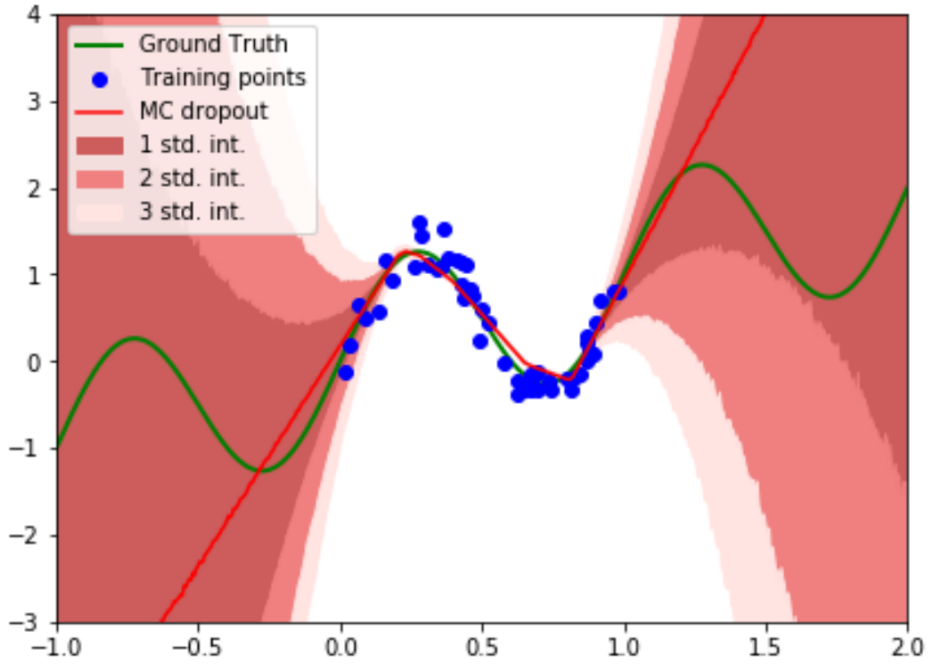
Sorbonne Université, Computer Science Master Données, Apprentissage et Connaissances (DAC)
Bayesian Deep Learning

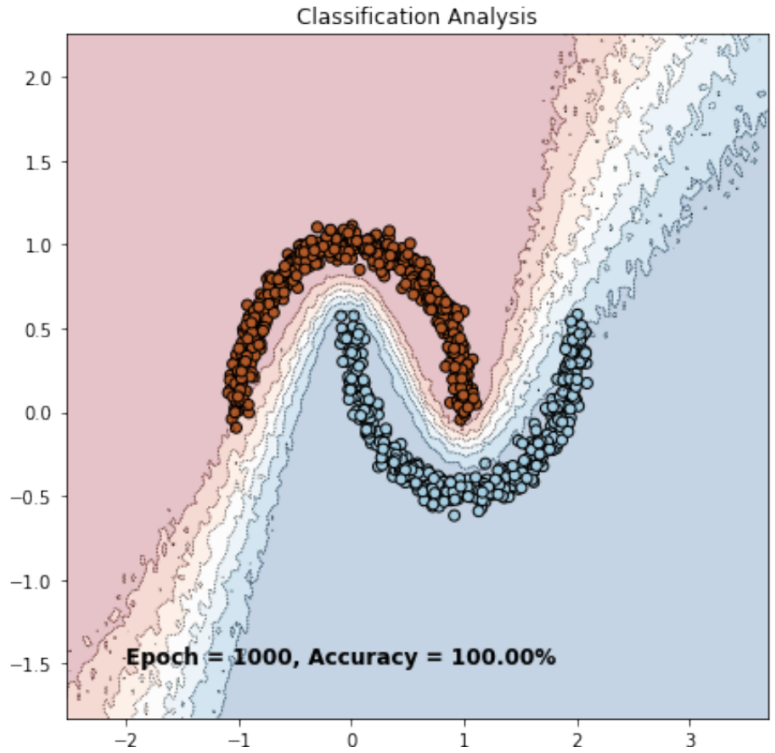
Nicolas Thome
Conservatoire National des Arts et Métiers (Cnam)
Laboratoire CEDRIC - équipe MSDMA

Outline

Recap: MCDropout [Gal and Ghahramani, 2016a]

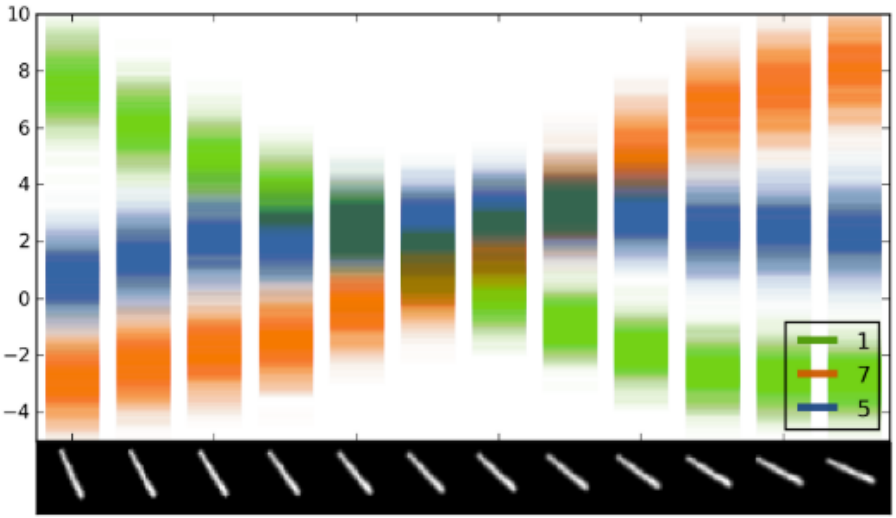
- Dropout: variational Bayesian NN approximation with particular prior and approximate posterior
- Consequence: MC Dropout sampling: approximating predictive distribution
  - Application for regression and binary classification on toy examples



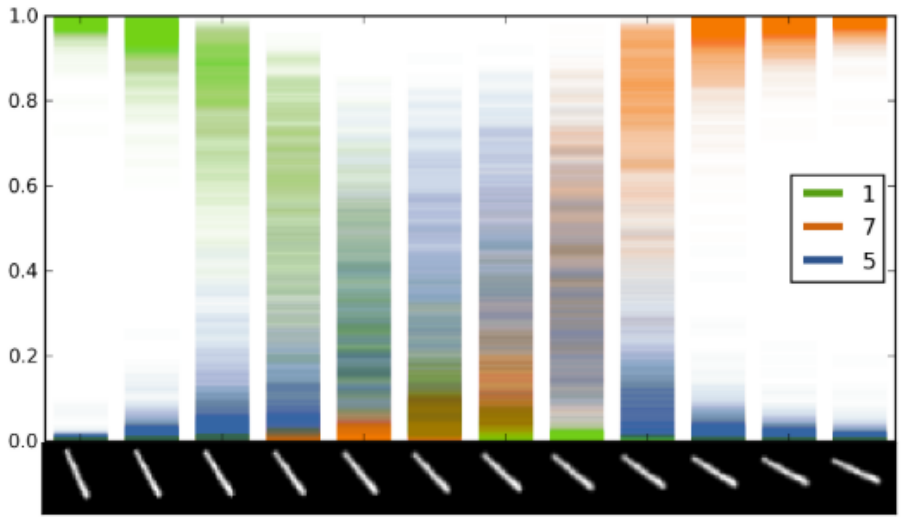


Applications: Classification

- Use of uncertainty in classification
  - Qualitative expe to illustrate importance of predictive distribution sampling
  - MNIST, LeNet CNN with **dropout only on last fc layer**, dropout probabilities $p = 0.5$, SGD with $LR = 0.01$ updated using momentum 0.9, weight decay $1e^{-06}$, $T = 100$ forward passes



(a) Softmax *input* scatter

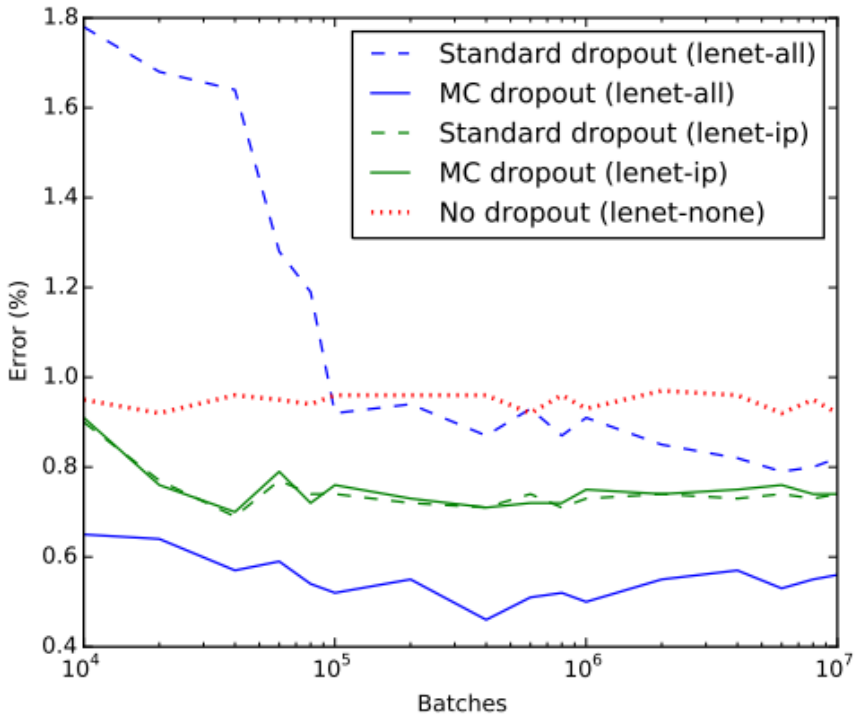


(b) Softmax *output* scatter

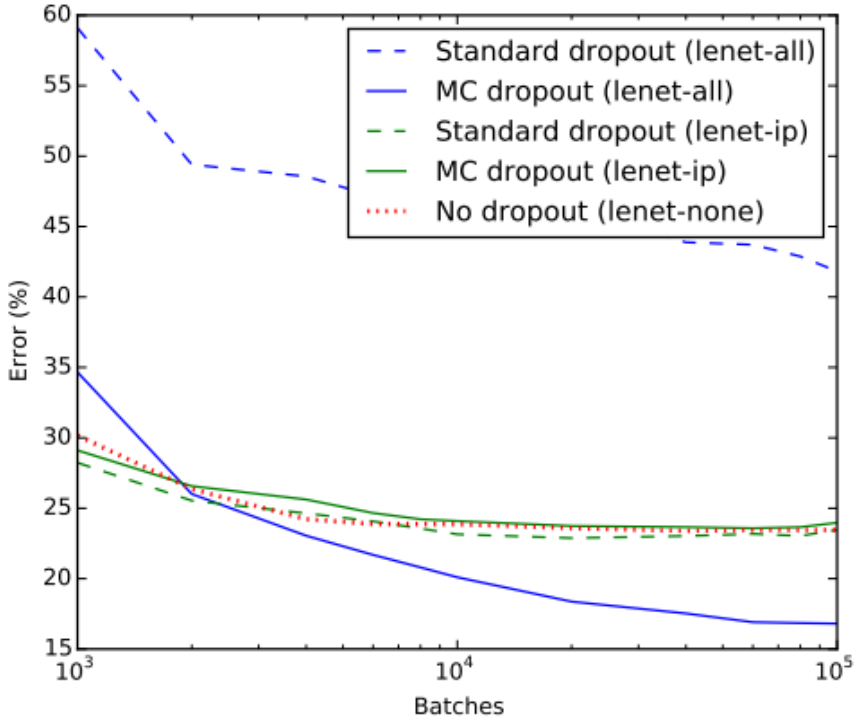
Pred: [1 1 1 1 1 5 5 7 7 7 7 7]

Applications: Classification

- Performances: MC dropout *vs* dropout [Gal and Ghahramani, 2015]
- Test error with LeNet, $T = 50$ forward passes
- Take-home message: dropout only propagates the mean; MC dropout propagates whole distribution



(a) MNIST



(b) CIFAR-10

Model uncertainty in classification

To summarize uncertainty, three approaches

1. **variation ratios**
   - For each stochastic forward pass $t \in \{1; T\}$, compute label from softmax probabilities
   - c^*: most frequent label over the T passes, with frequency $f_{\boldsymbol{x}}^{c^*}$
   - Compute variation-ratio var-ratio$[\boldsymbol{x}] = 1 - \frac{f_{\boldsymbol{x}}^{c^*}}{T}$
2. **predictive entropy**: captures the average amount of information contained in the predictive distribution.

$$\hat{\mathcal{H}}[y|x, \mathcal{D}_{train}] = -\sum_c \left(\frac{1}{T}\sum_t p(y = c|x, \hat{w}_t)\right) \log\left(\frac{1}{T}\sum_t p(y = c|x, \hat{w}_t)\right)$$

3. **mutual information** : maximise the mutual informations are points on which the model is uncertain on average

$$\hat{\mathcal{I}}[y, w|x, \mathcal{D}_{train}] = \hat{\mathcal{H}}[y|x, \mathcal{D}_{train}] + \frac{1}{T}\sum_{c,t} p(y = c|x, \hat{w}_t) \log p(y = c|x, \hat{w}_t)$$

