ON CALIBRATION OF MODERN NEURAL NETWORKS

Chuan Guo et al., ICML, 2017

VISION STUDY | 김강열 | 2020/08/18

DAVIAN

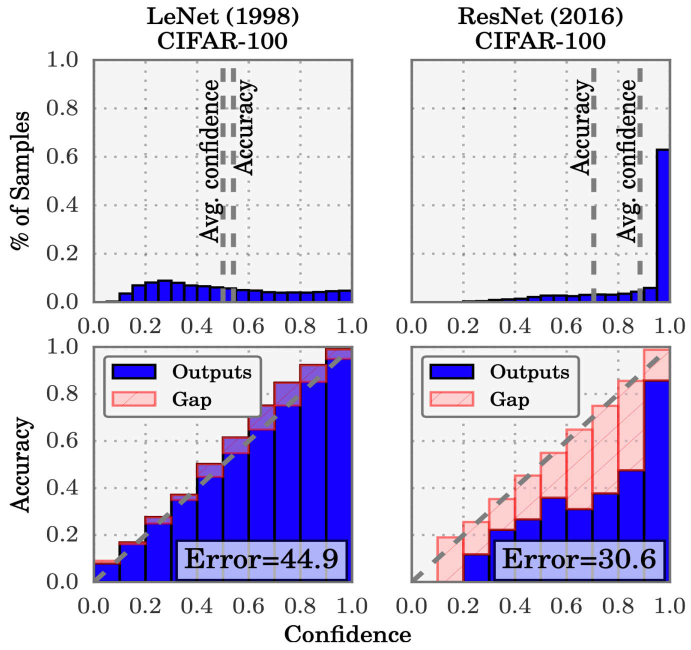
Data and Visual Analytics Lab

Short summary of the paper

- **Why this paper?**
  - Basic paper for uncertainty related research (citation: 768 as of Aug. 2020)
- **Problem statement**
  - **Uncalibrated classification outputs** of modern neural networks
  - Calibration? Probabilistic output of classifier should give probabilistic interpretation
  - e.g., score 0.8 for cat → the classifier made this decision with a confidence of 0.8
- **What makes the neural networks yield uncalibrated outputs?**
  - Depth
  - Batch normalization
  - *Removing* weight decay
- **Empirical experiment on post-hoc calibration methods**
  - Temperature scaling (*jjang-jjang-maen*)
  - Matrix / Vector scaling

Motivation

- **Why does calibration matter?**
  - Good confidence estimates provide a valuable extra bit of information to establish trustworthiness with the user
  - e.g., Medical diagnosis; we want to accept strongly supported prediction results. For this, there are several statistical methods to validate the results such as significance test
- **Does a softmax score tell us about the confidence of model ?**
  - In fact, **no** especially in the modern NN



Measuring calibration of predictions

- **What is well-calibrated predictions?**
- Ideally, perfect calibration can be defined as

$$\mathbb{P}\left(\hat{Y} = Y \mid \hat{P} = p\right) = p, \quad \forall p \in [0, 1]$$

- where Y, p is label and confidence (*i.e.*, softmax output) of the model respectively
- e.g., when there are 100 samples with $\hat{P}$= 0.8, above formula indicates 80 out of 100 samples are predicted correctly ($\hat{Y} = Y$).
- Naturally, p is a **continuous** random variable

Measuring calibration of predictions

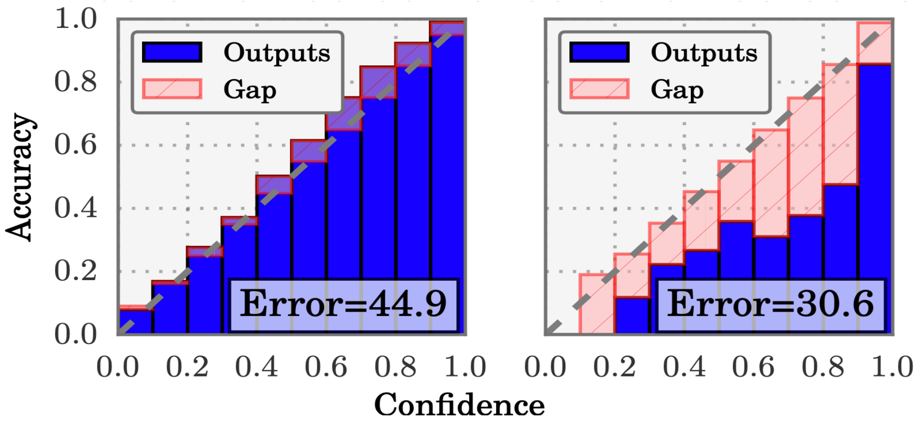
- **Measuring the degree of calibration** $\mathbb{P}\left(\hat{Y}=Y \mid \hat{P}=p\right)=p, \quad \forall p \in[0,1]$
- **Reliability Diagrams**
  - **Goal: want to see the number of $(\widehat{\boldsymbol{Y}}=\boldsymbol{Y})$ under $(\widehat{\boldsymbol{P}}=\boldsymbol{P})$**
  - Let's binning continuous distribution of p with *M* intervals bins
  - **Accuracy**

$$\operatorname{acc}(B_m)=\frac{1}{|B_m|} \sum_{i \in B_m} \mathbf{1}(\hat{y}_i=y_i),$$

