Calibrated uncertainty:
machine learning methods that
know how well they know

Meelis Kull
Institute of Computer Science
University of Tartu, Estonia
Why do we care about uncertainty? (or confidence?)

- Any machine learning model sometimes makes errors (inevitably)

- Knowing how well it knows allows to reduce the harm of errors:
  - an autonomous car can slow down if its perception system has high uncertainty
  - a doctor can run additional tests if the diagnostic tool is highly uncertain

- Knowing how well it knows = calibrated uncertainty \(\sim\) calibrated confidence

- More examples: energy forecasting; fraud detection; predicting client behaviour
Confidence and uncertainty

- **Confidence:** “I am 70% sure max temperature is above 30°C tomorrow”

- **Uncertainty:** “Tomorrow’s max temperature is:
  - with 70%: above 30°C
  - with 20%: in the range 25°C to 30°C
  - with 10%: below 25°C

- **Confidence:** a probability attached to the predicted event (to a point estimate)
- **Uncertainty:** a probability distribution over possible events
- For the first part of the talk, we will focus on classifier confidence
Making classifiers more trustworthy

A classifier with 60% accuracy on a set of instances (suppose this classifier does not report uncertainty or confidence)
Making classifiers more trustworthy

a classifier with 60% accuracy on a set of instances

(suppose this classifier does not report uncertainty or confidence)
Making classifiers more trustworthy

A classifier with 60% accuracy on a set of instances if the classifier reports class probabilities

\[ \hat{p} = (\hat{p}_1, \hat{p}_2, \ldots, \hat{p}_k) \]
Making classifiers more trustworthy

A classifier with 60% accuracy on a set of instances. If the classifier reports class probabilities

$$\hat{p} = (\hat{p}_1, \hat{p}_2, \ldots, \hat{p}_k)$$

Then we get instance-specific confidence

$$\text{confidence} = \max(\hat{p})$$
Trustworthy if confidence-calibrated
Trustworthy if confidence-calibrated
Trustworthy if confidence-calibrated
Trustworthy if confidence-calibrated
Trustworthy if confidence-calibrated

\[ \max \hat{p}(X) = 0.9 \]
Trustworthy if confidence-calibrated

\[ Y = \arg \max \hat{p}(X) \quad \max \hat{p}(X) = 0.9 \]
Trustworthy if confidence-calibrated

\[ P(\hat{Y} = \arg \max \hat{p}(X) \mid \max \hat{p}(X) = 0.9) = 0.9 \]
Trustworthy if confidence-calibrated

\[ P(Y = \arg \max \hat{p}(X) \mid \max \hat{p}(X) = 0.9) = 0.9 \]

Confidence-calibrated:

\[ P(Y = \arg \max \hat{p}(X) \mid \max \hat{p}(X) = c) = c \]
Deep nets are usually over-confident

Confidence-calibrated:
\[ P(Y = \arg \max \hat{p}(X) \mid \max \hat{p}(X) = c) = c \]

Experimental setup:
CIFAR-10
ResNet Wide 32

Accuracy:
Overall: 94%
At 90% confidence: 58%
Deep nets are usually over-confident

Confidence-calibrated:

\[ P(Y = \arg \max \hat{p}(X) \mid \max \hat{p}(X) = c) = c \]

Experimental setup:
CIFAR-10
ResNet Wide 32

Accuracy:
Overall: 94%
At 90% confidence: 58%
Example: uncalibrated predictions

Confidence-calibrated:

\[ P(Y = \arg \max \hat{p}(X) \mid \max \hat{p}(X) = c) = c \]
Example: uncalibrated predictions

Confidence-calibrated:

\[ P(Y = \arg \max \hat{p}(X) | \max \hat{p}(X) = c) = c \]

Experimental setup:
CIFAR-10
ResNet Wide 32

Accuracy:
Overall: 94%
At 90% confidence: 58%
Example: uncalibrated predictions

Confidence-calibrated:

\[ P(Y = \arg \max \hat{p}(X) \mid \max \hat{p}(X) = c) = c \]

Experimental setup:
CIFAR-10
ResNet Wide 32

Accuracy:
Overall: 94%
At 90% confidence: 58%
Types of over-confidence

● When evaluating on the training set, the models are typically calibrated

● Models are typically over-confident on all other data:
  ○ On a test set from the same distribution as the training set
  ○ On adversarial examples
  ○ On out-of-distribution data

