max planck institut
informatik

Leveraging uncertainty information from deep neural networks for disease detection

C. Leibig, V. Allken, P. Berens, and S. Wahl

bioRxiv:084210 (2.8.2017), doi:10.1101/084210

Deep learning reading group

Presented by Lisa Handl

August 31, 2017

Motivation

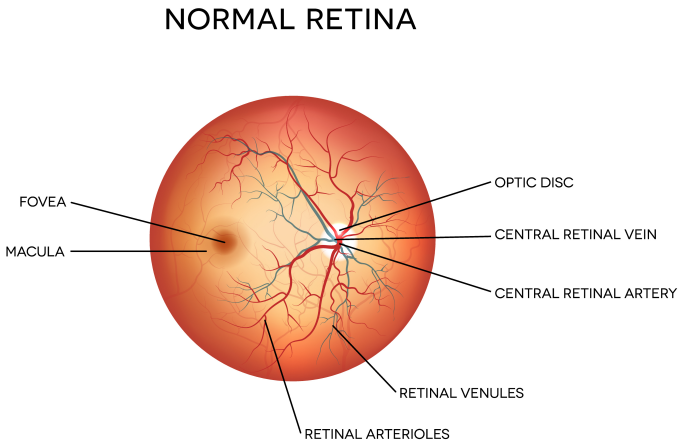
- Deep neural networks provide outstanding prediction accuracy on many challenging problems
- Can and should be used in medical applications
  - Medical image classification
  - Automatic diagnosis

Motivation

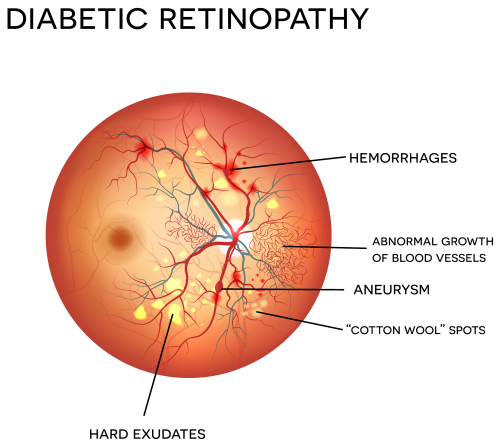
- Deep neural networks provide outstanding prediction accuracy on many challenging problems
- Can and should be used in medical applications
  - Medical image classification
  - Automatic diagnosis
- Drawback: no measure of uncertainty
- This is crucial for medical applications
  - Difficult diagnostic cases
  - No clear-cut boundary between healthy/diseased
- If uncertainty information was available, difficult cases could receive additional attention

Diabetic Retinopathy (DR)

- One of the leading causes of blindness in the working-age population of the developed world
- Can be diagnosed based on fundus images
- Early detection allows to stop progression to vision impairment
- Increasing incidence of diabetes demands effective screening



1



[1] Images from http://wjscottmd.com/wp-content/uploads/2015/11/Diabetic-Retinopathy.jpg

How Can Uncertainty Be Measured?

- Bayesian perspective could be useful
- Combinations of deep NNs and Bayesian ideas are an active topic of research ([23]-[34] in the paper)

[2] Y. Gal, Z. Ghahramani, Bayesian Convolutional Neural Networks with Bernoulli Approximate Variational Inference, ICLR 2016, arXiv:1506.02158

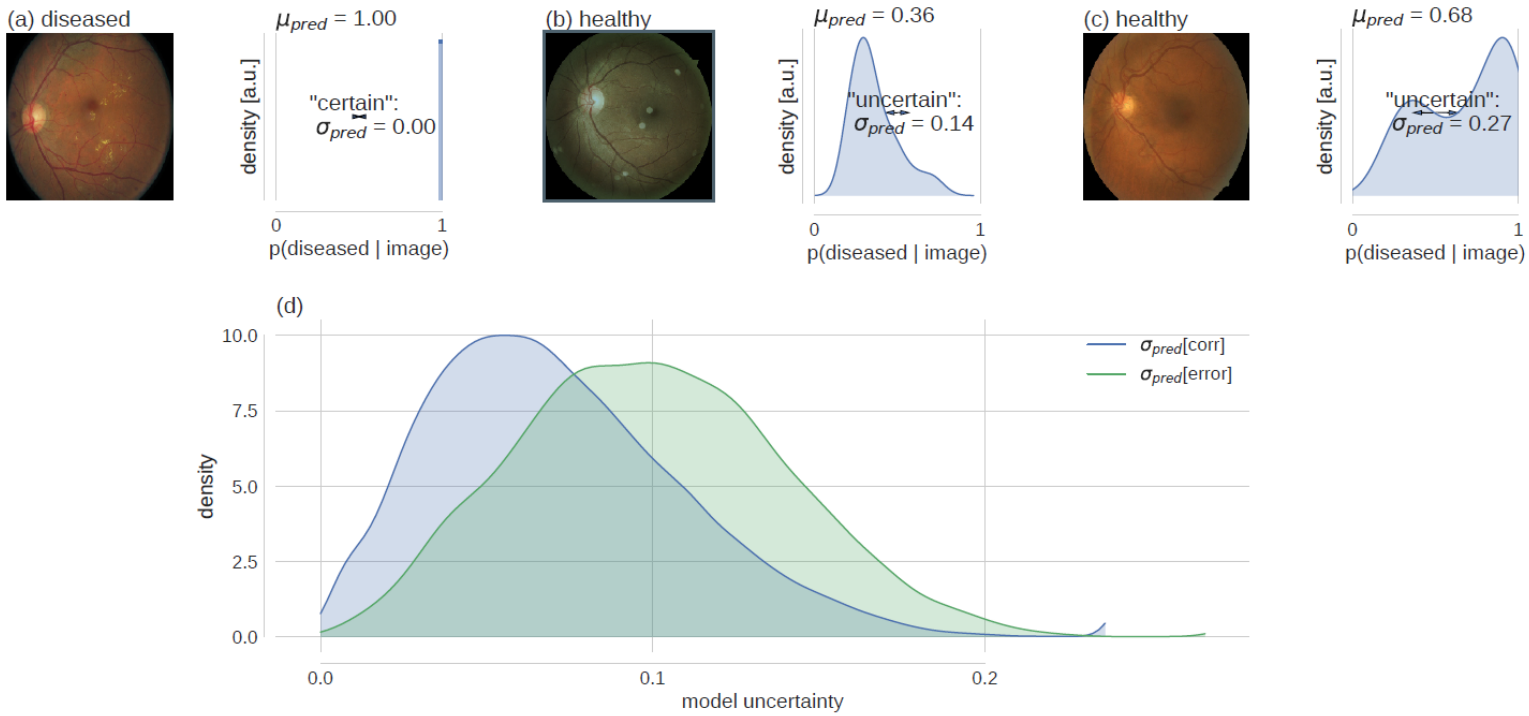
How Can Uncertainty Be Measured?

