UNIVERSITY OF
OXFORD

Uncertainty Estimation Using a Single Deep Deterministic Neural Network

Paper ID: 4538

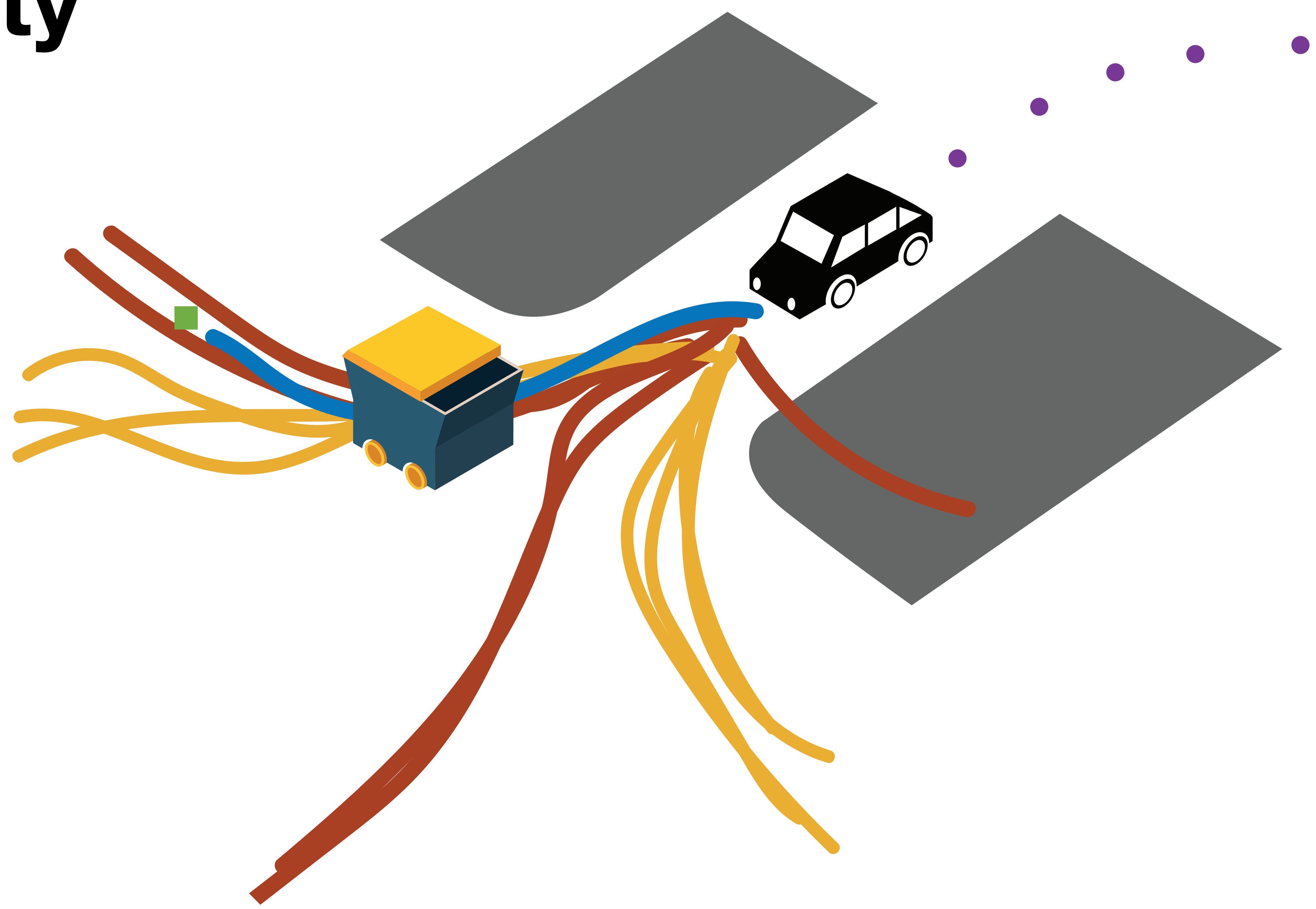
Joost van Amersfoort, Lewis Smith, Yee Whye Teh, Yarin Gal
Email: hi@joo.st

DUQ - 4 min Overview

Why do we want uncertainty?

Many applications need uncertainty

- **Self-driving cars**
- **Active Learning**
- **Exploration in RL**

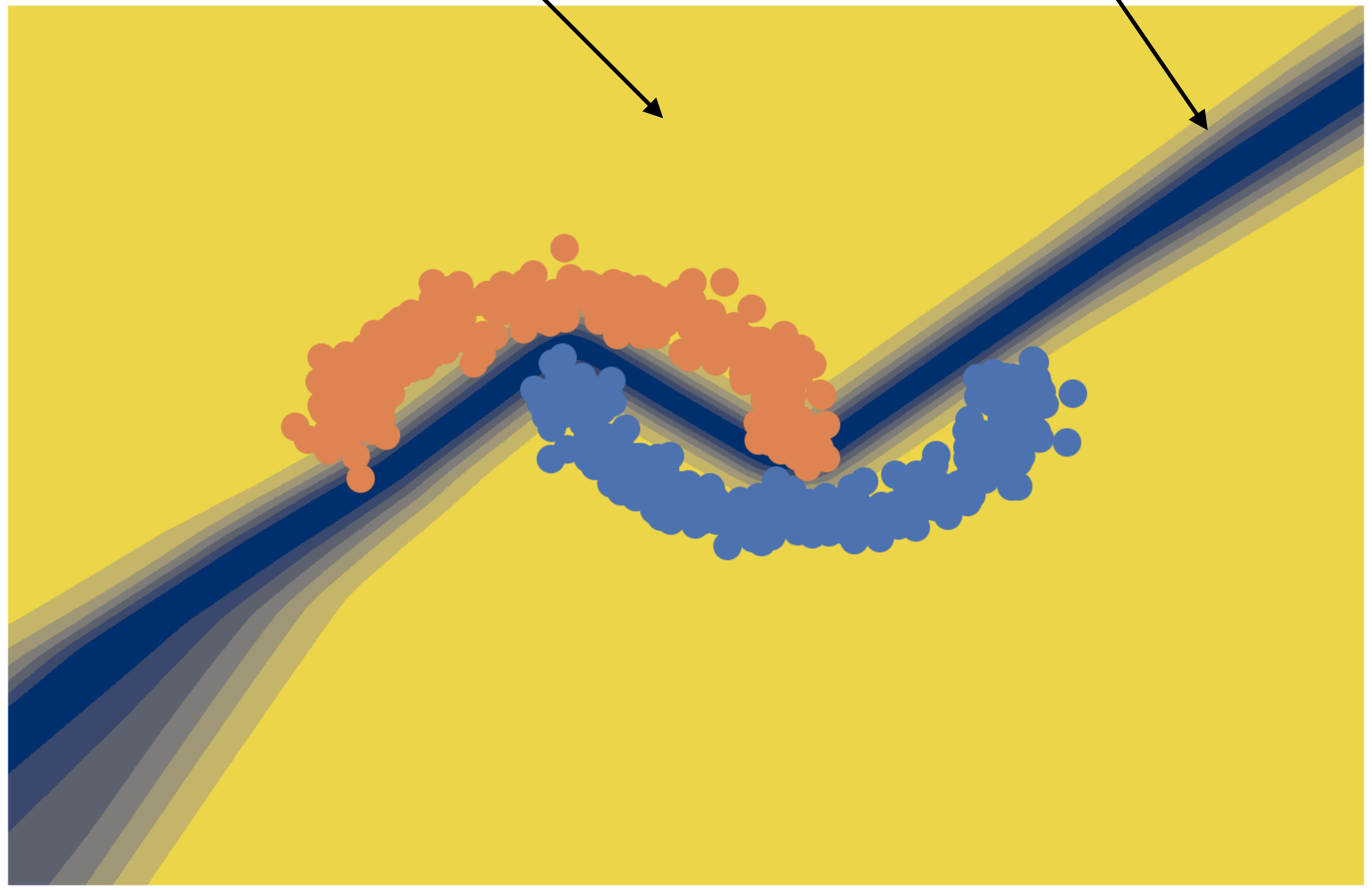


Deterministic Uncertainty Quantification (DUQ)