  - **Confidence**

$$\operatorname{conf}(B_m)=\frac{1}{|B_m|} \sum_{i \in B_m} \hat{p}_i,$$

  - Where B_m is the number of samples of the interval
  - A perfectly calibrated model will have **acc = conf**



Measuring calibration of predictions

- **Measuring the degree of calibration** $\mathbb{P}\left(\hat{Y} = Y \mid \hat{P} = p\right) = p, \quad \forall p \in [0, 1]$
- **Expected Calibration Error (ECE)**
  - **Goal: want to see the scalar summary statistics of calibration**
  - Let's compute the weighted error of 'Reliability Diagrams'

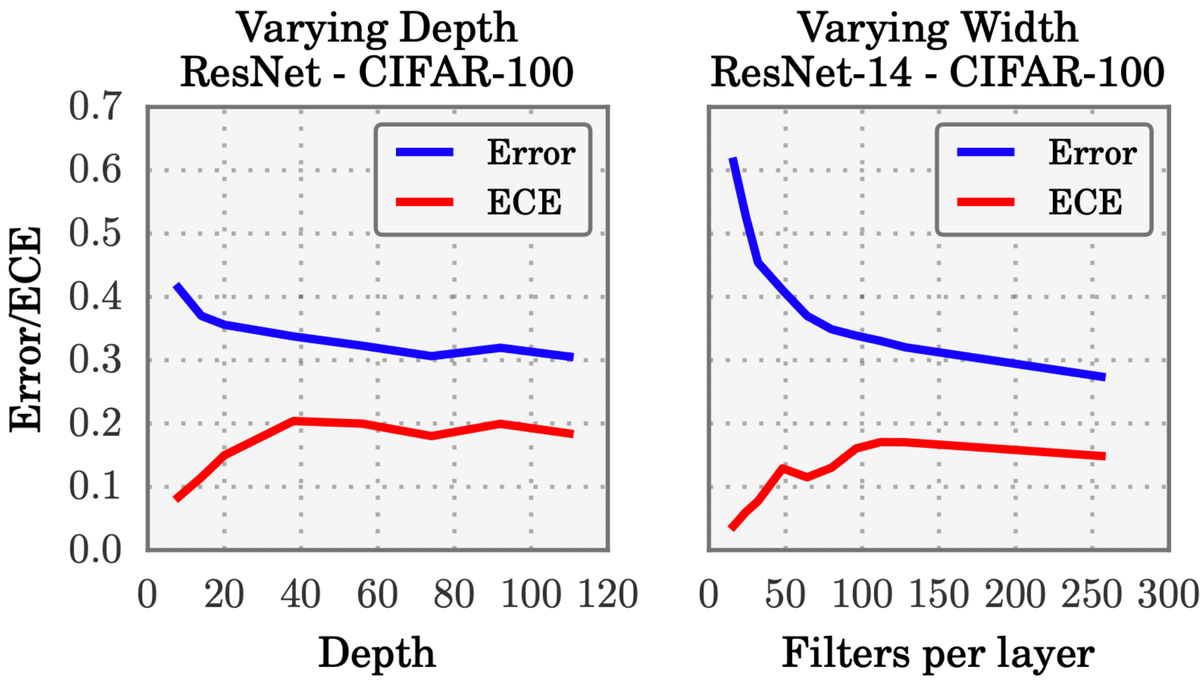
$$\mathrm{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{n} \left| \operatorname{acc}(B_m) - \operatorname{conf}(B_m) \right|,$$

- **Maximum Calibration Error (MCE)**
  - Goal: minimize the worst-case deviation between confidence and accuracy

$$\mathrm{MCE} = \max_{m \in \{1, \ldots, M\}} \left| \operatorname{acc}(B_m) - \operatorname{conf}(B_m) \right|.$$