● In the current presentation, we focus on the 1st case, i.e. the goal is to achieve calibrated confidence without any dataset shift
Example: uncalibrated predictions

Confidence-calibrated:

\[ P(Y = \text{arg max} \hat{p}(X) \mid \text{max } \hat{p}(X) = c) = c \]

Experimental setup:
CIFAR-10
ResNet Wide 32

Accuracy:
Overall: 94%
At 90% confidence: 58%
Example: after calibration with temperature scaling

Confidence-calibrated:
\[ P(Y = \arg \max \hat{p}(X) | \max \hat{p}(X) = c) = c \]

Experimental setup:
- CIFAR-10
- ResNet Wide 32

Accuracy:
- Overall: 94%
- At 90% confidence: 58%

Accuracy after Temp.Scal:
- Overall: 94%
- At 90% confidence: 88%
Multi-class classifier without calibration

ANY FEED-FORWARD NETWORK

Input layer

Last hidden layer

logits

\[ z_1 \]

\[ z_2 \]

\[ \ldots \]

\[ z_k \]

Softmax

class probabilities

\[ \hat{p}_1 \]

\[ \hat{p}_2 \]

\[ \ldots \]

\[ \hat{p}_k \]
Multi-class classifier with temperature scaling

Multi-class classifier with temperature scaling

Example: uncalibrated predictions

Confidence-calibrated:
\[ P(Y = \arg \max \hat{p}(X) \mid \max \hat{p}(X) = c) = c \]

Experimental setup:
CIFAR-10
ResNet Wide 32

Accuracy:
Overall: 94%
At 90% confidence: 58%
Example: after calibration with temperature scaling

Confidence-calibrated:

\[ P(Y = \arg \max \hat{p}(X) \mid \max \hat{p}(X) = c) = c \]

Experimental setup:
CIFAR-10
ResNet Wide 32

Accuracy:
Overall: 94%
At 90% confidence: 58%

Accuracy after Temp. Scal:
Overall: 94%
At 90% confidence: 88%
Example: after calibration with temperature scaling

Confidence-calibrated:

$$P(Y = \arg\max \hat{p}(X) \mid \max \hat{p}(X) = c) = c$$

Classwise-calibrated:

$$P(Y = i \mid \hat{p}_i(X) = c) = c$$

Experimental setup:
CIFAR-10
ResNet Wide 32

Accuracy:
Overall: 94%
At 90% confidence: 58%

Accuracy after Temp. Scal:
Overall: 94%
At 90% confidence: 88%
At 90% class 2 prob: 76%
Example: after calibration with Dirichlet calibration

Confidence-calibrated:
\[ P(Y = \arg \max \hat{p}(X) | \max \hat{p}(X) = c) = c \]

Classwise-calibrated:
\[ P(Y = i | \hat{p}_i(X) = c) = c \]

Experimental setup:
CIFAR-10
ResNet Wide 32

Accuracy:
Overall: 94%
At 90% confidence: 58%

Accuracy after Temp.Scal:
Overall: 94%
At 90% confidence: 88%
At 90% class 2 prob: 76%

Accuracy after Dir.Calib:
At 90% class 2 prob: 90%
How to calibrate a multi-class classifier?

ANY FEED-FORWARD NETWORK

Input layer

Last hidden layer

logits

\[ z_1 \]

\[ z_2 \]

\[ \ldots \]

\[ z_k \]

class probabilities

\[ \hat{p}_1 \]

\[ \hat{p}_2 \]

\[ \ldots \]

\[ \hat{p}_k \]
Temperature scaling

Vector scaling

Parameters: \((w, b) \in \mathbb{R}^{k+k}\)

Matrix scaling

Input layer → frozen

ANY
FEED-FORWARD NETWORK

Last hidden layer → logits

Matrix scaling

\( (Wz + b)_1 \)

\( (Wz + b)_2 \)

\( \ldots \)

\( (Wz + b)_k \)

Matrix scaling

\( (Wz + b)_1 \)

\( (Wz + b)_2 \)

\( \ldots \)

\( (Wz + b)_k \)

Softmax

\( \hat{p}_1 \)

\( \hat{p}_2 \)

\( \ldots \)

\( \hat{p}_k \)

Parameters: \((W, b) \in \mathbb{R}^{k \times k+k}\)

