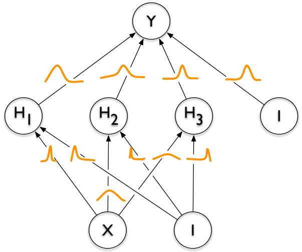
- Approximate $\mathcal{F}$ using unbiased MC samples:

$$\mathcal{F}(\theta,x,y) \approx \sum_i \log q(\theta_i|w) - \log p(\theta_i) - \log p(y|x,\theta_i)$$

- Motivate the usage of a scale-mixture prior:

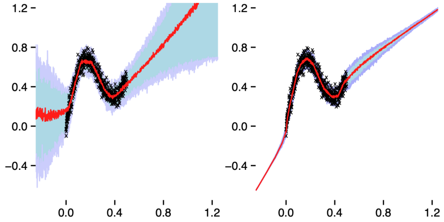
$$p(\theta) = \prod_j \pi\mathcal{N}(\theta_j|0,\sigma_1^2) + (1-\pi)\mathcal{N}(\theta_j|0,\sigma_2^2)$$





Weight Uncertainty in Neural Networks [BCKW15] (cont'd)

Method	**# Units/Layer**	**# Weights**	**Test Error**
SGD, no regularisation	800	1.3m	1.6%
SGD, dropout			≈ 1.3%
SGD, dropconnect	800	1.3m	**1.2%***
SGD	400	500k	1.83%
	800	1.3m	1.84%
	1200	2.4m	1.88%
SGD, dropout	400	500k	1.51%
	800	1.3m	1.33%
	1200	2.4m	1.36%
Bayes by Backprop, Gaussian	400	500k	1.82%
	800	1.3m	1.99%
	1200	2.4m	2.04%
Bayes by Backprop, Scale mixture	400	500k	1.36%
	800	1.3m	1.34%
	1200	2.4m	**1.32%**



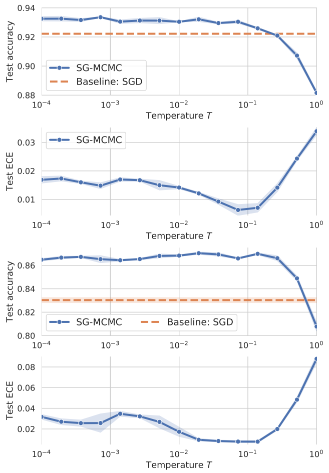
How Good is the Bayes Posterior in DNNs Really? [WRV+20]

- Empirical evidence shows that cooling the posterior, $T \ll 1$, improves predictive performance.
- A temperature $T = 1$ yields the true posterior while a temperature $T = 0$ corresponds to the maximum a-posteriori estimate (posterior sharpening).
- Rewrite the posterior distribution over θ,

$$p(\theta|x, y) \propto \exp(-U(\theta)/T)$$

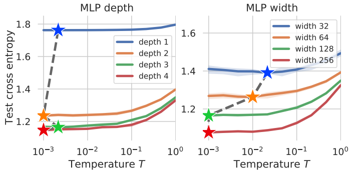
in terms of an energy function:

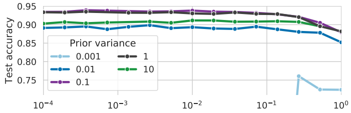
$$U(\theta) := -\sum_{i=1}^{n} \log p(y_i|x_i, \theta) - \log p(\theta),$$

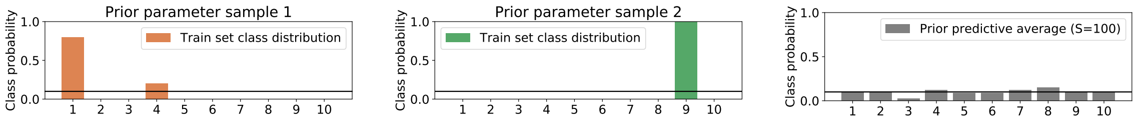


How Good is the Bayes Posterior in DNNs Really? [WRV+20] (cont'd)

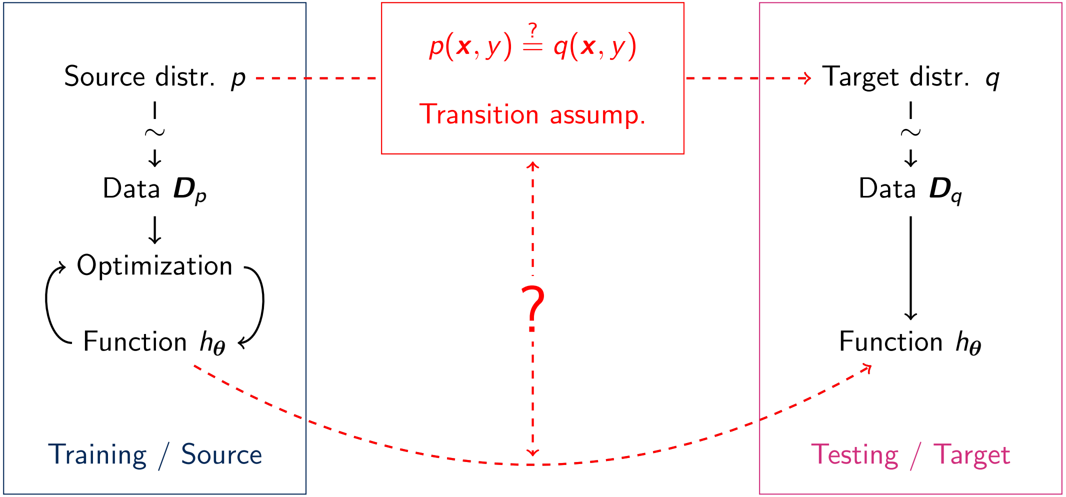
- Where does this effect come from? Potential candidates: inference procedure, likelihood, prior.
- **Prior**:
  - The typical Gaussian prior $p(\theta) = \mathcal{N}(0, I)$ is unintentionally informative as it produces concentrated class distributions.
  - It intensifies the cold posterior effect with increasing model depth and width.
  - Modifying the variance of the Gaussian was not found to mitigate this effect.







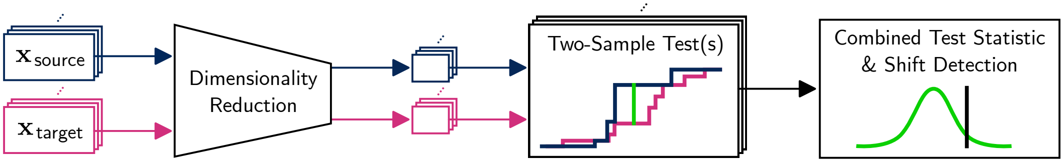
The Problem of Distribution Shift



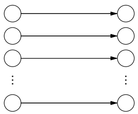
Empirical Study of Methods for Detecting Shift [RGL18]

Faced with distribution shift we want to **fail loudly**. Our goals are three-fold:

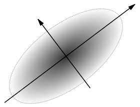
1. Detect when distribution shift occurs from as few examples as possible;
2. Characterize the shift in a qualitative manner; and
3. Provide some guidance on whether the shift is harmful or not.


$\mathbf{x}_{\text{source}}$
$\mathbf{x}_{\text{target}}$
Dimensionality Reduction
Two-Sample Test(s)
Combined Test Statistic & Shift Detection

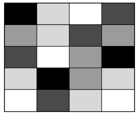
Empirical Study of Methods for Detecting Shift [RGL18](cont'd)



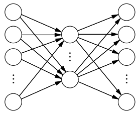
NoRed ○



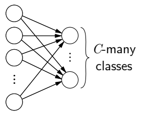
PCA ⬡



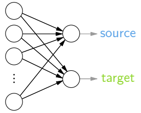
SRP ⬠



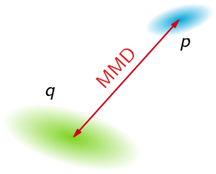
TAE ◇
UAE □



BBSDs ◁
BBSDh ▷

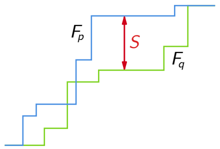


Classif ×



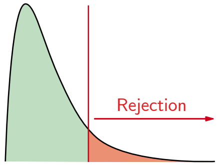
$$||\boldsymbol{\mu}_p - \boldsymbol{\mu}_q||^2_{\mathcal{F}}$$

MMD



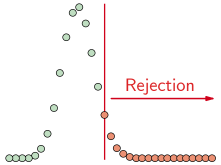
$$\sup_z |F_p(\boldsymbol{z}) - F_q(\boldsymbol{z})|$$

KS + Bonferroni



$$\sum_{i=1}^{2} \sum_{j=1}^{C} \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$$

χ^2

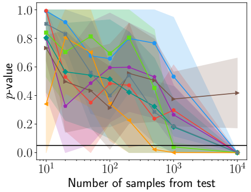


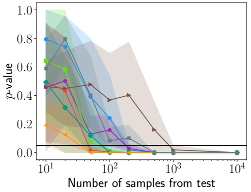
$$\frac{\mathrm{TP+TN}}{\mathrm{TP+TN+FP+FN}}$$

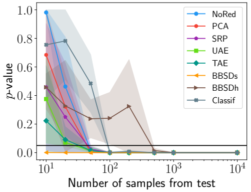
Binom

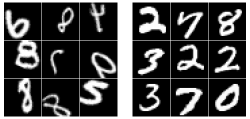
Empirical Study of Methods for Detecting Shift [RGL18](cont'd)

Test	DR	Number of samples from test			
		10	100	1,000	10,000
KS + Bonf.	NoRed	0.03	0.36	0.54	0.72
	PCA	0.11	0.36	0.54	0.63
	SRP	0.15	0.27	0.55	0.68
	UAE	0.12	0.33	0.56	0.77
	TAE	0.18	0.38	0.55	0.69
	BBSDs	**0.19**	**0.47**	**0.70**	**0.79**
χ^2	*BBSDh*	0.03	0.22	*0.46*	*0.57*
Bin	*Classif*	*0.01*	*0.21*	0.51	0.67
MMD	NoRed	0.14	*0.28*	0.55	–
	PCA	0.15	0.38	0.55	–
	SRP	*0.12*	0.31	0.54	–
	UAE	**0.20**	**0.43**	**0.61**	–
	TAE	0.18	0.38	0.59	–
	BBSDs	0.16	0.35	*0.50*	–









Evaluating Predictive Uncertainty Under Shift [OFR$^+$19]

Common misbelief: Current uncertainty quantification and calibration techniques are robust under distribution shift.