- Bayesian perspective could be useful
- Combinations of deep NNs and Bayesian ideas are an active topic of research ([23]-[34] in the paper)
- Link between dropout and approximate Bayesian inference [2]
  - Draw multiple predictions with dropout left on at test time
  - Interpretable as approximate Bayesian inference
- Advantages:
  - Efficient and easy to do with existing software implementations
  - Can be applied to already trained networks (with dropout)

[2] Y. Gal, Z. Ghahramani, Bayesian Convolutional Neural Networks with Bernoulli Approximate Variational Inference, ICLR 2016, arXiv:1506.02158

Idea for Uncertainty Estimation

- Sample from the approximate predictive posterior
  - Mean μ_{pred}: used for prediction
  - Standard deviation σ_{pred}: measure of uncertainty



Theoretical Foundation

- We will follow the paper by Yarin Gal and Zoubin Ghahramani [2]
  - Approach from the Bayesian viewpoint
  - Optimization is equivalent to dropout training
  - Predict using the Bayesian perspective
- Some details that I skip can be found in [3,4]
  - Connection to (deep) Gaussian processes
  - Some approximations and derivations

---

[2] Y. Gal, Z. Ghahramani, Bayesian Convolutional Neural Networks with Bernoulli Approximate Variational Inference, ICLR 2016, arXiv:1506.02158

[3] Y. Gal, Z. Ghahramani, Dropout as a Bayesian approximation: Representing model uncertainty in deep learning, ICML 2016, arXiv:1506.02142

[4] Y. Gal, Z. Ghahramani, Dropout as a Bayesian Approximation: Appendix, ICML 2016, arXiv:1506.02157

Non-Probabilistic vs. Bayesian Neural Net

- In a standard NN for classification, the softmax layer outputs $p(y^* = k \mid x^*, \omega)$ for one set of NN parameters ω
- In a fully Bayesian setting, we are interested in $p(y^* = k \mid x^*, \mathcal{W})$, where $\mathcal{W}$ is a random variable with distribution $p(\omega \mid X, Y)$

  $\Rightarrow$ distribution over class probabilities

Non-Probabilistic vs. Bayesian Neural Net

- In a standard NN for classification, the softmax layer outputs $p(y^* = k \mid x^*, \omega)$ for one set of NN parameters ω
- In a fully Bayesian setting, we are interested in $p(y^* = k \mid x^*, \mathcal{W})$, where $\mathcal{W}$ is a random variable with distribution $p(\omega \mid X, Y)$

  $\Rightarrow$ distribution over class probabilities
- For predicting in the Bayesian setting, we can integrate over ω:

$$p(y^* = k \mid x^*, X, Y) = \int p(y^* = k \mid x^*, \omega)\, p(\omega \mid X, Y)\, \mathrm{d}\omega$$

- The variance in the distribution of class probabilities carries (un)certainty information

Variational Inference

- Usually, $p(\omega \mid X, Y)$ is hard to compute
- Idea: Define an easy to evaluate *variational* distribution $q(\omega)$ to approximate $p(\omega \mid X, Y)$ and predict with

$$p(y^* \mid X, Y, x^*) = \int p(y^* \mid x^*, \omega)\, q(\omega)\, \mathrm{d}\omega$$

- Minimize $\mathrm{KL}\left(q(\omega) \,||\, p(\omega \mid X, Y)\right)$
- Equivalent to maximizing

$$\mathcal{L}_{VI} = \int q(\omega) \log p(Y \mid X, \omega)\, \mathrm{d}\omega - \mathrm{KL}(q(\omega) \,||\, p(\omega))$$

(log evidence lower bound)

Link Between Dropout Networks and Approximate Bayesian Inference (1)

- In a Bayesian neural net $\omega = (W_i)_{i=1}^{L}$ is all weight matrices
- Gaussian prior: $\omega \sim \mathrm{N}(0, I)$
- We approximate $p(\omega \mid X, Y)$ with a variational distribution
- For every layer i we define $q(W_i)$ as

$$W_i = M_i \cdot \mathrm{diag}\left((z_{i,j})_{j=1}^{K_i}\right)$$
$$z_{i,j} \sim \mathrm{Bernoulli}(p_i)$$

- Here p_i are predefined and M_i are variational parameters

Link Between Dropout Networks and Approximate Bayesian Inference (2)

- For this variational distribution $\mathcal{L}_{VI}$ is not tractable
- However, we can approximate it using Monte Carlo integration:

$$\hat{\mathcal{L}}_{VI} = \sum_{i=1}^{N} E(y_i, \hat{f}(x_i, \hat{\omega}_i)) - KL(q(\omega) \,||\, p(\omega)) \quad \text{with} \quad \hat{\omega}_i \sim q(\omega),$$

  where $E(.,.)$ is the loss for softmax

- Sampling from $q(W_i)$ is equivalent to performing dropout on layer i of the network with weights $(M_i)_{i=1}^{L}$
- Regularization with weight decay approximates $KL(q(\omega)||p(\omega))$

Predicting in the Bayesian Framework

- Training with dropout and maximimzing $\hat{\mathcal{L}}_{VI}$ results in the same optimal parameters (M_i)
- We can approximate the Bayesian predictions using $q(\omega)$ and Monte Carlo integration:

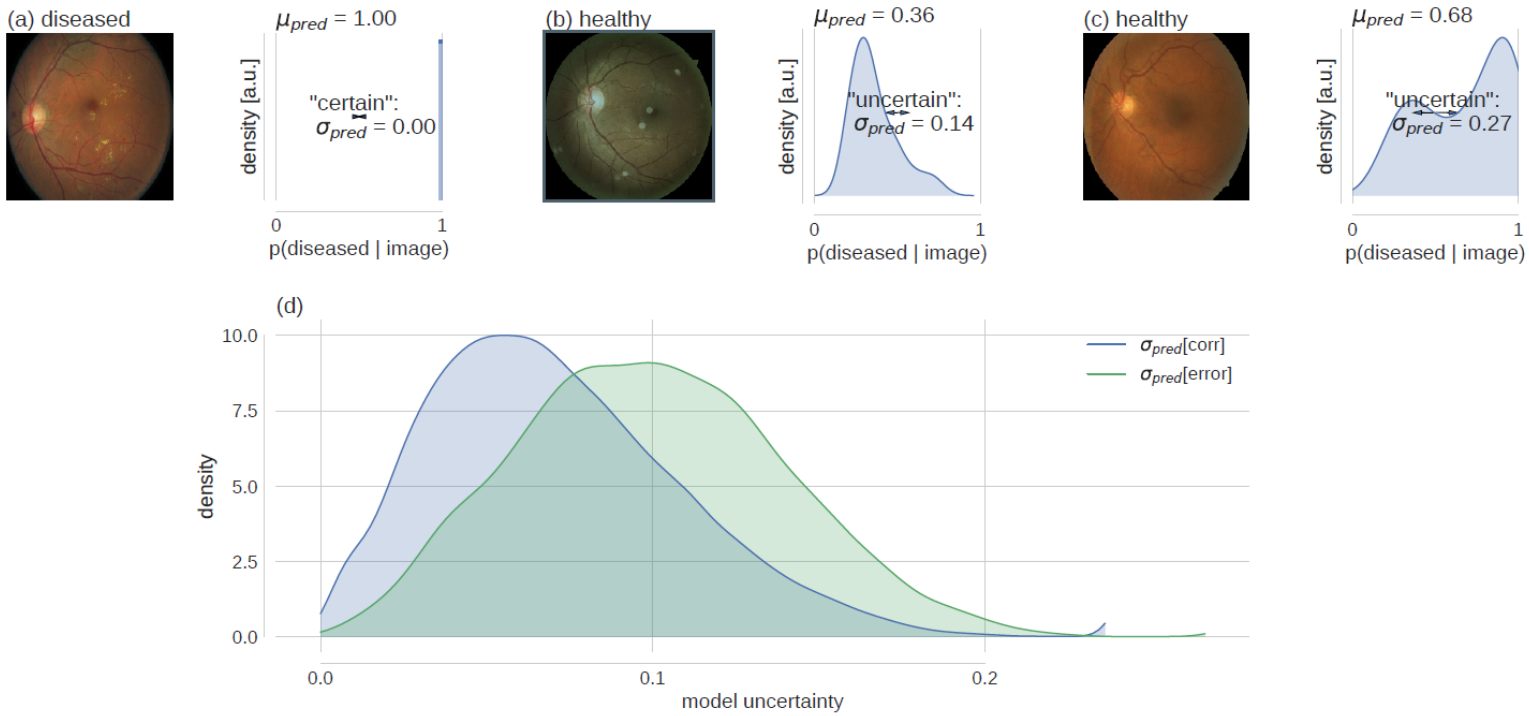
$$p(y^* \mid x^*, X, Y) \approx \int p(y^* \mid x^*, \omega) q(\omega) \, \mathrm{d}\omega \approx \frac{1}{T} \sum_{t=1}^{T} p(y^* \mid x^*, \hat{\omega}_t)$$

with $\hat{\omega}_t \sim q(\omega)$ (MC dropout)

- Alternatively, $p(y^* \mid x^*, \hat{\omega}_1), \ldots, p(y^* \mid x^*, \hat{\omega}_T)$ is a sample from the predictive posterior

Idea for Uncertainty Estimation

- Sample from the approximate predictive posterior
  - Mean μ_{pred}: used for prediction
  - Standard deviation σ_{pred}: measure of uncertainty



Bayesian Convolutional Neural Networks

- In this section we have only considered fully connected layers
  $\Rightarrow$ Does it extend to convolutional / pooling layers?

Bayesian Convolutional Neural Networks

- In this section we have only considered fully connected layers
  $\Rightarrow$ Does it extend to convolutional / pooling layers?
- By rearranging the data (extracting patches, vectorize), the convolution can be written as a dot product
- Pooling is a nonlinear operation that is performed afterwards

$\Rightarrow$ Everything extends if dropout is performed after the convolutional layer and before pooling

Bayesian Convolutional Neural Networks

- In this section we have only considered fully connected layers
  $\Rightarrow$ Does it extend to convolutional / pooling layers?
- By rearranging the data (extracting patches, vectorize), the convolution can be written as a dot product
- Pooling is a nonlinear operation that is performed afterwards