Model uncertainty in classification

Let's see three concrete examples:

- **all equal to 1** (*i.e.* the probability vectors collected are $\{(1,0),...,(1,0)\}$)
  → *high pred confidence, low model uncertainty*

$$\text{var-ratio} = \hat{\mathcal{H}}[y|x,\mathcal{D}_{train}] = \hat{\mathcal{I}}[y,w|x,\mathcal{D}_{train}] = 0$$

- **all equal to 0.5** (*i.e.* the probability vectors collected are $\{(0.5,0.5),...,(0.5,0.5)\}$) → *low pred confidence, low model uncertainty*

$$\text{var-ratio} = \hat{\mathcal{H}}[y|x,\mathcal{D}_{train}] = 0.5$$

$$\hat{\mathcal{I}}[y,w|x,\mathcal{D}_{train}] = 0$$

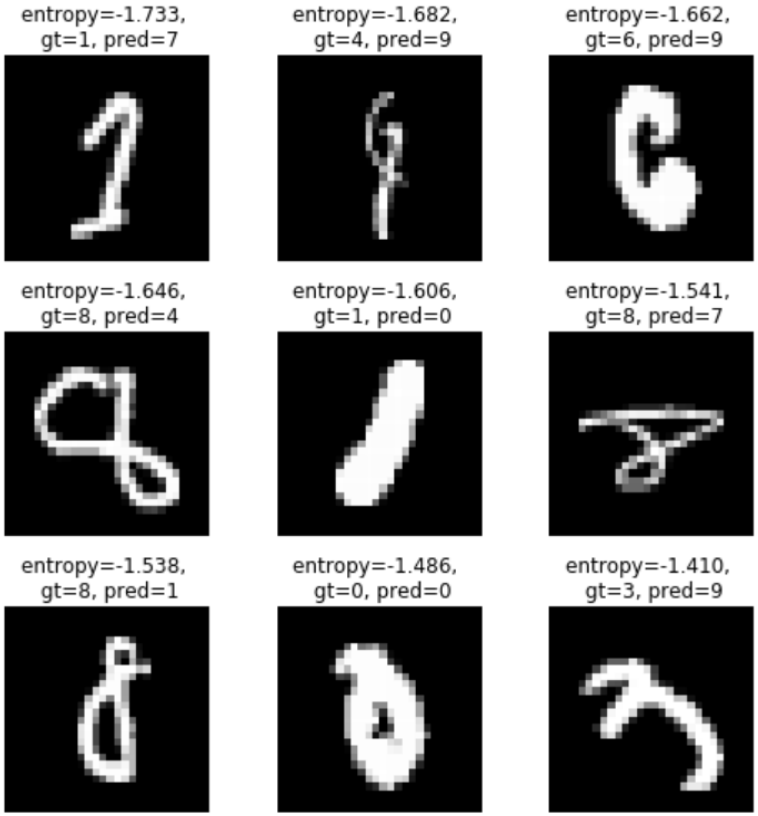
- **half-half**, *i.e.* probability vectors collected are $\{(1,0),(0,1),(0,1),...,(1,0)\}$
  → *low pred confidence, high model uncertainty*

$$\text{var-ratio} = \hat{\mathcal{H}}[y|x,\mathcal{D}_{train}] = \hat{\mathcal{I}}[y,w|x,\mathcal{D}_{train}] = 0.5$$

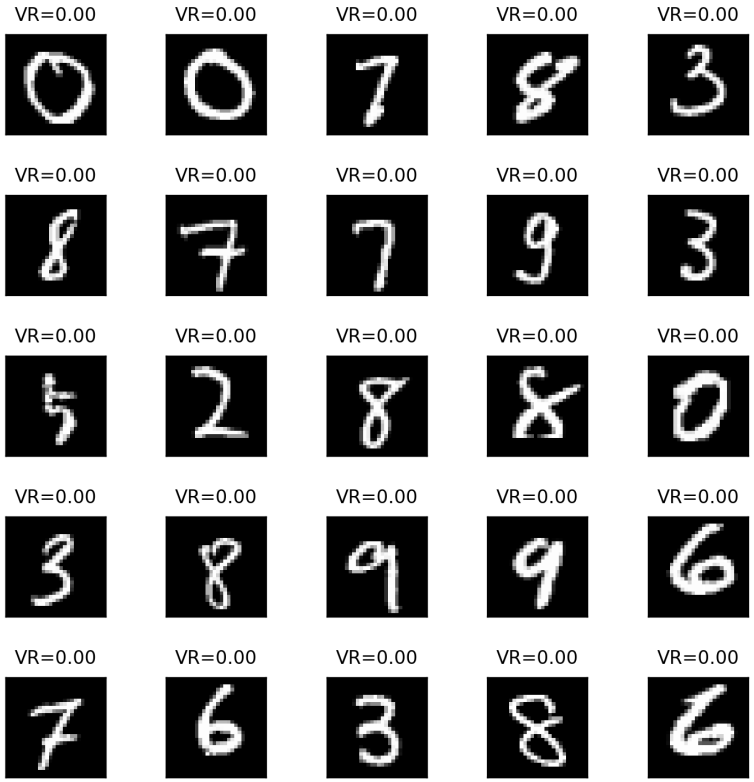
Practical Session: MC dropout on MNIST for classification

- MC dropout regularization on MNIST
  - Training a convnet with dropout ⇒ improved training performances
  - Improved Training performances with activating dropout at test time
- Use variation ratio to measure uncertainty

Most incertain test set examples



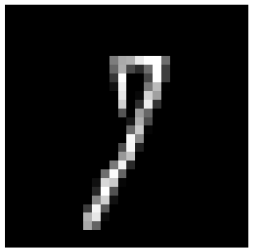
Least incertain test set examples

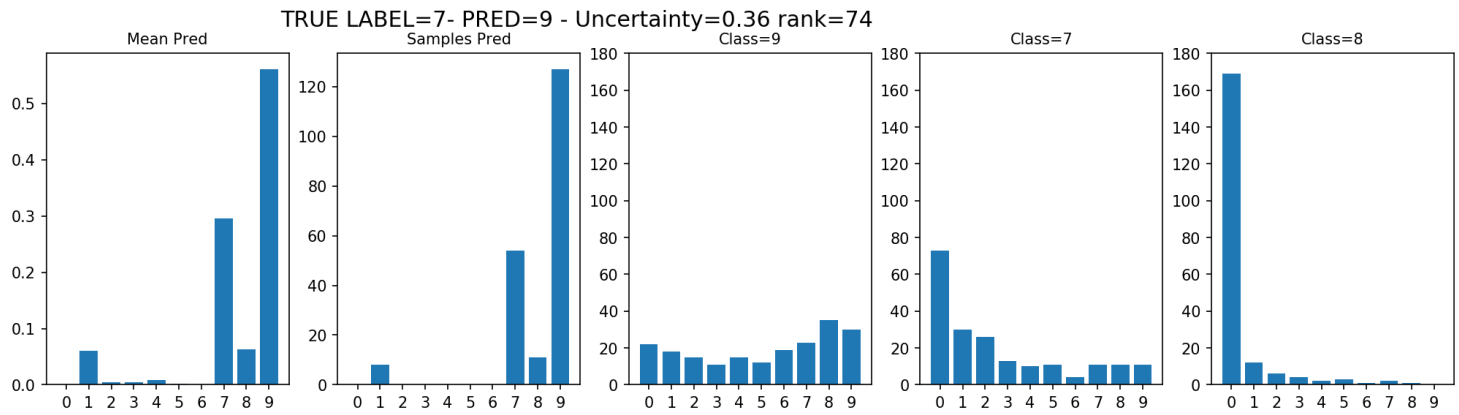


Practical Session: MC dropout on MNIST for classification

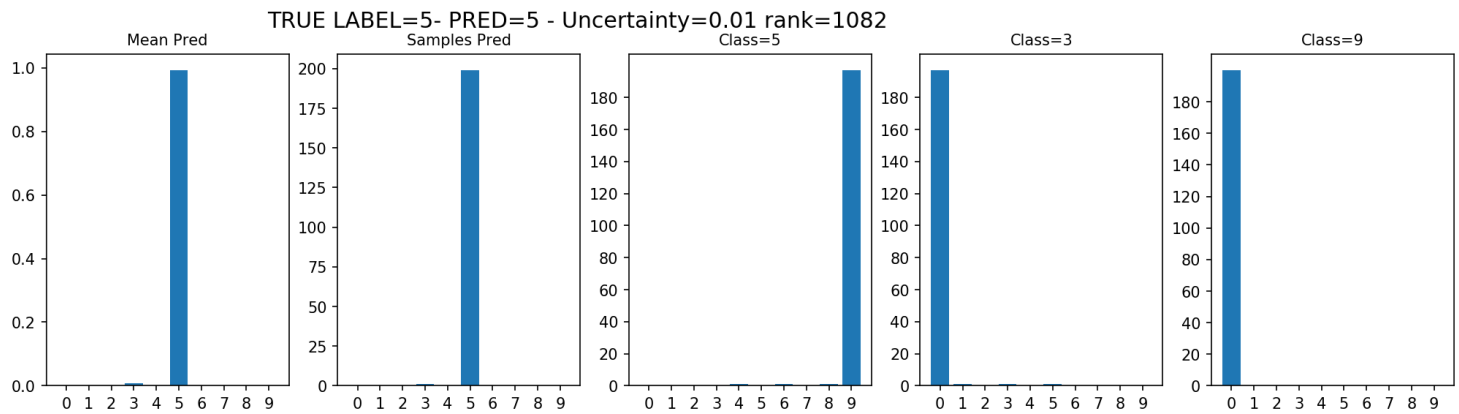
- Analyse MC sampling

Incertain example



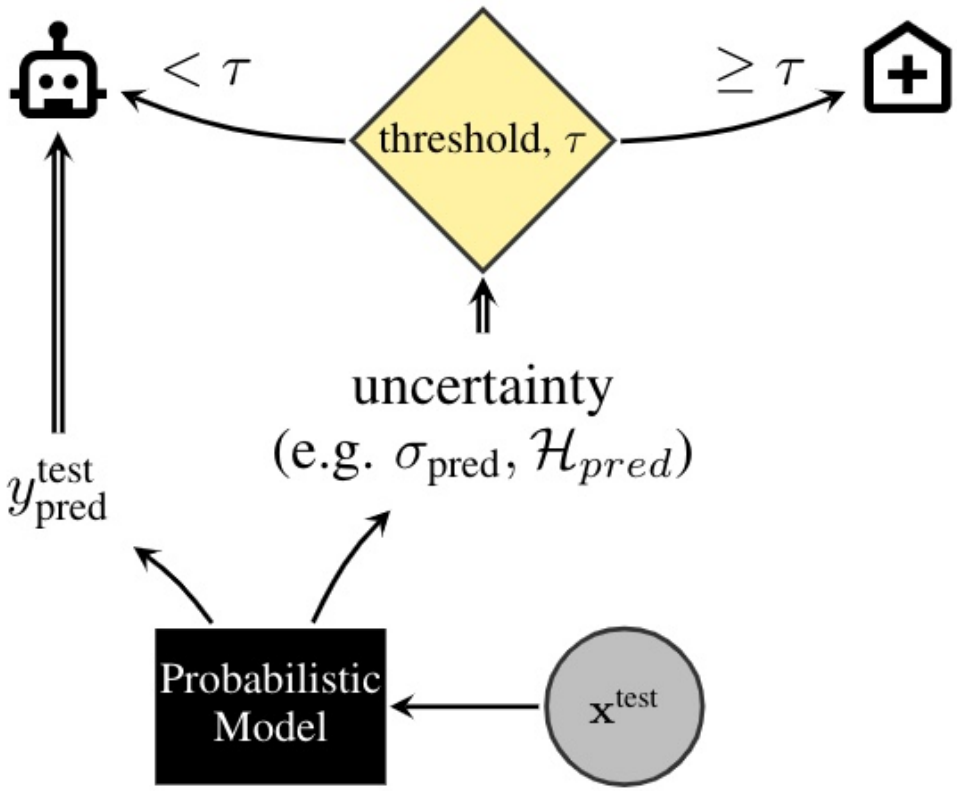


Confident example



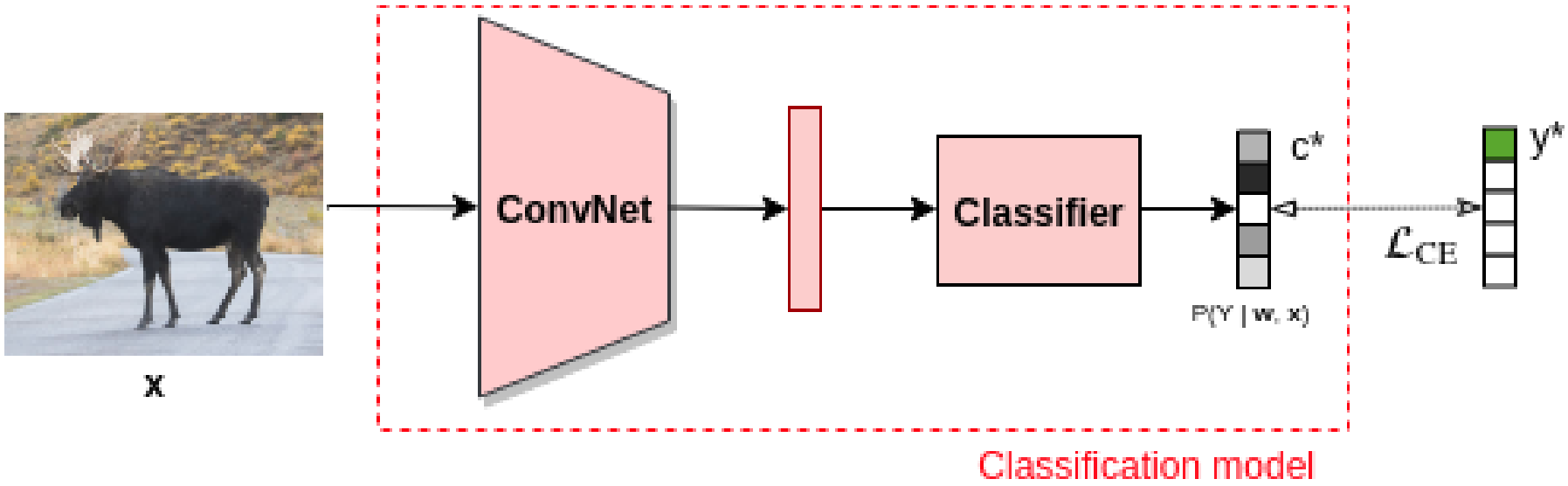
Applications: Failure Prediction

- Detecting failure of a system crucial in practice
- Use uncertainty estimate to accept / reject predictions