Methods for probabilistic deep learning:

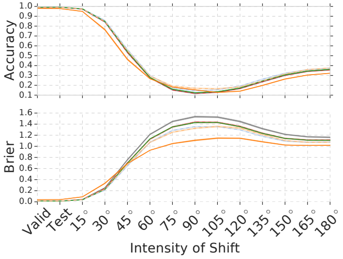
- Maximum softmax probability
- Temperature scaling
- Monte Carlo Dropout
- Deep Ensembles
- Stochastic Variational Inference
- Last Layer Dropout and Variational Inference

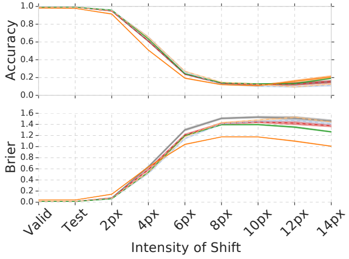
Uncertainty quality metrics:

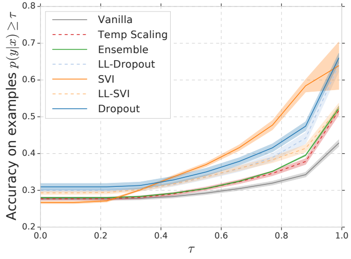
- Negative Log Likelihood (NLL): Evaluates the quality of model uncertainty on some held out set
- Brier Score: Assesses the accuracy of predicted probabilities.
- Expected Calibration Error (ECE): Measures the correspondence between predicted probabilities and empirical accuracy

Evaluating Predictive Uncertainty Under Shift [OFR$^+$19] (cont'd)

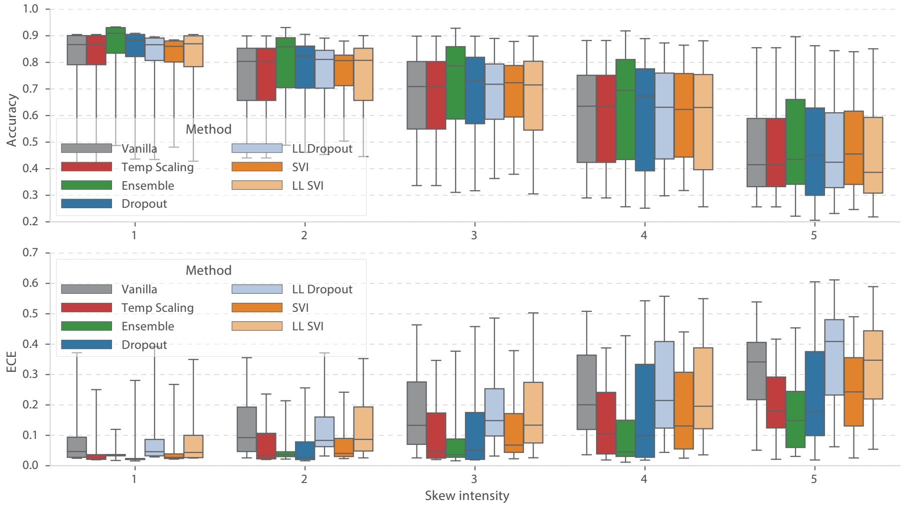
- Increasing the perturbation strength deteriorates both accuracy and uncertainty metrics.
- Post-hoc calibration on an iid validation set does not lead to (and might even hurt) well-calibrated predictions on OOD data.
- Deep Ensembles perform comparatively well, VI-based methods have mixed results.
- The results also show that the relative ranking of methods is mostly consistent throughout the experiments.







Evaluating Predictive Uncertainty Under Shift [OFR+19] (cont'd)



Conclusion

Uncertainty

- **Current State**:
  - Theoretical requirements are well understood.
  - A lot of approximate techniques exist.
- **Open Problems**:
  - Properly calibrated uncertainty is expensive, requiring approximations.
  - Inductive biases of current methods need more rigorous understanding.

Distribution Shifts

- **Current State**:
  - Detection is often possible without strong assumptions.
  - Correction is possible under assumptions.
- **Open Problems**:
  - Characterization, quantification, and correction under milder assumptions are mostly unsolved.
  - Deep connection with OOD approaches is needed.

OOD Detection

- **Current State**:
  - Multiple promising techniques have been proposed.
  - Strong performance on stark differences.
- **Open Problems**:
  - Many methods are supervised, requiring explicit access to OOD data.
  - Empirical evaluation mostly focusses on synthetic perturbations.

Thanks! :)

References I

- Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra, *Weight uncertainty in neural network*, International Conference on Machine Learning, PMLR, 2015, pp. 1613–1622.
- Nicholas Frosst, Nicolas Papernot, and Geoffrey Hinton, *Analyzing and improving representations with the soft nearest neighbor loss*, International Conference on Machine Learning, PMLR, 2019, pp. 2012–2020.
- Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson, *Why normalizing flows fail to detect out-of-distribution data*, arXiv preprint arXiv:2006.08545 (2020).
- Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin, *A simple unified framework for detecting out-of-distribution samples and adversarial attacks*, Advances in neural information processing systems **31** (2018).

References II

- Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell, *Simple and scalable predictive uncertainty estimation using deep ensembles*, Advances in neural information processing systems **30** (2017).
- David Madras and Richard Zemel, *Identifying and benchmarking natural out-of-context prediction problems*, Thirty-Fifth Conference on Neural Information Processing Systems, 2021.
- Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan, *Do deep generative models know what they don't know?*, arXiv preprint arXiv:1810.09136 (2018).
- Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua V Dillon, Balaji Lakshminarayanan, and Jasper Snoek, *Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift*, arXiv preprint arXiv:1906.02530 (2019).

References III

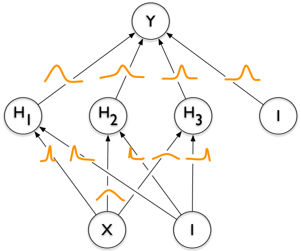
- Nicolas Papernot and Patrick McDaniel, *Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning*, arXiv preprint arXiv:1803.04765 (2018).
- Stephan Rabanser, Stephan Günnemann, and Zachary C Lipton, *Failing loudly: An empirical study of methods for detecting dataset shift*, arXiv preprint arXiv:1810.11953 (2018).
- Florian Wenzel, Kevin Roth, Bastiaan S Veeling, Jakub Swiatkowski, Linh Tran, Stephan Mandt, Jasper Snoek, Tim Salimans, Rodolphe Jenatton, and Sebastian Nowozin, *How good is the bayes posterior in deep neural networks really?*, arXiv preprint arXiv:2002.02405 (2020).

Backup

Weight Uncertainty in Neural Networks [BCKW15] (cont'd)

Training procedure

1. Sample $\epsilon \sim \mathcal{N}(0, I)$.
2. Compute weight $\theta_i = \mu + \log(1 + \exp(\rho)) \cdot \epsilon$.
3. Collect variational params in $w = (\mu, \rho)$.
4. Compute approximation of $\mathcal{F}$.
5. Update w using gradients of $\mathcal{F}$ wrt w.



Uncertainty Estimation using Deep Ensembles [LPB17]

1. Maximization of a proper scoring rule $S(p,q) = \mathbb{E}[S(p,(x,y))]$ for a scoring function $S(p,(x,y))$ ensuring that $S(p,q) \leq S(q,q)$ with equality iff $p(y|x) = q(y|x)$.
2. Employ adversarial training to smooth the predictive distribution.
3. Repeat process M-many times to yield an ensemble.

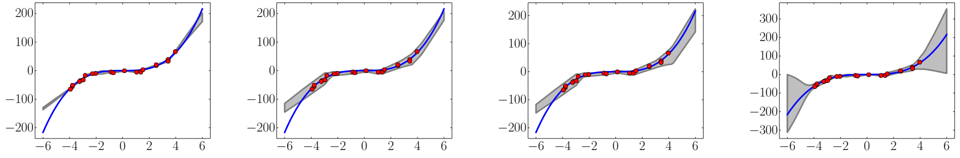
$$p(y|x) = \frac{1}{M}\sum_{m=1}^{M} p(y|x,\theta_m)$$

Approximate $p(y|x)$ as Gaussian for ease of computing quantiles:

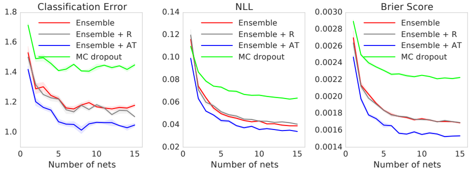
$$p(y|x) \approx \frac{1}{M}\sum_{m} \mathcal{N}\left(\mu_{\theta_m}(x), \sigma^2_{\theta_m}(x)\right)$$

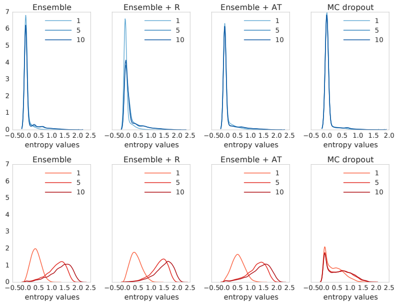
with $\mu_*(x) = \frac{1}{M}\sum_m \mu_{\theta_m}(x)$ and $\sigma^2_*(x) = \frac{1}{M}\sum_m \left(\sigma^2_{\theta_m}(x) + \mu^2_{\theta_m}(x)\right) - \mu^2_*(x)$.

Uncertainty Estimation using Deep Ensembles [LPB17] (cont'd)



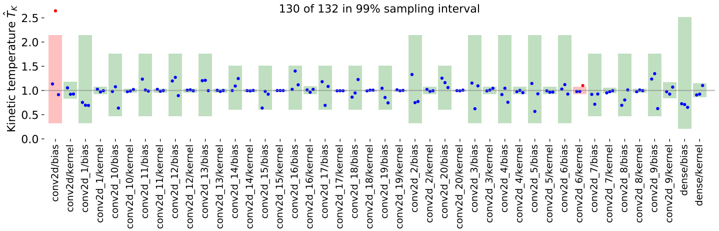
MSE NLL NLL + Adv Deep Ensembles





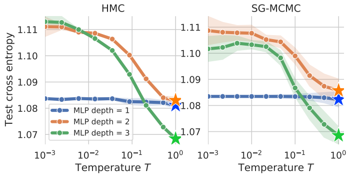
How Good is the Bayes Posterior in DNNs Really? [WRV+20] (cont'd)

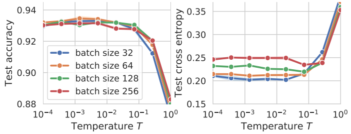
- **Inference procedure**:
  - Sampling $\theta \sim p(\theta|x, y)$ is based on Langevin dynamics over the parameters θ.
  - Employ Stochastic Gradient Markov Chain Monte Carlo (SG-MCMC) to approximate $\nabla_\theta U(\theta)$.
  - Two potential sources of error:
    - Mini-batch noise over $\nabla_\theta U(\theta)$.
    - Discretization errors incurred during discrete-time approximations in the SDE dynamics.



