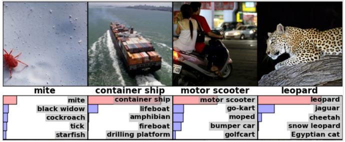
Addressing Failure Prediction by Learning Model Confidence

Charles Corbière
Conservatoire National des Arts et Métiers (CNAM)
Laboratoire CEDRIC - Equipe MSDMA
valeo.ai

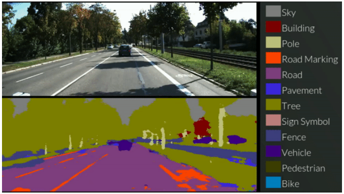
October 17, 2019

Deep Learning in Computer Vision

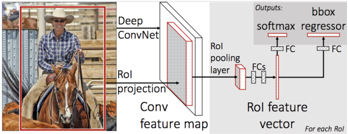
[*Krizhevsky*, 2012]



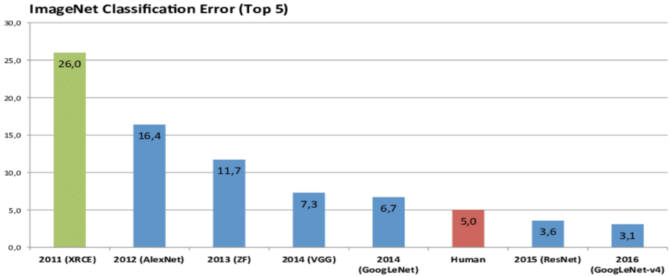
[*Kendall et al.* SegNet, 2015]



[*Girshick et al.* Fast R-CNN, 2015]

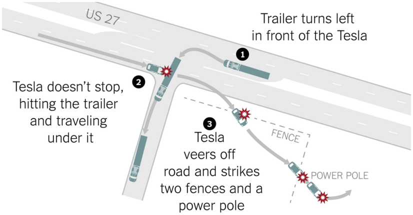


Brought significant improvements in multiple vision tasks



Robustness issues

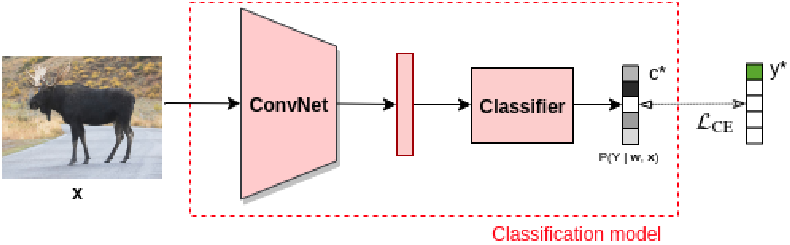
Tesla's car crash back in 2016, due to a confusion between white side of trailer and brightly lit sky



⇒ **Are neural network's predictions reliable? How much is the model certain about our output? How do we account for uncertainty?**

Confidence Estimation in Deep Learning

Classification framework
$\mathcal{D} = \{(\mathbf{x}_i, y_i^*)\}_{i=1}^N$ with $\mathbf{x}_i \in \mathbb{R}^D$ and $y_i^* \in \mathcal{Y} = \{1, ..., K\}$.
One can infer predicted class $\hat{y} = \text{argmax}_{k \in \mathcal{Y}}\, p(Y = k | \mathbf{w}, \mathbf{x})$.