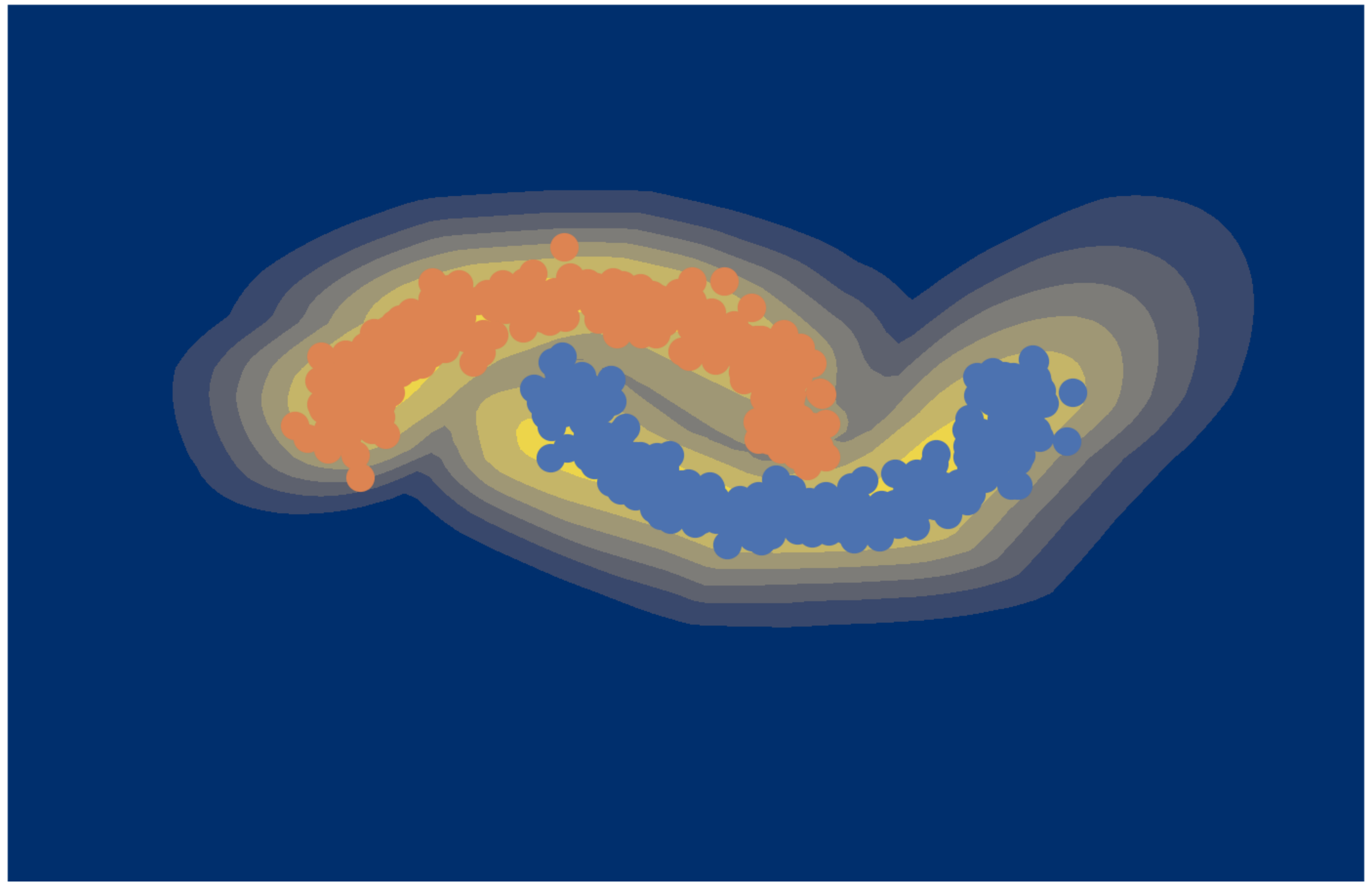
- A robust and powerful method to obtain uncertainty in deep learning
- Match or outperform Deep Ensembles uncertainty with the **runtime cost of a single network**
- Does not extrapolate arbitrarily and is able to detect OoD data

Two Moons

Certain Uncertain



Deep Ensembles[1]

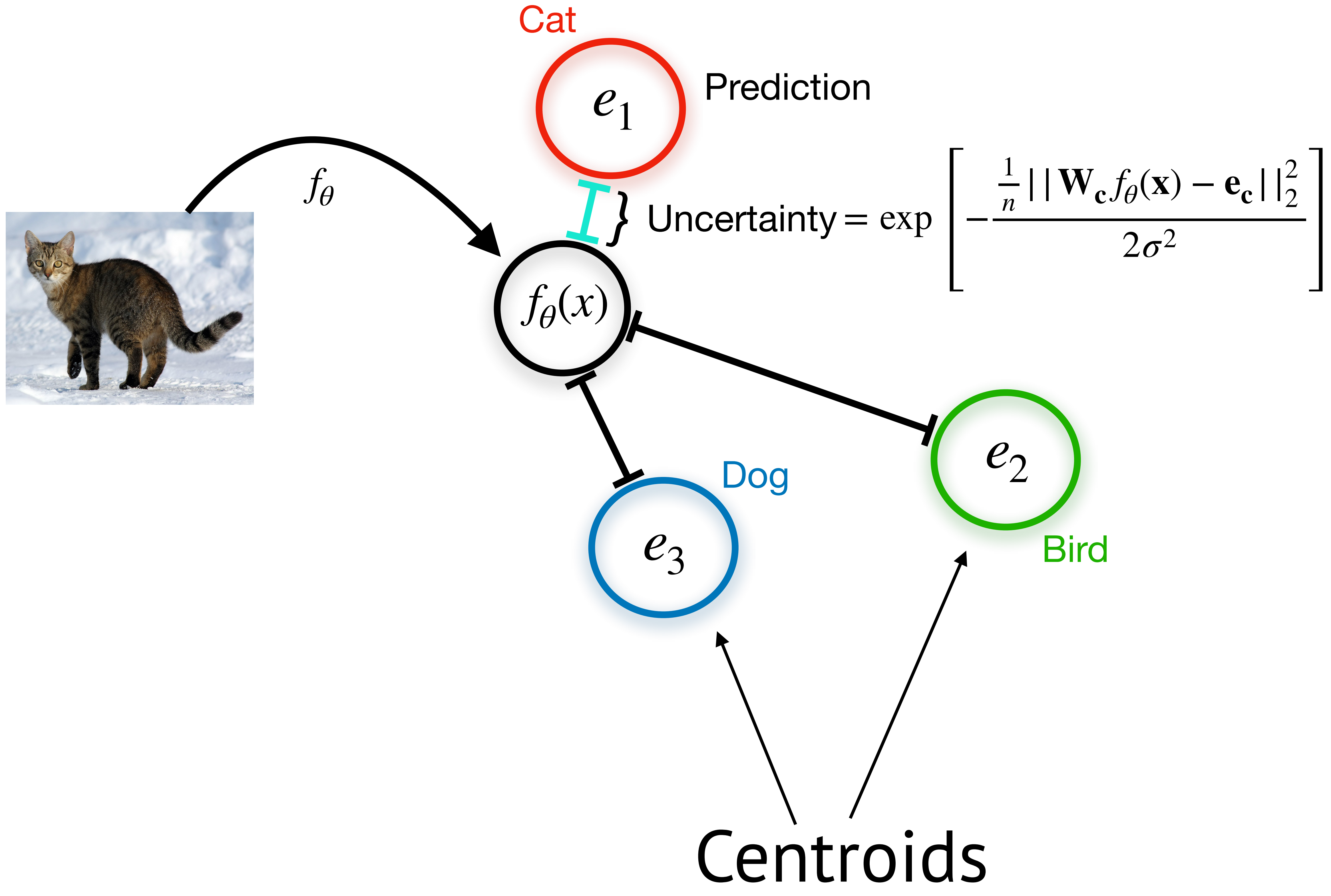


DUQ

(1) Lakshminarayanan, Balaji, Alexander Pritzel, and Charles Blundell. "Simple and scalable predictive uncertainty estimation using deep ensembles." *Advances in neural information processing systems*. 2017.

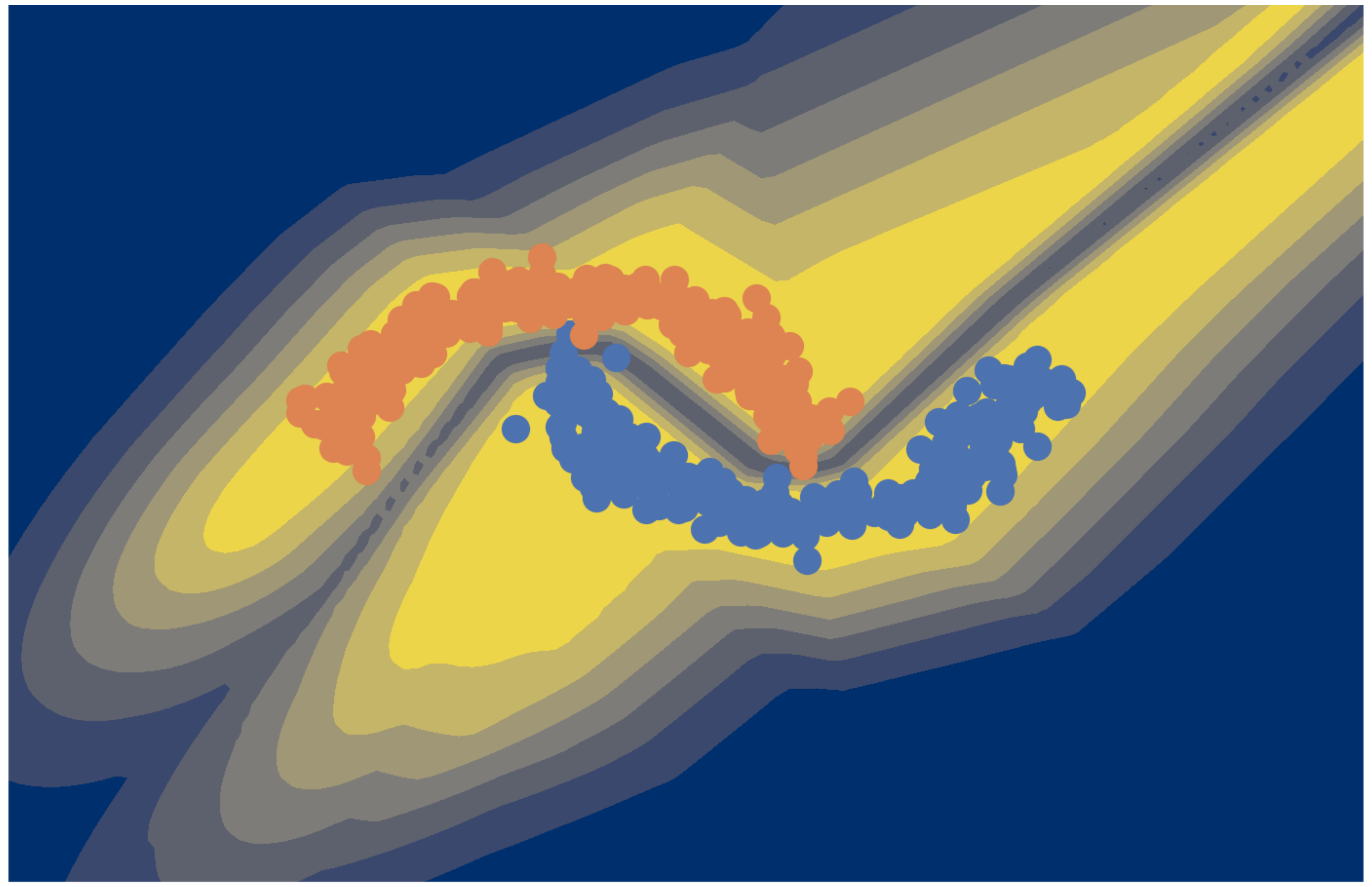
The Model

- Uncertainty = **distance between feature representation and closest centroid**
- Deterministic, cheap to calculate
- Old idea based on RBF networks