Failure Prediction: Model Calibration in Deep Learning

- Classification model trained on $\mathcal{D} = \{(\boldsymbol{x}_i, y_i^*)\}_{i=1}^N$



- Model prediction: $\hat{\boldsymbol{y}} = \arg\max_{k \in \mathcal{Y}} p(Y = k | \boldsymbol{w}, \boldsymbol{x})$
- Model confidence $\hat{C}(\boldsymbol{x})$:
  - Simple baseline for deep neural networks: $\boldsymbol{MCP}(\boldsymbol{x}) = \max_{k \in \mathcal{Y}} p(Y = k | \boldsymbol{w}, \boldsymbol{x})$
  - More advanced methods, *e.g.* MC dropout for classification
- **Threshold confidence (uncertainty) estimate to accept / reject predictions**

Model Calibration

- Test **Data**: $\mathcal{D} = (X, Y) = \{(x_1, y_1), ..., (x_N, y_N)\}$
- $(\hat{y}_i, \hat{C}(\boldsymbol{x}_i))$ class prediction and confidence level
- Perfect calibration:

$$p\big(\hat{Y} = Y | \hat{C} = p\big) = p, \quad \forall p \in [0, 1]$$

- Predicted confidence match actual accuracy
  - *e.g. given 100 predictions, each with confidence of 0.8, we expect that 80 should be correctly classified.*
  - Link to thresholding probabilities for failure prediction: non-calibrated probabilities $\Rightarrow$ arbitrary threshold!

Model Calibration in deep learning

Reliability Diagram

M interval bins.

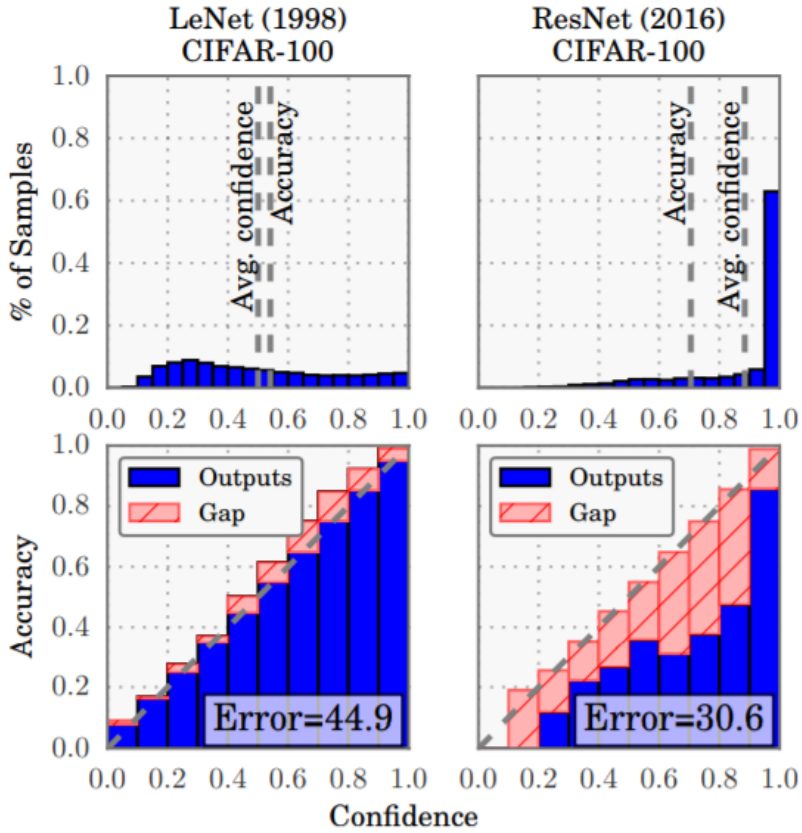
B_m set of samples whose predictions are in $I_m = (\frac{m-1}{M}, \frac{m}{M}]$

$$acc(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} \mathbb{1}(\hat{y}_i = y_i)$$

$$conf(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} \hat{p}_i$$

perfect calibration

$\iff acc(B_m) = conf(B_m) \quad \forall m$



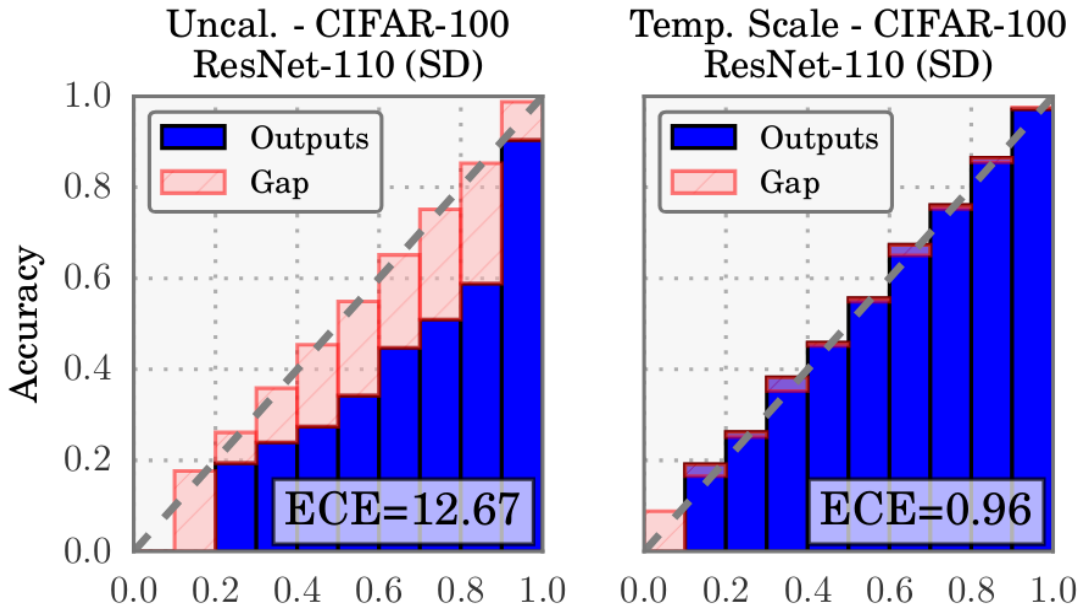
[Guo et al., 2017] **showed that modern neural networks are no longer well-calibrated!**

Model Calibration in deep learning

- Simple solution to over-confident prediction: temperature scaling [Guo et al., 2017]

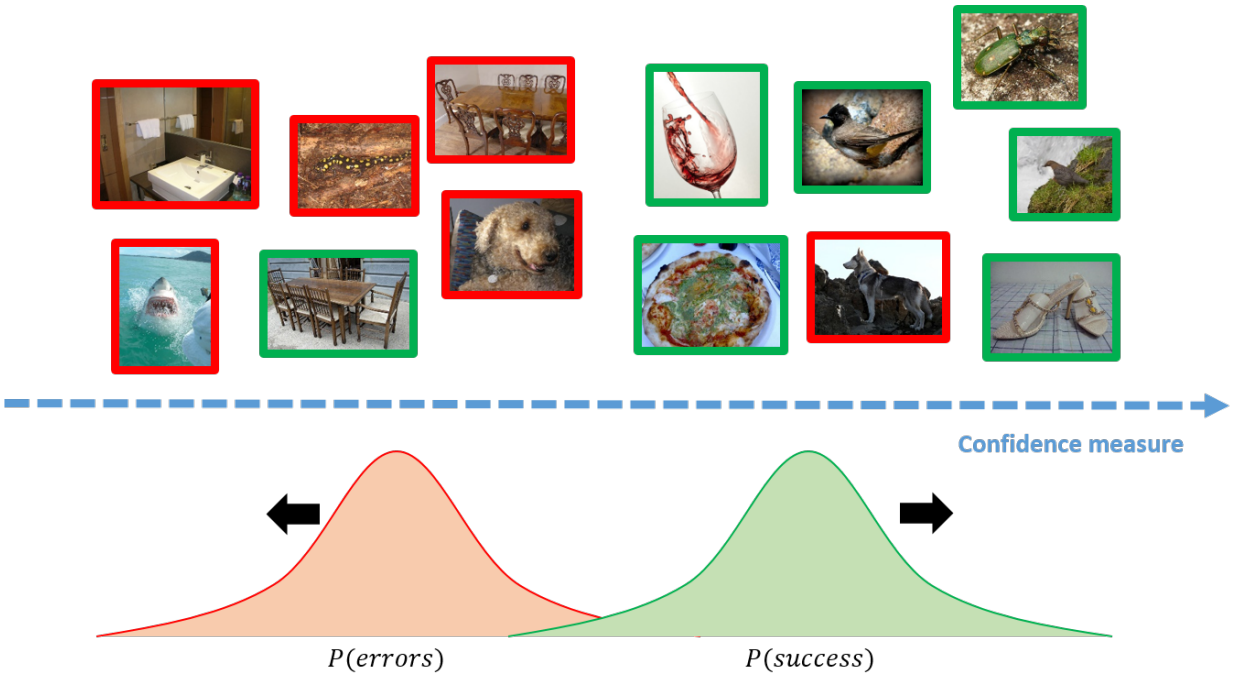
$$P(\hat{y}_k) = \frac{e^{s_k/T}}{\sum_{k'=1}^{K} e^{s_{k'}/T}}$$

- temperature T optimized on val set s.t. $acc(B_m) = conf(B_m)$



Failure Prediction in deep learning

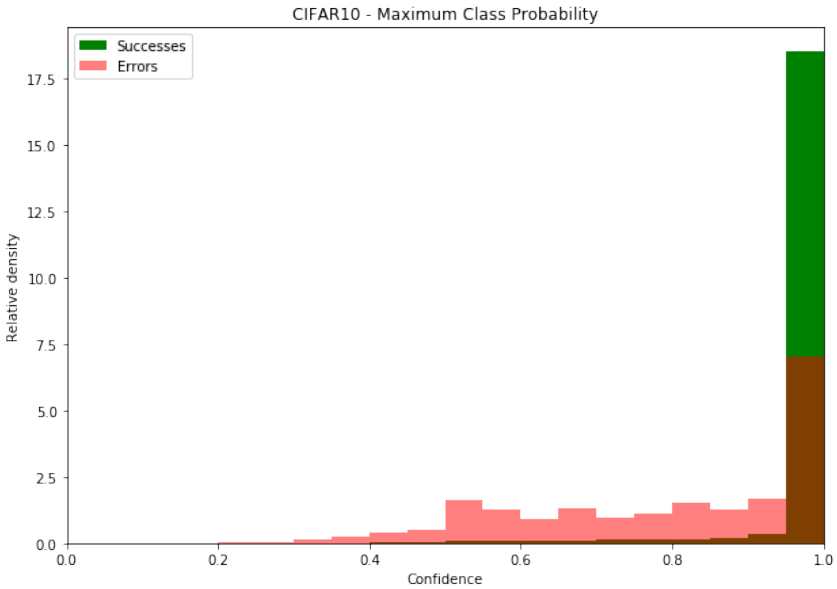
- Confidence estimate $\hat{C}(\boldsymbol{x}_i)$ goal: **distinguish correct from erroneous predictions**