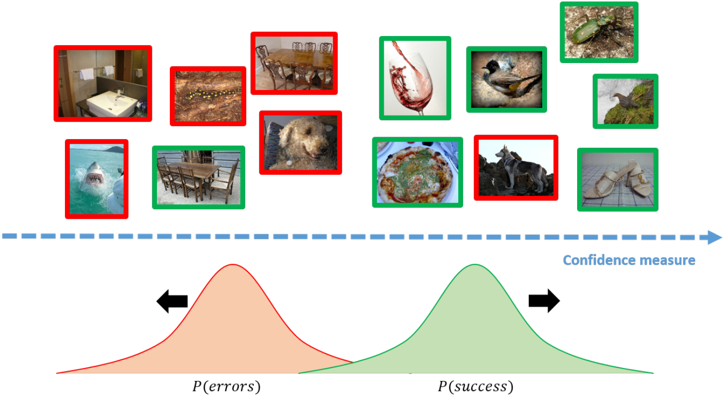
- **Maximum Class Probability** [Hendrycks and Gimpel, 2017]
  A confidence measure baseline for deep neural networks:

$$\mathrm{MCP}(\mathbf{x}) = \max_{k \in \mathcal{Y}} p(Y = k | \mathbf{w}, \mathbf{x})$$

Failure Prediction

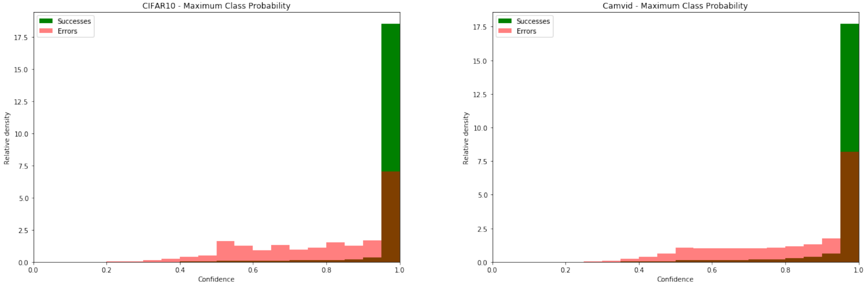
Goal

Provide **reliable confidence measures** over model's predictions whose ranking among samples enables to **distinguish correct from erroneous predictions.**



MCP, a sub-optimal ranking confidence measure

$$\mathrm{MCP}(\mathbf{x}) = \max_{k \in \mathcal{Y}} p(Y = k | \mathbf{w}, \mathbf{x})$$



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

(also, overconfident values for both distributions)

Our Approach: True Class Probability

When missclassifying, MCP $\Leftrightarrow$ probability of the wrong class.
$\Rightarrow$ **what if we had taken the probability of the true class?**

True Class Probability

Given a sample $(\mathbf{x}, y^*)$ and a model parametrized by $\mathbf{w}$, *True Class Probability* writes as:

$$\mathrm{TCP}(\mathbf{x}, y^*) = p(Y = y^* | \mathbf{w}, \mathbf{x})$$

Theoretical guarantees:

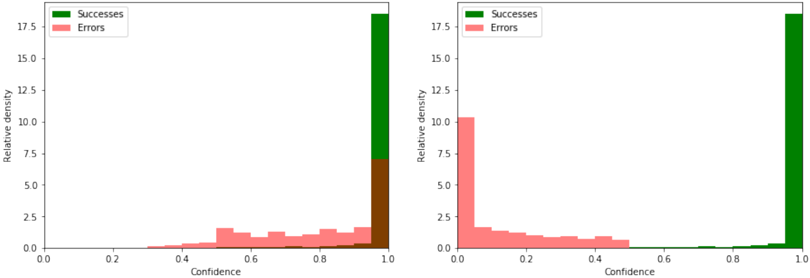
- $\mathrm{TCP}(\mathbf{x}, y^*) > 1/2 \Rightarrow \hat{y} = y^*$
- $\mathrm{TCP}(\mathbf{x}, y^*) < 1/K \Rightarrow \hat{y} \neq y^*$

N.B: a normalized variant present stronger guarantees:

$$TCP^r(\mathbf{x}, y^*) = p(Y = y^* | \mathbf{w}, \mathbf{x}) / p(Y = \hat{y} | \mathbf{w}, \mathbf{x})$$