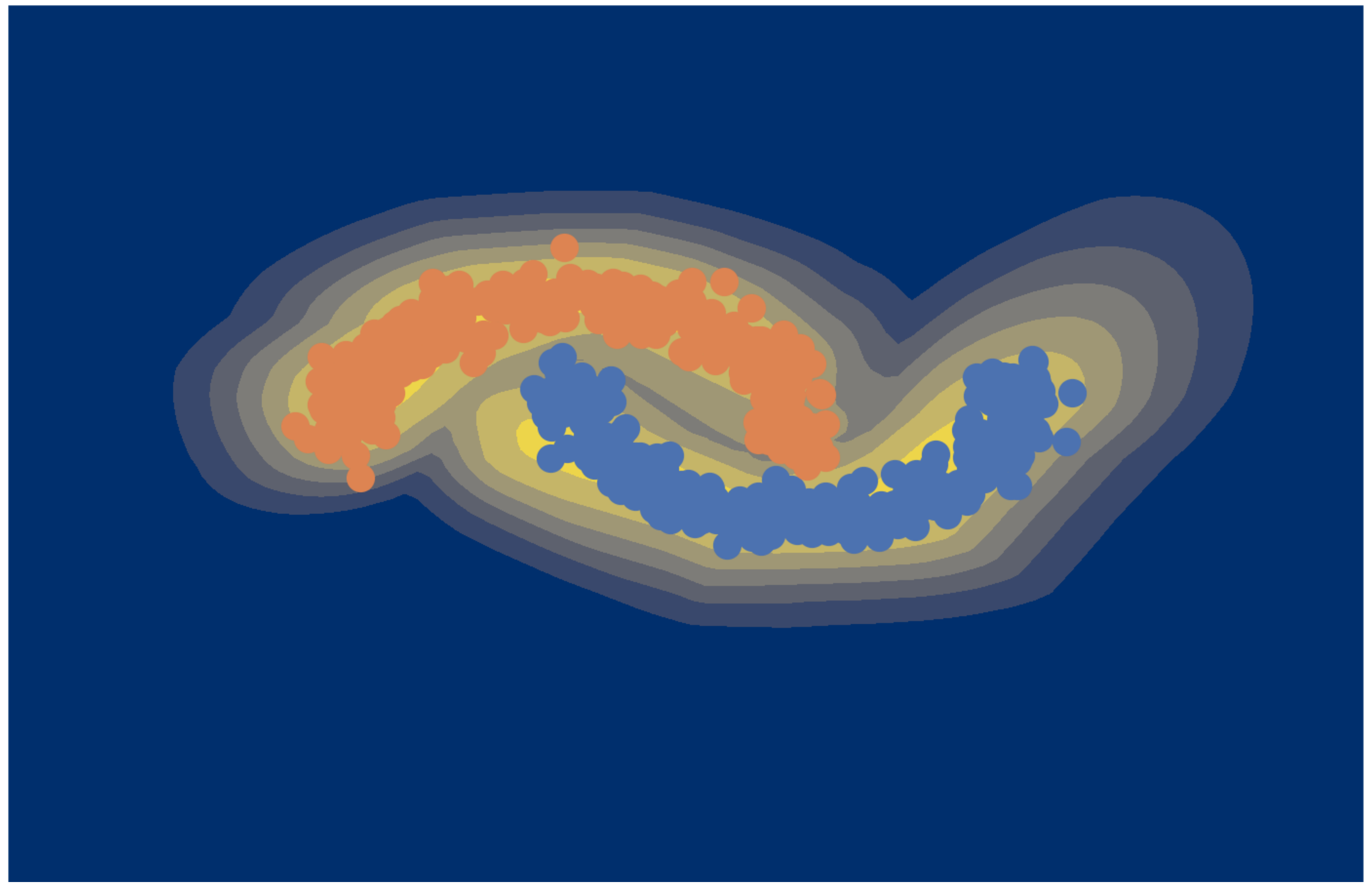


Overview

- Use "One vs Rest" loss function to update model $f_\theta(x)$
- Update centroids with exponential moving average
- Regularise centroids to stay close to origin
- Need $f_\theta(x)$ to be well behaved → penalty on the Jacobian



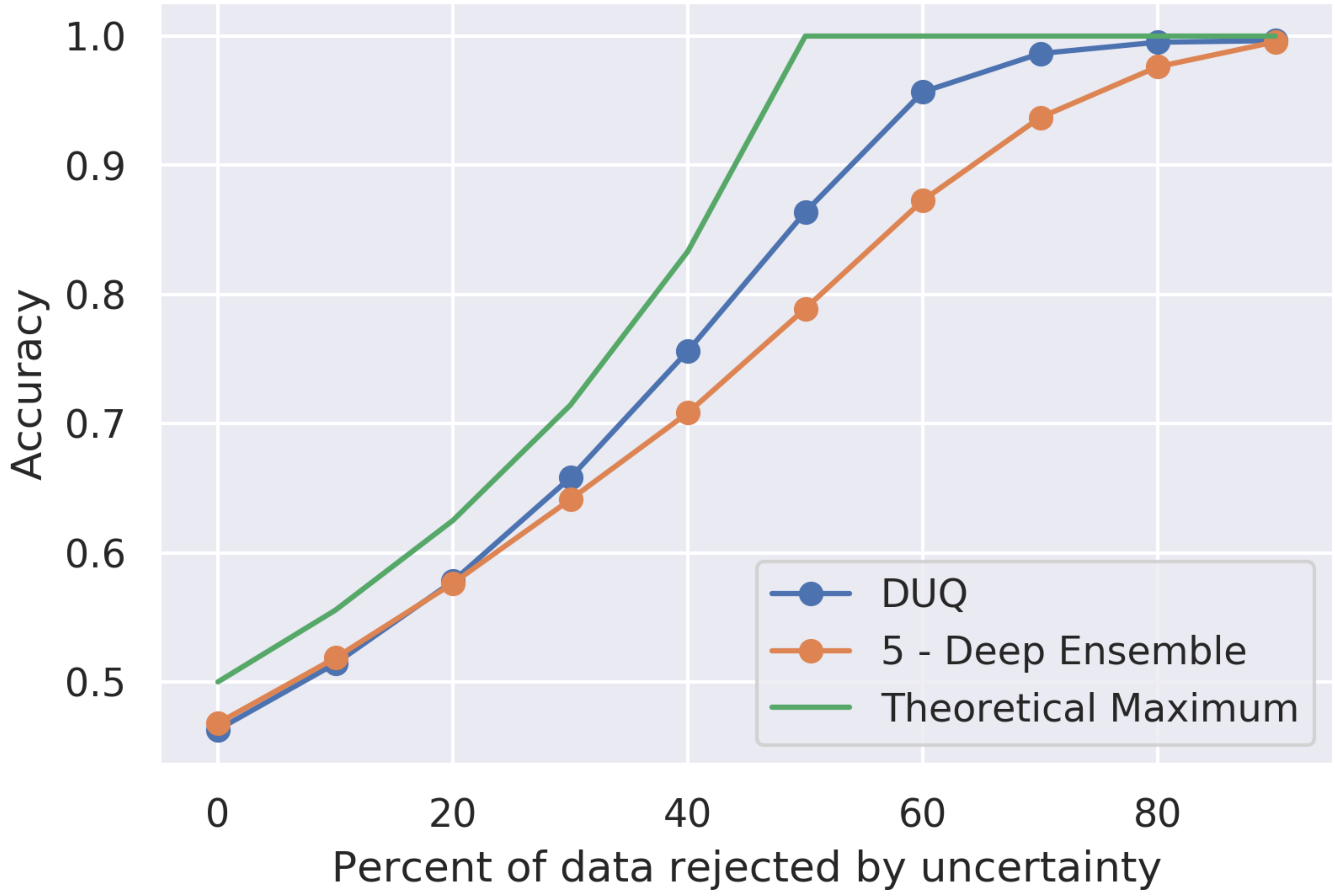
Standard RBF



DUQ

Results

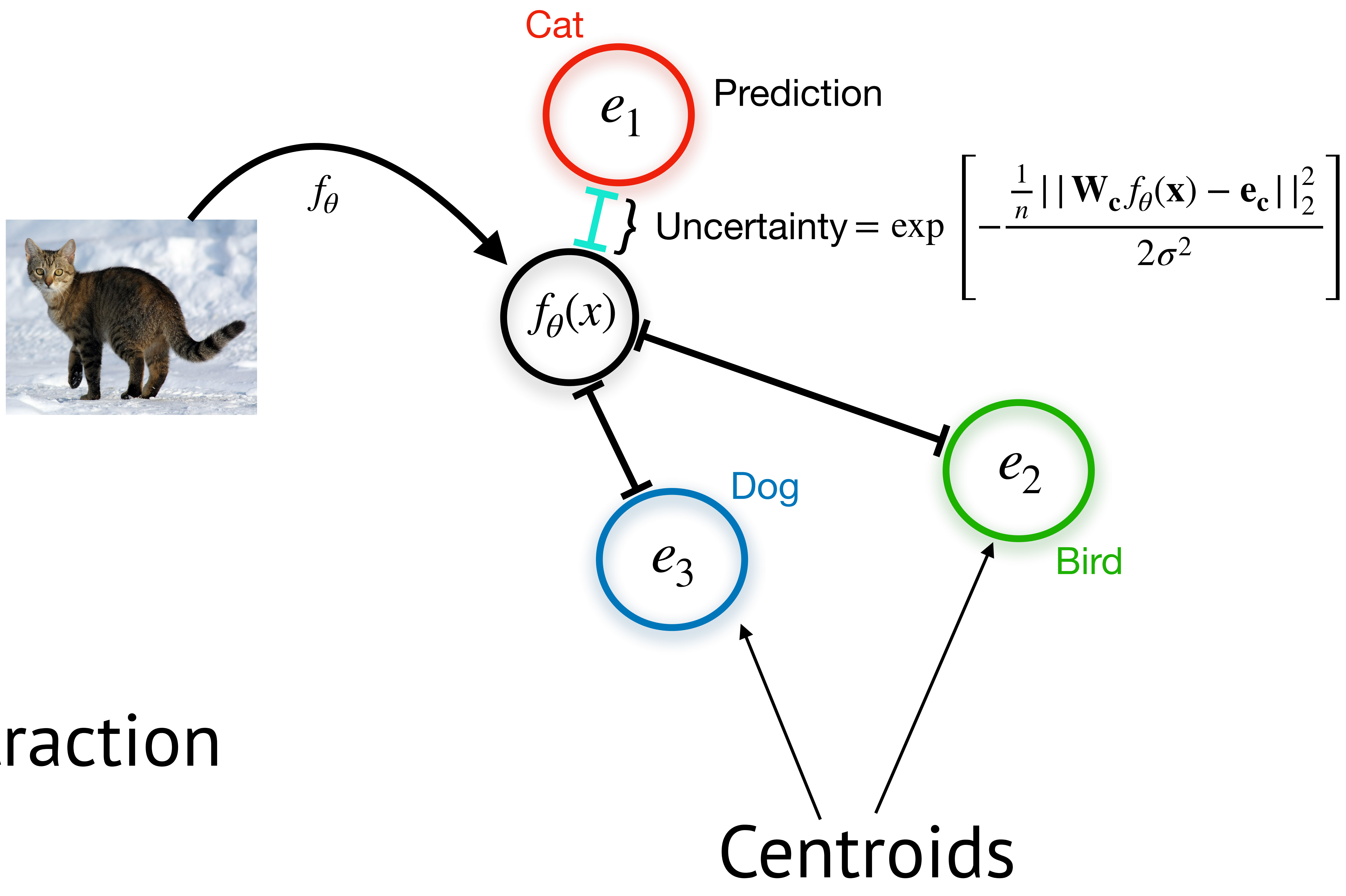
- Training is easy and stable
- Accuracy same as common softmax networks
- Match or outperform Deep Ensembles uncertainty with the **runtime cost of a single network**