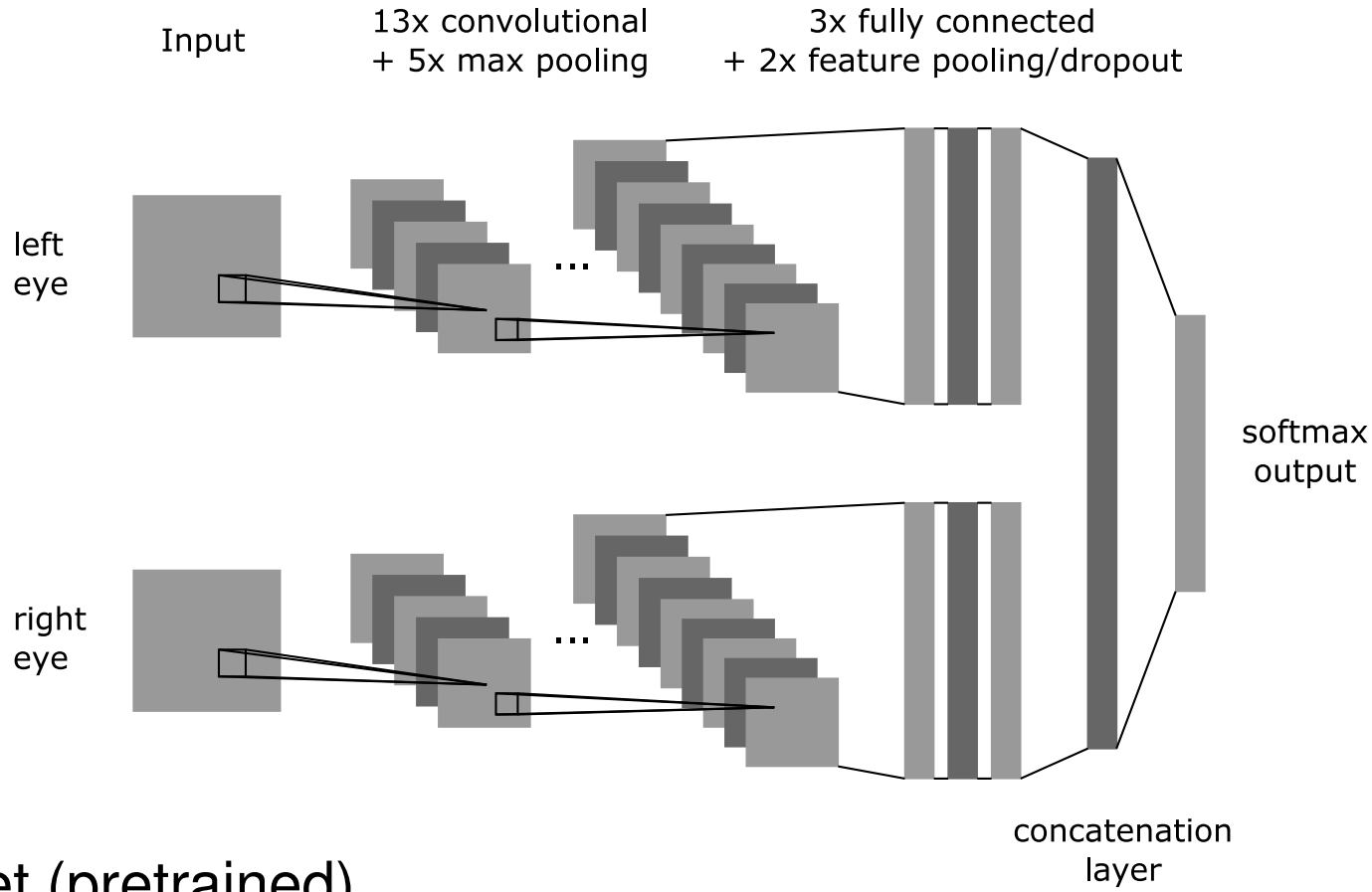
$\Rightarrow$ Everything extends if dropout is performed after the convolutional layer and before pooling

- Standard testing (rescaling M_i) does not perform well
  $\Rightarrow$ usually no dropout in conv. layers in practice

Experimental Setup

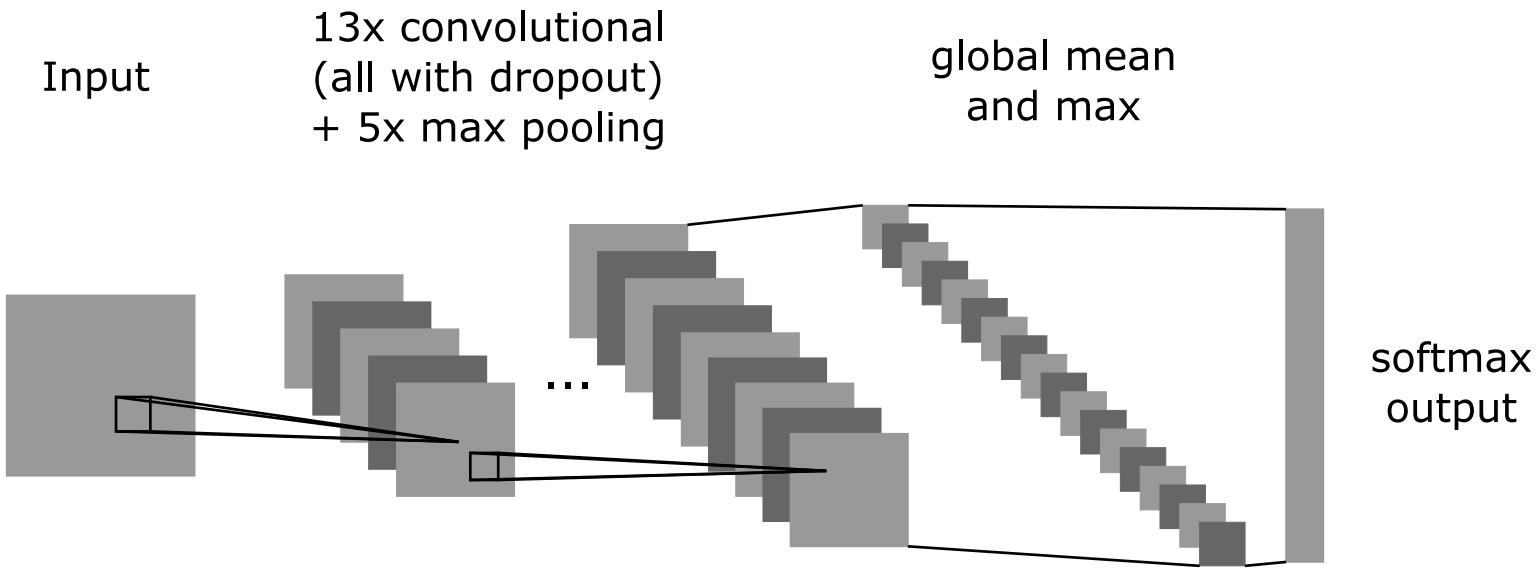
- Main dataset: Kaggle
  - 35,126 training images and 53,576 test images
  - Graded into 5 stages:
    0. No DR
    1. Mild DR
    2. Moderate DR
    3. Severe DR
    4. Proliferative DR
- Second dataset: Messidor
  - 1,200 images
  - Graded into 4 stages:
    0. No DR
    1. Mild non-proliferative DR
    2. Severe non-proliferative
    3. Most serious DR
- Classification task: Disease detection
  - Distinguish “healthy” and “diseased”
  - Disease onset varied between 1 and 2