TCP, a reliable confidence criterion

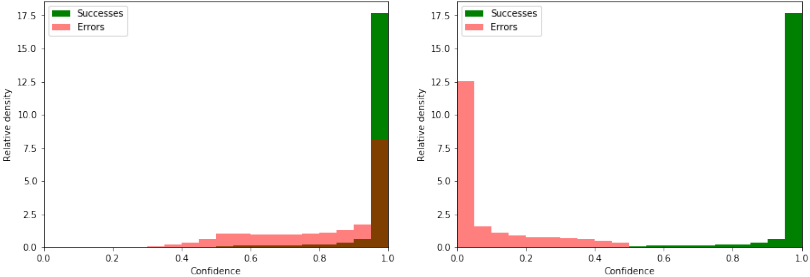
VGG16 on CIFAR-10



(a) Maximum Class Probability (b) Our Proposal (True Class Probability)

TCP, a reliable confidence criterion

SegNet on CamVid



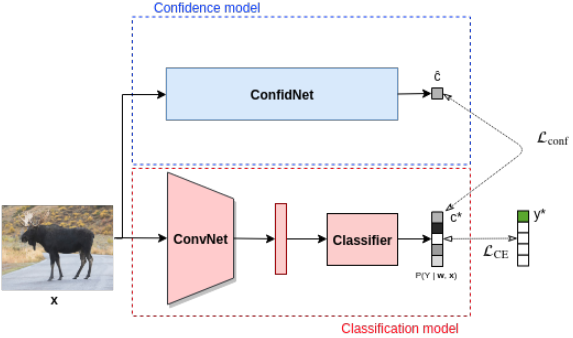
(a) Maximum Class Probability

(b) Our Proposal (True Class Probability)

ConfidNet: Learning TCP Model Confidence

However, $TCP(\mathbf{x}, y^*)$ is **unknown** at test time.

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

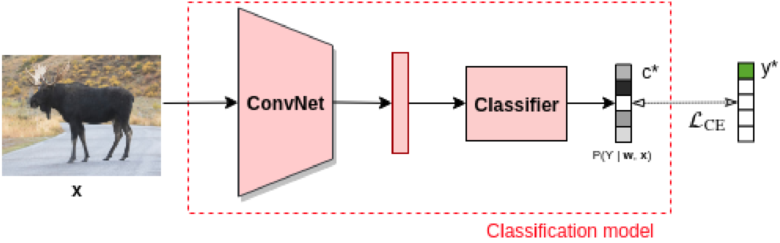


As $TCP(\mathbf{x}, y^*) \in [0, 1]$, we propose ℓ_2 loss to train ConfidNet:

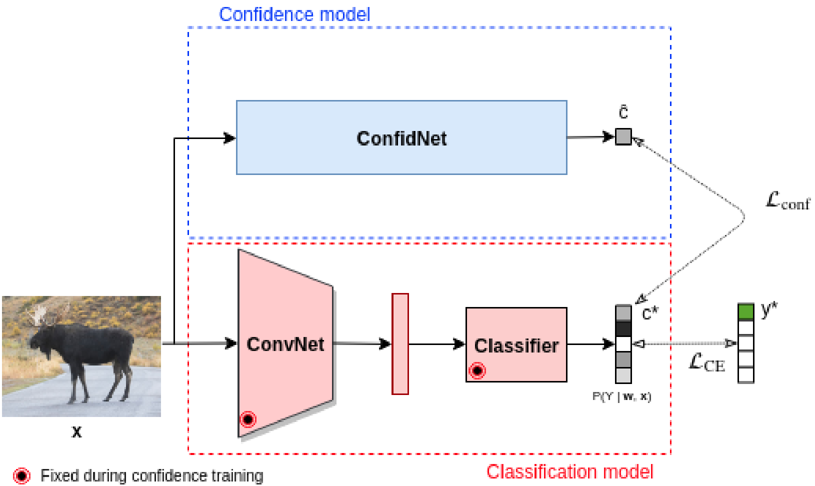
$$\mathcal{L}_{\text{conf}}(\theta; \mathcal{D}) = \frac{1}{N} \sum_{i=1}^{N} (\hat{c}(\mathbf{x}_i, \theta) - c^*(\mathbf{x}_i, y_i^*))^2$$