Train on FashionMNIST
Evaluate on FashionMNIST + MNIST

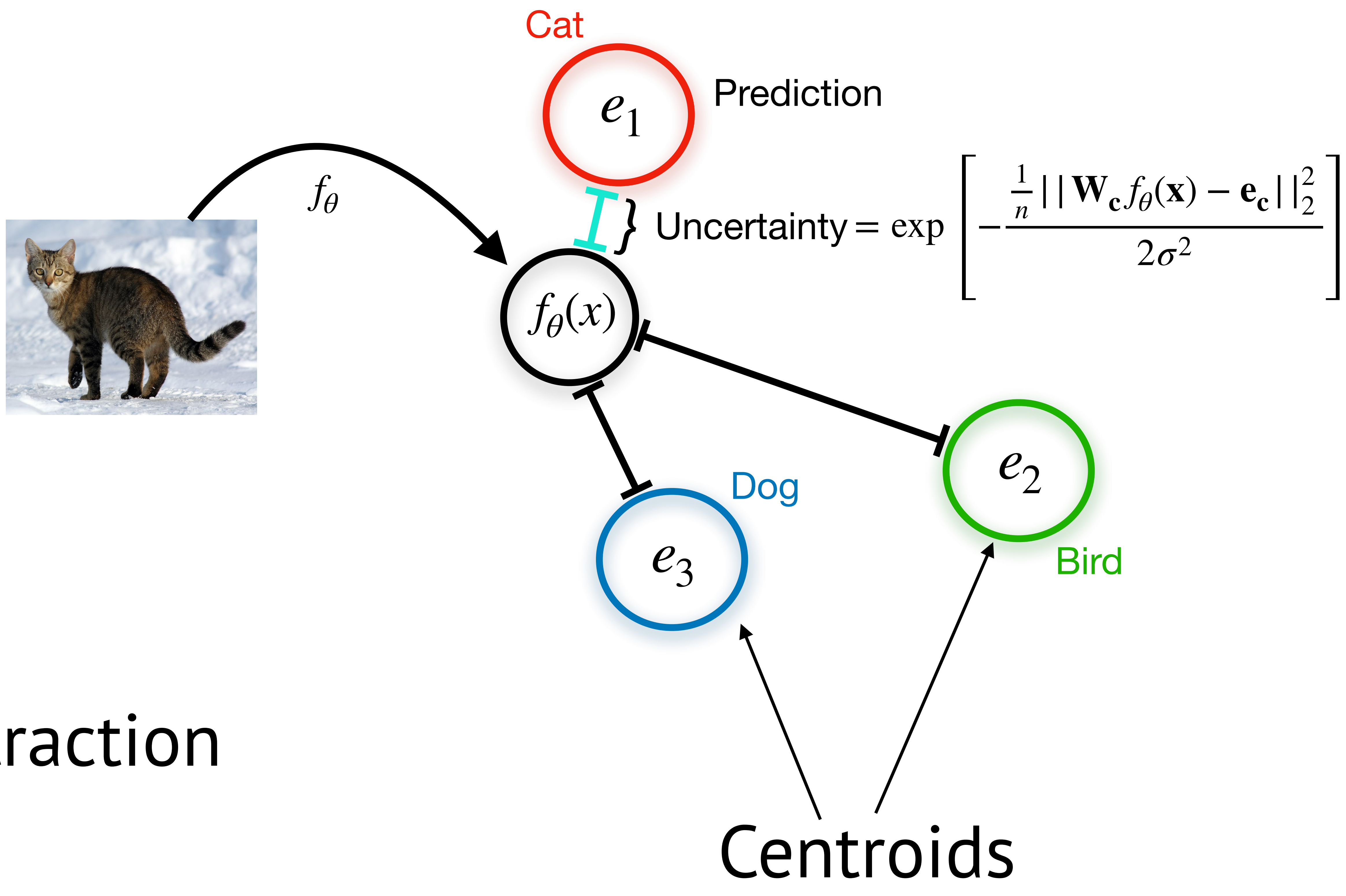
DUQ - Deep(er) Dive

Uncertainty Estimation



- Uncertainty estimation for classification
- Use a deep neural network for feature extraction
- Single centroid per class
- Define uncertainty as distance to closest centroid in feature space