The effect of modern NN's properties

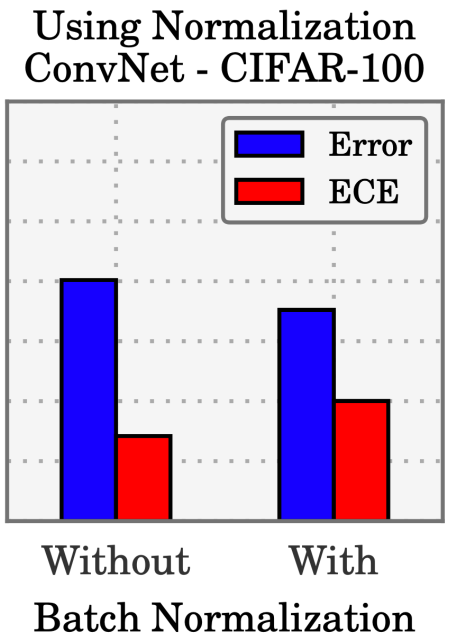
- **What makes the neural networks yield uncalibrated outputs?**
  - **(1) Model capacity**

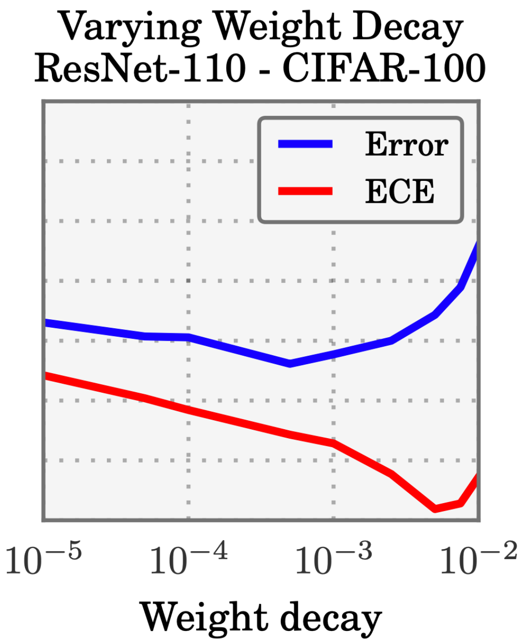


  - Why? (my idea): Excessive parameters amplify the extracted feature…?

The effect of modern NN's properties

- **What makes the neural networks yield uncalibrated outputs?**
  - **(2, 3) Batch normalization / Discarding weight decay**

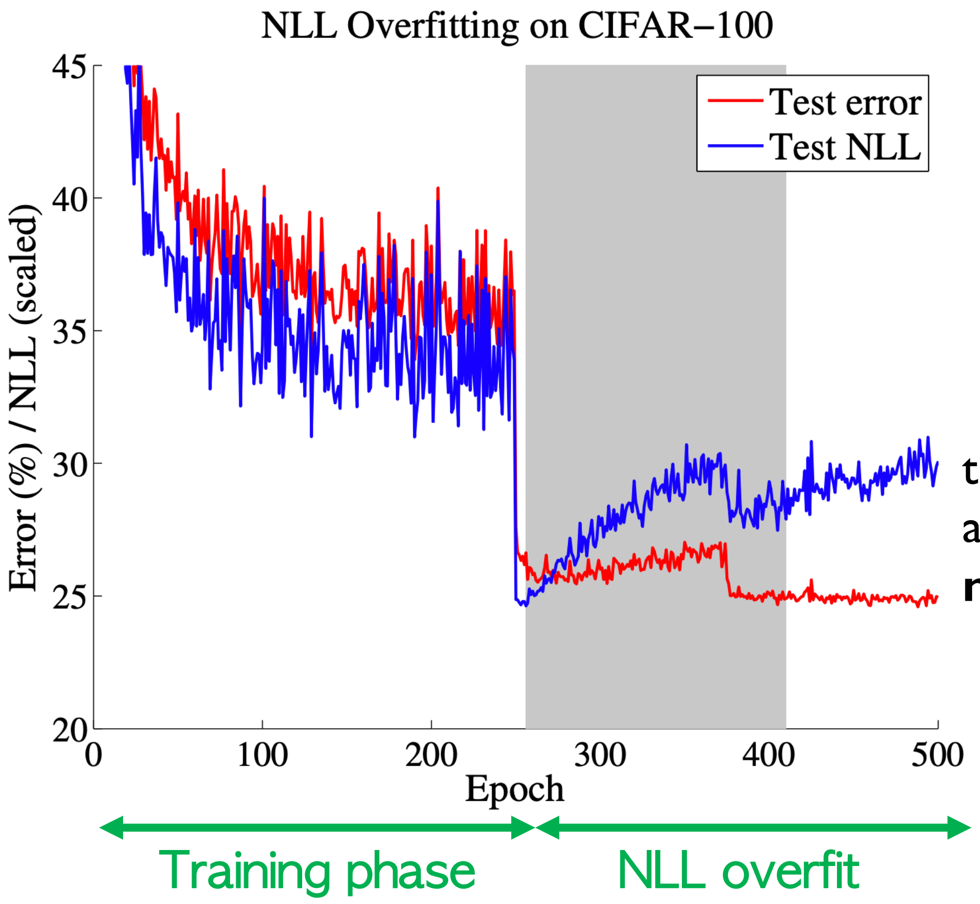




- Why? (my idea):
  - Batch Normalization: ??? (저자도 정확히는 모르겠대용)
  - Weight Decay: removing the trivial filters which cause the output to be close to 1…???