Key findings:

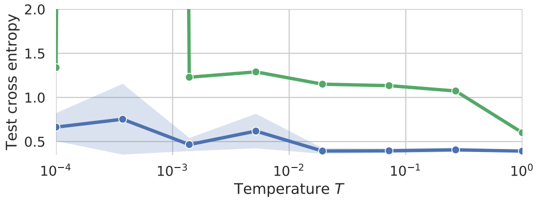
- The SDE simulation provides accurate samples.
- SG-MCMC is unbiased.
- Multiple choices for the mini-batch size all show best performance at $T < 1$.

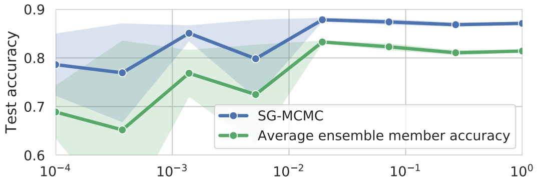




How Good is the Bayes Posterior in DNNs Really? [WRV+20] (cont'd)

- **Likelihood**:
  - Batch-normalization, dropout, or data augmentation could alter the likelihood in potentially unintended ways.
  - An evaluation performed by adding or removing these techniques still shows the presence of the cold posterior effect, even with clean likelihoods.





Empirical Study of Methods for Detecting Shift [RGL18](cont'd)

Covariate Shift

$$[p(\boldsymbol{x}) \neq q(\boldsymbol{x}) \wedge p(y|\boldsymbol{x}) = q(y|\boldsymbol{x})] \Rightarrow p(y|\boldsymbol{x})p(\boldsymbol{x}) \neq q(y|\boldsymbol{x})q(\boldsymbol{x}) \Rightarrow p(\boldsymbol{x}, y) \neq q(\boldsymbol{x}, y)$$

Label Shift

$$[p(y) \neq q(y) \wedge p(\boldsymbol{x}|y) = q(\boldsymbol{x}|y)] \Rightarrow p(\boldsymbol{x}|y)p(y) \neq q(\boldsymbol{x}|y)q(y) \Rightarrow p(\boldsymbol{x}, y) \neq q(\boldsymbol{x}, y)$$

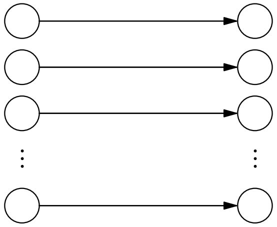
Concept Drift

$$[p(y|\boldsymbol{x}) \neq q(y|\boldsymbol{x}) \wedge p(\boldsymbol{x}) = q(\boldsymbol{x})] \Rightarrow p(y|\boldsymbol{x})p(\boldsymbol{x}) \neq q(y|\boldsymbol{x})q(\boldsymbol{x}) \Rightarrow p(\boldsymbol{x}, y) \neq q(\boldsymbol{x}, y)$$

$$[p(\boldsymbol{x}|y) \neq q(\boldsymbol{x}|y) \wedge p(y) = q(y)] \Rightarrow p(\boldsymbol{x}|y)p(y) \neq q(\boldsymbol{x}|y)q(y) \Rightarrow p(\boldsymbol{x}, y) \neq q(\boldsymbol{x}, y)$$

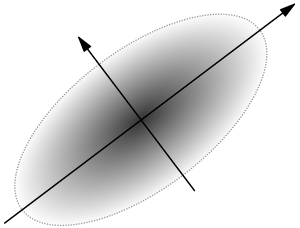
Empirical Study of Methods for Detecting Shift [RGL18](cont'd)

No Reduction (NoRed ○):



- To justify the use of any DR technique, our default baseline is to run tests on the original raw features.

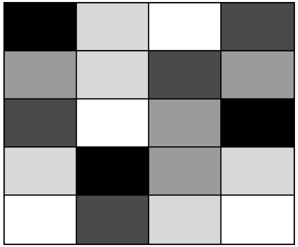
Principal Components Analysis (PCA ⬡):



- Find an optimal orthogonal transf. matrix such that points are linearly uncorrelated after transf.

Empirical Study of Methods for Detecting Shift [RGL18](cont'd)

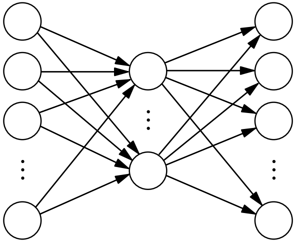
Sparse Random Projection (SRP ⬠):



$$R_{ij} = \begin{cases} +\sqrt{\frac{v}{K}} & \text{with prob. } \frac{1}{2v} \\ 0 & \text{with prob. } 1 - \frac{1}{v} \\ -\sqrt{\frac{v}{K}} & \text{with prob. } \frac{1}{2v} \end{cases}$$

with $v = \frac{1}{\sqrt{D}}$

Autoencoders (TAE ◇ and UAE ☐):

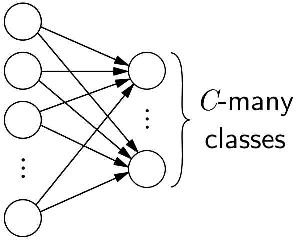


- Encoder $\phi : \mathcal{X} \to \mathcal{L}$
- Decoder $\psi : \mathcal{L} \to \mathcal{X}$

$$\phi, \psi = \arg\min_{\phi,\psi} \|\boldsymbol{X} - (\psi \circ \phi)\boldsymbol{X}\|^2$$

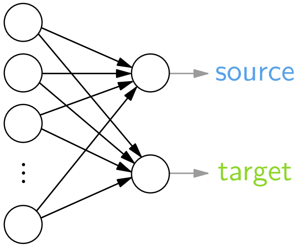
Empirical Study of Methods for Detecting Shift [RGL18](cont'd)

Label Classifiers (BBSDs ◁ and BBSDh ▷):



- Label classifier with softmax outputs (BBSDs ◁) or hard-thresholded predictions (BBSDh ▷).

Domain Classifier (Classif ×):

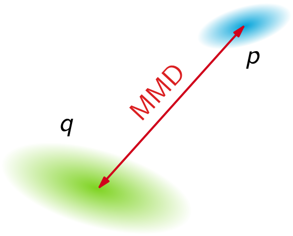


- Explicitly train a domain classifier to discriminate between data from source and target domains.

Empirical Study of Methods for Detecting Shift [RGL18](cont'd)

- Popular kernel-based technique for multivariate two-sample testing.
- Distinguish two distrib. based on their mean embeddings $\boldsymbol{\mu}_p$ and $\boldsymbol{\mu}_q$ in a reproducing kernel Hilbert space $\mathcal{F}$:

$$\mathrm{MMD}(\mathcal{F}, p, q) = ||\boldsymbol{\mu}_p - \boldsymbol{\mu}_q||_{\mathcal{F}}^2$$



- Empirical estimate:

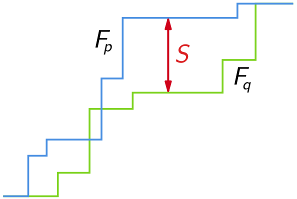
$$\begin{aligned}\mathrm{MMD}^2 = & \frac{1}{m(m-1)} \sum_{i=1}^{m} \sum_{j \neq i}^{m} \kappa(\mathbf{x}_i, \mathbf{x}_j) \\ & + \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{j \neq i}^{n} \kappa(\mathbf{x}'_i, \mathbf{x}'_j) \\ & - \frac{2}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} \kappa(\mathbf{x}_i, \mathbf{x}'_j)\end{aligned}$$

- Kernel: $\kappa(\mathbf{x}_1, \mathbf{x}_2) = e^{-\frac{1}{\sigma}\|\mathbf{x}_1 - \mathbf{x}_2\|^2}$
- Used with NoRed ○, PCA ⬡, SRP ⬠, TAE ◇, UAE □, and BBSDs ◁.

Empirical Study of Methods for Detecting Shift [RGL18](cont'd)

- Test each of the K dimensions separately (instead of jointly) using the Kolmogorov-Smirnov (KS) test.
- Largest difference S of the cumulative density functions over all values $\boldsymbol{z}$:

$$S = \sup_z |F_p(\boldsymbol{z}) - F_q(\boldsymbol{z})|$$



- Multiple hypothesis testing: we must subsequently combine the p-values from the K-many test.
- Problem: We cannot make strong assumptions about the (in)dependence among the tests.
- Solution: Bonferroni correction:
  - Does not assume (in)dependence.
  - Bounds the family-wise error rate, i.e. it is a conservative aggregation.
  - Rejects H_0 if $p_{\text{min}} \leq \frac{\alpha}{K}$.
- Used with NoRed ○, PCA ⬡, SRP ⬠, TAE ◇, UAE □, and BBSDs ◁.

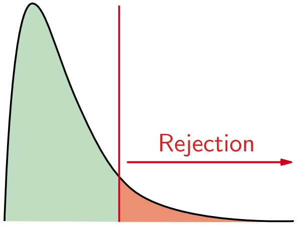
Empirical Study of Methods for Detecting Shift [RGL18](cont'd)

- Evaluate whether the freq. distr. of certain events observed in a sample is consistent with a particular theo. distr.
- Difference can be calculated as

$$X^2 = \sum_{i=1}^{2} \sum_{j=1}^{C} \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$$

  with observed counts O_{ij} and expected counts $E_{ij} = N_{\text{sum}} p_{i\bullet} p_{\bullet j}$ with
  - $p_{i\bullet} = \frac{n_{i\bullet}}{N_{\text{sum}}} = \sum_{j=1}^{c} \frac{n_{ij}}{N_{\text{sum}}}$ and
  - $p_{\bullet j} = \frac{n_{\bullet j}}{N_{\text{sum}}} = \sum_{i=1}^{r} \frac{n_{ij}}{N_{\text{sum}}}$.
- Under H_0, $X^2 \sim \chi^2_{C-1}$.