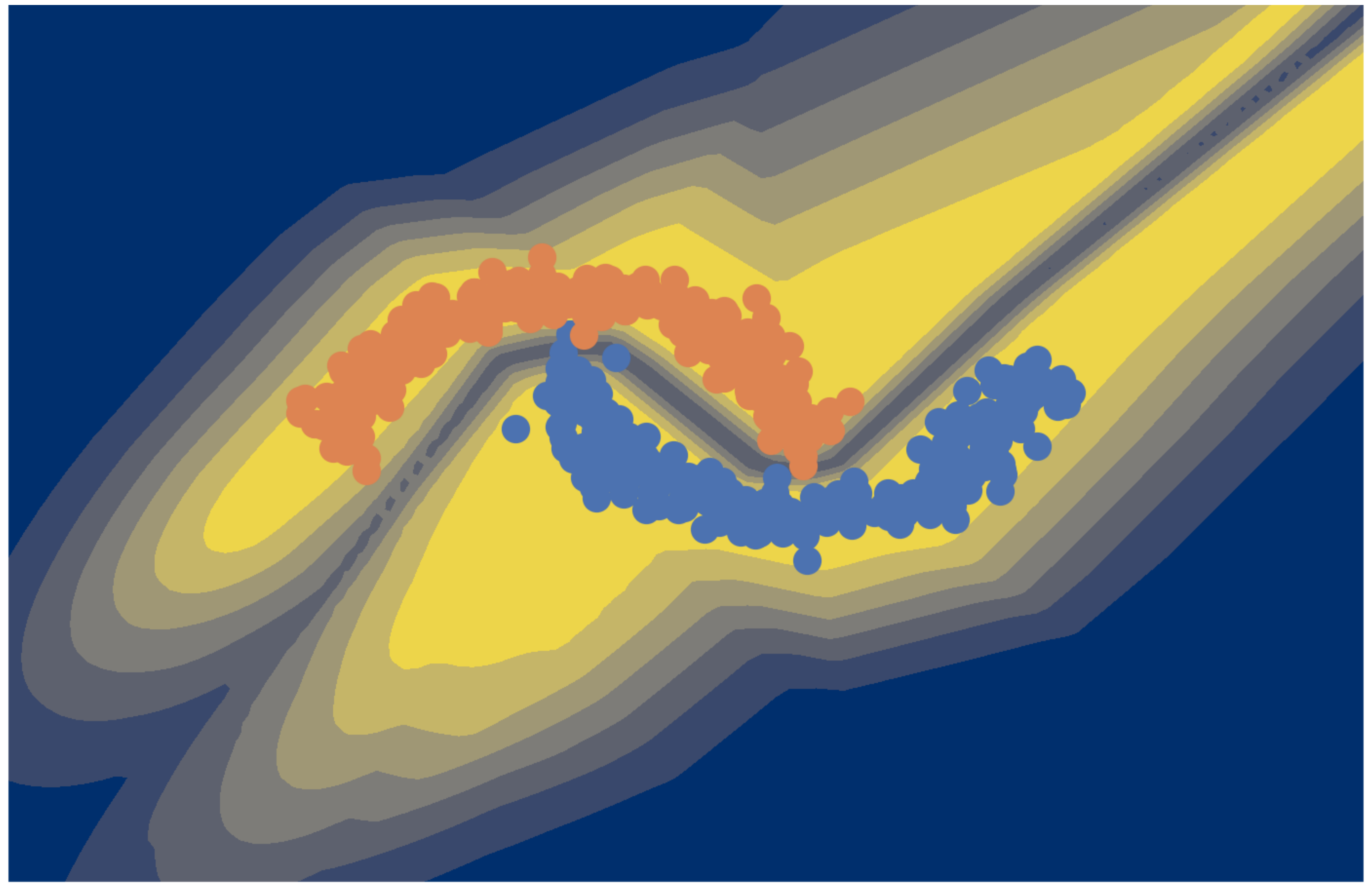
Uncertainty Estimation



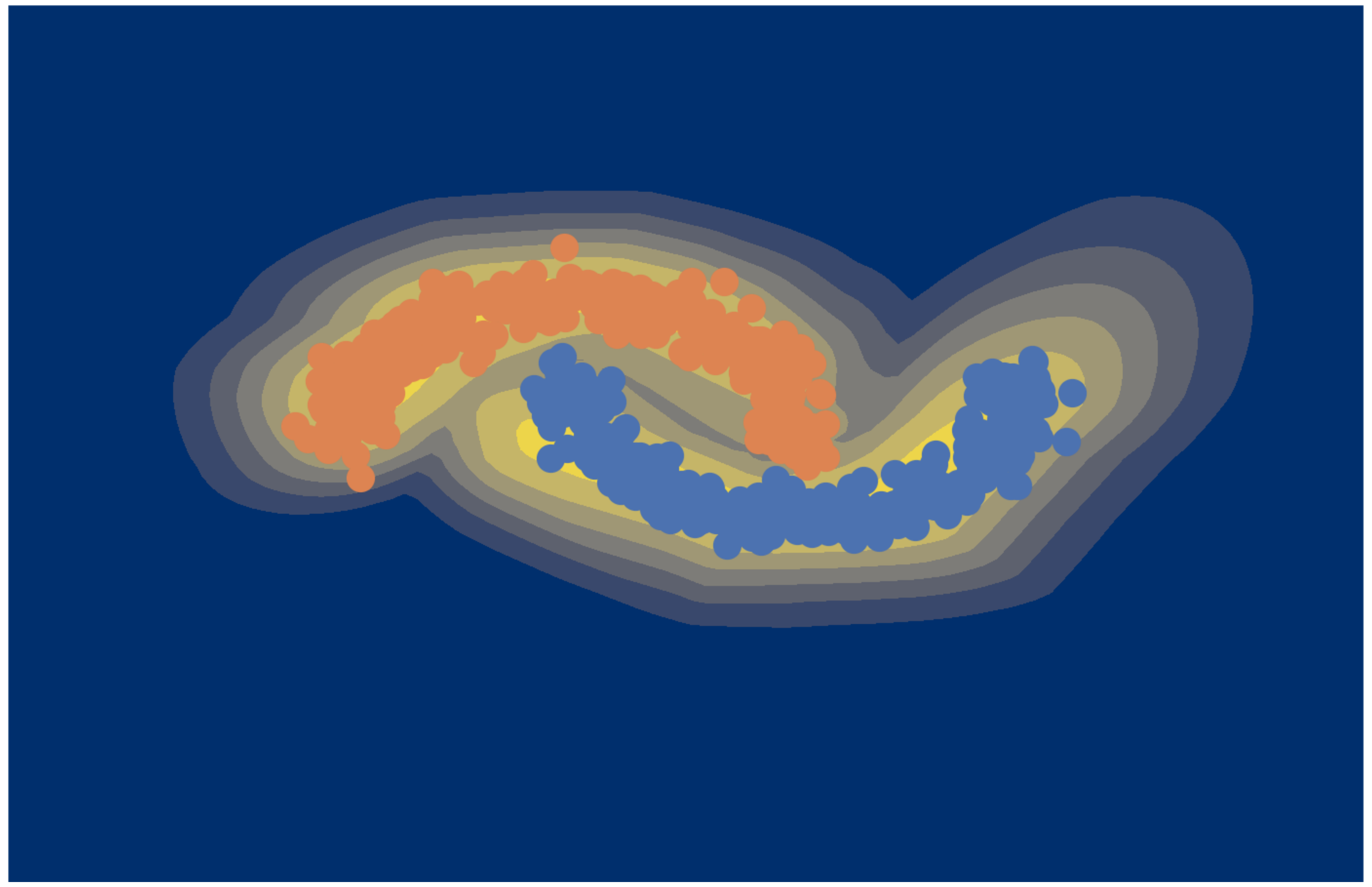
- Uncertainty estimation for classification
- Use a deep neural network for feature extraction
- Single centroid per class
- Define uncertainty as distance to closest centroid in feature space
- Deterministic and single forward pass!

DUQ - Overview

- **Use "One vs Rest" loss function to update model $f_\theta(x)$**
- **Update centroids with exponential moving average**
- Regularise centroids to stay close to origin
- Need $f_\theta(x)$ to be well behaved → penalty on the Jacobian



Standard RBF



DUQ

Learning the Model

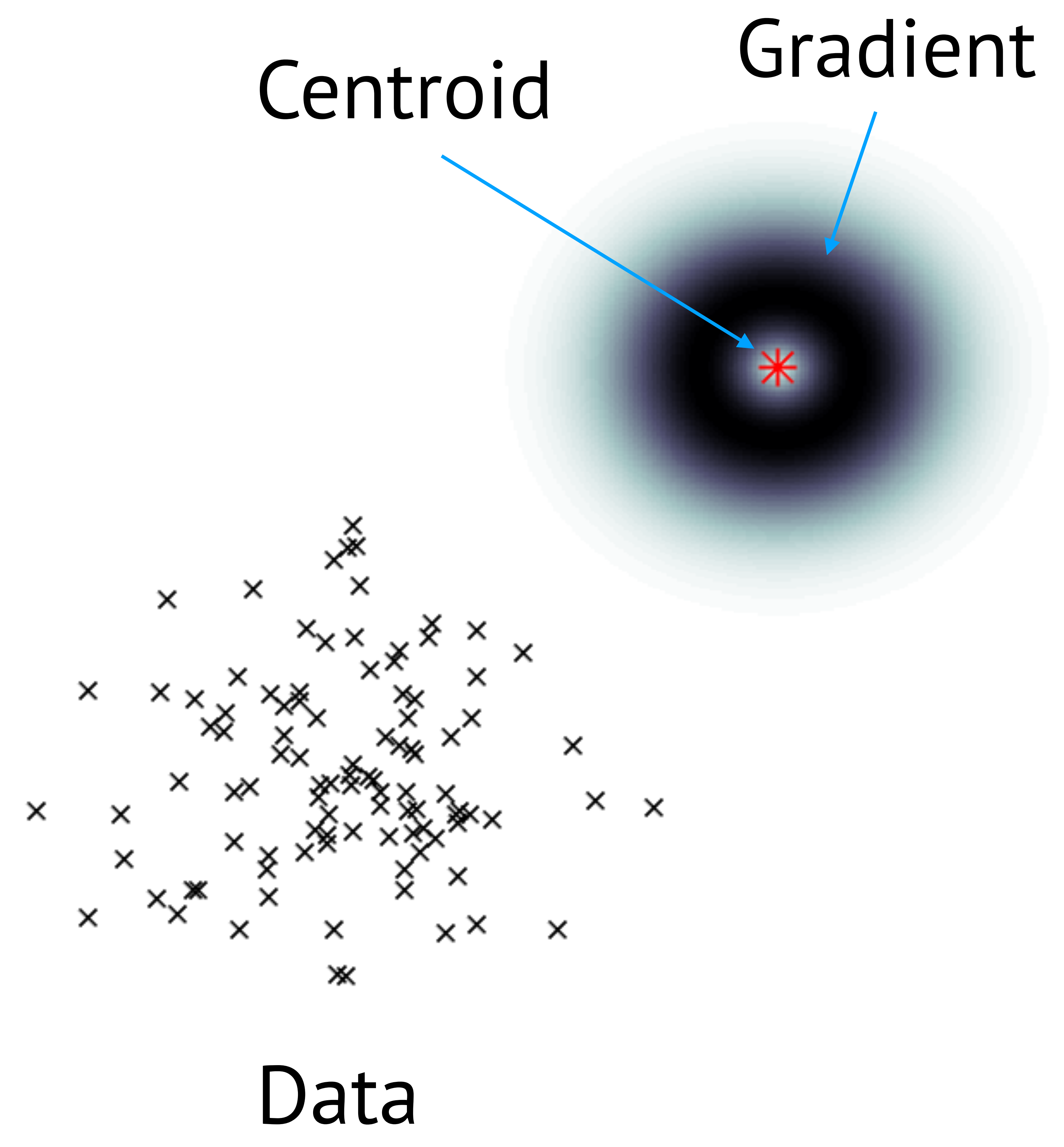
- “One vs Rest” loss function
- Decrease distance to correct centroid, while increasing it relative to all others
- Avoids centroids collapsing on top of each other
- Regularisation avoids centroids exploding

$$K_c(f_\theta(\mathbf{x}), \mathbf{e}_c) = \exp\left[-\frac{\frac{1}{n}\|\mathbf{W}_c f_\theta(\mathbf{x}) - \mathbf{e}_c\|_2^2}{2\sigma^2}\right]$$