Dirichlet calibration can calibrate any classifiers

Beyond temperature scaling: Obtaining well-calibrated multiclass probabilities with Dirichlet calibration. NeurIPS 2019
### Parametric calibration methods

<table>
<thead>
<tr>
<th></th>
<th>Logit space</th>
<th>Class probability space</th>
</tr>
</thead>
<tbody>
<tr>
<td>**Binary</td>
<td>Derived from Gaussian distribution</td>
<td>Derived from Beta distribution</td>
</tr>
<tr>
<td>classification</td>
<td><strong>Platt scaling</strong></td>
<td><strong>Beta calibration</strong>+ constrained variants</td>
</tr>
<tr>
<td>**Multi-class</td>
<td></td>
<td></td>
</tr>
<tr>
<td>classification</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Parametric calibration methods

<table>
<thead>
<tr>
<th></th>
<th>Logit space</th>
<th>Class probability space</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Binary classification</strong></td>
<td>Derived from Gaussian distribution</td>
<td>Derived from Beta distribution</td>
</tr>
<tr>
<td></td>
<td><strong>Platt scaling</strong>[^1]</td>
<td><strong>Beta calibration</strong>[^2]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(+ constrained variants)</td>
</tr>
<tr>
<td><strong>Multi-class classification</strong></td>
<td></td>
<td><strong>Dirichlet calibration</strong>[^3]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(+ constrained variants)</td>
</tr>
</tbody>
</table>

## Parametric calibration methods

<table>
<thead>
<tr>
<th></th>
<th>Logit space</th>
<th>Class probability space</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Binary classification</strong></td>
<td>Platt scaling[1]</td>
<td>Beta calibration[2]</td>
</tr>
<tr>
<td></td>
<td>Derived from Gaussian distribution</td>
<td>Derived from Beta distribution</td>
</tr>
<tr>
<td></td>
<td>( + vector scaling, temperature scaling)</td>
<td>( + constrained variants)</td>
</tr>
<tr>
<td><strong>Multi-class classification</strong></td>
<td>Matrix scaling[4]</td>
<td>Dirichlet calibration[3]</td>
</tr>
<tr>
<td></td>
<td>Derived from Dirichlet distribution</td>
<td></td>
</tr>
</tbody>
</table>

\[2\] M. Kull, T. Silva Filho, P. Flach. Beta calibration: a well-founded and easily implemented improvement on logistic calibration for binary classifiers. AISTATS 2017
\[4\] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On Calibration of Modern Neural Networks. ICML 2017
Dirichlet calibration

ANY PROBABILISTIC MULTI-CLASS CLASSIFIER

\[ \hat{p}_1 \rightarrow \ln \hat{p}_1 \rightarrow \left( W \ln \hat{p} + b \right)_1 \rightarrow \hat{p}'_1 \]
\[ \hat{p}_2 \rightarrow \ln \hat{p}_2 \rightarrow \left( W \ln \hat{p} + b \right)_2 \rightarrow \hat{p}'_2 \]
\[ \vdots \qquad \vdots \qquad \vdots \qquad \vdots \]
\[ \hat{p}_k \rightarrow \ln \hat{p}_k \rightarrow \left( W \ln \hat{p} + b \right)_k \rightarrow \hat{p}'_k \]

Parameters: \( (W, b) \in \mathbb{R}^{k \times k + k} \)

Regularisation:
- L2
- ODIR (Off-Diagonal and Intercept Regularisation)
Non-neural experiments

- 21 datasets x 11 classifiers = 231 settings

Classifiers:

- Logistic Regression
- Naïve Bayes
- Decision tree
- Random forest
- Adaboost
- SVC-linear
- SVC-RBF
- LDA
- QDA
- KNN
- MLP