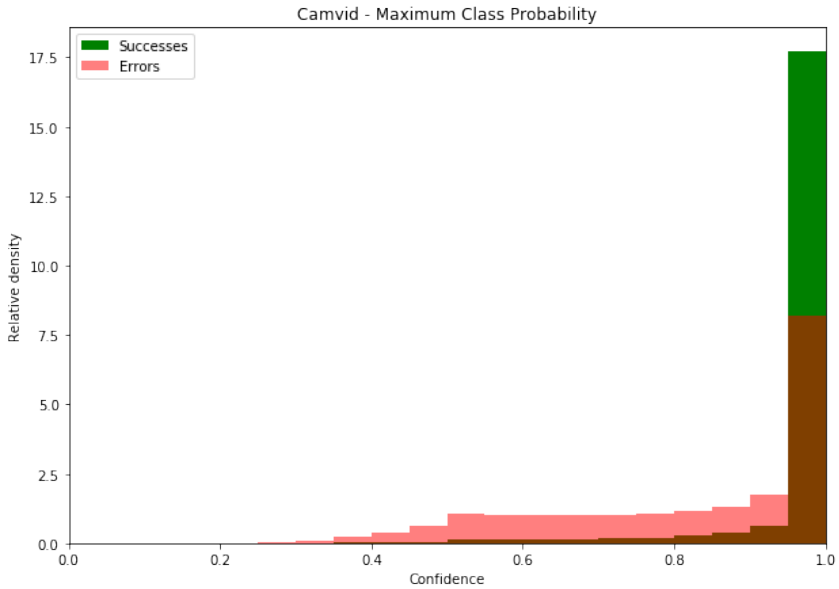


- Sort examples wrt $\hat{C}(\boldsymbol{x}_i)$
  - Evaluate capacity of $\hat{C}$ to assign larger prediction values for correct predictions than for errors

Failure Prediction

- MCP: $\boldsymbol{MCP}(\boldsymbol{x}) = \max_{k \in \mathcal{Y}} p(Y = k | \boldsymbol{w}, \boldsymbol{x})$ unreliable confidence criterion
  - For failure prediction: by design assign largest probability

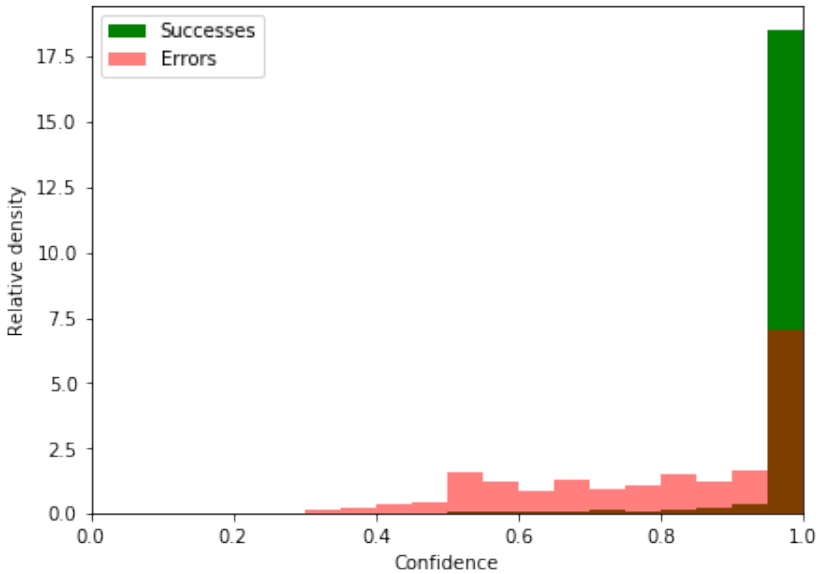




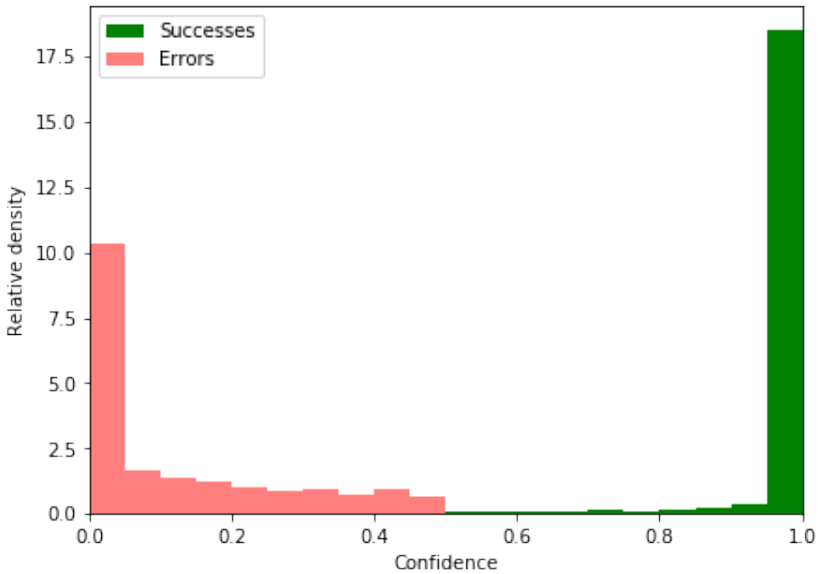
- **overlapping distributions** between successes vs. errors
  $\Rightarrow$ hard to distinguish

Failure Prediction with TCP [Corbière et al., 2019]

- True Class Probability (TCP): $\boldsymbol{TCP}(\boldsymbol{x}, y^*) = p(Y = y^* | \boldsymbol{w}, \boldsymbol{x})$
  unreliable confidence criterion
  - For failure prediction: assign lower probability for errors



(a) Maximum Class Probability

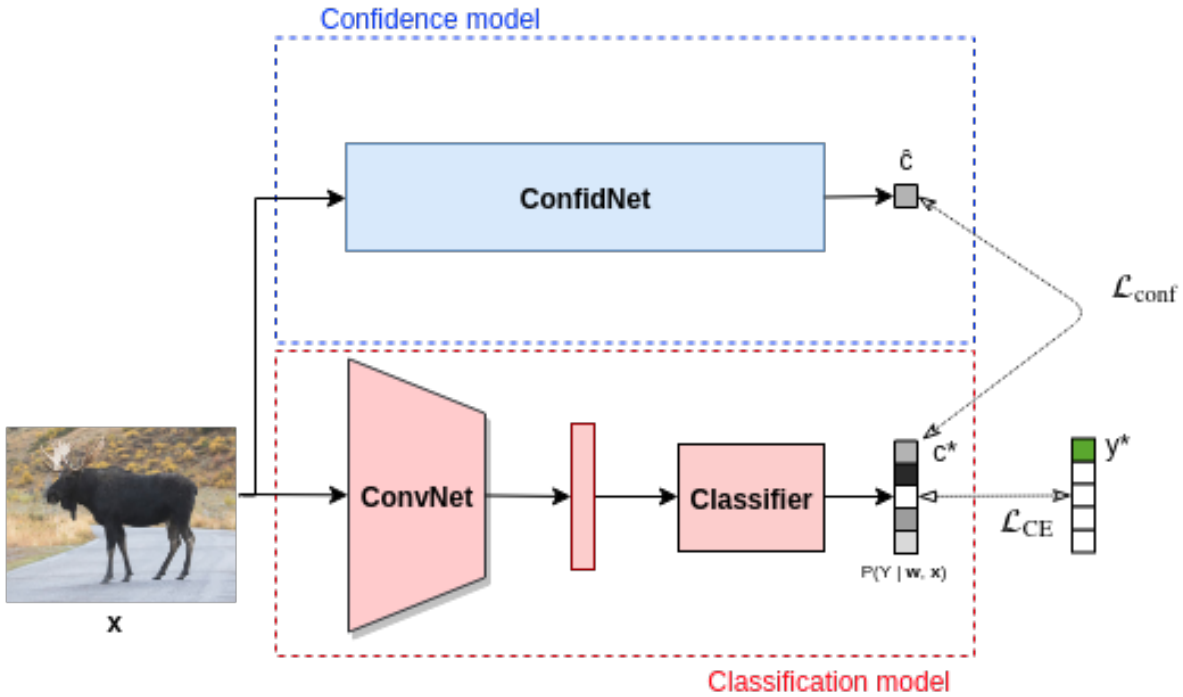


(b) Our Proposal (True Class Probability)

ConfidNet [Corbière et al., 2019]

- However, $TCP(\boldsymbol{x}, y^*)$ is **unknown** at test time.
- ConfidNet [Corbière et al., 2019]: Learning TCP Model Confidence

Given $\mathcal{D}_{train}$, **learn a confidence model** with parameters θ such that $\forall \boldsymbol{x} \in \mathcal{D}_{train}$, its scalar output $\hat{c}(\boldsymbol{x}, \theta)$ close to $TCP(\boldsymbol{x}, y^*)$

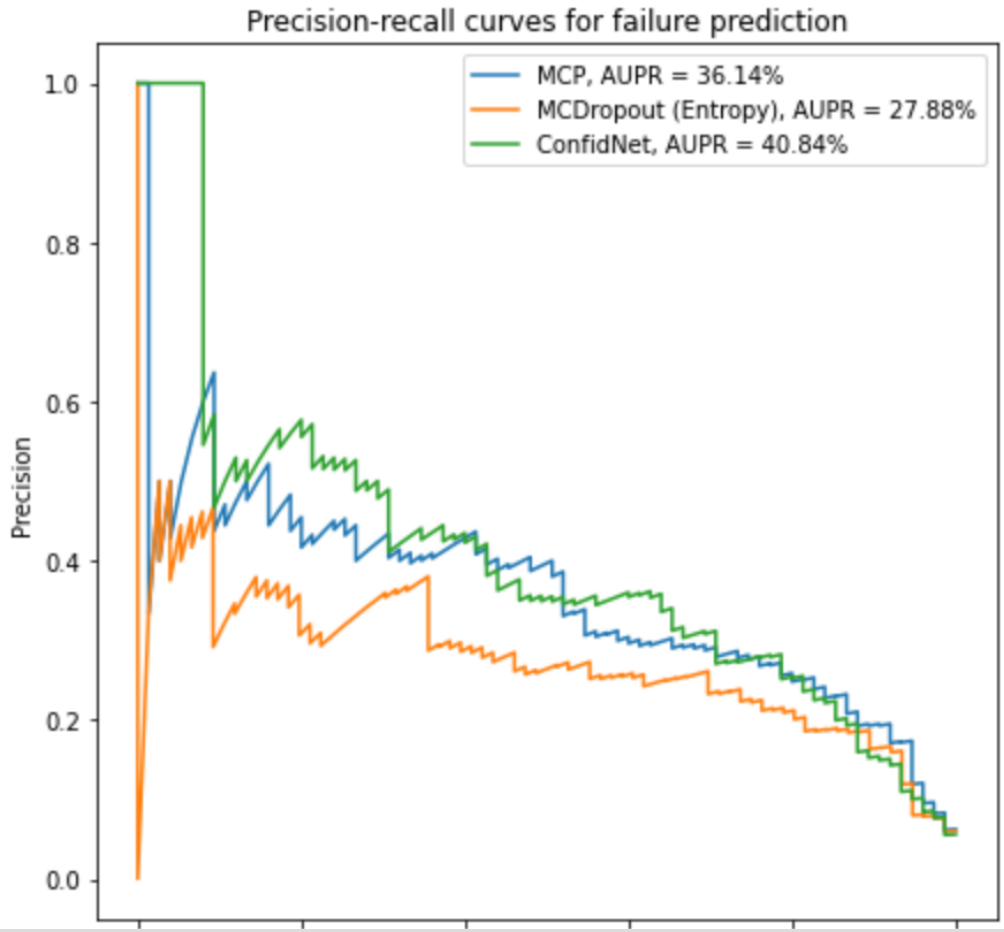


As $TCP(\boldsymbol{x}, y^*) \in [0, 1]$, we propose ℓ_2 loss to train ConfidNet:

$$\mathcal{L}_{\boldsymbol{conf}}(\theta; \mathcal{D}) = \frac{1}{N} \sum_{i=1}^{N} (\hat{c}(\boldsymbol{x}_i, \theta) - c^*(\boldsymbol{x}_i, y_i^*))^2$$

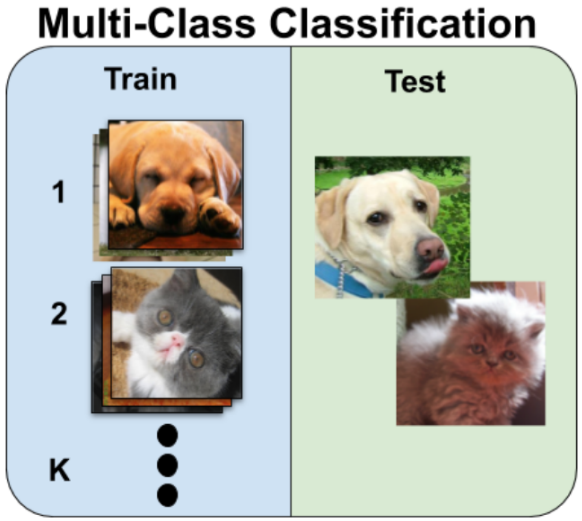
Practical Session: Failure Prediction on MNSIT

- Compute difference confidence estimate $\hat{C}$ on MNIST test data
  - MCP $\boldsymbol{MCP}(\boldsymbol{x}) = \max_{k \in \mathcal{Y}} p(Y = k | \boldsymbol{w}, \boldsymbol{x})$, MC dropout, ConfdNet
- Compare confidence criterion quality
  - Rank test examples wrt uncertainty criterion
  - Compute Precision/Recall (PR) curve and AP_{errors}