Sample	Cat 1	$\cdots$	Cat C	$\sum$
p	n_{p1}	$\cdots$	n_{pC}	$n_{p\bullet}$
q	n_{q1}	$\cdots$	n_{qC}	$n_{q\bullet}$
$\sum$	$n_{\bullet 1}$	$\cdots$	$n_{\bullet C}$	N_{sum}



- Used with BBSDh ▷.

Empirical Study of Methods for Detecting Shift [RGL18](cont'd)

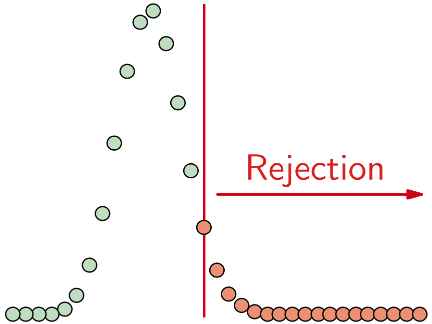
- Compare difference classifier accuracy (acc) on held-out data to random chance via a binomial test.

$$H_0 : \text{acc} = 0.5 \quad \text{vs} \quad H_A : \text{acc} > 0.5$$

- Under H_0, the acc follows a binomial distribution

$$\text{acc} \sim \text{Bin}(N_{\text{hold}}, 0.5)$$

where N_{hold} corresponds to the number of held-out samples.



- Used with Classif ×.

Empirical Study of Methods for Detecting Shift [RGL18](cont'd)

- Recall: our detection framework does not detect outliers but rather aims at capturing top-level shift dynamics.
- We can not decide whether any given sample is in- or out-of-distribution.
- But: we can harness domain assignments from the domain classifier.
- It is easy to identify the exemplars which the domain classifier was most confident in assigning to the target domain.

- Other shift detectors compare entire distributions against each other.
- Identification of samples which if removed would lead to a large increase in the overall p-value was not successful.

Empirical Study of Methods for Detecting Shift [RGL18](cont'd)

- Distribution shifts can cause arbitrarily severe degradation in performance.
- In practice distributions shift constantly and often these changes are benign.
- Goal: distinguishing malignant shifts from benign shifts.
- Problem: although prediction quality can be assessed easily on source data, we are not able compute the target error directly without labels.

- Heuristic methods for approximating the target performance:
  - **Difference classifier assignments:** assess black-box model's accuracy on the labeled top anomalous samples (*implicit* shift characterization).
  - **Domain expert:** Get hints on the target accuracy by evaluating the classifier on held-out source data that has been *explicitly* perturbed by a function determined by a domain expert.

Empirical Study of Methods for Detecting Shift [RGL18](cont'd)

Family-Wise Error Rate (FWER)

The most stringent control is given by procedures controlling the FWER, which limits the probability of making at least one false positive, formally

$$\mathsf{FWER} = P(V \geq 1) < \alpha$$

where V is the total amount of false discoveries.

False Discovery Rate (FDR)

A less stringent but more powerful alternative to the FWER is the FDR, which limits the expected proportion of false positives, formally

$$\mathsf{FDR} = \mathbb{E}\left[\frac{V}{M}\right] < \alpha$$

where M is the total amount of discoveries.

Empirical Study of Methods for Detecting Shift [RGL18](cont'd)

- Core experiments: synthetic shifts on MNIST and CIFAR-10 image datasets.
- Autoencoders: convolutional architecture with 3 convolutional layers.
- BBSD and Classif: ResNet-18 architecture.
- Network training (TAE $\diamond$, BBSDs $\triangleleft$, BBSDh $\triangleright$, Classif $\times$): SGD with momentum in batches of 128 examples over 200 epochs with early stopping.
- Dimensionality reduction to $K = 32$ (PCA ⬡, SRP ⬠, UAE $\square$, and TAE $\diamond$), $C = 10$ (BBSDs $\triangleleft$), and 1 (BBSDh $\triangleright$ and Classif $\times$).
- Evaluate shift detection at a significance level of $\alpha = 0.05$.
- Shift detection performance is averaged over a total of 5 random splits.
- Randomly split the data into training, validation, and test sets and then apply a particular shift to the test set only.
- Evaluate the models with various amounts of samples from the test set $s \in \{10, 20, 50, 100, 200, 500, 1000, 10000\}$.

Empirical Study of Methods for Detecting Shift [RGL18](cont'd)

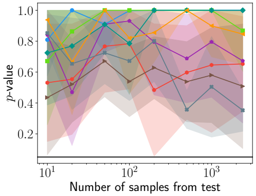
For each shift type (as appropriate) we explored three levels of shift intensity and various percentages of affected data $\delta \in \{0.1, 0.5, 1.0\}$.

- **Adversarial (adv)**: We turn a fraction δ of samples into adversarial samples via FGSM;
- **Knock-out (ko)**: We remove a fraction δ of samples from class 0, creating class imbalance;
- **Gaussian noise (gn)**: We corrupt covariates of a fraction δ of test set samples by Gaussian noise with standard deviation $\sigma \in \{1, 10, 100\}$ (denoted *s_gn*, *m_gn*, and *l_gn*);
- **Image (img)**: We also explore more natural shifts to images, modifying a fraction δ of images with combinations of random rotations $\{10, 40, 90\}$, (x, y)-axis-translation percentages $\{0.05, 0.2, 0.4\}$, as well as zoom-in percentages $\{0.1, 0.2, 0.4\}$ (denoted *s_img*, *m_img*, and *l_img*);
- **Image + knock-out (m_img+ko)**: We apply a fixed medium image shift with $\delta_1 = 0.5$ and a variable knock-out shift δ;

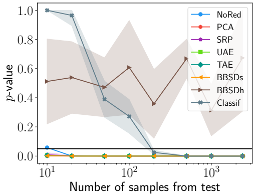
Empirical Study of Methods for Detecting Shift [RGL18](cont'd)

- **Only-zero + image (oz+m_img)**: Here, we only include images from class 0 in combination with a variable medium image shift affecting only a fraction δ of the data;
- **Original splits**: We evaluate our detectors on the original source/target splits provided by the creators of MNIST, CIFAR-10, Fashion MNIST, and SVHN datasets (assumed to be i.i.d.);
- **Real shift datasets**:
  - Domain adaptation from MNIST (source) to USPS (target).
  - COIL-100 dataset where images between 0° and 175° are sampled by the source and images between 180° and 355° are sampled by the target distribution.

Empirical Study of Methods for Detecting Shift [RGL18](cont'd)

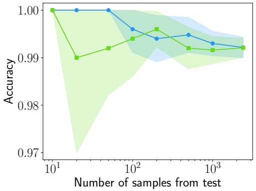


(a) Test random.

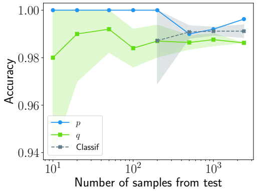


(b) Test partitioned.

(c) Top different.



(d) Acc. random.



(e) Acc. partitioned.

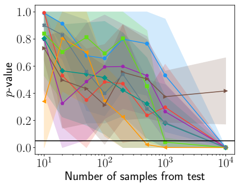
(f) Top similar.

Empirical Study of Methods for Detecting Shift [RGL18](cont'd)

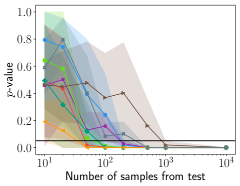
Table: Detection accuracy for small, medium, and large simulated shifts and low (10%), medium (50%), and high (100%) percentages of perturbed target samples on MNIST and CIFAR-10. Reported accuracy values are results of the best DR technique (univariate: BBSDs, multivariate: average of UAE and TAE). Underlined entries indicate accuracy values > 0.5.

Test	Intensity	Number of samples from test							
		10	20	50	100	200	500	1,000	10,000
Univariate	Small	0.00	0.00	0.14	0.14	0.18	0.36	0.40	$\underline{0.54}$
	Medium	0.14	0.21	0.39	0.38	0.42	$\underline{0.57}$	$\underline{0.66}$	$\underline{0.76}$
	Large	0.32	$\underline{0.54}$	$\underline{0.78}$	$\underline{0.82}$	$\underline{0.83}$	$\underline{0.92}$	$\underline{0.96}$	$\underline{1.00}$
	10%	0.11	0.15	0.24	0.25	0.28	0.44	$\underline{0.54}$	$\underline{0.66}$
	50%	0.14	0.28	$\underline{0.52}$	$\underline{0.53}$	$\underline{0.60}$	$\underline{0.68}$	$\underline{0.72}$	$\underline{0.85}$
	100%	0.26	0.41	$\underline{0.61}$	$\underline{0.64}$	$\underline{0.70}$	$\underline{0.82}$	$\underline{0.84}$	$\underline{0.86}$
Multivariate	Small	0.11	0.11	0.12	0.14	0.20	0.23	0.33	–
	Medium	0.11	0.19	0.23	0.27	0.32	0.42	0.44	–
	Large	0.34	0.45	$\underline{0.57}$	$\underline{0.68}$	$\underline{0.72}$	$\underline{0.82}$	$\underline{0.93}$	–
	10%	0.12	0.13	0.21	0.26	0.27	0.31	0.44	–
	50%	0.19	0.27	0.41	0.41	0.47	$\underline{0.57}$	$\underline{0.60}$	–
	100%	0.29	0.41	0.44	$\underline{0.53}$	$\underline{0.60}$	$\underline{0.70}$	$\underline{0.78}$	–

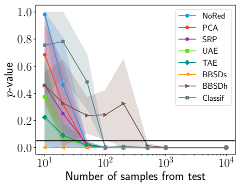
Empirical Study of Methods for Detecting Shift [RGL18](cont'd)



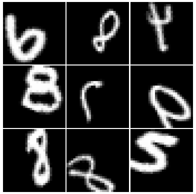
(a) Test w/ 10%.



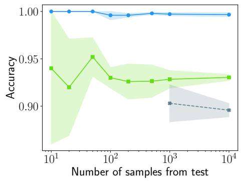
(b) Test w/ 50%.



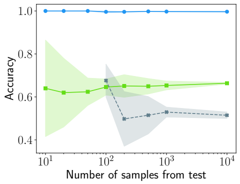
(c) Test w/ 100%.



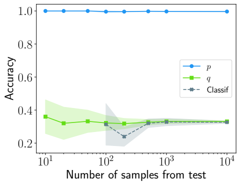
(d) Top different.