$$L(\mathbf{x}, \mathbf{y}) = -\sum_c y_c \log(K_c) + (1 - y_c)\log(1 - K_c)$$

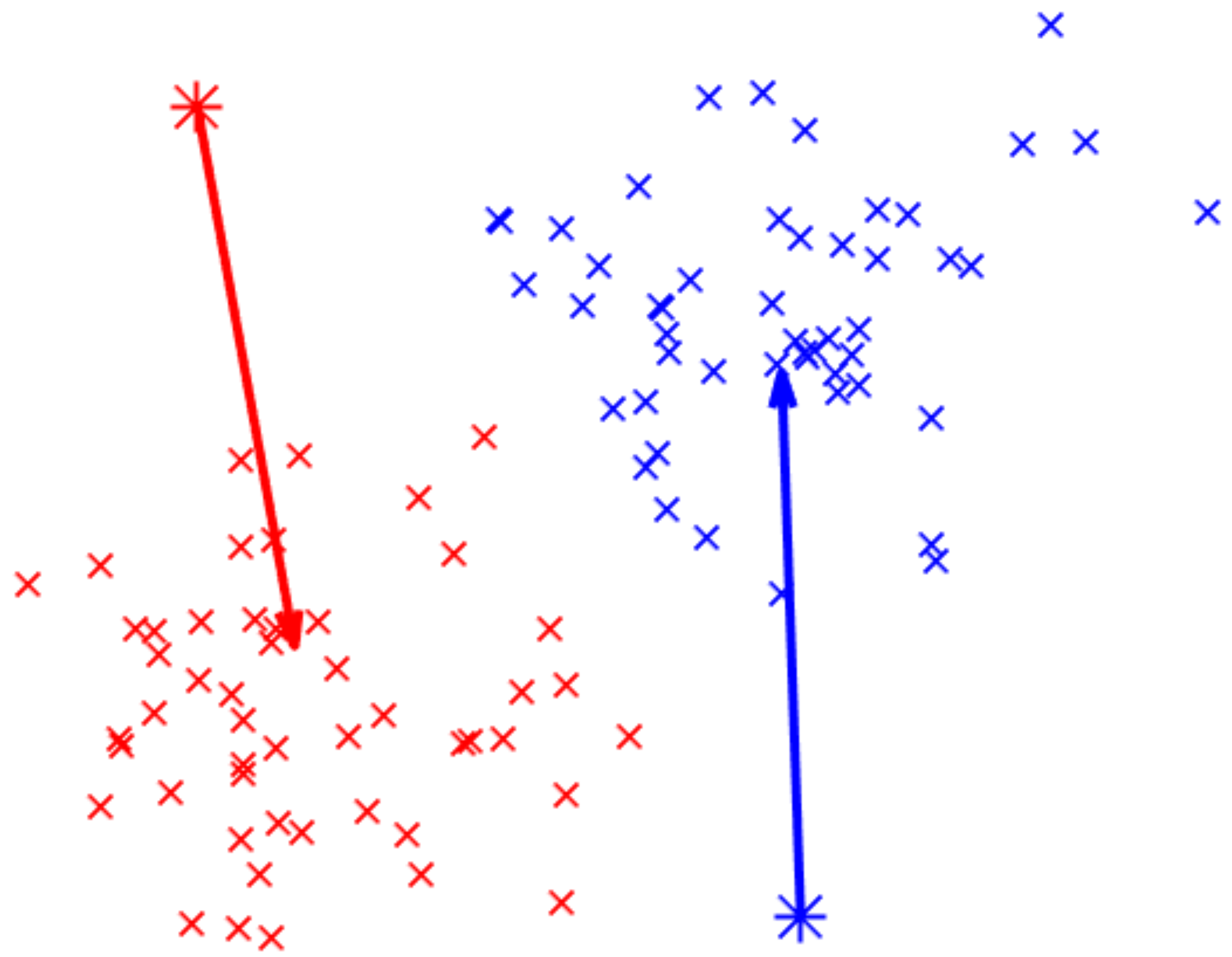
Learning the Centroids

- Exponential distance from centroid is bad for gradient based learning
- When far away from the correct centroid, the gradient goes to zero
- No learning signal for model



Learning the Centroids

- Move each centroid to the mean of the feature vector of that class
- Use exponential moving average with heavy momentum to make this work with mini-batches.



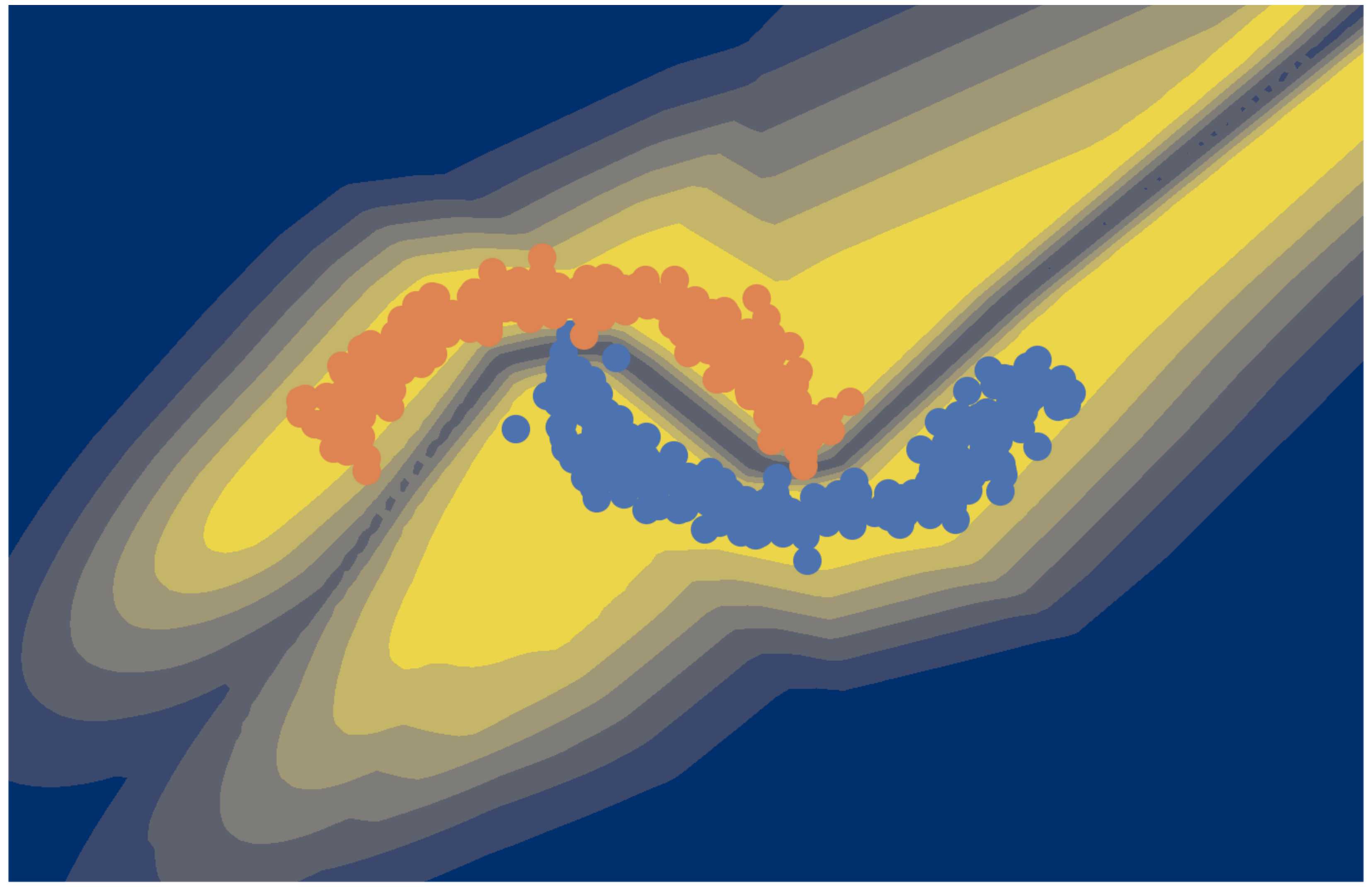
Centroid moves towards the data

$$n_{c,t} = \gamma * n_{c,t-1} + (1-\gamma) * n_{c,t}$$
$$\mathbf{m}_{c,t} = \gamma * \mathbf{m}_{c,t-1} + (1-\gamma) \sum_i \mathbf{W}_c f_\theta(\mathbf{x}_{c,t,i})$$
$$\mathbf{e}_{c,t} = \frac{\mathbf{m}_{c,t}}{n_{c,t}}$$

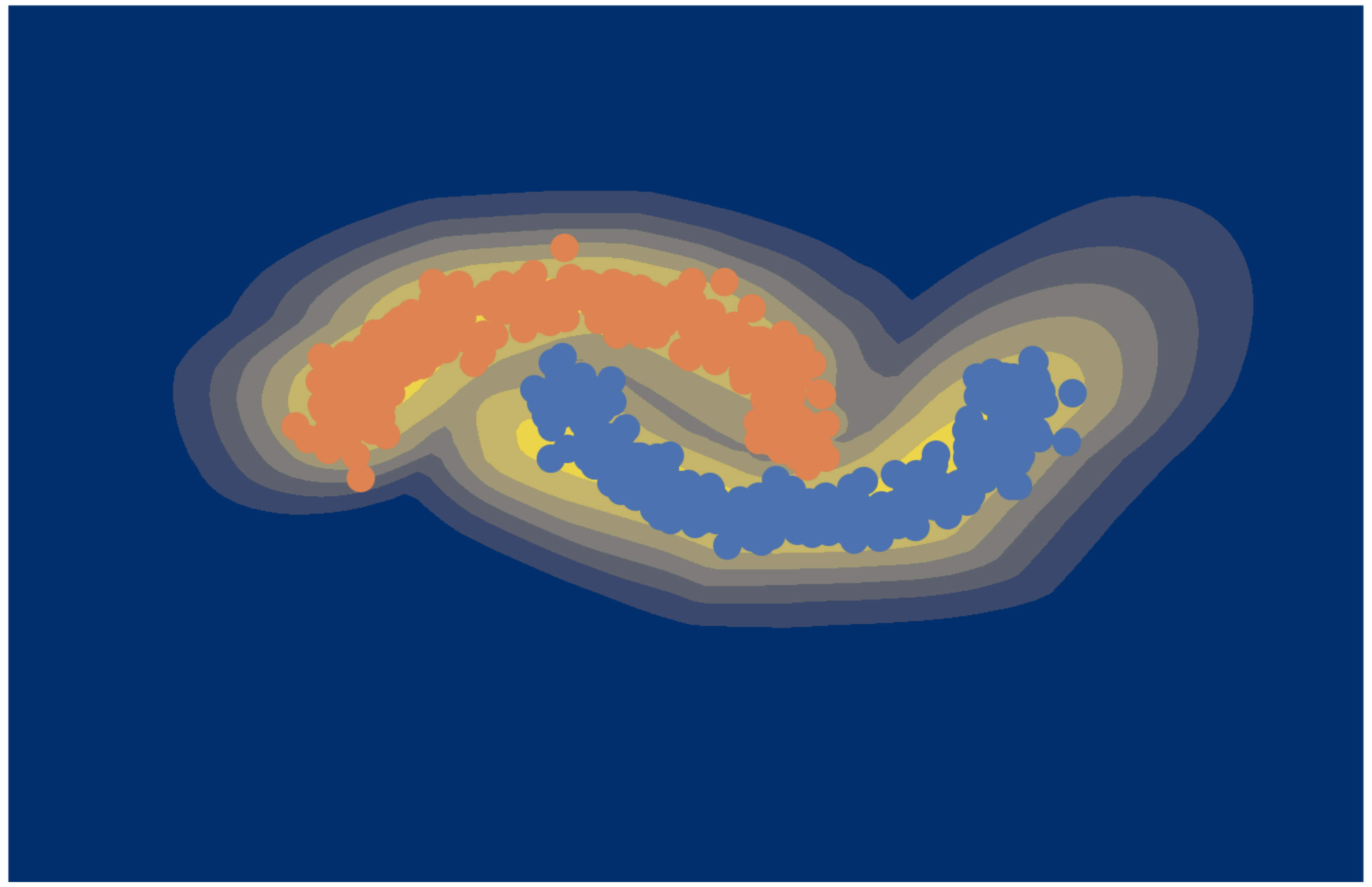
Set to 0.99(9)