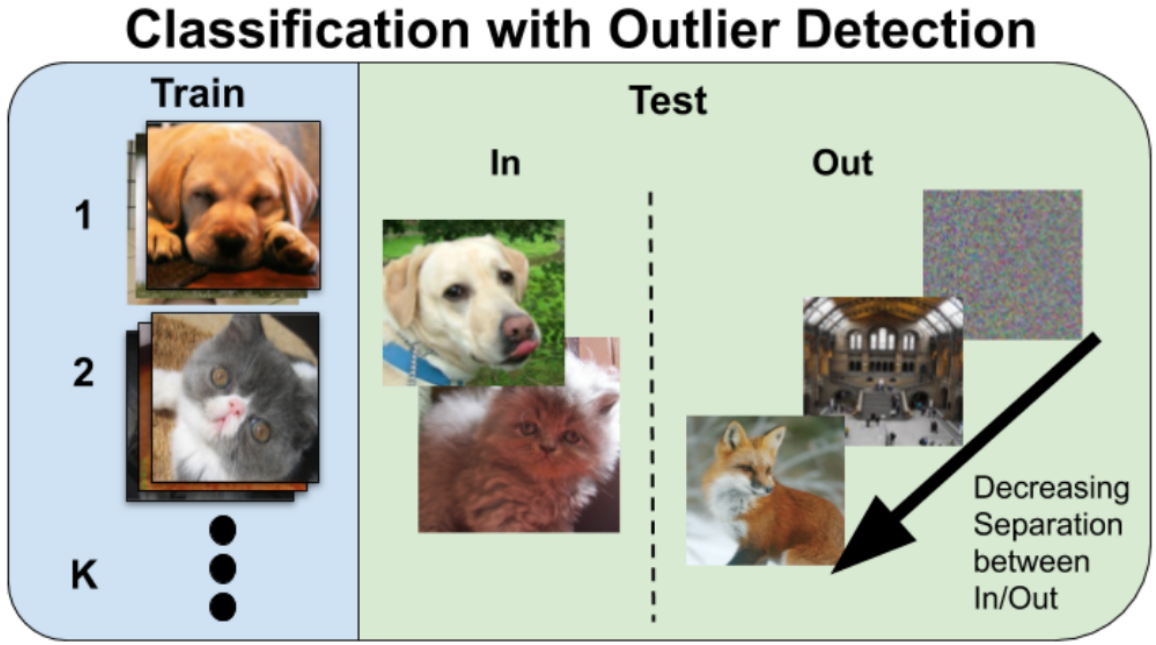


Applications: Out of Distribution Detection

- Standard classification: same classes during train and test

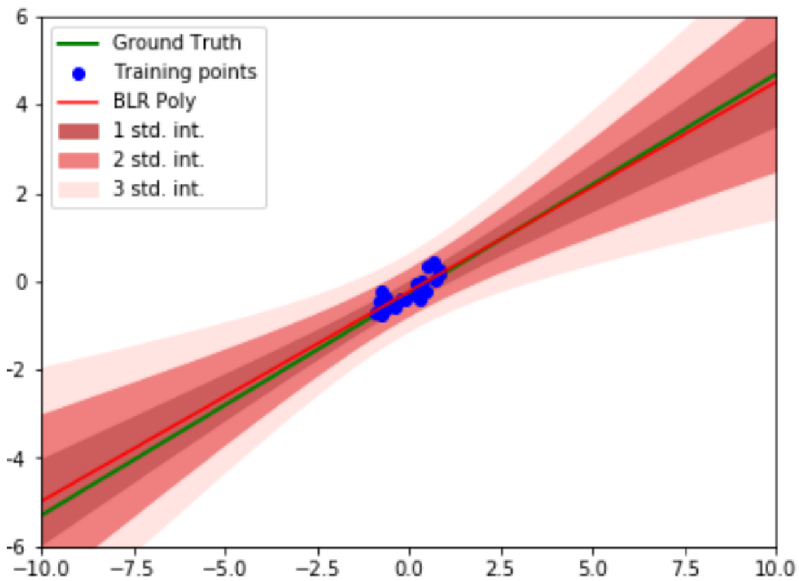


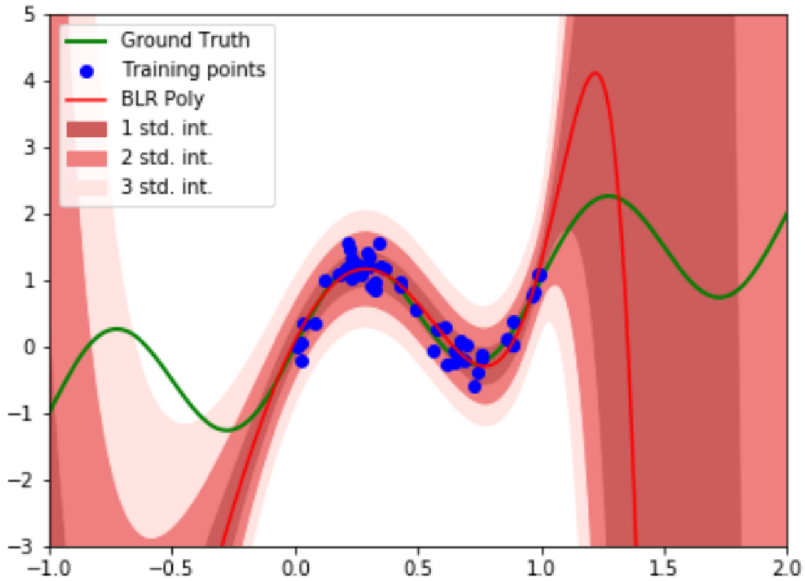
- Out of Distribution Detection: detecting unknown classes, far from training distribution



Out of Distribution Detection

- Detecting examples far from training distribution
  $\Rightarrow$ natural use of Bayesian confidence estimates

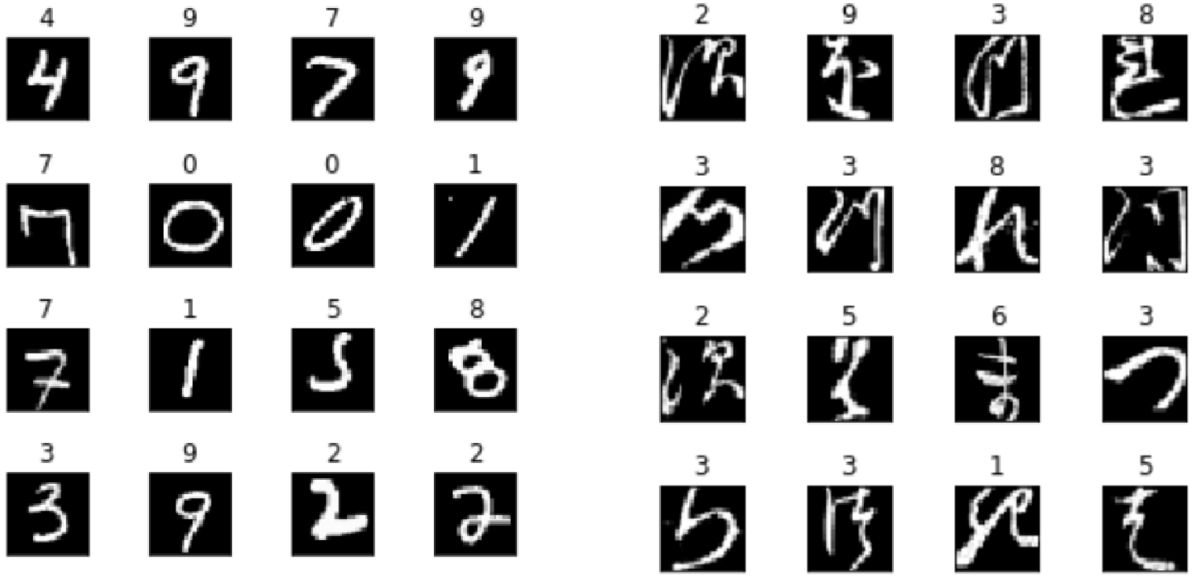




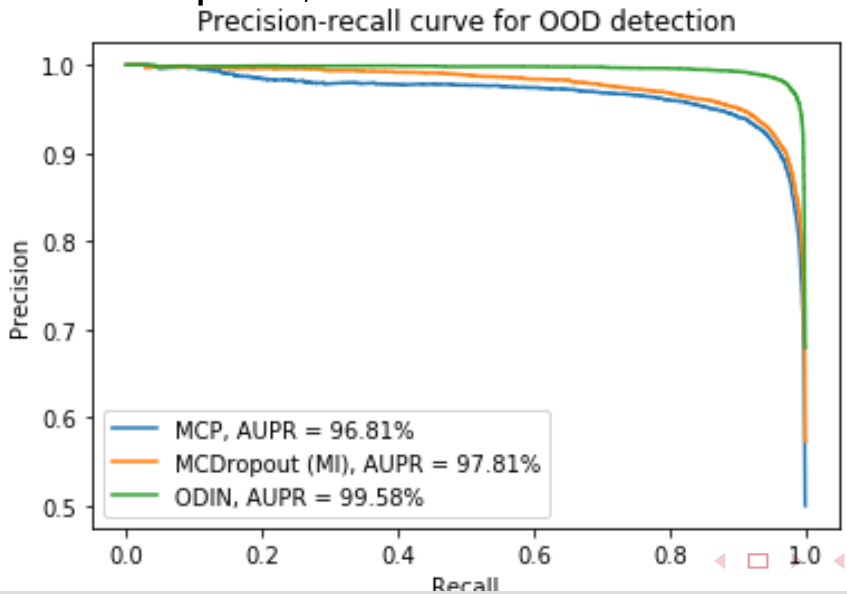
- Other methods specifically learn dataset-specific OOD methods
  - Ex: ODIN [Liang et al., 2017]

Out of Distribution Example

- Model trained on MNIST - Out: Kuzushiii-MNIST

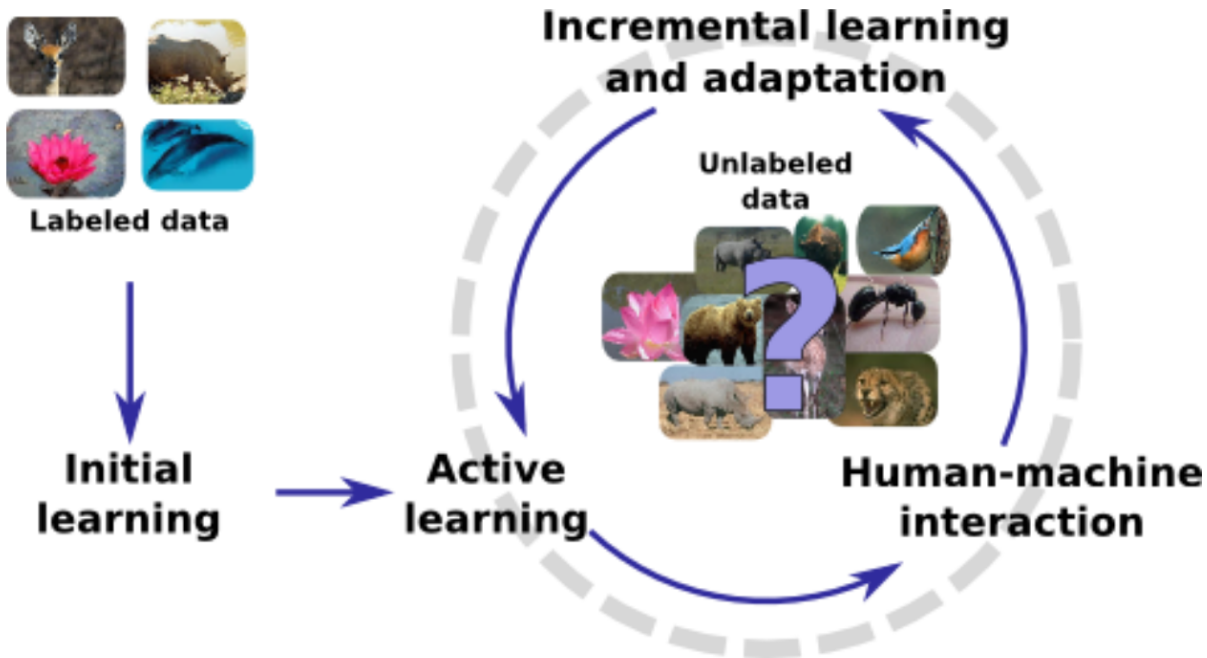


- Comparing MCP, MC dropout, ODIN for OOD detection



Applications: Active Learning

- Active/ interactive learning: learning a model with few data annotated by users
- Challenge: determining most informative data to annotate
  - Most uncertain data: optimal convergence [Tong and Koller, 2002]



Applications: Active Learning

- Use of uncertainty in classification: active learning [Gal et al., 2017]