N.B: $c^*(\mathbf{x}, y^*) = TCP(\mathbf{x}, y^*)$ (or $TCP^r(\mathbf{x}, y^*)$)

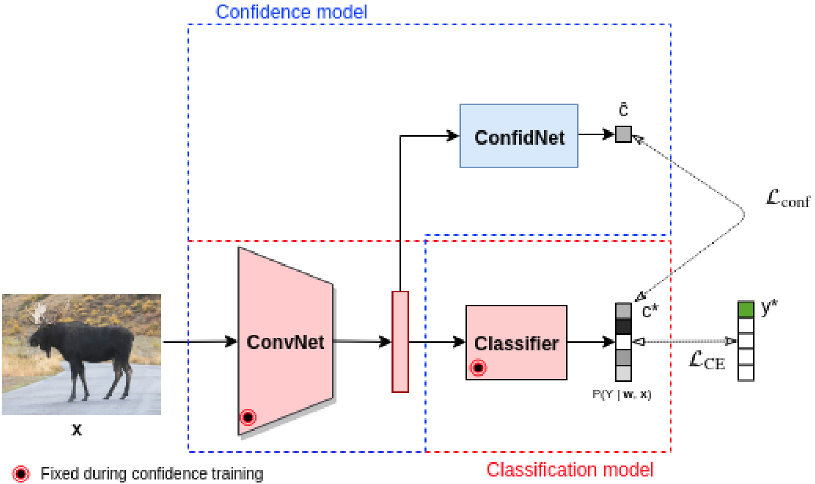
ConfidNet learning scheme



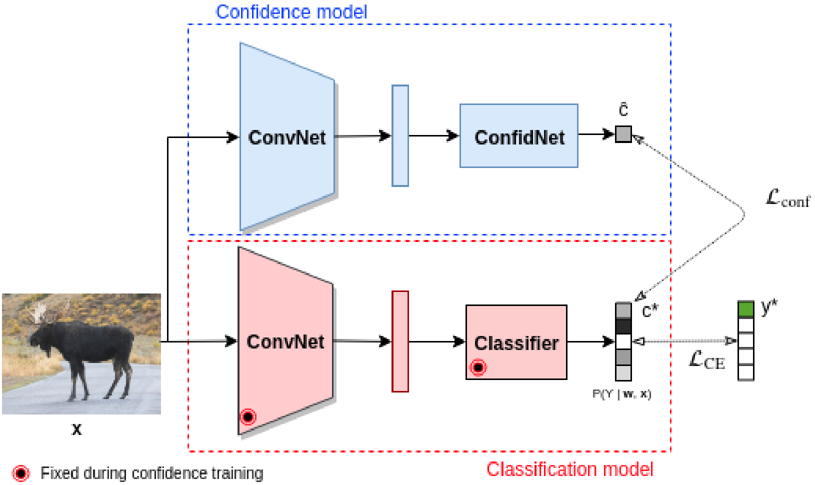
ConfidNet learning scheme



Efficient ConfidNet learning scheme (1/2)



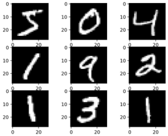
Efficient ConfidNet learning scheme (2/2)

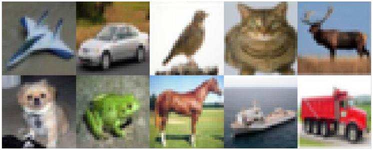


Experiments

Traditional public **image classification** and **semantic segmentation** datasets

- **MNIST**: 32x32 BW, 10 classes, 60K training + 10K test
- **SVHN**: 32x32 RGB , 10 classes, 73K training + 26K test
- **CIFAR-10 & CIFAR-100**: 32x32 RGB, *10 / 100 classes*, 50K training + 10K test
- **CamVid**: *semantic segmentation* , 360x480, 11 classes