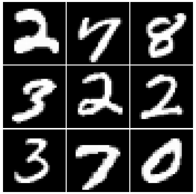
(e) Acc. w/ 10%.



(f) Acc. w/ 50%.

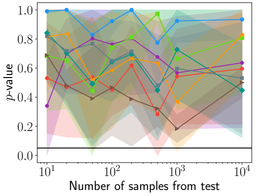


(g) Acc. w/ 100%.

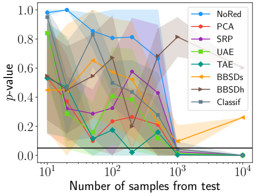


(h) Top similar.

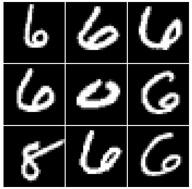
Empirical Study of Methods for Detecting Shift [RGL18](cont'd)



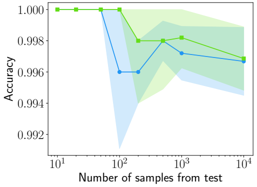
(a) Test random.



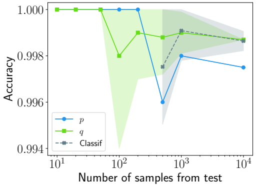
(b) Test original.



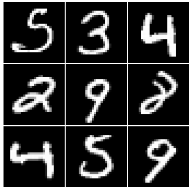
(c) Top different.



(d) Acc. random.



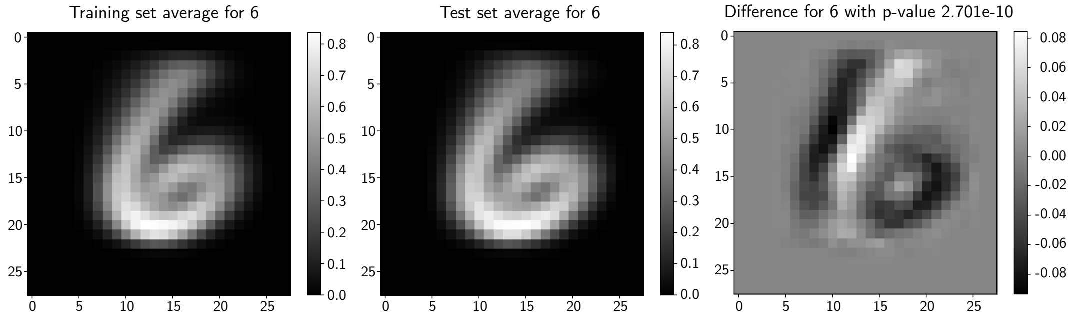
(e) Acc. original.



(f) Top similar.

Empirical Study of Methods for Detecting Shift [RGL18](cont'd)

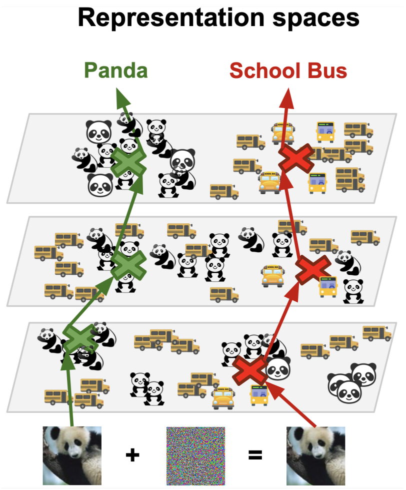
- The original splits from the MNIST dataset appear to exhibit a dataset shift.
- We observed that the top anomalous samples depicted the digit 6.
- This particular shift does not look significant to the human eye and is also declared harmless by our malignancy detector.



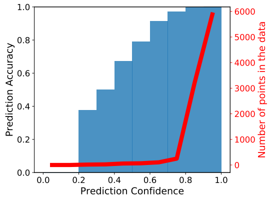
Deep k Nearest Neighbors [PM18]

- Perform a nearest neighbor analysis in each induced latent space.
- Legitimate inputs from a class should be close to points from the same class.

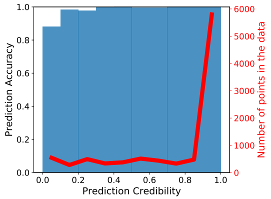
1. Obtain $\{f_\lambda(x) \mid \lambda \in 1..l\}$.
2. Find labels of nearest neighbors: $\Omega_\lambda \leftarrow \{Y_i : i \in \Gamma\}$.
3. Compute calibration $A = \{\alpha(x,y) : (x,y) \in (X^c, Y^c)\}$.
4. Compute nonconformity $\alpha(z,j) \leftarrow \sum_{\lambda \in 1..l} |i \in \Omega_\lambda : i \neq j|$.
5. Computer empirical p-value $p_j(z) = \frac{|\{\alpha \in A : \alpha \geq \alpha(z,j)\}|}{|A|}$.



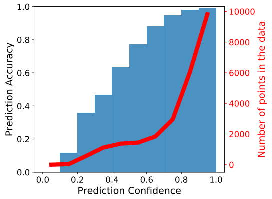
Deep *k* Nearest Neighbors [PM18] (cont'd)



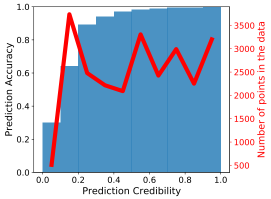
Softmax - MNIST test



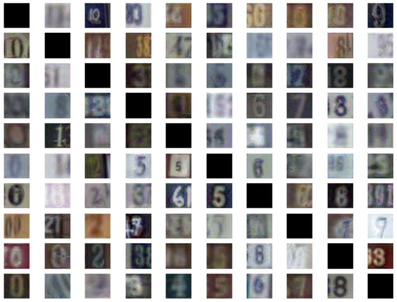
DkNN - MNIST test

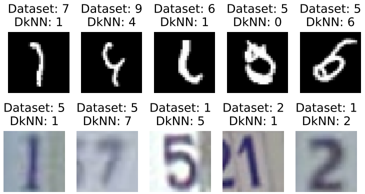


Softmax - SVHN test



DkNN - SVHN test





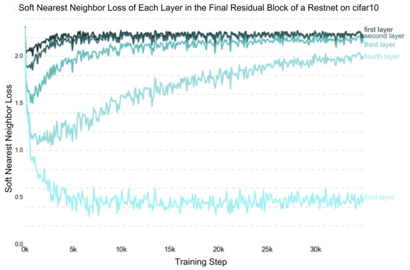
Representations Using Soft Nearest Neighbor Loss [FPH19]

- Measure Entanglement via Soft Nearest Neighbor Loss:

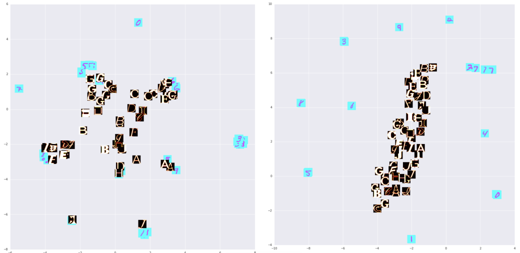
$$l_{sn}(x, y, T) = -\frac{1}{b} \sum_{i \in 1..b} \log \left(\frac{\sum\limits_{\substack{j \in 1..b \\ j \neq i \\ y_i = y_j}} e^{-\frac{||x_i - x_j||^2}{T}}}{\sum\limits_{\substack{k \in 1..b \\ k \neq i}} e^{-\frac{||x_i - x_k||^2}{T}}} \right)$$

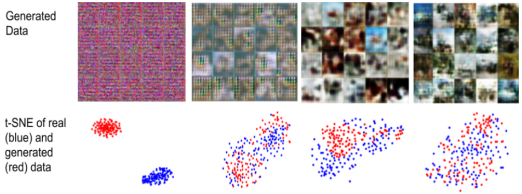
- Entangling lower layers leads to better generalization ability.
- Disentangling higher layers leads to better linear separation.

- Yielded representations are better suited to detect ambiguous/OOD data points as they are projected off the data manifold.

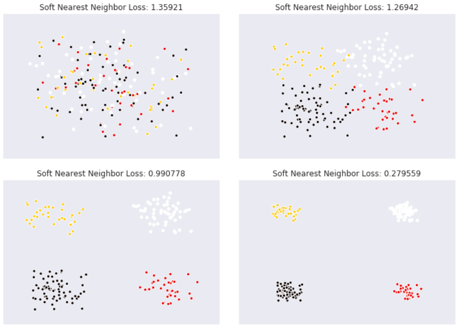


Representations Using Soft Nearest Neighbor Loss [FPH19] (cont'd)

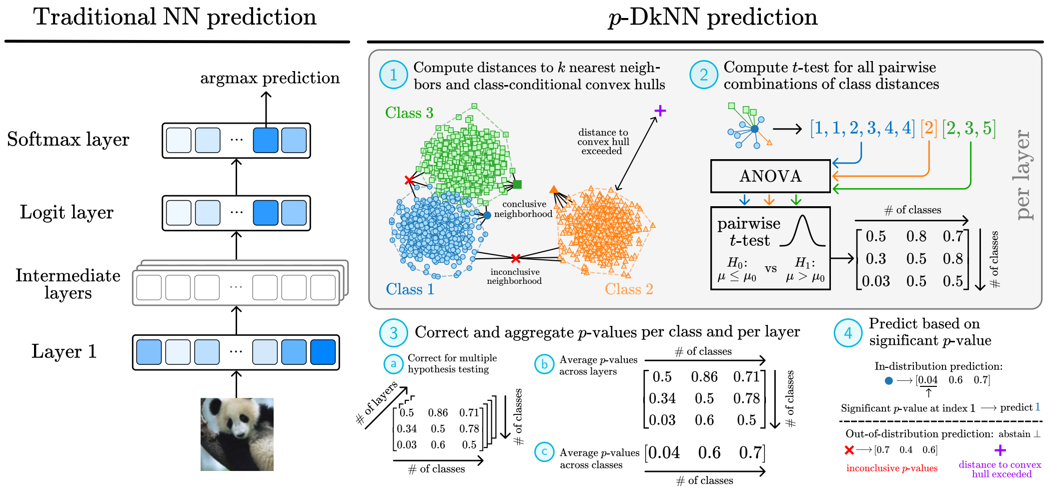




Training Step	0	400	800	2000
Soft Nearest Neighbor Loss	0.5617	0.6196	0.6208	0.6487



p-DkNN


Traditional NN prediction
argmax prediction
Softmax layer
Logit layer
Intermediate layers
Layer 1
p-DkNN prediction
1 Compute distances to k nearest neighbors and class-conditional convex hulls
Class 3
Class 1
Class 2
distance to convex hull exceeded
conclusive neighborhood
inconclusive neighborhood
2 Compute t-test for all pairwise combinations of class distances
[1, 1, 2, 3, 4, 4] [2] [2, 3, 5]
ANOVA
pairwise t-test
H0: μ ≤ μ0 vs H1: μ > μ0
# of classes
0.5 0.8 0.7
0.3 0.5 0.8
0.03 0.5 0.5
per layer
3 Correct and aggregate p-values per class and per layer
a Correct for multiple hypothesis testing
# of layers
0.5 0.86 0.71
0.34 0.5 0.78
0.03 0.6 0.5
b Average p-values across layers
c Average p-values across classes
[0.04 0.6 0.7]
4 Predict based on significant p-value
In-distribution prediction:
→ [0.04 0.6 0.7]
Significant p-value at index 1 → predict 1
Out-of-distribution prediction: abstain ⊥
→ [0.7 0.4 0.6]
inconclusive p-values
distance to convex hull exceeded

