Addressing Failure Prediction by Learning Model Confidence

Charles Corbière[1,2], Nicolas Thome[1], Avner Bar-Hen[1], Matthieu Cord[2,3], Patrick Pérez[2]

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

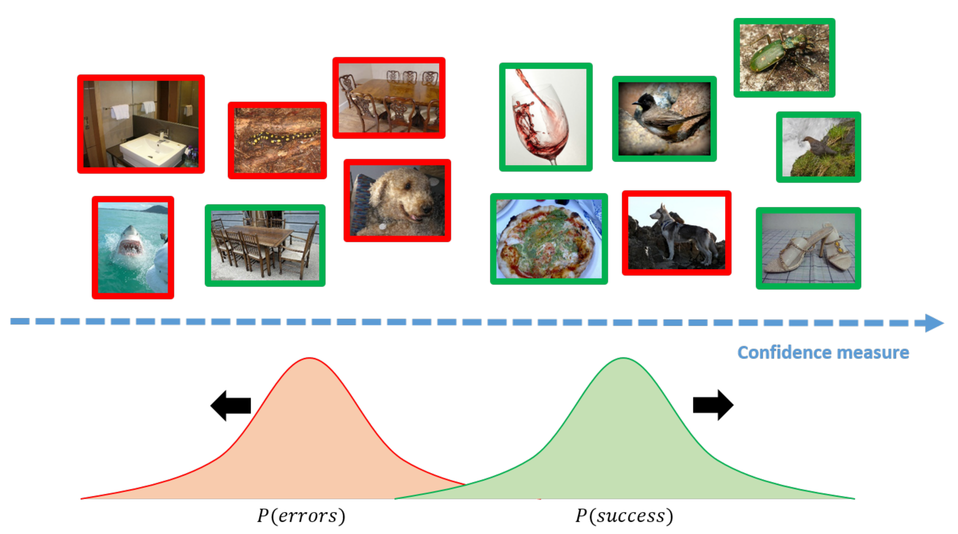
Context

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:

$$\hat{y} = \underset{k \in \mathcal{Y}}{\operatorname{argmax}}\, p(Y = k|\mathbf{w}, \mathbf{x})$$

Failure Prediction

- Provide reliable confidence measures
- Distinguish correct from erroneous predictions



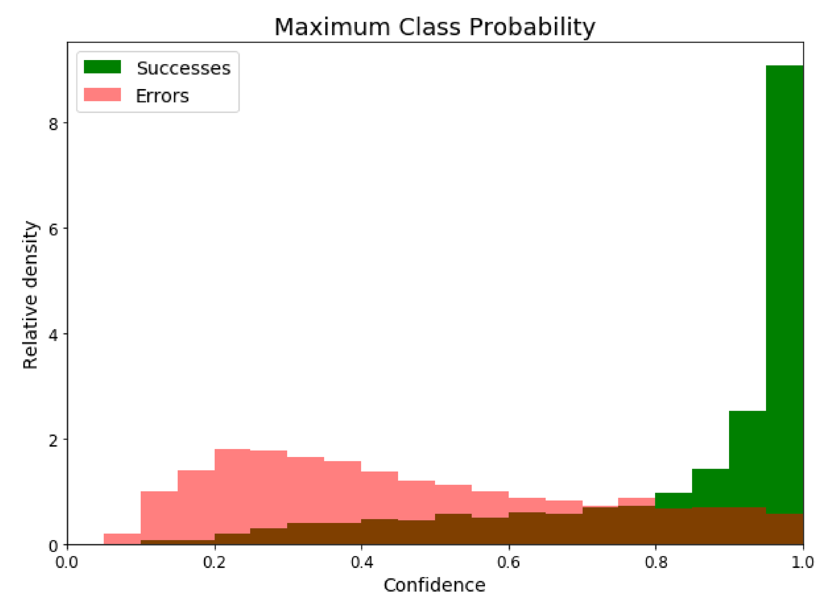
- Applications in critical systems, e.g. autonomous driving, medical diagnosis, nuclear power plant monitoring

True Class Probability (TCP)

- **Maximum Class Probability**, widely used baseline with DNN for measuring confidence [1]:

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

- Issue: **overlapping distributions** between successes and errors
$\Rightarrow$ hard to distinguish
- Calibration does not affect ranking