Network Architectures – JFnet



- JFnet (pretrained)
  - Ranked 5th out of 661 in Kaggle competition
  - Nonlinearities: ReLUs or Leaky ReLUs

Network Architectures – BCNN



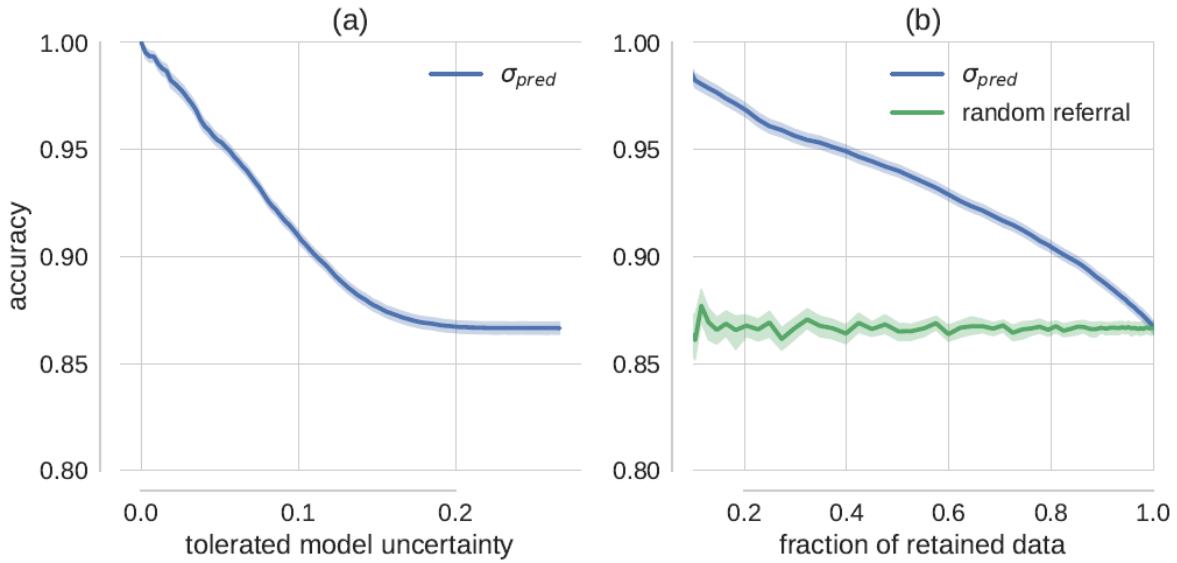
- Own architecture: BCNN
  - Inspired by the monocular part of JFnet
  - Global mean and max instead of fully connected part
  - Dropout after each convolutional layer ($p_{\text{drop}} = 0.2$)

Network Training

- BCNNs were trained on Kaggle training data
  - 80% used for optimization
  - 20% held out as validation set
- Weights from the pretrained JFnet used for initialization
- SGD with Nesterov updates (momentum: 0.9)
- L1-regularization in the last, L2-regularization in all other layers
- Data augmentation for 50% of the data (zooming, translating, rotating, flipping)
- Dropout in convolutional layers $p_{drop} = 0.2$

Uncertainty-Informed Decision Referral

- Does it improve performance to refer uncertain cases?
- Accuracy on remaining data for varied threshold for referral:

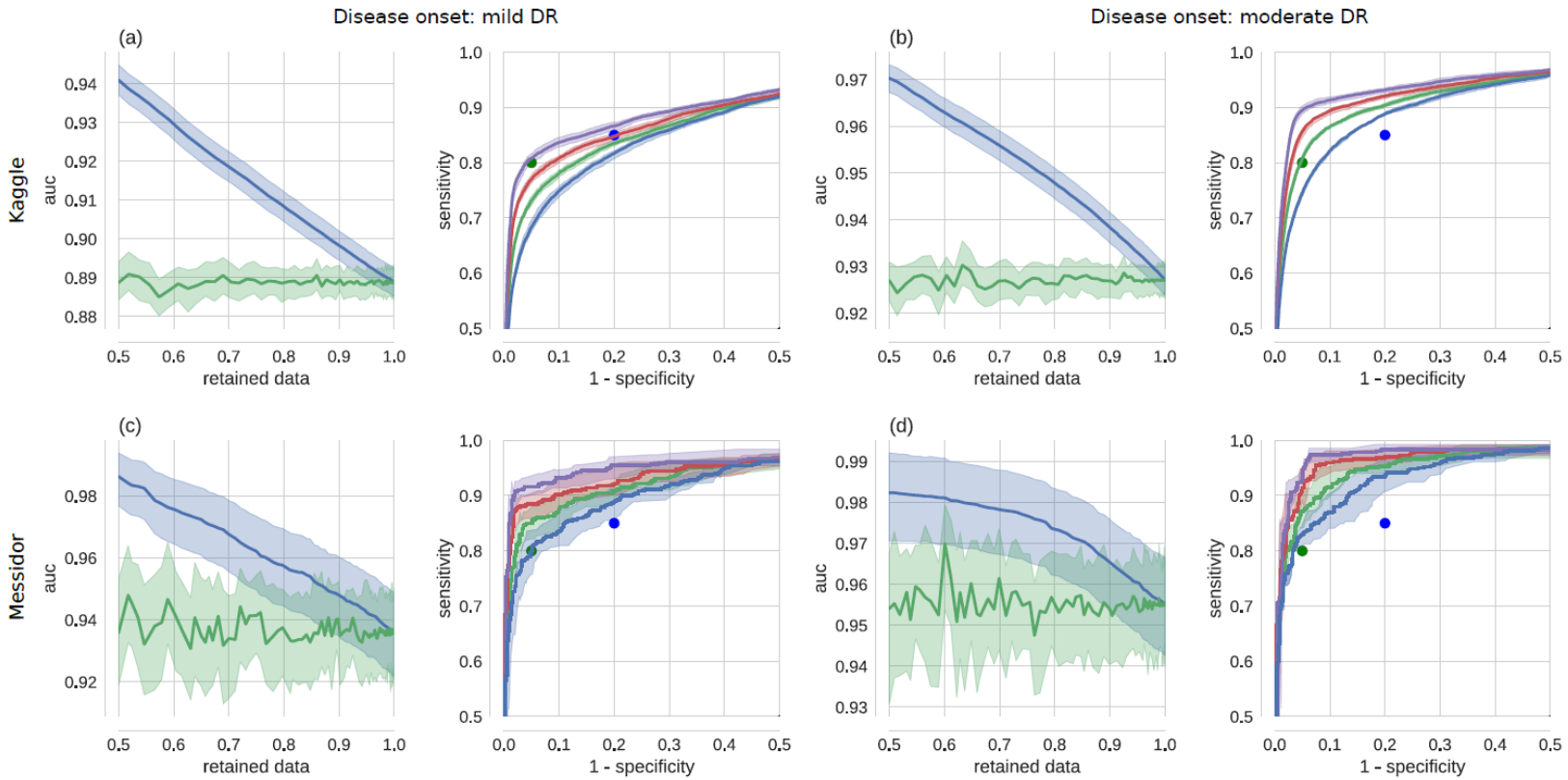


- BCNN
- Disease onset 1
- Kaggle dataset

$\Rightarrow$ Yes, significant improvement when referring 2% or more

Uncertainty-Informed Decision Referral

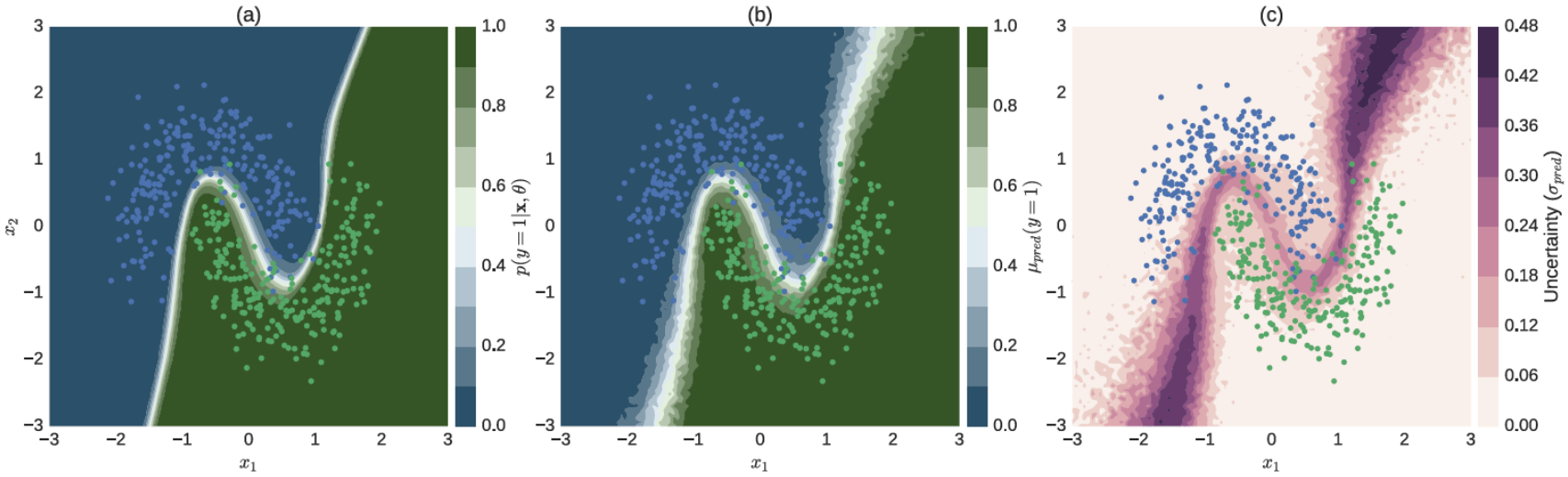
- ROC for multiple tasks and datasets:



- Curves are shown for 0%, 10%, 20% and 30% referred data

What Causes Uncertainty?

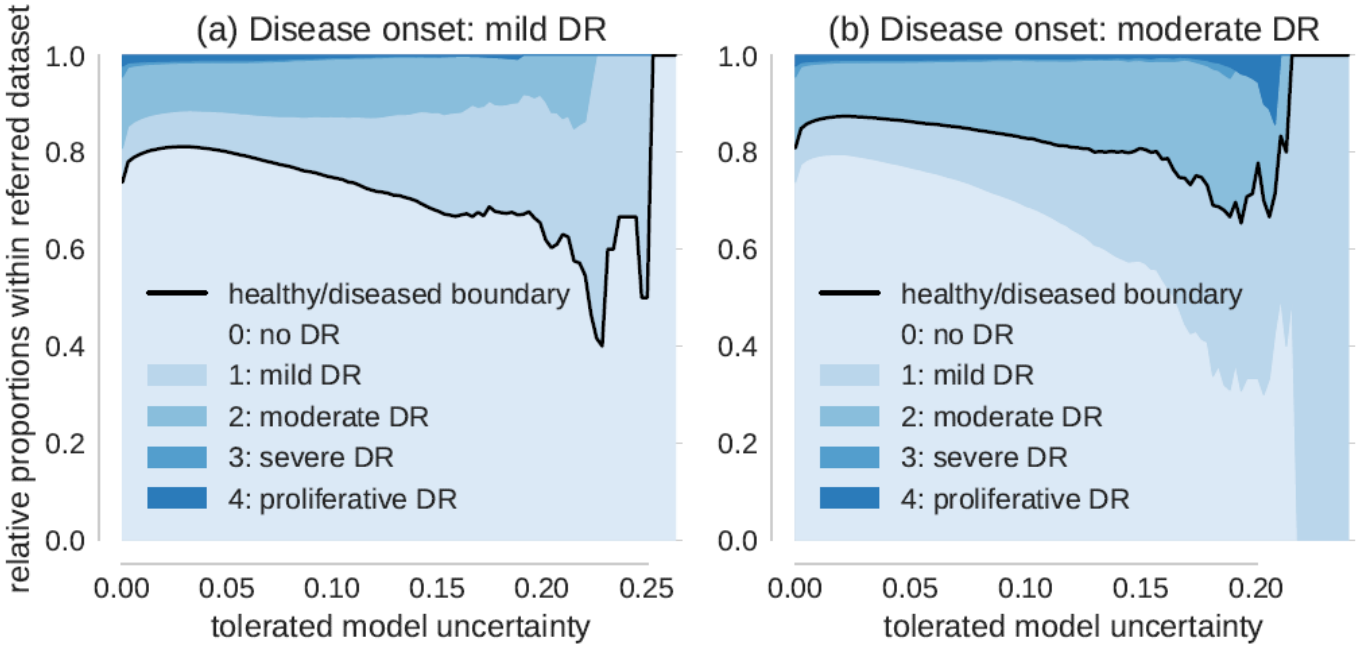
- Simple Bayesian NN trained on 2D toy problem
  - (a) Traditional prediction (dropout off)
  - (b) Predictive mean (BNN)
  - (c) Predictive standard deviation / uncertainty (BNN)