The effect of modern NN's properties

- **What makes the neural networks yield uncalibrated outputs?**
  - **(4) Negative Log Likelihood (NLL) loss = Cross entropy**



the network learns better classification accuracy **at the expense of well-modeled probabilities**

Calibration Method

- **How to calibrate the prediction outputs then?**
  - ~~Histogram binning~~
  - ~~Isotonic regression~~
  - ~~Bayesian Binning into Quantiles (BBQ)~~
  - ~~Platt Scaling~~
  - ~~Matrix and vector scaling~~
  - **Temperature scaling (KING GOD)**

Calibration Method

- **How to calibrate the prediction outputs then?**
  - Goal: Obtaining calibrated probability q from p
  - Importantly, temperature scaling does not affect the model's accuracy

Given the logit vector $\mathbf{z}_i$, the new confidence prediction is

$$\hat{q}_i = \max_k \ \sigma_{\mathrm{SM}}(\mathbf{z}_i/T)^{(k)}. \tag{9}$$

Calibration Method

- **How to train temperature T**

```python
# First: collect all the logits and labels for the validation set
logits_list = []
labels_list = []
with torch.no_grad():
    for input, label in valid_loader:
        input = input.cuda()
        logits = self.model(input)
        logits_list.append(logits)
        labels_list.append(label)
    logits = torch.cat(logits_list).cuda()
    labels = torch.cat(labels_list).cuda()
```

```python
def eval():
    loss = nll_criterion(self.temperature_scale(logits), labels)
    loss.backward()
    return loss
optimizer.step(eval)
```

Experiments

- **ECE on various datasets and models**

Dataset	Model	Uncalibrated	Hist. Binning	Isotonic	BBQ	Temp. Scaling	Vector Scaling	Matrix Scaling
Birds	ResNet 50	9.19%	4.34%	5.22%	4.12%	**1.85%**	3.0%	21.13%
Cars	ResNet 50	4.3%	**1.74%**	4.29%	1.84%	2.35%	2.37%	10.5%
CIFAR-10	ResNet 110	4.6%	0.58%	0.81%	**0.54%**	0.83%	0.88%	1.0%
CIFAR-10	ResNet 110 (SD)	4.12%	0.67%	1.11%	0.9%	**0.6%**	0.64%	0.72%
CIFAR-10	Wide ResNet 32	4.52%	0.72%	1.08%	0.74%	**0.54%**	0.6%	0.72%
CIFAR-10	DenseNet 40	3.28%	0.44%	0.61%	0.81%	**0.33%**	0.41%	0.41%
CIFAR-10	LeNet 5	3.02%	1.56%	1.85%	1.59%	**0.93%**	1.15%	1.16%
CIFAR-100	ResNet 110	16.53%	2.66%	4.99%	5.46%	**1.26%**	1.32%	25.49%
CIFAR-100	ResNet 110 (SD)	12.67%	2.46%	4.16%	3.58%	0.96%	**0.9%**	20.09%
CIFAR-100	Wide ResNet 32	15.0%	3.01%	5.85%	5.77%	**2.32%**	2.57%	24.44%
CIFAR-100	DenseNet 40	10.37%	2.68%	4.51%	3.59%	1.18%	**1.09%**	21.87%
CIFAR-100	LeNet 5	4.85%	6.48%	2.35%	3.77%	**2.02%**	2.09%	13.24%
ImageNet	DenseNet 161	6.28%	4.52%	5.18%	3.51%	**1.99%**	2.24%	-
ImageNet	ResNet 152	5.48%	4.36%	4.77%	3.56%	**1.86%**	2.23%	-
SVHN	ResNet 152 (SD)	0.44%	**0.14%**	0.28%	0.22%	0.17%	0.27%	0.17%
20 News	DAN 3	8.02%	**3.6%**	5.52%	4.98%	4.11%	4.61%	9.1%
Reuters	DAN 3	0.85%	1.75%	1.15%	0.97%	0.91%	**0.66%**	1.58%
SST Binary	TreeLSTM	6.63%	1.93%	**1.65%**	2.27%	1.84%	1.84%	1.84%
SST Fine Grained	TreeLSTM	6.71%	2.09%	**1.65%**	2.61%	2.56%	2.98%	2.39%

Table 1. ECE (%) (with $M = 15$ bins) on standard vision and NLP datasets before calibration and with various calibration methods. The number following a model's name denotes the network depth.

Experiments

- **Reliability diagrams**