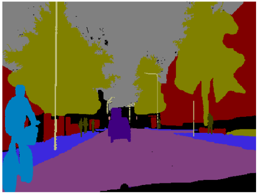
Quantitative results

Dataset	Model	FPR-95%-TPR	AUPR-Error	AUPR-Success	AUC
MNIST MLP	Baseline (MCP)	14.87	37.70	99.94	97.13
	MCDropout	15.15	38.22	99.94	97.15
	TrustScore	12.31	52.18	99.95	97.52
	ConfidNet (Ours)	**11.79**	**57.37**	**99.95**	**97.83**
MNIST Small ConvNet	Baseline (MCP)	5.56	35.05	99.99	98.63
	MCDropout	5.26	38.50	99.99	98.65
	TrustScore	10.00	35.88	99.98	98.20
	ConfidNet (Ours)	**3.33**	**45.89**	**99.99**	**98.82**
SVHN Small ConvNet	Baseline (MCP)	31.28	48.18	99.54	93.20
	MCDropout	36.60	43.87	99.52	92.85
	TrustScore	34.74	43.32	99.48	92.16
	ConfidNet (Ours)	**28.58**	**50.72**	**99.55**	**93.44**
CIFAR-10 VGG16	Baseline (MCP)	47.50	45.36	99.19	91.53
	MCDropout	49.02	46.40	**99.27**	92.08
	TrustScore	55.70	38.10	98.76	88.47
	ConfidNet (Ours)	**44.94**	**49.94**	99.24	**92.12**
CIFAR-100 VGG16	Baseline (MCP)	67.86	71.99	92.49	85.67
	MCDropout	64.68	72.59	**92.96**	86.09
	TrustScore	71.74	66.82	91.58	84.17
	ConfidNet (Ours)	**62.96**	**73.68**	92.68	**86.28**
CamVid SegNet	Baseline (MCP)	63.87	48.53	96.37	84.42
	MCDropout	62.95	49.35	96.40	84.58
	TrustScore		20.42	92.72	68.33
	ConfidNet (Ours)	**61.52**	**50.51**	**96.58**	**85.02**

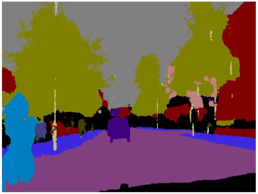
Qualitative results

Failure detection for **semantic segmentation** on CamVid dataset

(a) Input Image

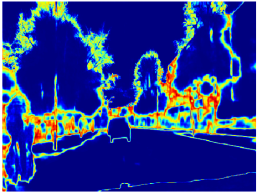


(b) Ground truth

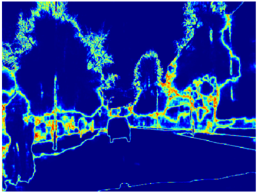


(c) Prediction

(d) Model Errors



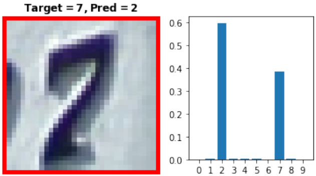
(e) ConfidNet



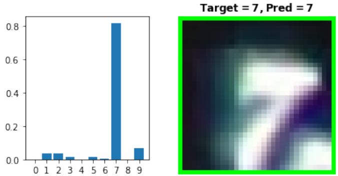
(f) MCP

Qualitative results

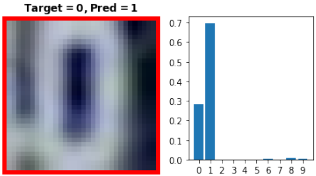
Entropy as a confident estimate, such as in MC-Dropout, may not always be adequate



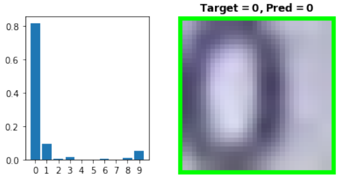
(a) MCP=0.596, MCDropout=-0.787, *ConfidNet*=0.449