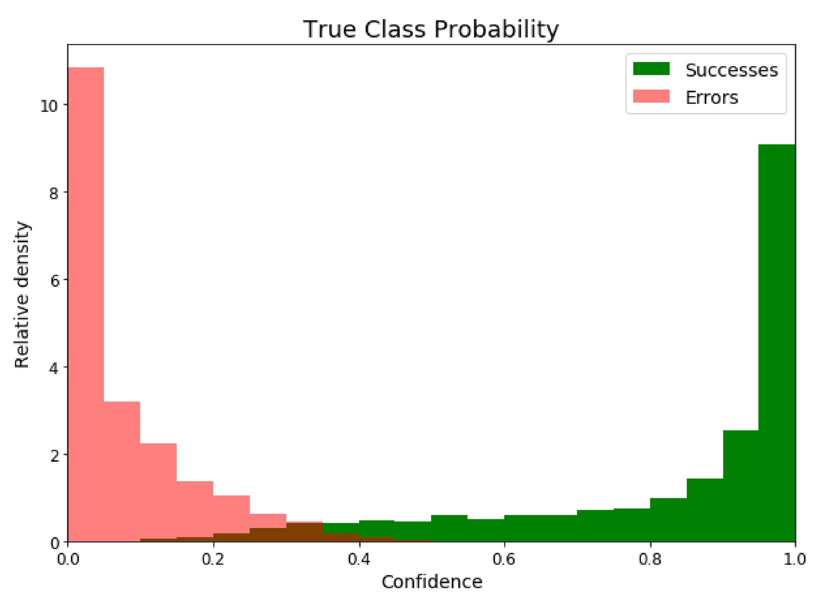


When missclassifying, MCP $\Longleftrightarrow$ probability of the wrong class
$\Rightarrow$ **What if we had taken the probability of the true class?**

True Class Probability

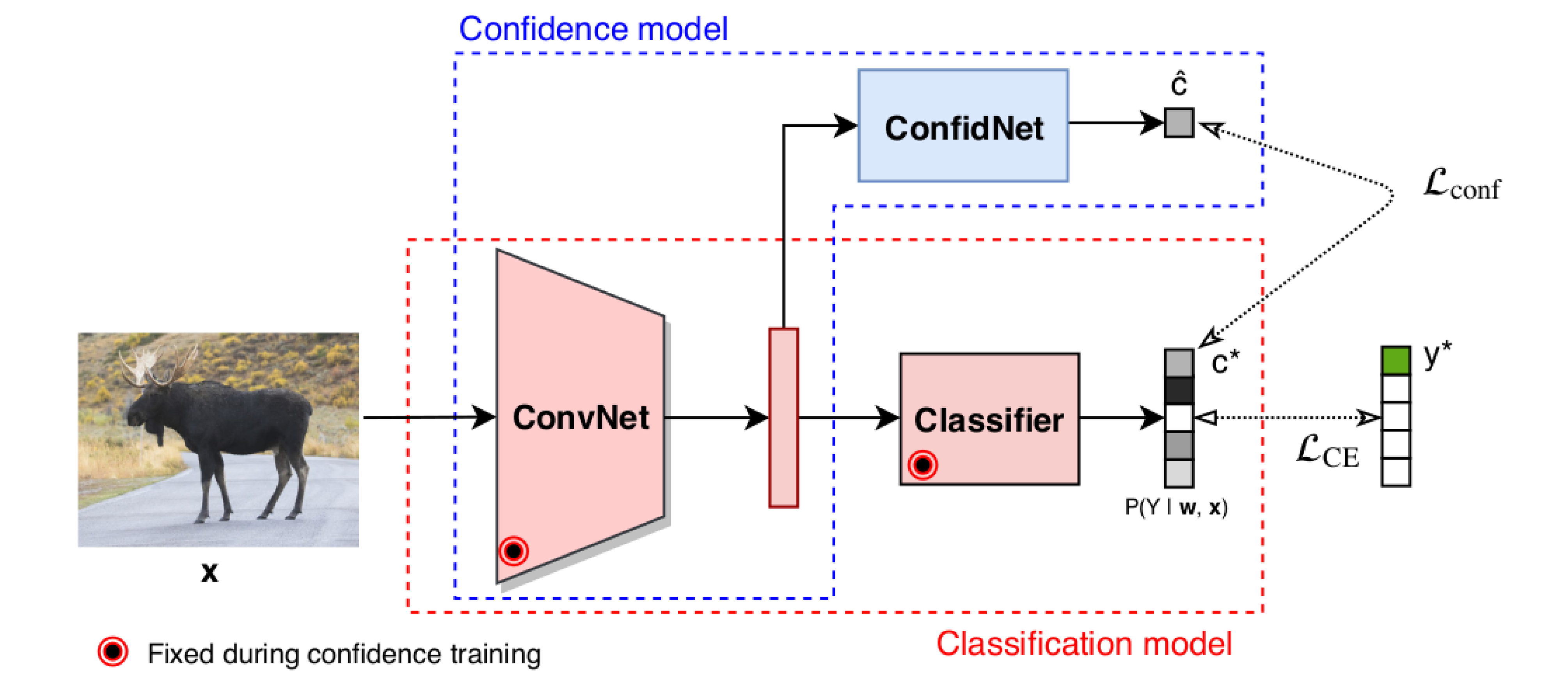
$$\text{TCP}: \mathbb{R}^d \times \mathcal{Y} \rightarrow \mathbb{R}$$
$$(\mathbf{x}, y^*) \rightarrow p(Y = y^*|\mathbf{w}, \mathbf{x})$$

