Introduction

- **Goal**: having a measure of the network’s prediction reliability
- **Motivation**: essential in many applications (e.g., pedestrian detection in autonomous driving)
- **Challenge**:
  - Even though the standard softmax layer ensures output scores lie in the (0,1) interval, it is not a good estimator of the posterior probability $p(y | x)$
  - In fact, the standard softmax is typically overconfident in its output score

Output Calibration

- We adopt Reliability Diagrams (RD) to evaluate the quality of the calibration
- The RD is a histogram that plots true ratio of correct samples vs. output score
- For example, given 100 samples from a test set with output score 0.6, we'd expect 60 samples to be correctly classified and 40 samples to be classified incorrectly; we'd consider such method calibrated
- If only 30/100 test samples with score 0.6 were correctly classified, the method is over-confident, if 70/100 samples were correctly classified, the method is conservative

- Quantitative evaluation metrics based on the Reliability Diagram
  - Average Calibration Error (ACE)
  - Expected Calibration Error (ECE)
  - Maximal Calibration Error (MCE)
  - see the paper for the exact definition

Relaxed Softmax

- Standard approach: neural network outputs are converted into probabilities using the softmax operator
- Training minimizes the log-likelihood $L_{\alpha_k}$ of the ground truth label $y_i$
  \[
  L_{\alpha_k}(y_i, z_i) = -\sum_{j=1}^{K} \log \sigma_j(y_i, z_i)
  \]
- In theory this is fine, but in practice softmax output is typically over-confident; we think this is because it forces ambiguous (hard) training samples towards 1.0
- Our method: use a temperature scaling model to soften predictions, by predicting an extra scalar value $T_i$ for each sample $i$
  \[
  \sigma_j(z_i, T_i) = \frac{\exp(\frac{z_i}{T_i})}{\sum_{k=1}^{K} \exp(\frac{z_k}{T_i})}
  \]
  - For $T_i > 1$ this spreads the probability mass to other classes as well, in the extreme when $T_i \to \infty$ this puts equal probability behind each class
  - For $T_i = 1$ we obtain the original softmax
- The network can use $T_i$ to directly modulate confidence
- **Relaxed Softmax**: equivalent formulation that avoids instabilities when $T_i \to 0$
  \[
  \hat{\sigma}_j(z, T_i, \alpha_i) = \frac{\exp(\alpha_i z_j)}{\sum_{k=1}^{K} \exp(\alpha_i z_k)}, \quad \alpha_i := \frac{1}{T_i}
  \]
- A plug-in replacement for the standard softmax (just add 1 dimension to the softmax input)
- Trained using vanilla SGD, using the standard cross-entropy loss
- The model learns to predict the $T_i / \alpha_i$ values in a completely unsupervised manner

Experiments on Caltech Pedestrians dataset

- We use the SDS-RCNN\(^1\) pedestrian detector, replace the softmax with Relaxed Softmax, and train the model from scratch on Caltech Pedestrians\(^3\)

- Vanilla softmax is over-confident, even if we try to subsequently calibrate the output scores by linear/temporal scaling on the validation set
- Vanilla Relaxed Softmax is better calibrated, linear scaling on a validation set improves that even further
- The Miss Rate is slightly higher probably because the network assigns lower scores to harder samples, ranking higher a few hard positives

Experiments on NightOwls dataset

- The NightOwls dataset\(^4\) contains 279k right images of pedestrians, data is more ambiguous than Caltech

- Vanilla Relaxed Softmax is again over-confident even with subsequent calibration
- Vanilla Relaxed Softmax is well-calibrated already, subsequent scaling has no significant impact
- We suggest that this is because there are more ambiguous samples in the dataset, so that the model has more data to learn from for the whole (0,1) interval

References