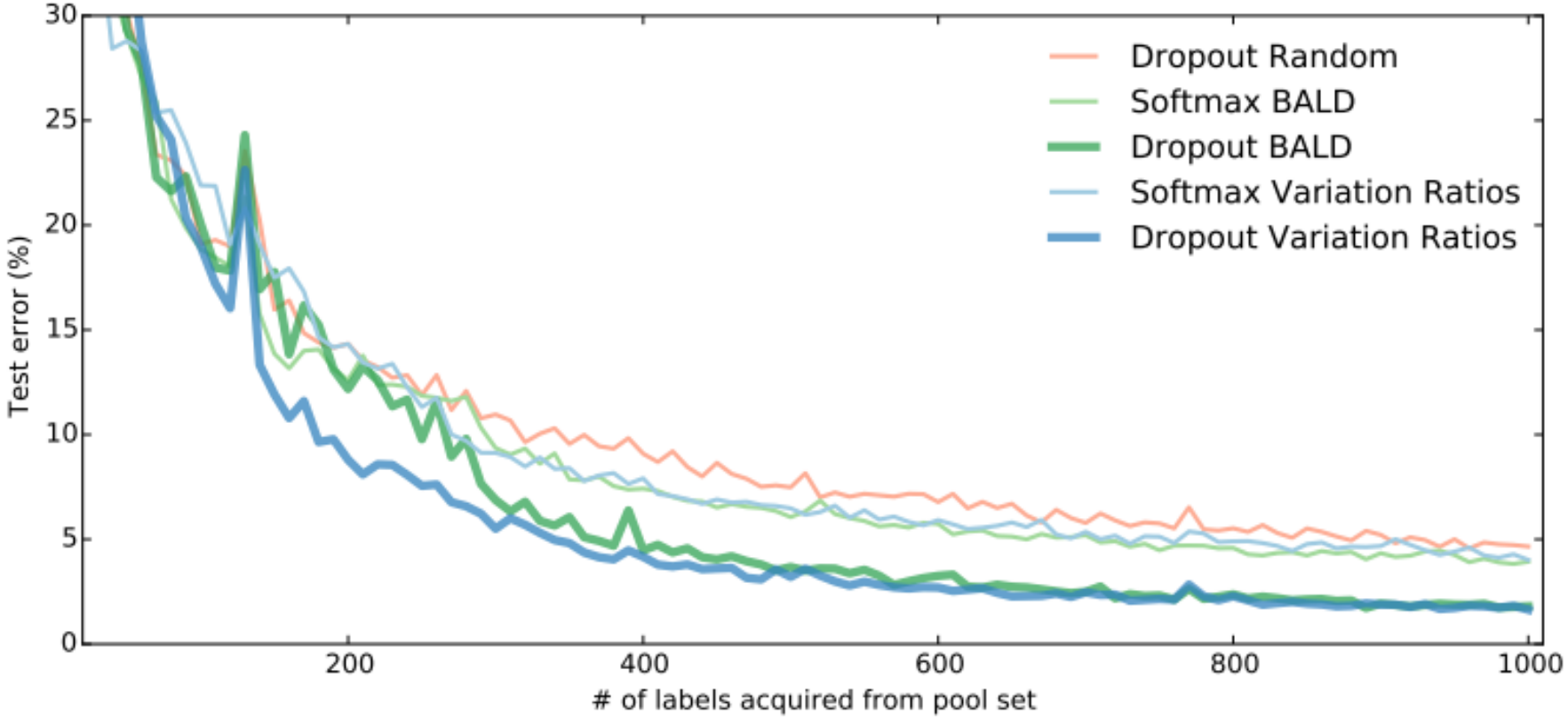
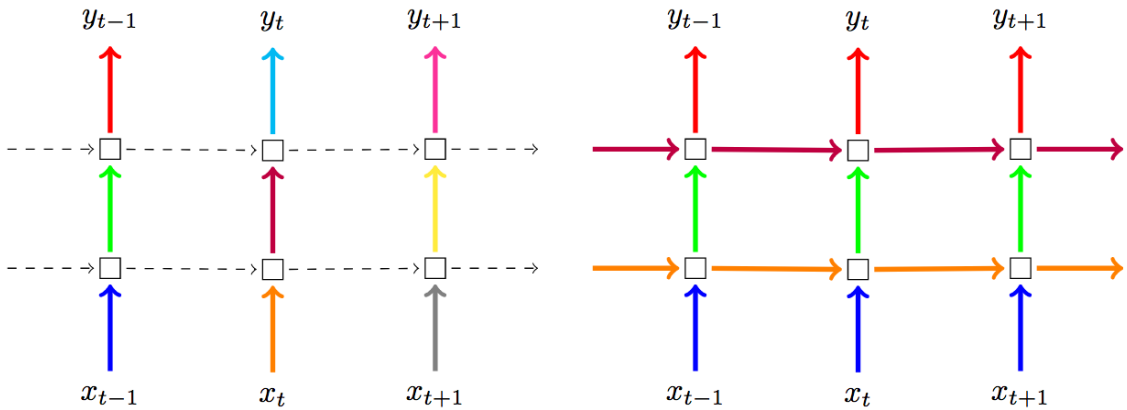


Fig. 5.1 **Test error on MNIST as a function of number of labels acquired from the pool set.** Two acquisition functions (*BALD* and *Variation Ratios*) evaluated with two approximating distributions — delta (*Softmax*) and Bernoulli (*Dropout*) — are compared to a *random* acquisition function.

Applications

- Use of uncertainty in classification: beyond MC nets
  - Application to ConvNets
  - and RNNs : BPTT: Bayesian Dropout [Gal and Ghahramani, 2016b]

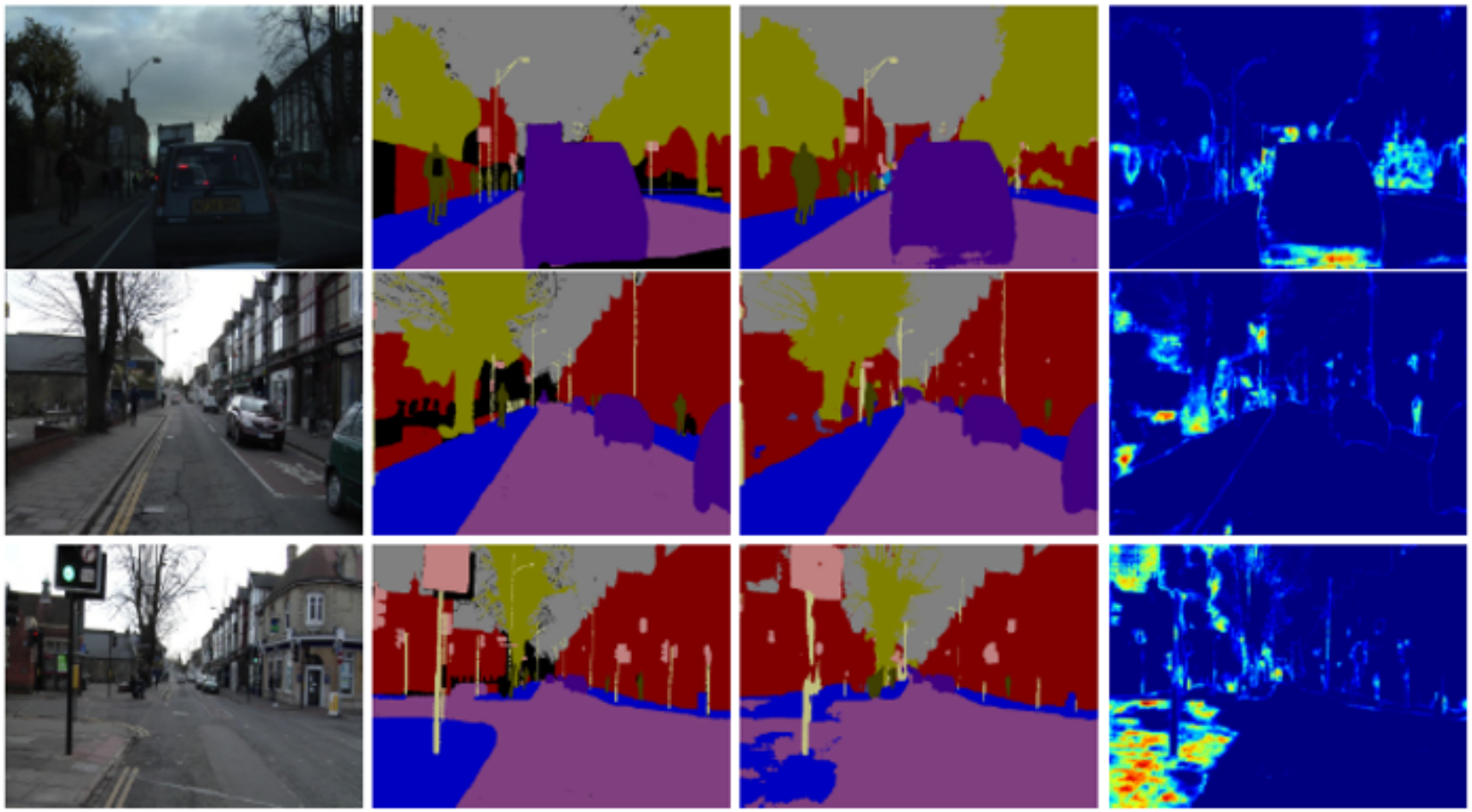


(a) Naive dropout RNN (b) Variational RNN

Figure 1: **Depiction of the dropout technique following our Bayesian interpretation (right) compared to the standard technique in the field (left).** Each square represents an RNN unit, with horizontal arrows representing time dependence (recurrent connections). Vertical arrows represent the input and output to each RNN unit. Coloured connections represent dropped-out inputs, with different colours corresponding to different dropout masks. Dashed lines correspond to standard connections with no dropout. Current techniques (naive dropout, left) use different masks at different time steps, with no dropout on the recurrent layers. The proposed technique (Variational RNN, right) uses the same dropout mask at each time step, including the recurrent layers.

Application in Semantic Segmentation

From [Kendall and Gal, 2017]



(a) Input Image (b) Ground Truth (c) Semantic Segmentation (e) Epistemic Uncertainty

Application in Reinforcement Learning

Based on uncertainty estimates given by a **dropout Q-network**, we can perform Thomspon sampling to help an agent decide when to exploit rewards or when to explore its environment [Gal, 2016]

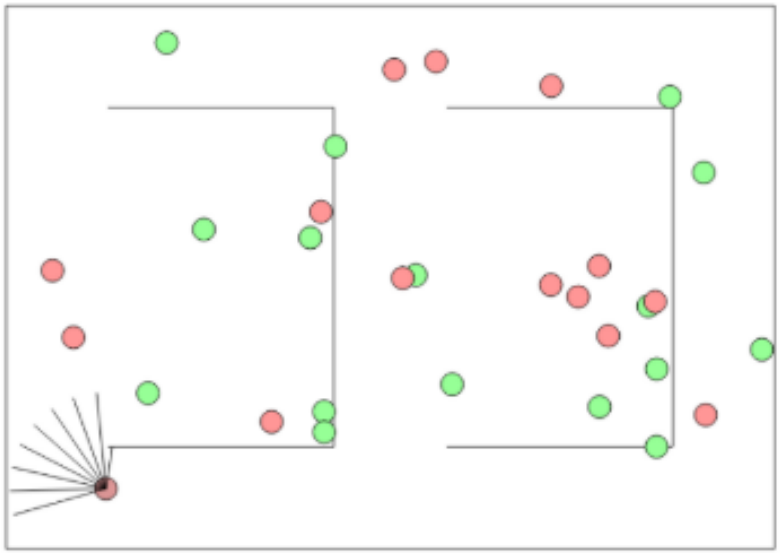


Fig. 5.3 Depiction of the reinforcement learning problem used in the experiments. The agent is in the lower left part of the maze, facing north-west.

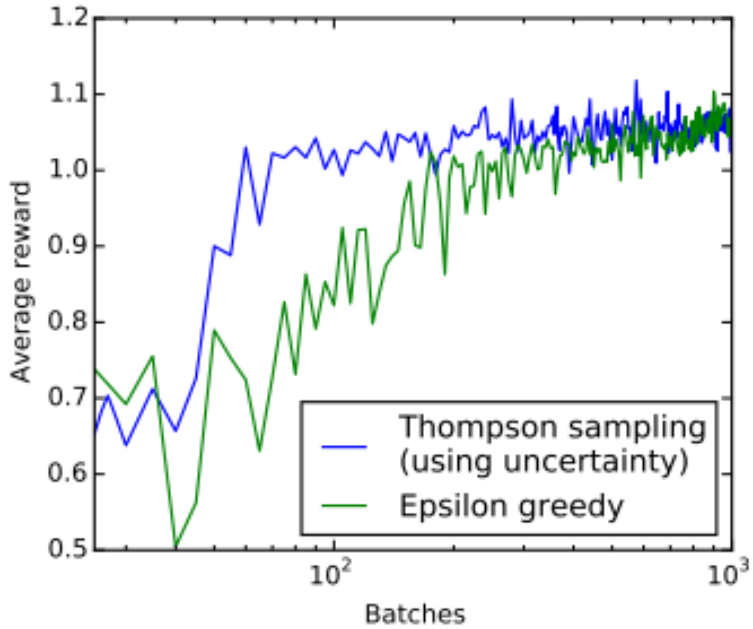
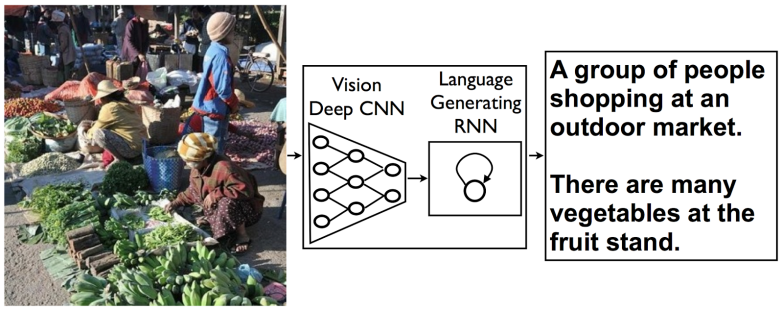
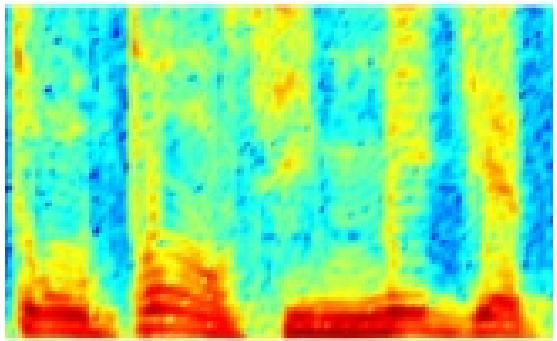


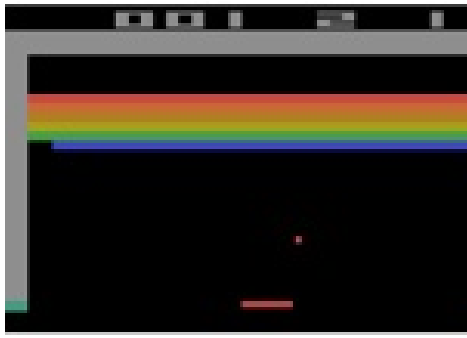
Fig. 5.4 **Log plot of average reward** obtained by both epsilon greedy (in green) and our approach (in blue), as a function of the number of batches.