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



ConfidNet: Learning TCP Confidence

$TCP(\mathbf{x}, y^*)$ **unknown** at test time
$\Rightarrow$ **Train a confidence neural network to learn TCP**



- **Learning scheme**: 1- fix classifier weights, 2- learn ConfidNet layers with $\mathcal{L}_{\text{conf}}$, 3- duplicate and fine-tune encoder ConvNet+ConfidNet

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

- **Architecture**: succession of dense layers + final sigmoid activation
- **ConfidNet**, a model- and task-agnostic training method to learn TCP

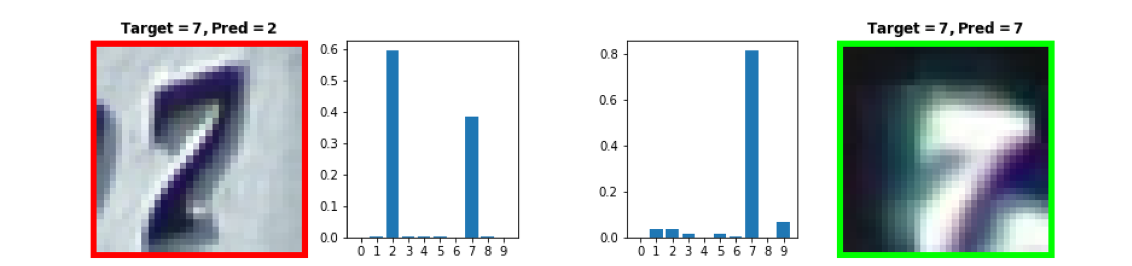
Experiments and visualisations

Comparative experiments

- Traditional public **image classification** and **semantic segmentation** datasets
- **ConfidNet outperforms confidence and uncertainty estimation baseline approaches**

AUPR_Error (%)	MNIST MLP	MNIST Small ConvNet	SVHN Small ConvNet	CIFAR-10 VGG16	CIFAR-100 VGG16	CamVid SegNet
Baseline (MCP) [1]	37.70	35.05	48.18	45.36	71.99	48.53
MCDropout [2]	38.22	38.50	43.87	46.40	72.59	49.35
TrustScore [3]	52.18	35.88	43.32	38.10	66.82	20.42
ConfidNet (Ours)	**57.37**	**45.89**	**50.72**	**49.94**	**73.68**	**50.51**

- **ConfidNet improves over direct failure prediction**: +0.72pt on SVHN, +1.99pt on CIFAR-10, +1.55pt on CamVid
- **Entropy** not always adequate