DUQ - Overview

- Use "One vs Rest" loss function to update model $f_\theta(x)$
- Update centroids with exponential moving average
- Regularise centroids to stay close to origin
- **Need $f_\theta(x)$ to be well behaved → penalty on the Jacobian**



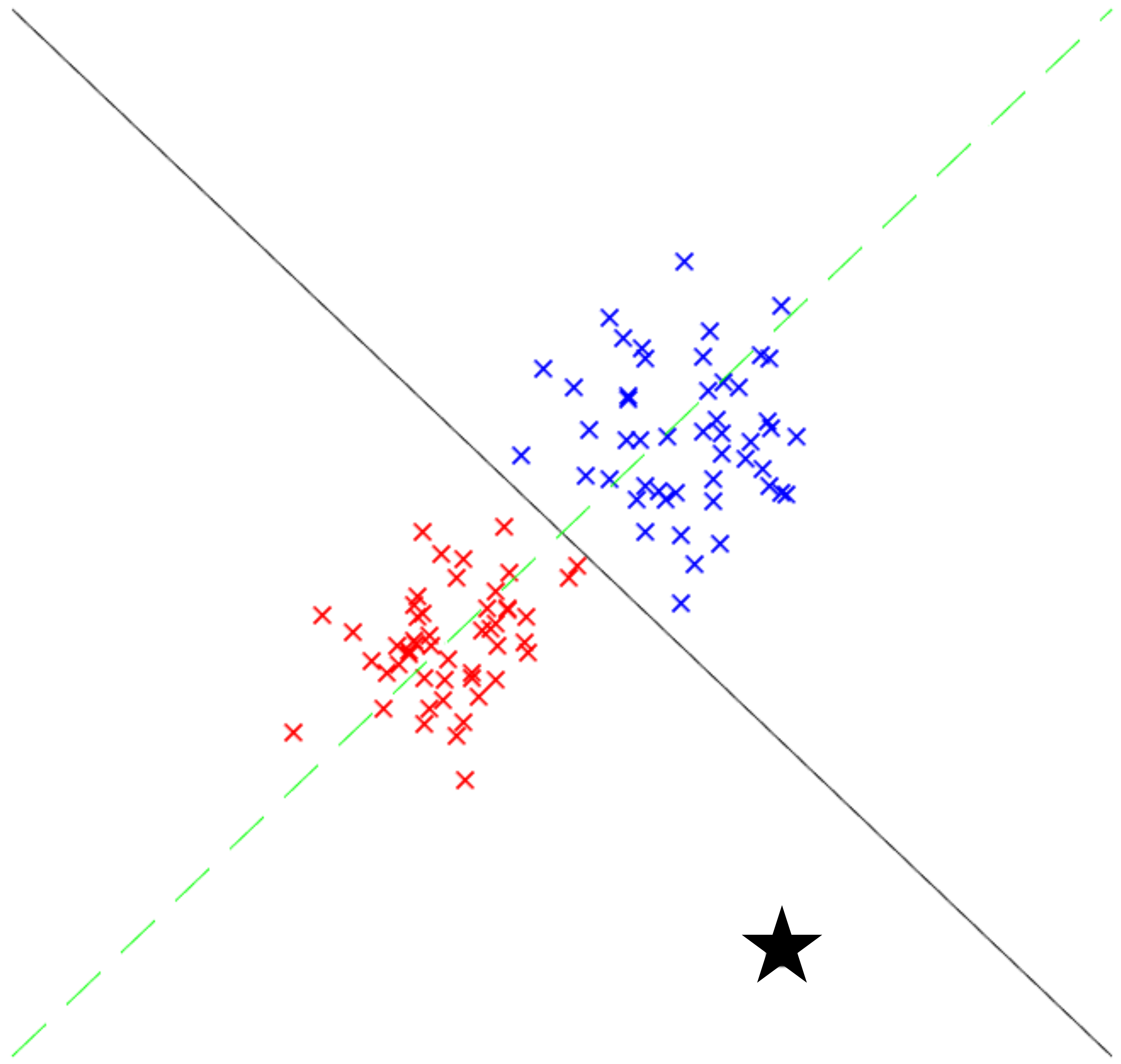
Standard RBF



DUQ

Why do we need to regularise *f*?

- Classification is at odds with being able to detect OoD input
- **Is the black star OoD?**
- Classification means we **ignore features that don't affect the class**



Stability & Sensitivity

- Two-sided gradient penalty
- From **above**: low Lipschitz constant - commonly used
- From **below:** *sensitive* to changes in the input