(b) MCP=0.816, MCDropout=-0.786, *ConfidNet*=0.894



(c) MCP=0.696, MCDropout=-0.726, *ConfidNet*=0.436



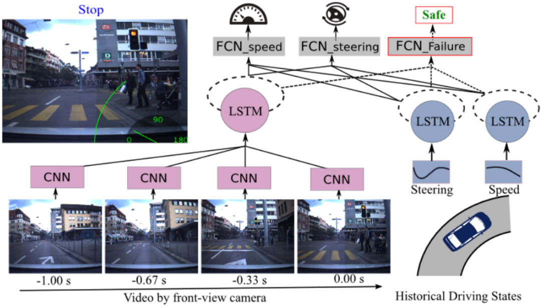
(d) MCP=0.814, MCDropout=-0.725, *ConfidNet*=0.886

Perspectives

- We defined TCP, a specific **criterion for failure detection**

$$\mathrm{TCP}(\mathbf{x}, y^*) = p(Y = y^* | \mathbf{w}, \mathbf{x})$$

- We proposed **ConfidNet** a model- and task-agnostic training method to learn TCP



- **Application in various domains**: autonomous driving, medical diagnosis, nuclear plant monitoring, etc...

Thank you for your attention.

Addressing Failure Prediction by Learning Model Confidence

Charles Corbière[1,2]
charles.corbiere@valeo.com

Nicolas Thome[1]
nicolas.thome@cnam.fr

Avner Bar-Hen[1]
avner@cnam.fr

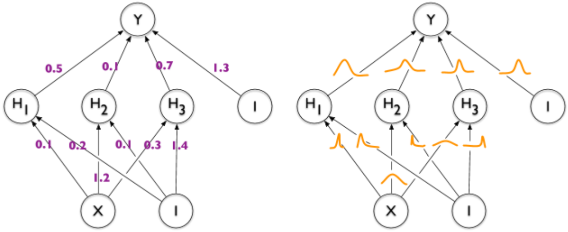
Matthieu Cord[2,3]
matthieu.cord@lip6.fr

Patrick Pérez[2]
patrick.perez@valeo.com

[1]CEDRIC, Conservatoire National des Arts et Métiers, Paris, France
[2]valeo.ai, Paris, France
[3]Sorbonne University, Paris, France

Related work

Bayesian Monte-Carlo Dropout
We attach probability distributions to network weights (*Bayesian Neural Network*)



- compute posterior $p(w|\mathcal{D})$ with **variational inference** approximation
- sample from posterior to obtain predictive distribution

[Gal and Ghahramani, 2016] showed that **a NN with Dropout can be seen as a variational Bayesian approximation**

Related work

How to estimate uncertainty with Bayesian MCDropout?

- train a model with Dropout units
- given a point x, repeat T times:
  - keep Dropout units at test time
  - compute output prediction $f_w(\mathbf{x})$
- Compute Softmax mean output and entropy

$$p(y = c|\mathbf{x}, \mathcal{D}) \approx \frac{1}{T}\sum_{t=1}^{T} \text{Softmax}(f_{\hat{w}_t}(\mathbf{x}))$$

$$\hat{\mathcal{H}}[y|x, \mathcal{D}] = -\sum_{c} \left(\frac{1}{T}\sum_{t} p(y = c|\mathbf{x}, \hat{w}_t)\right) \cdot \log\left(\frac{1}{T}\sum_{t} p(y = c|\mathbf{x}, \hat{w}_t)\right)$$

Related work

Trust Score [Jiang et al., 2018]
Measure the agreement between the classifier and a modified nearest-neighbor classifier on the testing example

1. Define k-NN radius

$$r_k(x) := \inf\{r > 0 : |B(\mathbf{x}, r) \cap X| \geq k\}$$

and $\varepsilon := \inf\{r > 0 : |\{\mathbf{x} \in X : r_k(\mathbf{x}) > r\}| \leq \alpha N\}$