<table>
<thead>
<tr>
<th>dataset</th>
<th>n_samples</th>
<th>n_features</th>
<th>n_classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>abalone</td>
<td>4177</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>balance-scale</td>
<td>625</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>car</td>
<td>1728</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>cleveland</td>
<td>297</td>
<td>13</td>
<td>5</td>
</tr>
<tr>
<td>dermatology</td>
<td>358</td>
<td>34</td>
<td>6</td>
</tr>
<tr>
<td>glass</td>
<td>214</td>
<td>9</td>
<td>6</td>
</tr>
<tr>
<td>iris</td>
<td>150</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>landsat-satellite</td>
<td>6435</td>
<td>36</td>
<td>6</td>
</tr>
<tr>
<td>libras-movement</td>
<td>360</td>
<td>90</td>
<td>15</td>
</tr>
<tr>
<td>mfeat-karhunen</td>
<td>2000</td>
<td>64</td>
<td>10</td>
</tr>
<tr>
<td>mfeat-morphological</td>
<td>2000</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>mfeat-zernike</td>
<td>2000</td>
<td>47</td>
<td>10</td>
</tr>
<tr>
<td>optdigits</td>
<td>5620</td>
<td>64</td>
<td>10</td>
</tr>
<tr>
<td>page-blocks</td>
<td>5473</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>pendigits</td>
<td>10992</td>
<td>16</td>
<td>10</td>
</tr>
<tr>
<td>segment</td>
<td>2310</td>
<td>19</td>
<td>7</td>
</tr>
<tr>
<td>shuttle</td>
<td>101500</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>vehicle</td>
<td>846</td>
<td>18</td>
<td>4</td>
</tr>
<tr>
<td>vowel</td>
<td>990</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>waveform-5000</td>
<td>5000</td>
<td>40</td>
<td>3</td>
</tr>
<tr>
<td>yeast</td>
<td>1484</td>
<td>8</td>
<td>10</td>
</tr>
</tbody>
</table>

5-fold cross-validation:

- 1/5 test
- 4/5 train

3-fold cross-validation:

- 1/3 train calibrator
- 2/3 train classifier and validate calibrator
Non-neural experiments

- 21 datasets x 11 classifiers = 231 settings
- Average rank
  - Classwise-ECE
  - Log-loss
  - Error rate

Classifiers:
- Logistic Regression
- Naïve Bayes
- Decision tree
- Random forest
- Adaboost
- SVC-linear
- SVC-RBF
- LDA
- QDA
- KNN
- MLP

<table>
<thead>
<tr>
<th>dataset</th>
<th>n_samples</th>
<th>n_features</th>
<th>n_classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>abalone</td>
<td>4177</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>balance-scale</td>
<td>625</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>car</td>
<td>1728</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>cleveland</td>
<td>297</td>
<td>13</td>
<td>5</td>
</tr>
<tr>
<td>dermatology</td>
<td>358</td>
<td>34</td>
<td>6</td>
</tr>
<tr>
<td>glass</td>
<td>214</td>
<td>9</td>
<td>6</td>
</tr>
<tr>
<td>iris</td>
<td>150</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>landsat-satellite</td>
<td>6435</td>
<td>36</td>
<td>6</td>
</tr>
<tr>
<td>libras-movement</td>
<td>360</td>
<td>90</td>
<td>15</td>
</tr>
<tr>
<td>mfeat-kar handwritten</td>
<td>2000</td>
<td>64</td>
<td>10</td>
</tr>
<tr>
<td>mfeat-morphological</td>
<td>2000</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>mfeat-zerinski</td>
<td>2000</td>
<td>47</td>
<td>10</td>
</tr>
<tr>
<td>optdigits</td>
<td>5620</td>
<td>64</td>
<td>10</td>
</tr>
<tr>
<td>page-blocks</td>
<td>5473</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>pendigits</td>
<td>10992</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>segment</td>
<td>2310</td>
<td>19</td>
<td>7</td>
</tr>
<tr>
<td>shuttle</td>
<td>101500</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>vehicle</td>
<td>846</td>
<td>18</td>
<td>4</td>
</tr>
<tr>
<td>vowel</td>
<td>990</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>waveform-5000</td>
<td>5000</td>
<td>40</td>
<td>3</td>
</tr>
<tr>
<td>yeast</td>
<td>1484</td>
<td>8</td>
<td>10</td>
</tr>
</tbody>
</table>

5 times 5-fold cross-validation 3-fold cross-validation
Original Dataset 1/5 test 1/3 train classifier and validate calibrator
4/5 train 1/5 test calibrator

41
Which classifiers are calibrated? (not significantly non-calibrated)
Which classifiers are calibrated? (not significantly non-calibrated)
Neural experiments

- Datasets: CIFAR-10, CIFAR-100, SVHN
- 11 CNNs trained as in Guo et al + 3 pretrained