⇒ Bayesian uncertain regions are wider, taking alternative reasonable decision boundaries into account

Uncertainty and Disease Level

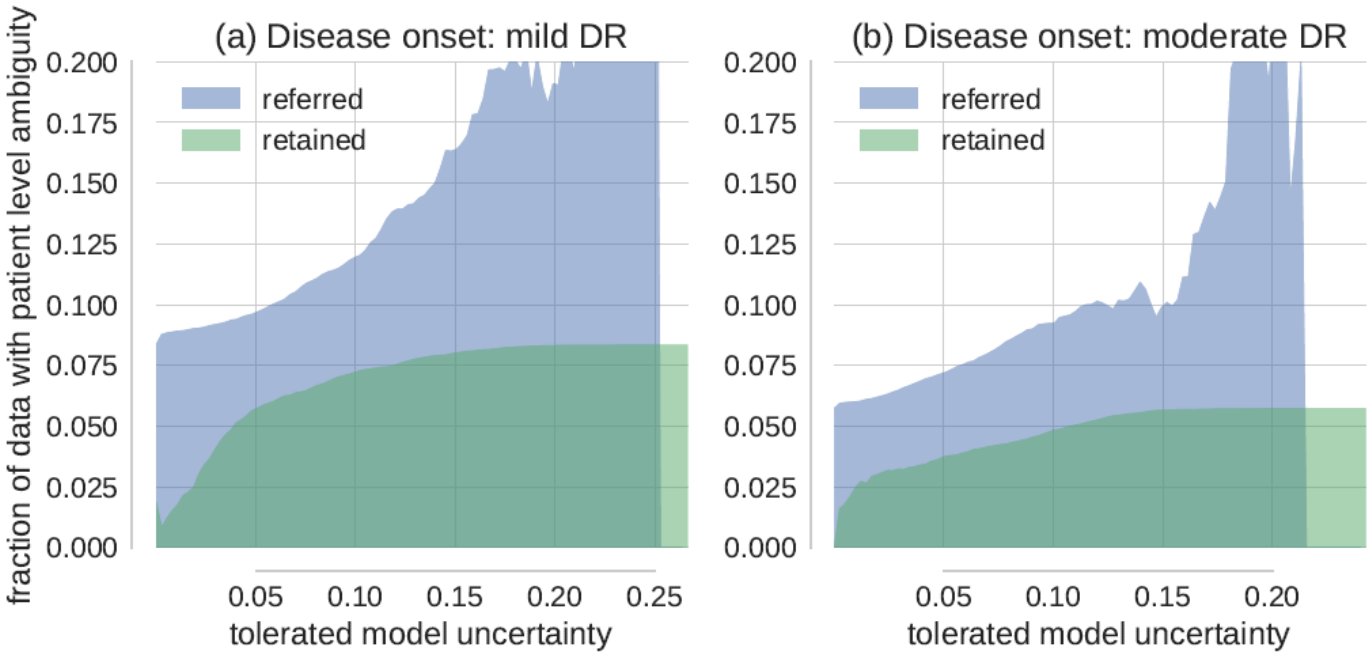
- Are disease levels next to the healthy/diseased boundary more uncertain than others?
- Proportion of disease levels in referred data:



⇒ They dominate for low referral rate, but not in general

Uncertainty of Ambiguous Cases

- Kaggle data contains 5-10% ambiguous cases
  - Patients with two eyes in different healthy/diseased labels
  - Represent cases where physicians were uncertain
- Are these more uncertain than unambiguous cases?