Deep Mahalanobis Detector [LLLS18]

- Parameterize each class in each hidden hidden layer via a multivariate Gaussian:

$$p(f_l(x)|y = c) = \mathcal{N}(f_l(x)|\mu_{c,l}, \Sigma_l)$$

- Estimation of $\hat{\mu}_{c,l} \approx \mu_{c,l}$ and $\hat{\Sigma}_l \approx \Sigma_l$ via empirical mean and covariance.
- Mahalanobis score at layer l is then given by

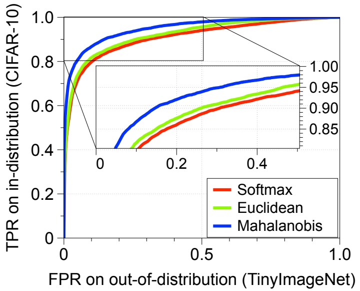
$$M_l(x) = \max_c -(f_l(x)-\hat{\mu}_{c,l})^\top \hat{\Sigma}_l^{-1}(f_l(x)-\hat{\mu}_{c,l}).$$

- Pre-processing the inputs using adversarial-like perturbations:

$$\hat{x} = x + \epsilon \cdot \text{sign}(\nabla_x M_L(x))$$

- Aggregate Mahalanobis score across all layers:

$$M(x) = \sum_l \alpha_l M_l(x).$$



Deep Mahalanobis Detector [LLLS18] (cont'd)

In-dist (model)	OOD	Validation on OOD samples			Validation on adversarial samples		
		TNR at TPR 95%	AUROC	Detection acc.	TNR at TPR 95%	AUROC	Detection acc.
		Maximum softmax / ODIN / Mahalanobis (ours)			Maximum softmax / ODIN / Mahalanobis (ours)		
CIFAR-10 (DenseNet)	SVHN	40.2 / 86.2 / **90.8**	89.9 / 95.5 / **98.1**	83.2 / 91.4 / **93.9**	40.2 / 70.5 / **89.6**	89.9 / 92.8 / **97.6**	83.2 / 86.5 / **92.6**
	TinyImageNet	58.9 / 92.4 / **95.0**	94.1 / 98.5 / **98.8**	88.5 / 93.9 / **95.0**	58.9 / 87.1 / **94.9**	94.1 / 97.2 / **98.8**	88.5 / 92.1 / **95.0**
	LSUN	66.6 / 96.2 / **97.2**	95.4 / 99.2 / **99.3**	90.3 / 95.7 / **96.3**	66.6 / 92.9 / **97.2**	95.4 / 98.5 / **99.2**	90.3 / 94.3 / **96.2**
CIFAR-100 (DenseNet)	SVHN	26.7 / 70.6 / **82.5**	82.7 / 93.8 / **97.2**	75.6 / 86.6 / **91.5**	26.7 / 39.8 / **62.2**	82.7 / 88.2 / **91.8**	75.6 / 80.7 / **84.6**
	TinyImageNet	17.6 / 42.6 / **86.6**	71.7 / 85.2 / **97.4**	65.7 / 77.0 / **92.2**	17.6 / 43.2 / **87.2**	71.7 / 85.3 / **97.0**	65.7 / 77.2 / **91.8**
	LSUN	16.7 / 41.2 / **91.4**	70.8 / 85.5 / **98.0**	64.9 / 77.1 / **93.9**	16.7 / 42.1 / **91.4**	70.8 / 85.7 / **97.9**	64.9 / 77.3 / **93.8**
SVHN (DenseNet)	CIFAR-10	69.3 / 71.7 / **96.8**	91.9 / 91.4 / **98.9**	86.6 / 85.8 / **95.9**	69.3 / 69.3 / **97.5**	91.9 / 91.9 / **98.8**	86.6 / 86.6 / **96.3**
	TinyImageNet	79.8 / 84.1 / **99.9**	94.8 / 95.1 / **99.9**	90.2 / 90.4 / **98.9**	79.8 / 79.8 / **99.9**	94.8 / 94.8 / **99.8**	90.2 / 90.2 / **98.9**
	LSUN	77.1 / 81.1 / **100**	94.1 / 94.5 / **99.9**	89.1 / 89.2 / **99.3**	77.1 / 77.1 / **100**	94.1 / 94.1 / **99.9**	89.1 / 89.1 / **99.2**
CIFAR-10 (ResNet)	SVHN	32.5 / 86.6 / **96.4**	89.9 / 96.7 / **99.1**	85.1 / 91.1 / **95.8**	32.5 / 40.3 / **75.8**	89.9 / 86.5 / **95.5**	85.1 / 77.8 / **89.1**
	TinyImageNet	44.7 / 72.5 / **97.1**	91.0 / 94.0 / **99.5**	85.1 / 86.5 / **96.3**	44.7 / 69.6 / **95.5**	91.0 / 93.9 / **99.0**	85.1 / 86.0 / **95.4**
	LSUN	45.4 / 73.8 / **98.9**	91.0 / 94.1 / **99.7**	85.3 / 86.7 / **97.7**	45.4 / 70.0 / **98.1**	91.0 / 93.7 / **99.5**	85.3 / 85.8 / **97.2**
CIFAR-100 (ResNet)	SVHN	20.3 / 62.7 / **91.9**	79.5 / 93.9 / **98.4**	73.2 / 88.0 / **93.7**	20.3 / 12.2 / **41.9**	79.5 / 72.0 / **84.4**	73.2 / 67.7 / **76.5**
	TinyImageNet	20.4 / 49.2 / **90.9**	77.2 / 87.6 / **98.2**	70.8 / 80.1 / **93.3**	20.4 / 33.5 / **70.3**	77.2 / 83.6 / **87.9**	70.8 / 75.9 / **84.6**
	LSUN	18.8 / 45.6 / **90.9**	75.8 / 85.6 / **98.2**	69.9 / 78.3 / **93.5**	18.8 / 31.6 / **56.6**	75.8 / 81.9 / **82.3**	69.9 / 74.6 / **79.7**
SVHN (ResNet)	CIFAR-10	78.3 / 79.8 / **98.4**	92.9 / 92.1 / **99.3**	90.0 / 89.4 / **96.9**	78.3 / 79.8 / **94.1**	92.9 / 92.1 / **97.6**	90.0 / 89.4 / **94.6**
	TinyImageNet	79.0 / 82.1 / **99.9**	93.5 / 92.0 / **99.9**	90.4 / 89.4 / **99.1**	79.0 / 80.5 / **99.2**	93.5 / 92.9 / **99.3**	90.4 / 90.1 / **98.8**
	LSUN	74.3 / 77.3 / **99.9**	91.6 / 89.4 / **99.9**	89.0 / 87.2 / **99.5**	74.3 / 76.3 / **99.9**	91.6 / 90.7 / **99.9**	89.0 / 88.2 / **99.5**

Deep Mahalanobis Detector [LLLS18] (cont'd)

Model	Dataset (model)	Score	Detection of known attack				Detection of unknown attack			
			FGSM	BIM	DeepFool	CW	FGSM (seen)	BIM	DeepFool	CW
DenseNet	CIFAR-10	KD+PU	85.96	96.80	68.05	58.72	85.96	3.10	68.34	53.21
		LID	98.20	99.74	**85.14**	80.05	98.20	94.55	70.86	71.50
		Mahalanobis (ours)	**99.94**	**99.78**	83.41	**87.31**	**99.94**	**99.51**	**83.42**	**87.95**
	CIFAR-100	KD+PU	90.13	89.69	68.29	57.51	90.13	66.86	65.30	58.08
		LID	99.35	98.17	70.17	73.37	99.35	68.62	69.68	72.36
		Mahalanobis (ours)	**99.86**	**99.17**	**77.57**	**87.05**	**99.86**	**98.27**	**75.63**	**86.20**
	SVHN	KD+PU	86.95	82.06	89.51	85.68	86.95	83.28	84.38	82.94
		LID	99.35	94.87	91.79	94.70	99.35	92.21	80.14	85.09
		Mahalanobis (ours)	**99.85**	**99.28**	**95.10**	**97.03**	**99.85**	**99.12**	**93.47**	**96.95**
ResNet	CIFAR-10	KD+PU	81.21	82.28	81.07	55.93	83.51	16.16	76.80	56.30
		LID	99.69	96.28	88.51	82.23	99.69	95.38	71.86	77.53
		Mahalanobis (ours)	**99.94**	**99.57**	**91.57**	**95.84**	**99.94**	**98.91**	**78.06**	**93.90**
	CIFAR-100	KD+PU	89.90	83.67	80.22	77.37	89.90	68.85	57.78	73.72
		LID	98.73	96.89	71.95	78.67	98.73	55.82	63.15	75.03
		Mahalanobis (ours)	**99.77**	**96.90**	**85.26**	**91.77**	**99.77**	**96.38**	**81.95**	**90.96**
	SVHN	KD+PU	82.67	66.19	89.71	76.57	82.67	43.21	**84.30**	67.85
		LID	97.86	90.74	92.40	88.24	97.86	84.88	67.28	76.58
		Mahalanobis (ours)	**99.62**	**97.15**	**95.73**	**92.15**	**99.62**	**95.39**	72.20	**86.73**

Do DGMs Know What They Don't Know? [NMT+18]

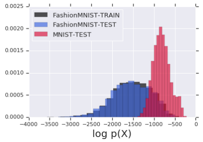
- Multiple DGMs have been found to assign higher likelihood to samples not from the training set.
- Paper investigates normalizing flows in particular.
- Deeper analysis on the change-of-variable objective finds that:
  - $p(z)$ behaves as expected.
  - $\log \left| \frac{\partial f}{\partial x} \right|$ is larger for OOD data.
- Constant inputs achieve the highest likelihood.
- Neither
  - changing the flow to be volume preserving; nor
  - robustifying the likelihood assignment using ensembling

  helps to mitigate the observed effect.