Outline

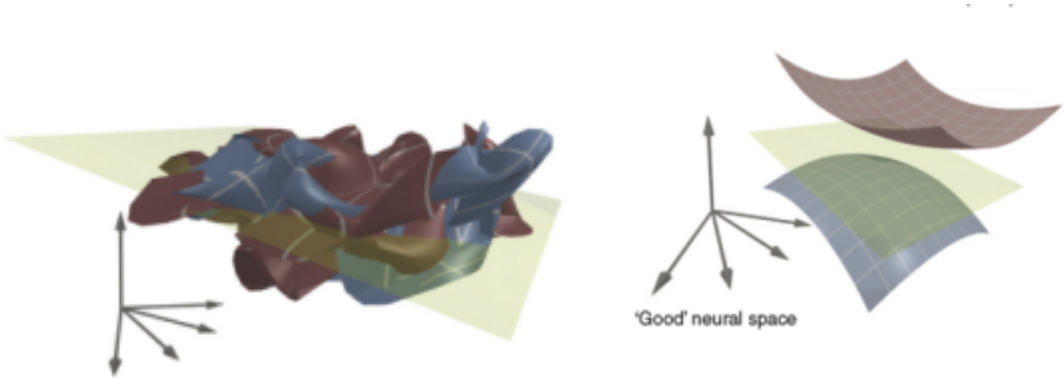
Deep Learning Theory







- Deep Learning: huge impact in terms of experimental results
- BUT: formal understanding still limited,
  - Optimization: non-convex problem
  - Generalization & over-fitting
  - Robustness

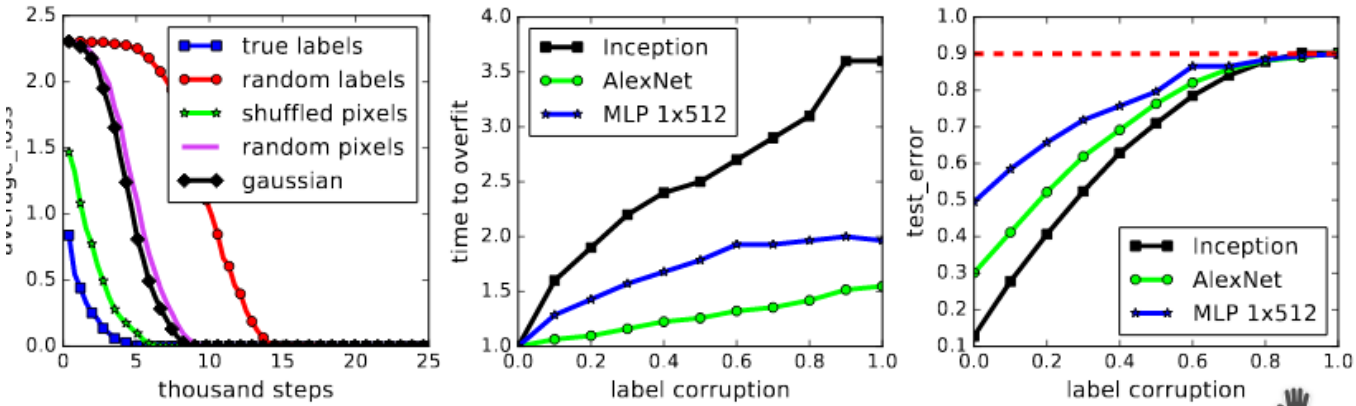


Deep Learning and generalization

- Rademacher complexity: capacity of a model to fit random label :

$$\mathcal{R}_n(\mathcal{H}) = E_\sigma \left[sup_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} \sigma_i h(x_i) \right]$$

- Rethinking generalization: Zhang *et. al.* ICLR17 [Zhang et al., 2017]



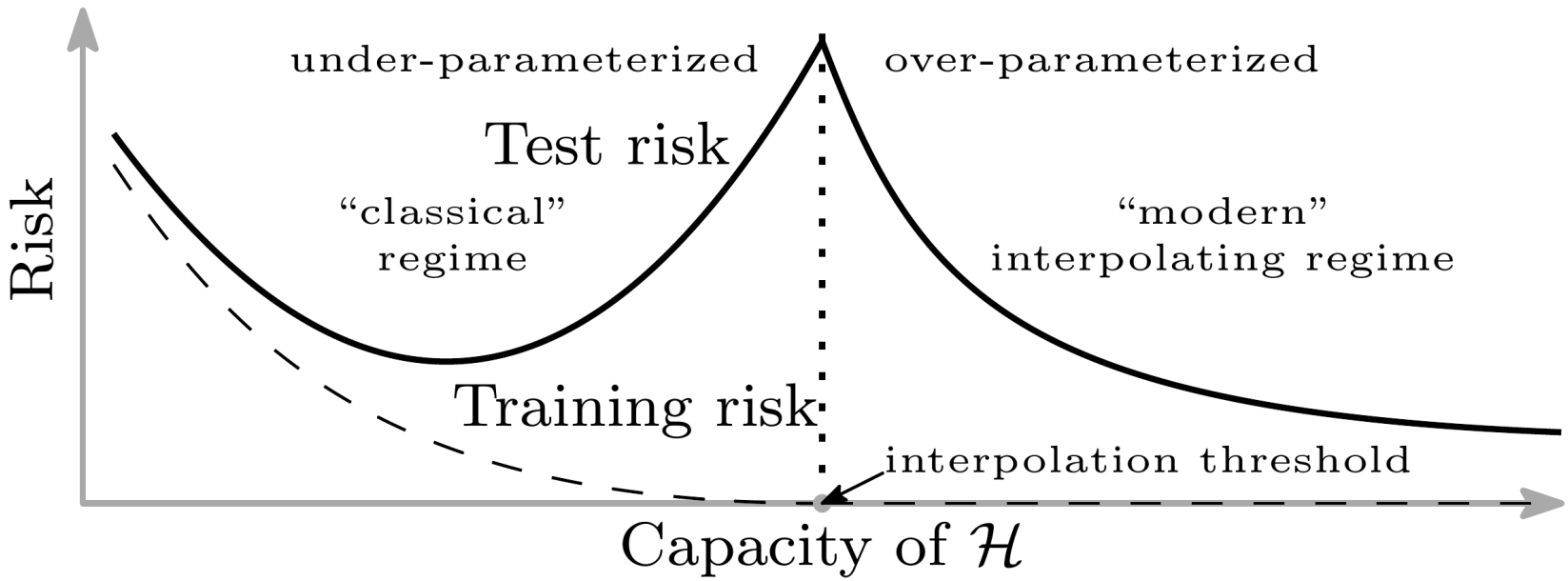
(a) learning curves (b) convergence slowdown (c) generalization error growth

  - Deep models easily fits random labels !!
  - $\mathcal{R}_n(\mathcal{H}) \approx 1 \Rightarrow$ no theoretical guarantee on generalization performances

- **Classical learning theory insufficient to explain the good generalization behavior of deep models**

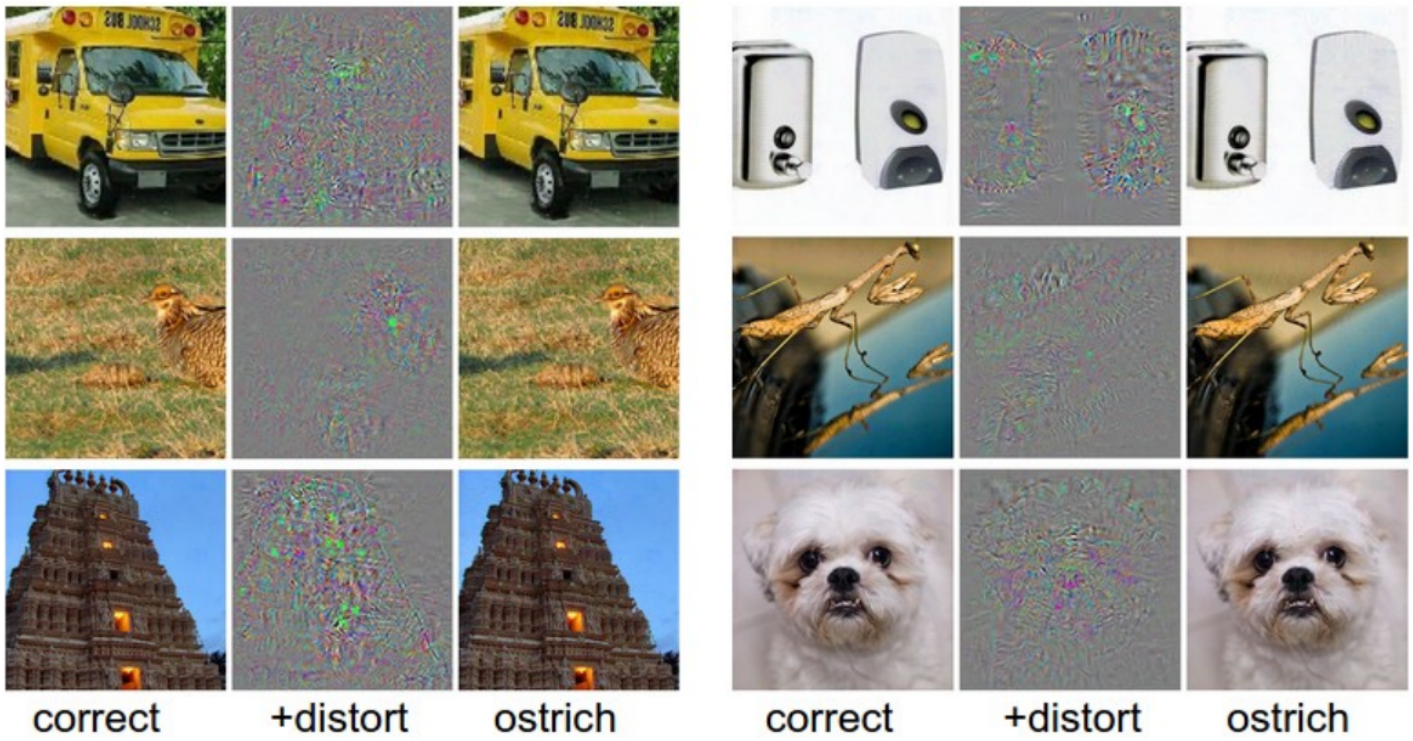
Generalization and over-parametrized models

- Double U-curve phenomena observed with deep models! [Belkin et al., 2019]



Deep Learning (DL) & Stability

- **Stability:** decision function with "controlled" variations
  - Small input variations $\Leftrightarrow$ reasonably small output variations on decision, *e.g.* Lipschitz property
  - **Decision function of deep Models not always stable**
    - **Ex: Adversarial Examples**