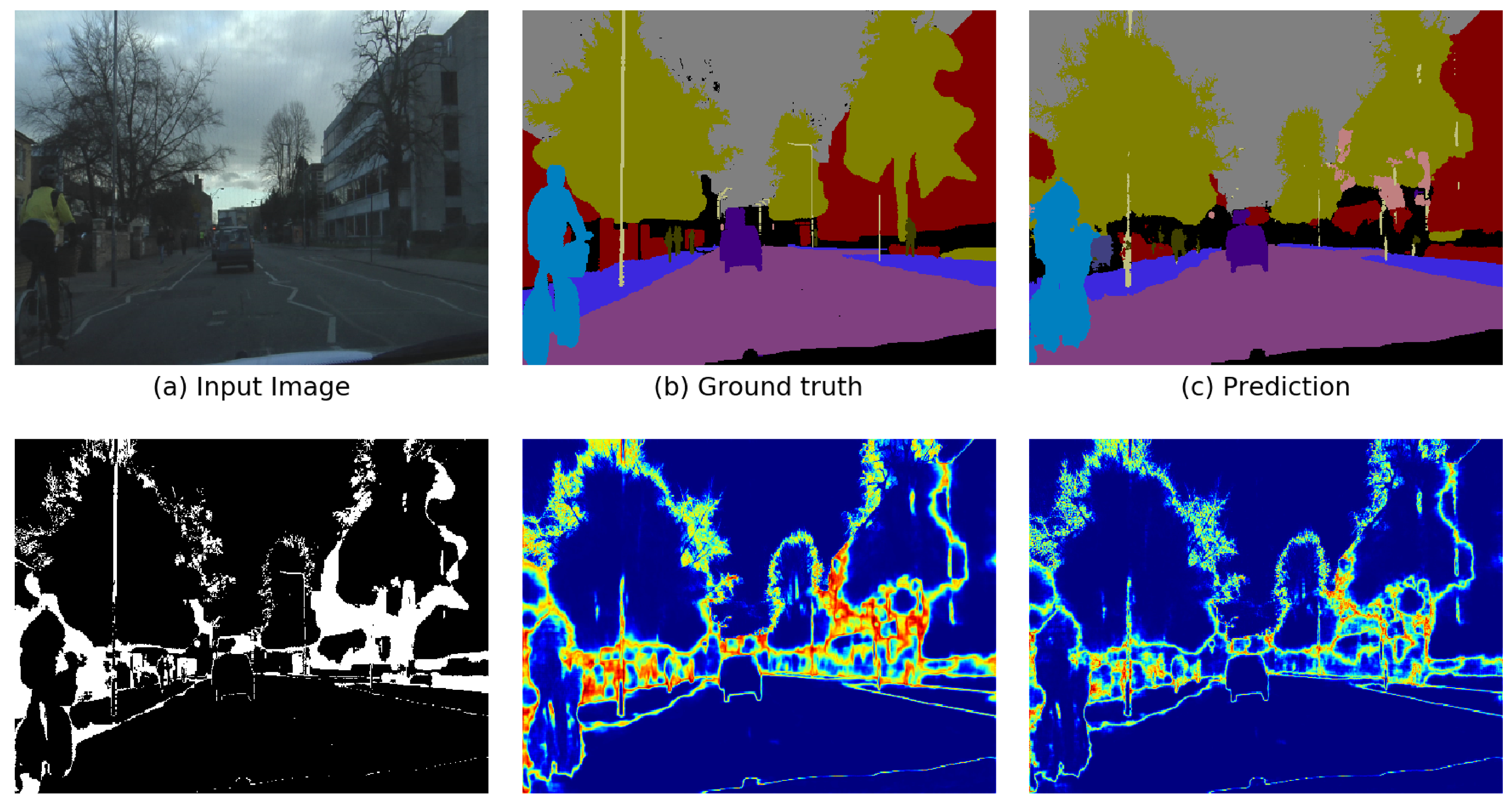


(a) MCP=0.596, MCDropout=-0.787, *ConfidNet*=0.449 (b) MCP=0.816, MCDropout=-0.786, *ConfidNet*=0.894

- Using a **val set** to train ConfidNet only improves if **low accuracy + large-scale**

Qualitative results

- Confidence estimation for semantic segmentation on CamVid dataset



(a) Input Image (b) Ground truth (c) Prediction
(d) Model Errors (e) ConfidNet (f) MCP

References

Paper Code

Code available: `https://github.com/valeoai/ConfidNet`

[1] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *ICLR*, 2017.

[2] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016.

[3] Heinrich Jiang, Been Kim, Melody Guan, and Maya Gupta. To trust or not to trust a classifier. In *NeurIPS*, 2018.