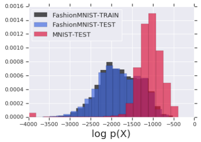
Data Set	Avg. Bits/Dimension
Glow Trained on FashionMNIST	
FashionMNIST-Train	2.902
FashionMNIST-Test	2.958
MNIST-Test	**1.833**
Glow Trained on MNIST	
MNIST-Test	1.262

Data Set	Avg. Bits/Dimension
Glow Trained on CIFAR-10	
CIFAR10-Train	3.386
CIFAR10-Test	3.464
SVHN-Test	**2.389**
Glow Trained on SVHN	
SVHN-Test	2.057

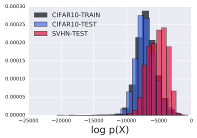
Do DGMs Know What They Don't Know? [NMT+18] (cont'd)



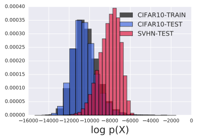
(a) **PixelCNN**: FMNIST vs MNIST



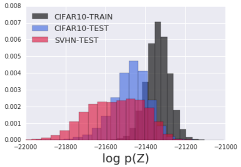
(b) **VAE**: FMNIST vs MNIST



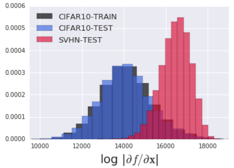
(c) **PixelCNN**: CIFAR-10 vs SVHN



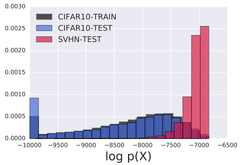
(d) **VAE**: CIFAR-10 vs SVHN



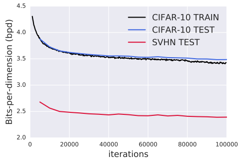
(a) CIFAR-10: $\log p(z)$



(b) CIFAR-10: Volume



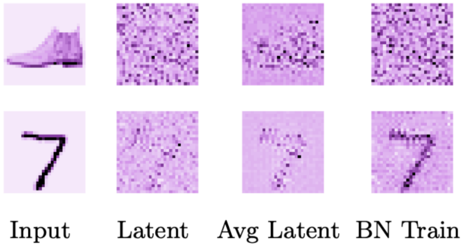
(c) CV-Glow Likelihoods

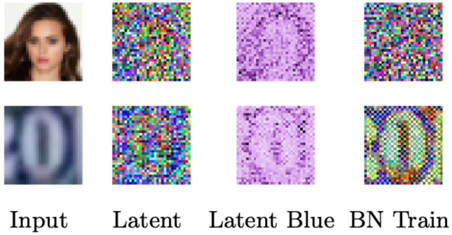


(d) Log-Likelihood vs Iter.

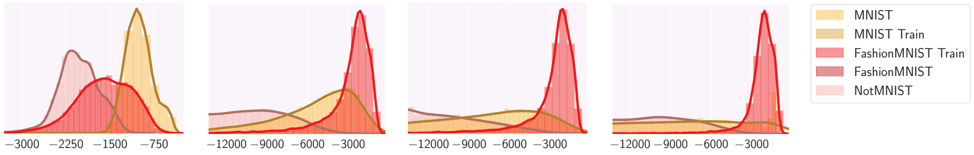
Why NFs Fail to Detect OOD Data [KIW20]

- Flows learn local pixel correlations rather than semantic properties.
- Paper studies masking in coupling layers (*st*-networks).
- Typical masks preserve local structure and do not learn global patterns.
- Proposed fixes:
  - Changing masking strategy to horizontal mask and cycle mask.
  - Introducing a bottleneck to learn global structure for reconstruction.
- Training flows on high-level semantic features enables OOD detection.

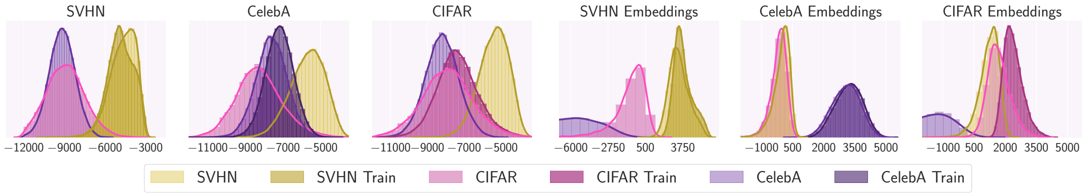




Why NFs Fail to Detect OOD Data [KIW20] (cont'd)



Baseline $\quad l = 100 \quad l = 50 \quad l = 10$



Identifying OOC Prediction Problems [MZ21]

- Progress in OOD detection is overestimated by using evidently OOD data in anomaly benchmarking setups.
- Unifying framework for Out-of-Context Prediction:
  1. Identify some existing auxiliary information C.
  2. Select a notion of OOC and define an OOC criterion by choosing a binary function ϕ of C.
  3. Restrict the test set to those examples where $\phi = 1$. Optionally, restrict the training set to examples where $\phi = 0$.
- Binary decision of determining object presence within a scene using COCO dataset.
- Two criterions:
  - Presence/absence of frequently co-occurring objects: measured using overlap of bounding boxes
  - Unusual scene gist: measured using BERT embeddings of image captions
- Enables creation of two types of samples:
  - Hard positives
  - Hard negatives

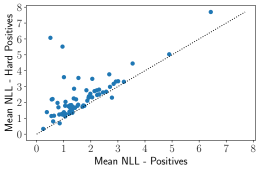
Identifying OOC Prediction Problems [MZ21] (cont'd)

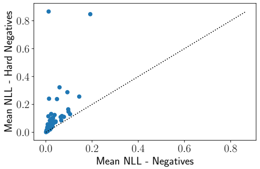
(a) hard positive (CE)

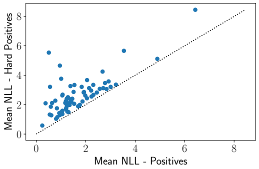
(b) hard positive (Gist)

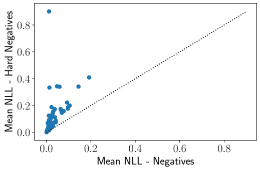
(c) hard negative (CE)

(d) hard negative (Gist)









Risk in Supervised Learning

Setup

- Dataset $D_p = \{(x_i, y_i)\}_{i=1}^N$ where $(x, y) \sim p$ over $\mathcal{D} = \mathcal{X} \times \mathcal{Y}$ with $x \in \mathcal{X}$, $y \in \mathcal{Y}$.
- Prediction function $h_\theta(x) : \mathcal{X} \to \mathcal{Y}$ producing labels $\hat{y} = h_\theta(x)$ with $h_\theta(\cdot) \in \mathcal{H}$.
- Loss function $\ell(\hat{y}, y)$ measuring prediction quality of $h_\theta(x)$.

Goal: By employing a learning algorithm $L : \mathcal{D} \to \mathcal{H}$ we want to produce a prediction function $h_\theta(\cdot)$ performing well on unseen test data $D'_p = \{(x_j, y_j)\}_{j=1}^M$, $(x, y) \sim p$, $D'_p \cap D_p = \varnothing$ as measured by our loss function $\ell(\cdot, \cdot)$.

(True) Risk

$$\mathcal{R}(h_\theta) := \mathbb{E}_{p(x,y)}[\ell(h_\theta(x), y)] = \int_{\mathcal{Y}} \int_{\mathcal{X}} p(x, y)\ell(h_\theta(x), y)dxdy$$

Empirical Risk Minimization (ERM)

$p(x, y)$ is typically not known or intractable to compute and as a result $\mathcal{R}(h_\theta)$ cannot be computed. But we can empirically approximate $\mathcal{R}(h_\theta)$ as $\hat{\mathcal{R}}(h_\theta)$ using samples from $p(x, y)$ (i.e. using D_p):

$$\mathcal{R}(h_\theta) := \mathbb{E}_{p(x,y)}[\ell(h_\theta(x), y)] \qquad \hat{\mathcal{R}}(h_\theta) := \frac{1}{N}\sum_{i=1}^{N} \ell(h_\theta(x_i), y_i)$$

Due to the law of large numbers we expect an increasingly better approximation of $\mathcal{R}(h_\theta)$ by $\hat{\mathcal{R}}(h_\theta)$ as more samples are provided to the learning algorithm L:

$$\hat{\mathcal{R}}(h_\theta) \approx \mathcal{R}(h_\theta) \qquad \hat{\mathcal{R}}(h_\theta) \xrightarrow{N\to\infty} \mathcal{R}(h_\theta) \qquad \arg\min_{h_\theta \in \mathcal{H}} \hat{\mathcal{R}}(h_\theta) \approx \arg\min_{h_\theta \in \mathcal{H}} \mathcal{R}(h_\theta)$$

The Problem of Distributional Shift

Revisiting our goal

Goal: By employing a learning algorithm $L : \mathcal{D} \to \mathcal{H}$ we want to produce a prediction function $h_\theta(\cdot)$ performing well on unseen test data $D'_p = \{(x_j, y_j)\}_{j=1}^M$, $(x, y) \sim p$, $D'_p \cap D_p = \varnothing$ as measured by our loss function $\ell(\cdot, \cdot)$.

A more realistic scenario

$$D'_q = \{(x_j, y_j)\}_{j=1}^M \qquad (x, y) \sim q,\ 0 \leq d(p, q) \leq \delta \qquad D'_q \cap D_p = \varnothing$$

$d(p, q)$ is a divergence measure between training distribution p and testing distribution q and is bounded by δ.