Deep Learning (DL) & Stability

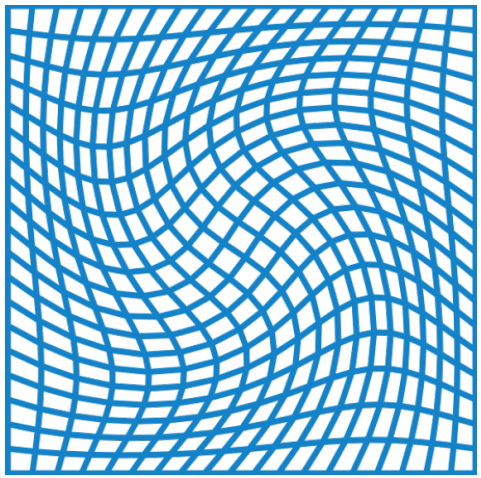
- Adversarial attacks in real-world

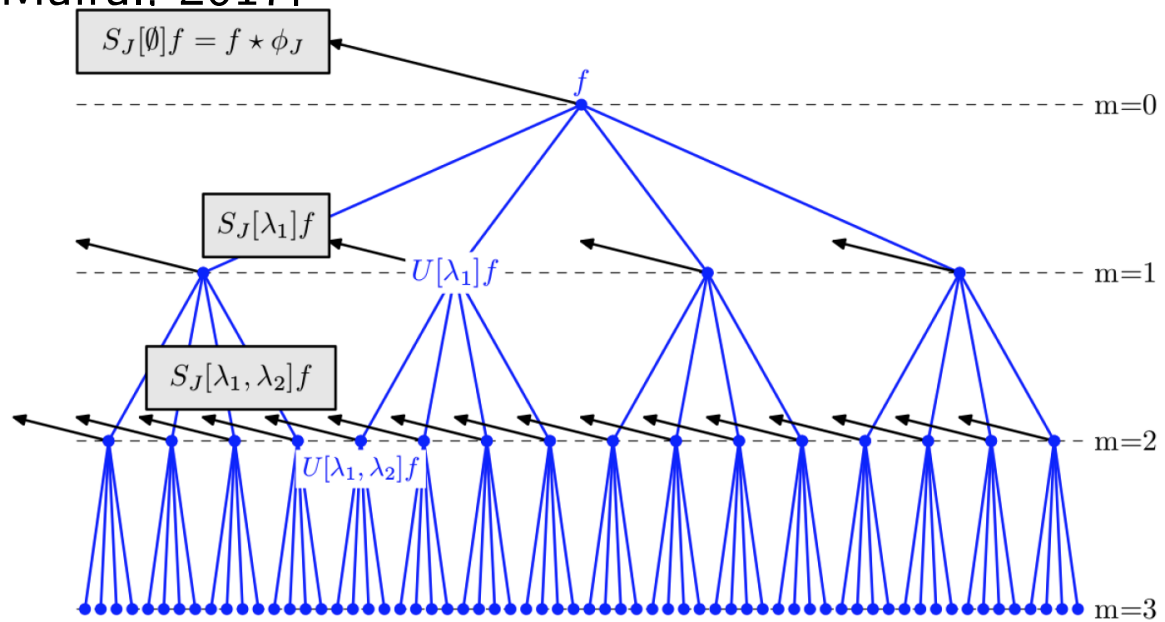
[Evtimov et al., 2017]

Deep Learning (DL) & Stability

Formal stability analysis of deep models

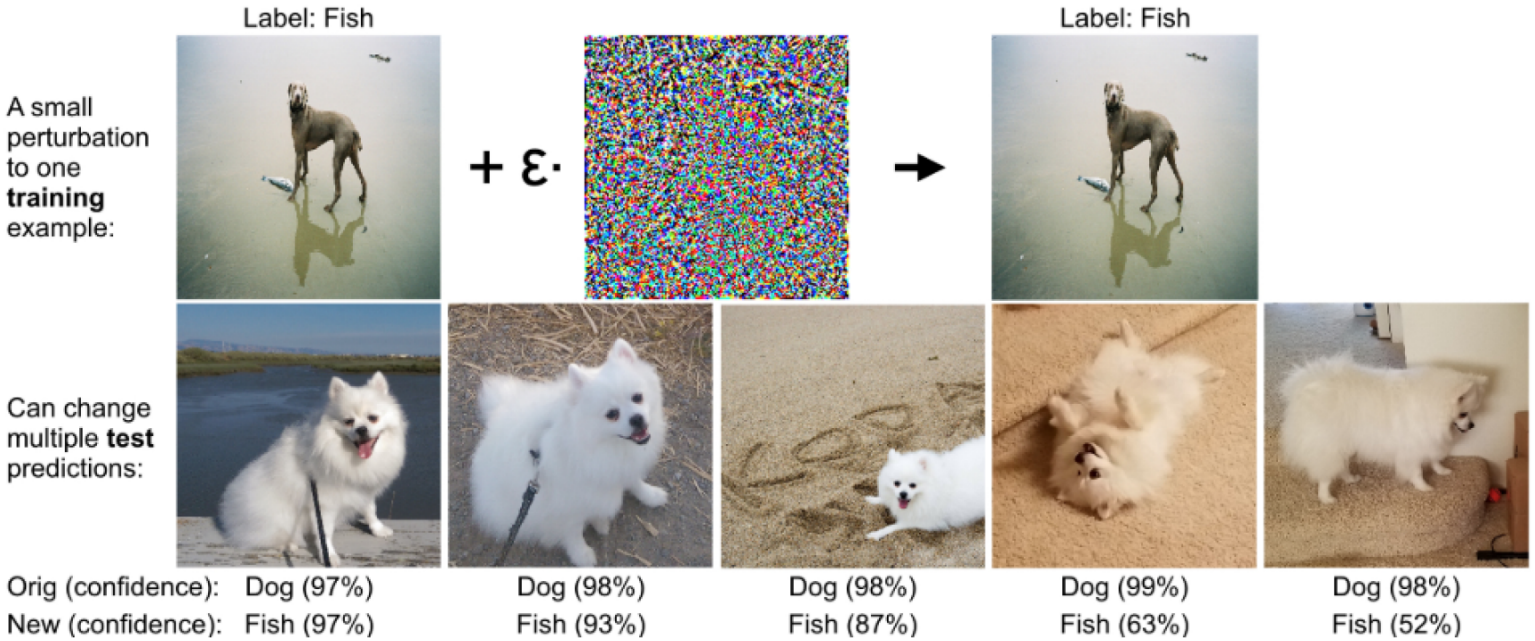
- Harmonic analysis in scattering operators [Mallat, 2012, Bruna and Mallat, 2013], *i.e.* "deep wavelets"
  - Show stability / invariance to diffeomorphisms
  - Stability bounds
- Generalized to deep kernel machines, closer to SoTA deep ConvNet architectures [Bietti and Mairal, 2017]





Formal stability analysis of deep models

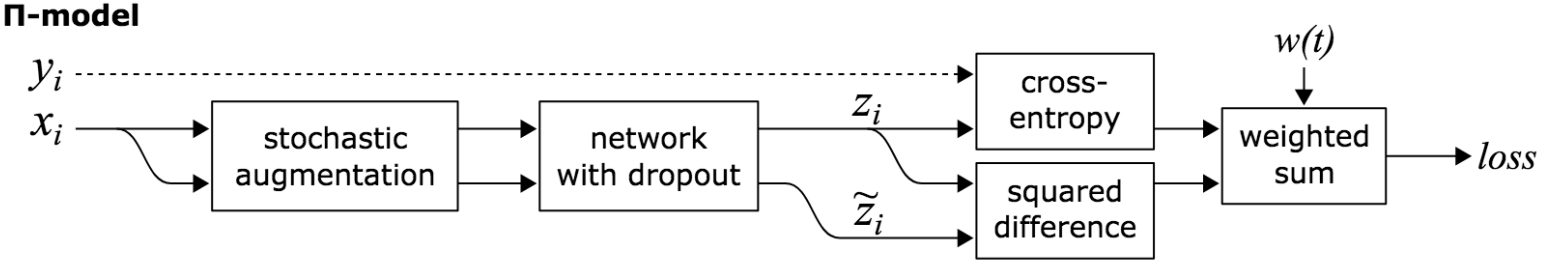
- Influence Functions [Cook and Weisberg, 1980]
  - Characterize decision function influence on training examples
    - Removing a training point: $\mathcal{I}_{up,loss}(z, z_{test}) = -\nabla_\theta L(z_{test}, \hat{\theta})^T H_\theta^{-1} \nabla_\theta L(z, \hat{\theta})$
    - Perturbing it: $\mathcal{I}_{pert,loss}(z, z_{test})^T = -\nabla_\theta L(z_{test}, \hat{\theta})^T H_\theta^{-1} \nabla_x \nabla_\theta L(z, \hat{\theta})$
  - Adapted / applied to deep networks [Koh and Liang, 2017]
- Data poisoning



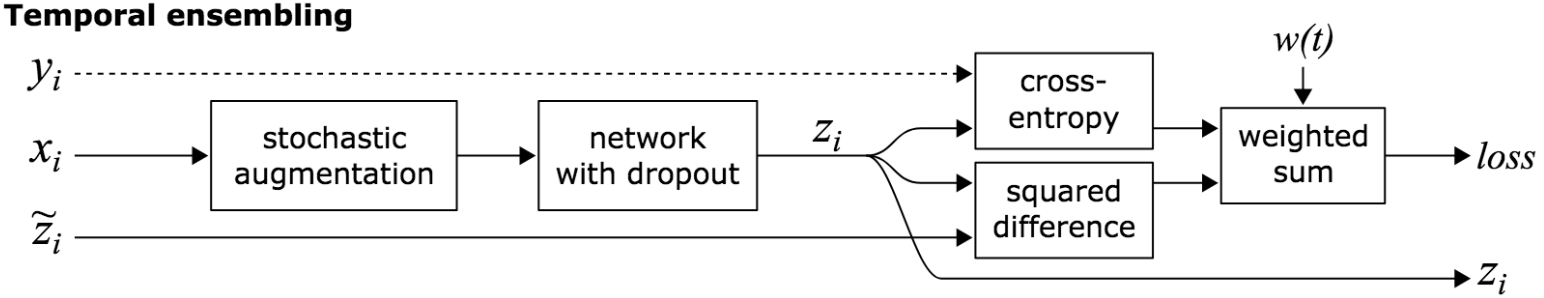
Ad hoc stability training

- Regularization criterion supporting learning stable decision function
  - Underlying model might not be stable, but helps to focus on a subet of stable functions of the family
- Robustness of the decision to transformations [Sajjadi et al., 2016], stability accross iterations [Laine and Aila. 2017. Tarvainen and Valpola. 2017]

Π-model



Temporal ensembling



References I

[Belkin et al., 2019] Belkin, M., Hsu, D., Ma, S., and Mandal, S. (2019).
Reconciling modern machine-learning practice and the classical bias–variance trade-off.
Proceedings of the National Academy of Sciences, 116(32):15849–15854.

[Bietti and Mairal, 2017] Bietti, A. and Mairal, J. (2017).
Invariance and stability of deep convolutional representations.
In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 6210–6220. Curran Associates, Inc.

[Bruna and Mallat, 2013] Bruna, J. and Mallat, S. (2013).
Invariant scattering convolution networks.
IEEE Trans. Pattern Anal. Mach. Intell., 35(8):1872–1886.

[Cook and Weisberg, 1980] Cook, R. and Weisberg, S. (1980).
Characterizations of an empirical influence function for detecting influential cases in regression.
Technometrics, 22(4):495–508.

[Corbière et al., 2019] Corbière, C., Thome, N., Bar-Hen, A., Cord, M., and Pérez, P. (2019).
Addressing Failure Detection by Learning Model Confidence.
In *Advances in Neural Information Processing Systems (NeurIPS)*, Vancouver, Canada.

[Evtimov et al., 2017] Evtimov, I., Eykholt, K., Fernandes, E., Kohno, T., Li, B., Prakash, A., Rahmati, A., and Song, D. (2017).
Robust physical-world attacks on machine learning models.
CoRR, abs/1707.08945.

[Gal, 2016] Gal, Y. (2016).
Uncertainty in Deep Learning.
PhD thesis, University of Cambridge.

[Gal and Ghahramani, 2015] Gal, Y. and Ghahramani, Z. (2015).
Bayesian convolutional neural networks with bernoulli approximate variational inference.
CoRR, abs/1506.02158.

References II

[Gal and Ghahramani, 2016a] Gal, Y. and Ghahramani, Z. (2016a).
Dropout as a bayesian approximation: Representing model uncertainty in deep learning.
In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pages 1050–1059. JMLR.org.

[Gal and Ghahramani, 2016b] Gal, Y. and Ghahramani, Z. (2016b).
A theoretically grounded application of dropout in recurrent neural networks.
In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pages 1027–1035, USA. Curran Associates Inc.

[Gal et al., 2017] Gal, Y., Islam, R., and Ghahramani, Z. (2017).
Deep Bayesian Active Learning with Image Data.
In *Proceedings of the 34th International Conference on Machine Learning (ICML-17)*.

[Guo et al., 2017] Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017).
On calibration of modern neural networks.
CoRR, abs/1706.04599.

[Kendall and Gal, 2017] Kendall, A. and Gal, Y. (2017).
What uncertainties do we need in bayesian deep learning for computer vision?
In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 5574–5584. Curran Associates, Inc.

[Koh and Liang, 2017] Koh, P. W. and Liang, P. (2017).
Understanding black-box predictions via influence functions.
In Precup, D. and Teh, Y. W., editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1885–1894, International Convention Centre, Sydney, Australia. PMLR.

[Laine and Aila, 2017] Laine, S. and Aila, T. (2017).
Temporal ensembling for semi-supervised learning.
In *International Conference on Learning Representations (ICLR)*.

References III

[Liang et al., 2017] Liang, S., Li, Y., and Srikant, R. (2017).
Enhancing the reliability of out-of-distribution image detection in neural networks.
In *ICLR*.

[Mallat, 2012] Mallat, S. (2012).
Group invariant scattering.
Communications in Pure and Applied Mathematics, 10:1331–1398.

[Sajjadi et al., 2016] Sajjadi, M., Javanmardi, M., and Tasdizen, T. (2016).
Regularization with stochastic transformations and perturbations for deep semi-supervised learning.
In *Advances in Neural Information Processing Systems (NIPS)*.

[Tarvainen and Valpola, 2017] Tarvainen, A. and Valpola, H. (2017).
Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results.
In *Advances in Neural Information Processing Systems (NIPS)*.

[Tong and Koller, 2002] Tong, S. and Koller, D. (2002).
Support vector machine active learning with applications to text classification.
J. Mach. Learn. Res., 2:45–66.

[Zhang et al., 2017] Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. (2017).
Understanding deep learning requires rethinking generalization.