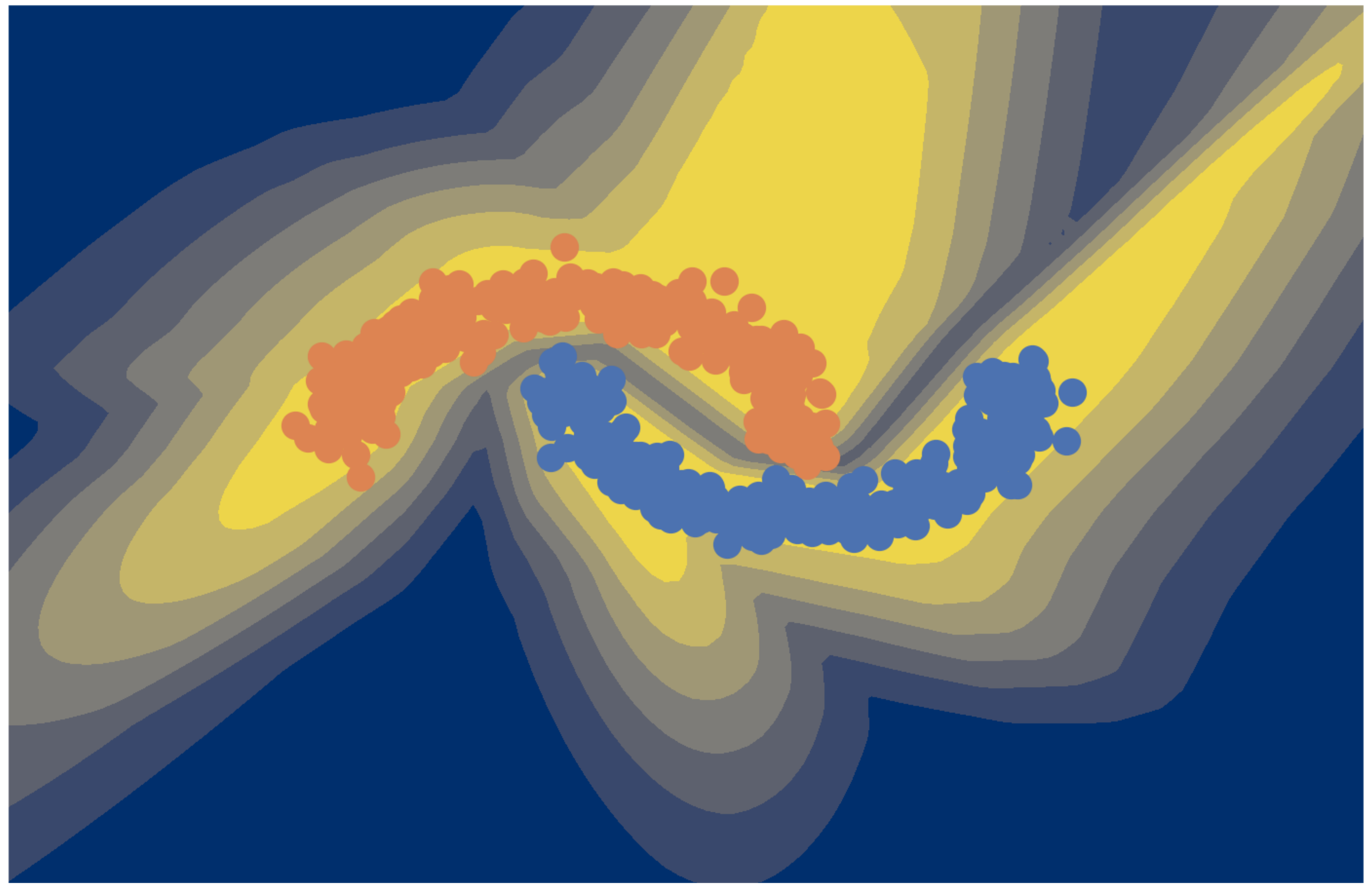
  $||f(x) - f(x + \delta)|| > L$

$$\lambda \cdot \left[||\nabla_x \sum_c K_c||_2^2 - L \right]^2$$

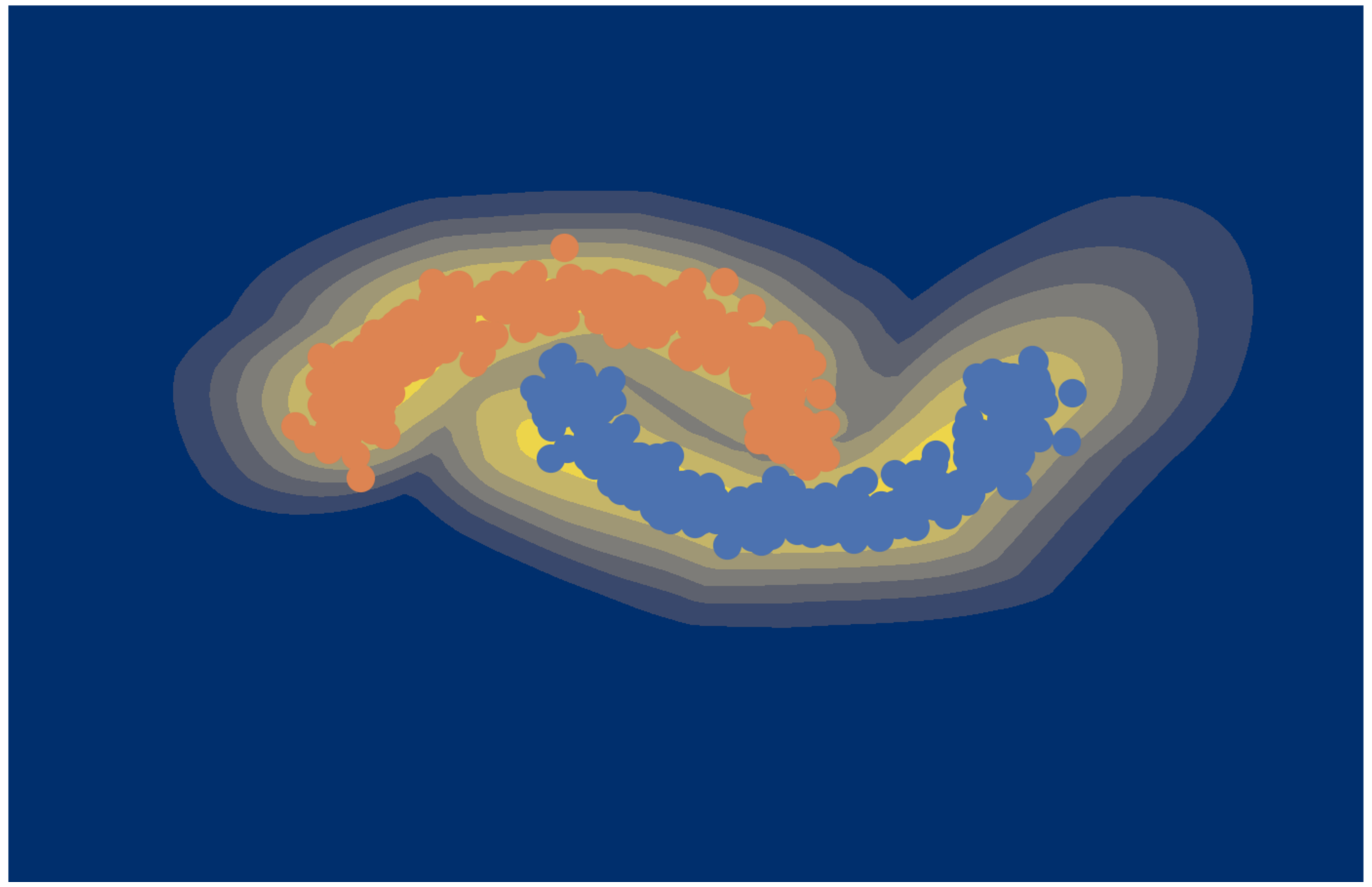
Stability & Sensitivity

- Two-sided gradient penalty
- From **above**: low Lipschitz constant - commonly used
- From **below:** *sensitive* to changes in the input

$||f(x) - f(x + \delta)|| > L$



DUQ - penalty from above



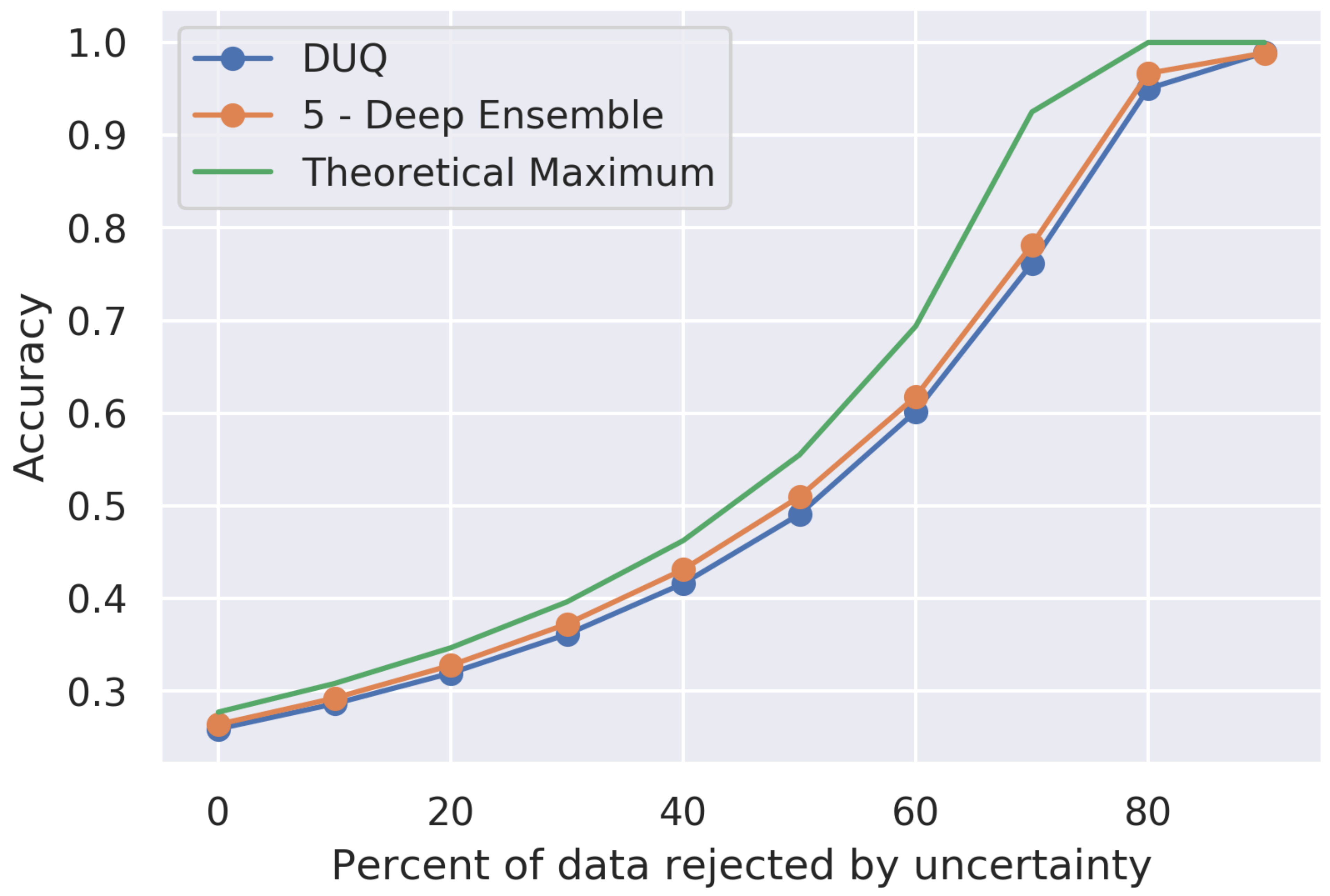
DUQ - two sided penalty

Results

- **FashionMNIST** vs **MNIST**
  Out of Distribution detection

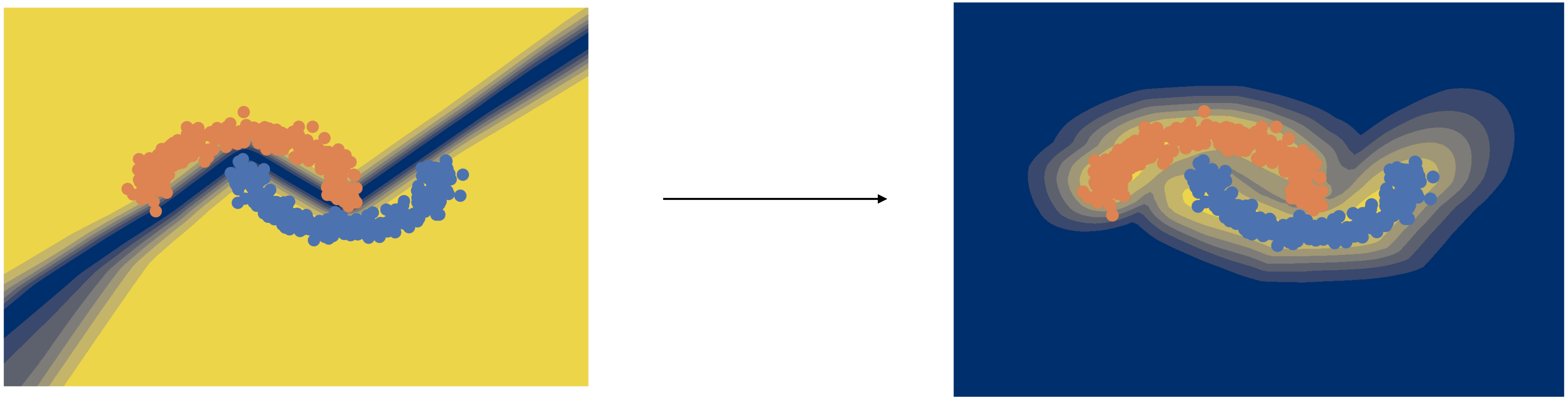
Method	AUROC
DUQ	0.955
Single model	0.843
5 - Deep Ensembles (ours)	0.861
5 - Deep Ensembles (ll)	0.839
Mahalanobis Distance (ll)	0.942

- Rejection Classification on **CIFAR-10** (training set) and **SVHN** (out of distribution set)



Summary

- A robust and powerful method to obtain uncertainty in deep learning
- Match or outperform Deep Ensembles[1] uncertainty with the **runtime cost of a single network**
- No arbitrary extrapolation and able to detect OoD data



Limitations and Future Work

- *Aleatoric Uncertainty*. DUQ is not able to estimate this. The one class per centroid system makes training stable, but does not allow assigning a data point to multiple classes.
- *Probabilistic Framework*. DUQ is not placed in a probabilistic framework, however there are interesting similarities to inducing point GPs with parametrised ("deep") kernels

Come chat with us

Time slots available on ICML website - Missed us? Email at hi@joo.st

Joost van Amersfoort

Lewis Smith

Yee Whye Teh

Yarin Gal