Distributionally Robust Optimization (DRO)

Instead of optimizing for good test time performance on the original distribution p, we optimize for good test time performance under the worst possible shift:

$$\mathcal{R}(h_\theta, q) := \max_{q \in \mathcal{Q}_p} \mathbb{E}_{q(x,y)}[\ell(h_\theta(x), y)]$$

$\mathcal{Q}_p$ is called our uncertainty set of data distributions. We constrain distribution realizations $q \in \mathcal{Q}_p$ to be absolutely continuous and bounded in divergence wrt p:

$$\mathcal{Q}_p = \{q \ll p \mid d(p, q) \leq \delta\}$$

Risk Minimization vs Distributionally Robust Optimization

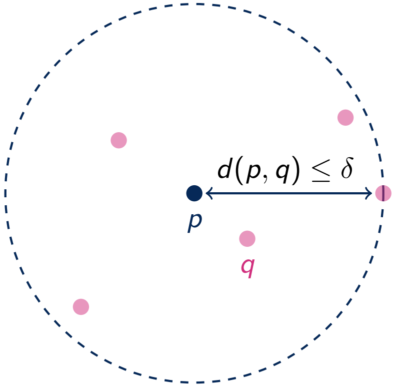
Risk Minimization

$$\arg\min_{h_\theta \in \mathcal{H}} \mathbb{E}_{p(x,y)}[\ell(h_\theta(x), y)]$$

Distributionally Robust Optimization

$$\arg\min_{h_\theta \in \mathcal{H}} \max_{q \in \mathcal{Q}_p} \mathbb{E}_{q(x,y)}[\ell(h_\theta(x), y)]$$

$$\mathcal{Q}_p = \{q \ll p \mid d(p,q) \leq \delta\}$$



Important: The distribution q that leads to the worst-case DRO loss does not necessarily correspond to be the distribution that maximizes $d(p,q)$!

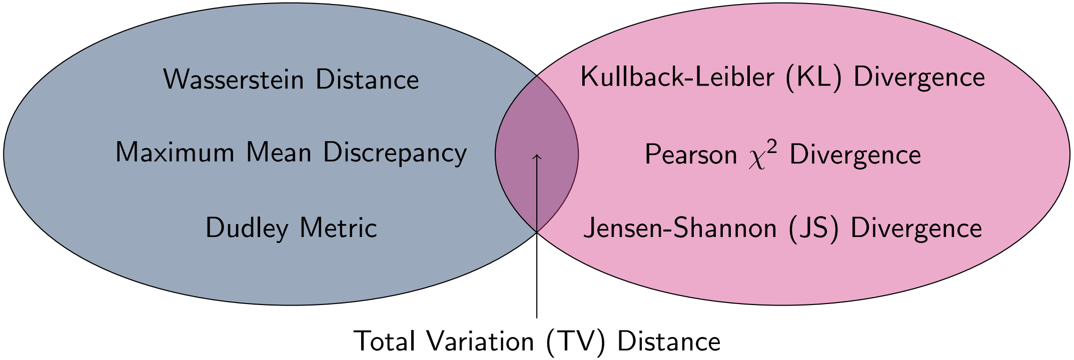
Divergences Between Probability Distributions

Integral Probability Metrics: $p - q$

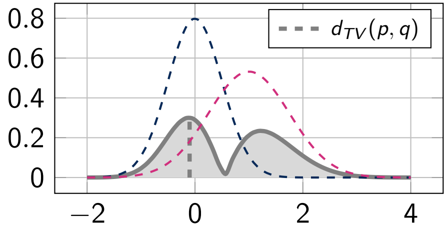
ϕ-divergences (f-divergences): $\frac{p}{q}$

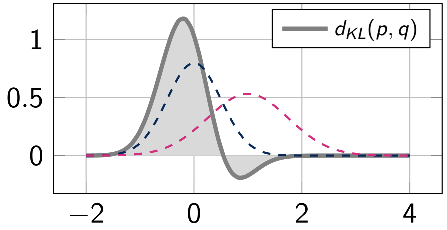
$d_{\mathcal{F}}(p, q) = \sup_{g \in \mathcal{F}} |\mathbb{E}_{X \sim p}[g(X)] - \mathbb{E}_{X' \sim q}[g(X')]|$

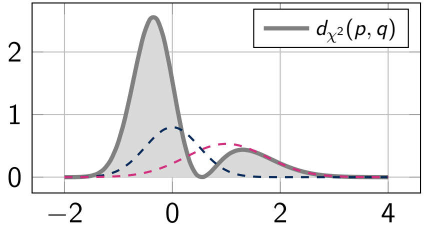
$d_{\phi}(p, q) = \int_{\mathcal{X}} q(x) \phi(\frac{p(x)}{q(x)}) dx$

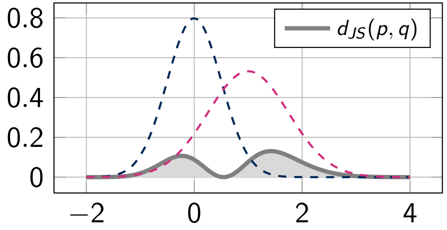

Wasserstein Distance
Maximum Mean Discrepancy
Dudley Metric
Kullback-Leibler (KL) Divergence
Pearson χ^2 Divergence
Jensen-Shannon (JS) Divergence
Total Variation (TV) Distance

ϕ-divergences: Choices for $\phi(\cdot)$









Application: ERM Generalization and Regularization

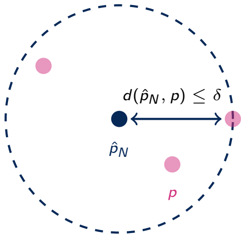
Recall the ERM definition:

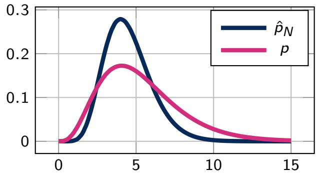
$$\hat{\mathcal{R}}_\lambda(h_\theta) := \frac{1}{N}\sum_{i=1}^{N} \ell(h_\theta(x_i), y_i) + \underbrace{\lambda\Omega(\theta)}_{\text{regularizer}}$$

By regularizing, we reduce overfitting on the sample distribution $\hat{p}_N$ and enable generalization to p.

Different divergences lead to different regularization:

- χ^2 penalizes $\mathbb{V}_{\hat{p}_N}[\ell(h_\theta(x), y)]$
- Wasserstein penalizes $||\nabla_x \ell(h_\theta(x), y)||$
- MMD penalizes $||\ell(h_\theta(x), y)||_{\mathcal{F}}$



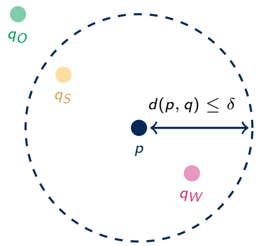


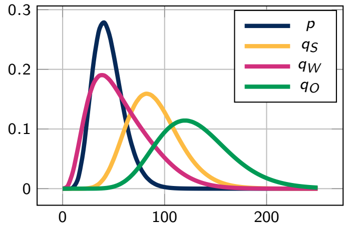
Application: Distribution Shifts in General

Example setting: You are building a predictive model for house prices based on square meters.

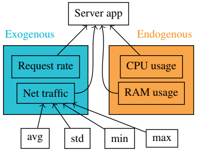
- p: square meters distribution in the inner city
- q_S: square meters distribution in the city's **s**uburbs
- q_W: square meters distribution in the **w**hole city
- q_O: square meters distribution of an**o**ther city

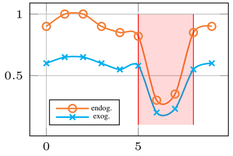
Goal: Generalize to the worst-case distribution within the city, i.e., q_S and q_W, but not to q_O.

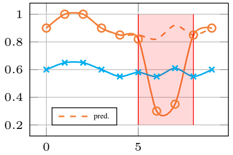


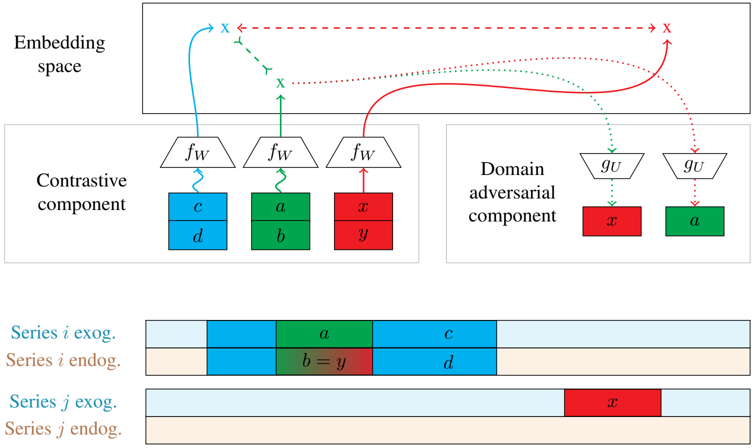


Unsupervised Contextual Anomaly Detection for Time Series



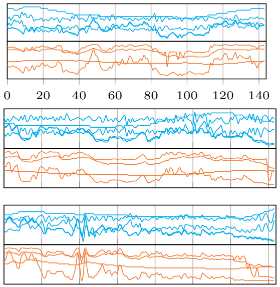


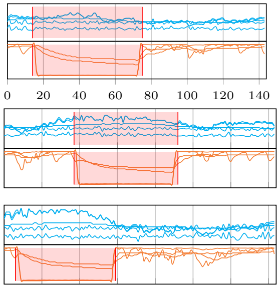


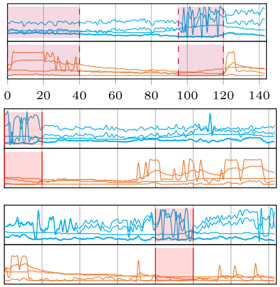


Unsupervised Contextual Anomaly Detection for Time Series (cont'd)

	BasicEmb	ResEmbRegr	ContInvEmb	ResTresh	Catch22	ResCatch22
Synthetic	0.512 (± 0.022)	**1.000** (± 0.000)	0.999 (± 0.002)	**1.000** (± 0.000)	0.494 (± 0.008)	**1.000** (± 0.000)
Pendulum	0.969 (± 0.013)	0.951 (± 0.015)	**0.980** (± 0.002)	0.510 (± 0.000)	0.904 (± 0.000)	0.891 (± 0.000)
DevOps	0.535 (± 0.041)	0.532 (± 0.036)	0.587 (± 0.007)	**0.619** (± 0.000)	0.573 (± 0.000)	0.573 (± 0.000)
Turbine	0.632 (± 0.015)	0.725 (± 0.018)	0.736 (± 0.022)	**0.845** (± 0.000)	0.512 (± 0.000)	0.680 (± 0.000)


0
20
40
60
80
100
120
140


0
20
40
60
80
100
120
140


0
20
40
60
80
100
120
140