<table>
<thead>
<tr>
<th>Classwise-ECE</th>
<th>Log-loss</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Uncal</strong></td>
<td><strong>general-purpose calibrators</strong></td>
</tr>
<tr>
<td>c10.convnet</td>
<td>0.1045</td>
</tr>
<tr>
<td>c10.densenet40</td>
<td>0.1146</td>
</tr>
<tr>
<td>c10.lenet5</td>
<td>0.1986</td>
</tr>
<tr>
<td>c10.resnet110</td>
<td>0.0985</td>
</tr>
<tr>
<td>c10.resnet110_SD</td>
<td>0.0866</td>
</tr>
<tr>
<td>c10.resnet_wide32</td>
<td>0.0956</td>
</tr>
<tr>
<td>c100.convnet</td>
<td>0.4246</td>
</tr>
<tr>
<td>c100.densenet40</td>
<td>0.4706</td>
</tr>
<tr>
<td>c100.lenet5</td>
<td>0.4736</td>
</tr>
<tr>
<td>c100.resnet110</td>
<td>0.4164</td>
</tr>
<tr>
<td>c100.resnet110_SD</td>
<td>0.3756</td>
</tr>
<tr>
<td>c100.resnet_wide32</td>
<td>0.4206</td>
</tr>
<tr>
<td>SVHN.convnet</td>
<td>0.1596</td>
</tr>
<tr>
<td>SVHN.resnet152_SD</td>
<td>0.0192</td>
</tr>
<tr>
<td><strong>Average rank</strong></td>
<td>5.71</td>
</tr>
</tbody>
</table>
Learning calibration maps in binary classification
How to obtain a calibration map?

1. Plot predicted scores vs actual labels on a validation dataset
2. Fit a line

Figure: Mari-Liis Allikivi
Isotonic calibration

Finds the monotonic curve that **minimises squared error** (or any other proper loss) on the training data
Isotonic calibration

Finds the monotonic curve that minimises squared error
(or any other proper loss) on the training data

Figure: Mari-Liis Allikivi
Isotonic calibration

Finds the monotonic curve that **minimises squared error** (or any other proper loss) on the training data.

Equivalently: **maximises likelihood** of the training data.

Figure: Mari-Liis Allikivi
Calibration methods

Logistic calibration

Beta calibration

Isotonic calibration

[Platt 1999]

[Kull, Silva Filho, Flach 2017]

[Zadrozny & Elkan 2002]

[Naenia & Cooper 2016]

Figure: Mari-Liis Allikivi
Stability of learning calibration maps (bias and variance)

- Fixed a true calibration function (red line)
- Generated 10 training sets with 1000 instances for calibration training
- 100K for testing

Mari-Liis Allikivi, Meelis Kull. ECML PKDD 2019
Non-parametric Bayesian Isotonic Calibration: Fighting Over-confidence in Binary Classification
Is my multiclass classifier calibrated?

Multiclass classifier: \( \hat{\mathbf{p}}(X) = (\hat{p}_1(X), \ldots, \hat{p}_k(X)) \in \Delta_k \subset [0, 1]^k \)

Actual class: \( Y \in \{1, \ldots, k\} \)

Multiclass-calibrated: \( P(Y = i \mid \hat{\mathbf{p}}(X) = \mathbf{q}) = q_i \) for \( \mathbf{q} \in \Delta_k; \ i = 1, \ldots, k \)

Classwise-calibrated: \( P(Y = i \mid \hat{p}_i(X) = q_i) = q_i \) for \( q_i \in [0, 1]; \ i = 1, \ldots, k \)

Confidence-calibrated: \( P(Y = \text{arg max } \hat{\mathbf{p}}(X) \mid \text{max } \hat{\mathbf{p}}(X) = c) = c \) for \( c \in [0, 1] \)
The probability simplex (for 3 classes)

Figure: Kaspar Valk
The probability simplex (for 3 classes), a synthetic task

Figure: Kaspar Valk
The probability simplex (for 3 classes), a synthetic task

Figure: Kaspar Valk
Reliability diagram for class 0

Figure: Kaspar Valk
The virtues of tilted-top reliability diagrams

Reliability diagrams

Figure: Kaspar Valk
Reliability diagrams

Figure: Kaspar Valk
Reliability diagrams

Figure: Kaspar Valk
Reliability diagrams

Figure: Kaspar Valk
The calibration map learned by temperature scaling

Figure: Kaspar Valk
Reliability diagrams after temperature scaling

Figure: Kaspar Valk
The calibration map learned by Dirichlet calibration

Figure: Kaspar Valk
Reliability diagrams after Dirichlet calibration

Figure: Kaspar Valk
Calibration and decision-making

- all possible loss functions as \( \mathcal{L}_{\text{all}} = \{ \ell : \mathcal{Y} \times \mathcal{A} \to \mathbb{R} \} \)