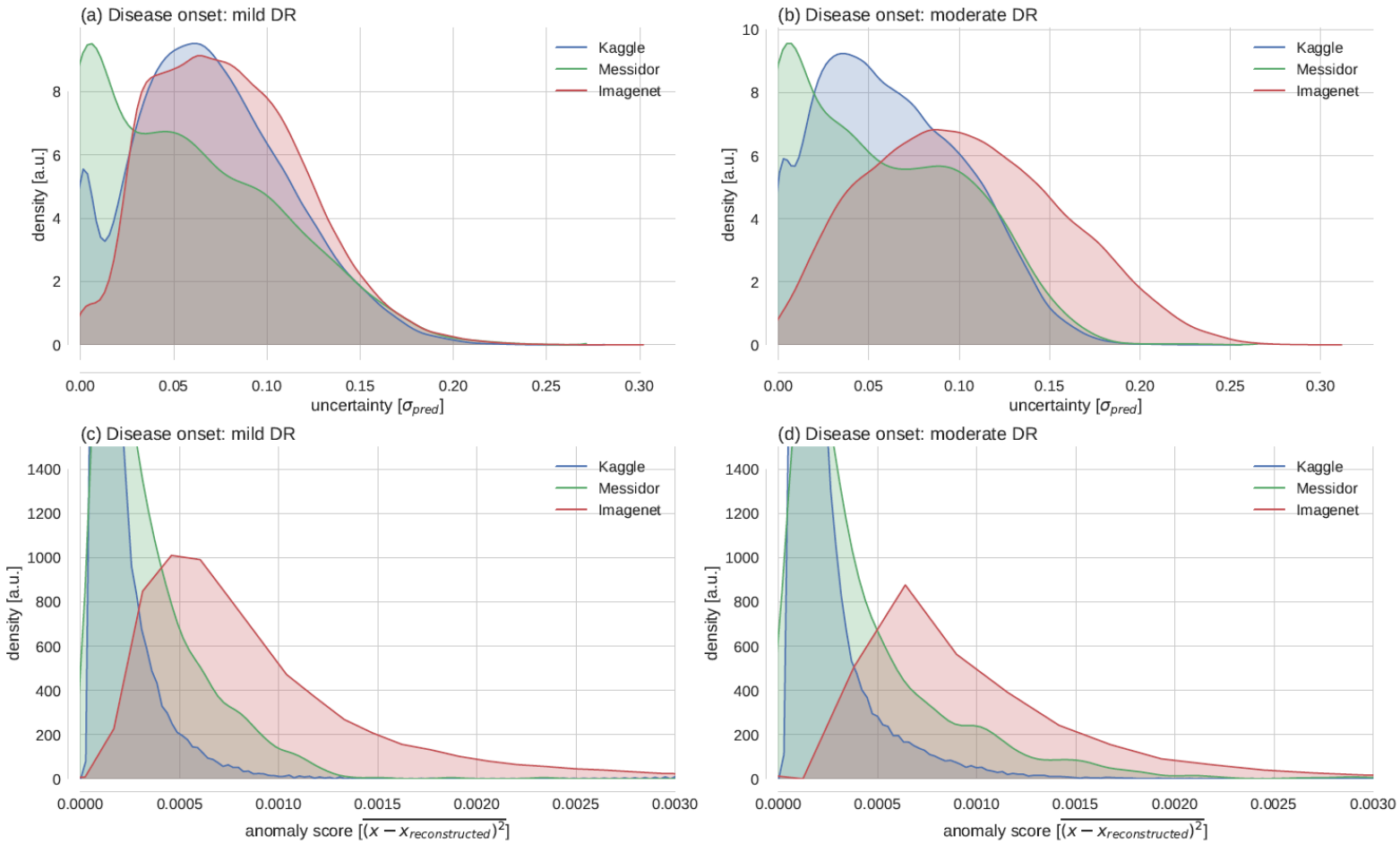


Uncertainty of Unfamiliar Samples

- Can uncertainty be used to identify unusable/unrelated data?
- DR prediction on 2012 Imagenet validation set
  - 49101 coloured, non-fundus images
  - 1000 different categories

  $\Rightarrow$ Higher uncertainty?
- Comparison to autoencoder reconstruction error
  - Trained on penultimate layer of DR detection network
  - Two encoding and two decoding layers (128/32/128/512 units)

Uncertainty of Unfamiliar Samples



⇒ Problem: uncertainty is highly task-dependent

Summary

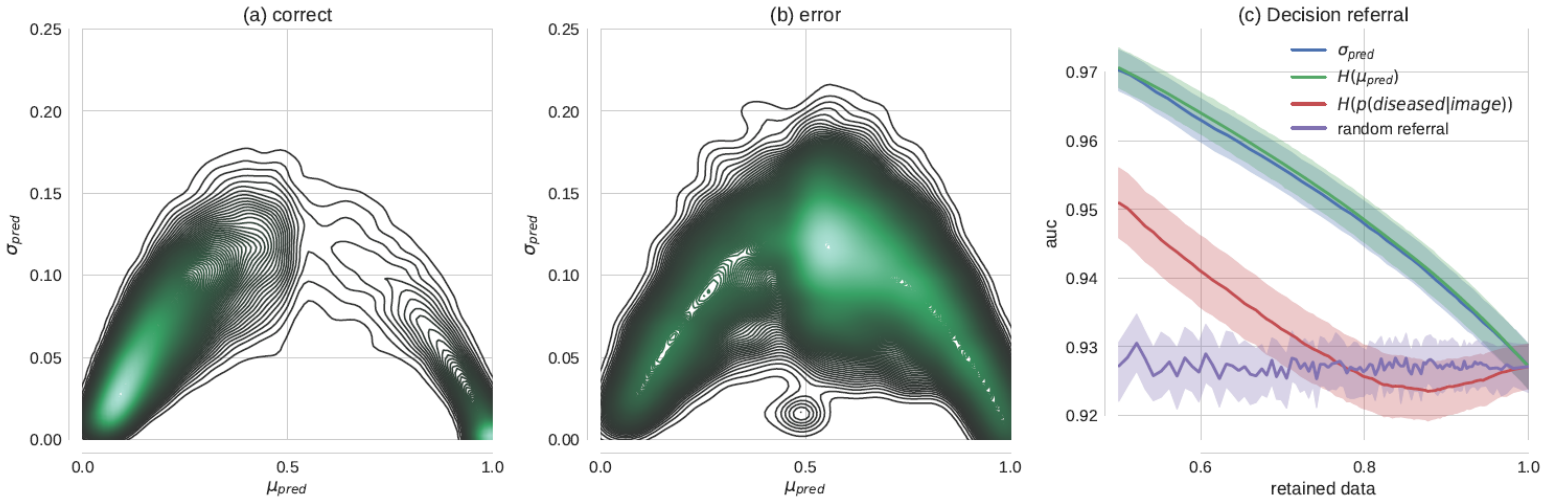
- Sampling multiple predictions with dropout turned on during test time provides useful uncertainty measurements
  - Sensitive to boundary cases
  - Sensitive to clinically relevant cases
- This has a nice interpretation as approximate variational inference in a Bayesian setting
- The method is simple, efficient and doesn't need extra software or labels for an *uncertain* category
- Not directly applicable to detect unknown classes

Open Questions

- How good are the approximations?
  - Variational distribution
  - KL divergence $\Leftrightarrow$ L2-regularization
  - Monte Carlo integration
- How different would the results be with a different variational distribution?
- Can the dropout probabilities be trained as well?

Why not use μ_{pred} instead of σ_{pred}?

- Indeed, μ_{pred} can give similar performance as σ_{pred}
- However, using the predicted probability of a traditional NN performs much worse



- $H(p) = -(p\log(p) + (1-p)\log(1-p))$ is the binary entropy