2. Estimate a α-high-density-set

$$H_\alpha(f) := \{\mathbf{x} \in X : r_k(\mathbf{x}) \leq \varepsilon\}$$

3. Compute Trust Score for classifier h

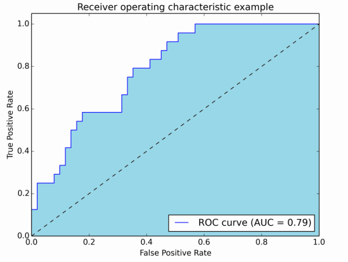
$$\xi(h, \mathbf{x}) := \frac{d\left(\mathbf{x}, H_\alpha(f_{\tilde{h}(\mathbf{x})})\right)}{d\left(\mathbf{x}, H_\alpha(f_{h(\mathbf{x})})\right)}$$

where $\tilde{h}(\mathbf{x}) = \mathrm{argmin}_{k \in \mathcal{Y}, k \neq h(\mathbf{x})}\, d\left(\mathbf{x}, H_\alpha(f_k)\right)$

Evaluation metrics

How to measure the quality of failure predictions?

1. **AUROC**: a threshold-independent evaluation, based on the ROC curve which plots the *True Positive Rate* (TPR = TP / (TP +FN)) against the *False Positive Rate* (FPR = FP / (FP+ TN)).

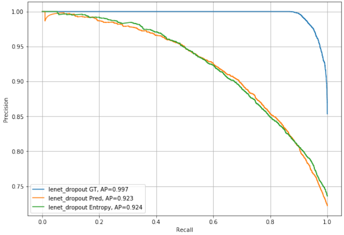


⇒ can be interpreted as **the probability that a positive example has a greater detector score than a negative example**

Evaluation metrics

But AUROC suffers from class inbalance

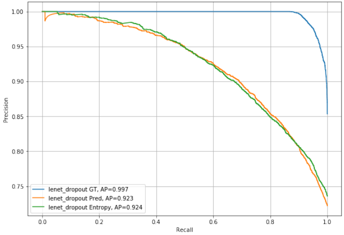
2. **AUPR_Success**: also threshold-independent, based on the PR curve which plots the *precision* (= TP / (TP+FP)) against *recall* (= TP / (TP+FN))



⇒ where the positive class are **correct predictions**

Evaluation metrics

3. **AUPR_Error**: also threshold-independent, based on the PR curve which plots the *precision* (= TP / (TP+FP)) against *recall* (= TP / (TP+FN))



⇒ where positive class are **errors** and scores are the negative of the confidence score

4. **FPR at 95% TPR**: measures the false positive rate (FPR) when the true positive rate (TPR) is equal to 95%.

Using validation set for training ConfidNet

With a high accuracy and a small validation set, we do not get a larger absolute number of errors using val set compared to train set.

Dataset	ConfidNet (train set)	ConfidNet (val set)
MLP	**57.34%**	33.41%
MNIST	**43.94%**	34.22%
SVHN	**50.72%**	47.96%
CIFAR-10	**49.94%**	48.93%
CIFAR-100	73.68%	**73.85%**
CamVid	**50.28%**	50.15%

Comparison with a BCE approach

TCP regularizes training by providing more fine-grained information about the quality of the classifier regarding a sample's prediction.

Dataset	TCP	BCE
SVHN	**50.72%**	50.00%
CIFAR-10	**49.94%**	47.95%
CamVid	**50.51%**	48.96%

$\Rightarrow$ difficult learning configuration where **very few error samples are available in training.**

References I

Gal, Y. and Ghahramani, Z. (2016).
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.

Hendrycks, D. and Gimpel, K. (2017).
A baseline for detecting misclassified and out-of-distribution examples in neural networks.
Proceedings of International Conference on Learning Representations.

References II

Jiang, H., Kim, B., Guan, M., and Gupta, M. (2018).
To trust or not to trust a classifier.
In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 31*, pages 5541–5552. Curran Associates, Inc.