- Possible action sets:
  \( \mathcal{A} \in \{ [1], [2], \ldots, [K], \ldots, \mathbb{N}, \mathbb{R} \} \)

- define the Bayes decision function as 
  \( \delta_\ell(\hat{p}(x)) = \arg \inf_{a \in \mathcal{A}} E_{\hat{Y} \sim \hat{p}(x)}[\ell(\hat{Y}, a)] \)

- subset \( \mathcal{L} \subset \mathcal{L}_{\text{all}} \) denote the set of all Bayes decision functions as 
  \( \Delta_{\mathcal{L}} := \{ \delta_\ell, \ell \in \mathcal{L} \} \)

**Definition 2 (Decision Calibration).** For any set of loss functions $\mathcal{L} \subseteq \mathcal{L}_{\text{all}}$ and set of decision rules $\Delta \subseteq \Delta_{\text{all}} := \{\Delta^C \to \mathcal{A}\}$, we say that a prediction $\hat{p}$ is $(\mathcal{L}; \Delta)$-decision calibrated (with respect to $p^*$) if $\forall \ell \in \mathcal{L}$ and $\delta \in \Delta$ with the same action space $\mathcal{A}$

$$E_X E_{\hat{Y} \sim \hat{p}(X)}[\ell(\hat{Y}, \delta(\hat{p}(X)))] = E_X E_{Y \sim p^*(X)}[\ell(Y, \delta(\hat{p}(X)))]$$

In particular, we say $\hat{p}$ is $\mathcal{L}$-decision calibrated if it is $(\mathcal{L}; \Delta_{\mathcal{L}})$-decision calibrated.
Decision-calibration generalizes many calibration definitions

<table>
<thead>
<tr>
<th>Existing Calibration Definitions</th>
<th>Associated Loss Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Confidence Calibration (Guo et al., 2017)</td>
<td>$\mathcal{L}_r := \left{ \ell(y, a) = \mathbb{I}(y \neq a \land a \neq \bot) + \beta \cdot \mathbb{I}(a = \bot) : a \in \mathcal{Y} \cup {\bot}, \forall \beta \in [0, 1] \right}$</td>
</tr>
<tr>
<td>$\Pr[Y = \text{arg max } \hat{p}(X) \mid \text{max } \hat{p}(X) = \beta] = \beta, \forall \beta \in [0, 1]$</td>
<td></td>
</tr>
<tr>
<td>Classwise Calibration (Kull et al., 2019)</td>
<td>$\mathcal{L}_{cr} := \left{ \ell_c(y, a) = \mathbb{I}(a = \bot) + \beta_1 \cdot \mathbb{I}(y = c \land a = T) : a \in {T, F, \bot}, \forall \beta_1, \beta_2 \in \mathbb{R}, c \in [C] \right}$</td>
</tr>
<tr>
<td>$\mathbb{E}[Y_c \mid \hat{p}_c(X) = \beta] = \beta, \forall c \in [C], \forall \beta \in [0, 1]$</td>
<td></td>
</tr>
<tr>
<td>Distribution Calibration (Kull &amp; Flach, 2015)</td>
<td>$\mathcal{L}_{all} = {\ell : \mathcal{Y} \times \mathcal{A} \mapsto \mathbb{R}}$</td>
</tr>
<tr>
<td>$\mathbb{E}[Y \mid \hat{p}(X) = q] = q, \forall q \in \Delta^C$</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: A prediction function $\hat{p}$ satisfies the calibration definitions on the left if and only if it satisfies $\mathcal{L}$-decision calibration for the loss function families on the right (Theorem 1).

For finite sets of actions, decision boundaries are linear

**Observation 2.** We observe that the partitions of $\Delta^C$ are defined by linear classification boundaries. Formally, we introduce a new notation for the linear multi-class classification functions

$$B^K = \{b_w \mid \forall w \in \mathbb{R}^{K \times C}\}$$

where $b_w(q, a) = \mathbb{1}(a = \arg\sup_{a' \in [K]} \langle q, w_{a'} \rangle)$

Calibration algorithm with guarantees (finite sets of actions)

**Algorithm 1:** Recalibration algorithm to achieve $L^K$ decision calibration.

1. Input current prediction function $\hat{p}$, tolerance $\epsilon$. Initialize $\hat{p}^{(0)} = \hat{p}$.
2. for $t = 1, 2, \cdots, T$ until output $\hat{p}^{(T)}$ when it satisfies Eq.(8) do
   3. Find $b \in B^K$ that maximizes $\sum_{a=1}^{K} \| \mathbb{E}[(Y - \hat{p}^{(t-1)}(X))b(\hat{p}^{(t-1)}(X), a)] \|$
   4. Compute the adjustments $d_a = \mathbb{E}[(Y - \hat{p}^{(t-1)}(X))b(\hat{p}^{(t-1)}(X), a)]/\mathbb{E}[b(\hat{p}^{(t-1)}(X), a)]$
   5. Set $\hat{p}^{(t)} : x \mapsto \hat{p}^{(t-1)}(x) + \sum_{a=1}^{K} b(\hat{p}^{(t-1)}(x), a) \cdot d_a$ (projecting onto $[0, 1]$ if necessary)
   6. end

The probability simplex (for 3 classes), a synthetic task

Figure: Kaspar Valk
The probability simplex (for 3 classes), a synthetic task

Figure: Kaspar Valk
Decision-calibration (K=2), calibration map after iteration 1

Figure: Kaspar Valk
Decision-calibration (K=2), reliability diagram after iteration 1

Figure: Kaspar Valk
Decision-calibration (K=2), reliability diagram after iteration 1

Figure: Kaspar Valk
Decision-calibration (K=2), calibration map after iteration 2

Figure: Kaspar Valk
Decision-calibration (K=2), reliability diagram after iteration 2

Figure: Kaspar Valk
Decision-calibration (K=2), reliability diagram after iteration 9

Figure: Kaspar Valk
Decision-calibration (K=2), reliability diagram after iteration 9

A sequence of simple models, every next aiming to fix the remaining error. Familiar?

Figure: Kaspar Valk
Decision-calibration (K=2), reliability diagram after iteration 9

A sequence of simple models, every next aiming to fix the remaining error. Familiar?

Boosting!

Figure: Kaspar Valk
Decision-calibration (K=5), reliability diagram after iteration 9

Figure: Kaspar Valk
Decision-calibration (K=5), reliability diagram after iteration 3

Figure: Kaspar Valk
Decision-calibration (K=2), calibration map after iteration 9

Figure: Kaspar Valk
Temp. scaling + decision-calibration (K=2), calibration map

Figure: Kaspar Valk
KNN, calibration map

Kaspar Valk, Meelis Kull (under review): Simple Non-Parametric Calibration of Multi-Class Classifiers
Temp. scaling + KNN, calibration map

Kaspar Valk, Meelis Kull (under review): *Simple Non-Parametric Calibration of Multi-Class Classifiers*
Future work

● How to combine non-parametric and parametric methods in a better way?

● How to achieve calibrated uncertainty out-of-distribution and robustness against adversarial examples?

● How to address other predictive tasks – regression, structured prediction
Summary

- Calibrated uncertainty – to be able to reduce the harm of errors
- Over-confidence – on same distribution, on adversarial examples, out-of-distribution
- Methods: temperature scaling, Dirichlet calibration, decision calibration
- Notions: confidence-calibrated, classwise-calibrated, distribution (or multi-class) calibrated, decision calibrated
- Decision-calibration for finite K has guarantees but can still overfit in practice
- Temperature scaling + KNN
Acknowledgements

- Estonian Research Council grant PRG1604
Summary

- Calibrated uncertainty – to be able to reduce the harm of errors
- Over-confidence – on same distribution, on adversarial examples, out-of-distribution
- Methods: temperature scaling, Dirichlet calibration, decision calibration
- Notions: confidence-calibrated, classwise-calibrated, distribution (or multi-class) calibrated, decision calibrated
- Decision-calibration for finite K has guarantees but can still overfit in practice
- Temperature scaling + KNN