Comparison of Temperature Scaling and Dirichlet Output Layer methods for the Calibration of Deeplab v3+ Neural Network

Yahia Brini
TU Darmstadt
yahia.brini@stud.tu-darmstadt.de

Tomás Pinto
TU Darmstadt
tomas.pinto@stud.tu-darmstadt.de

Moritz Nottebaum
TU Darmstadt
moritz.nottebaum@stud.tu-darmstadt.de

Abstract

Deep neural networks have seen their utilisation explode especially in the recent years. The main focus of researchers was to build models with higher accuracy, thus the architecture of these neural networks has become more complex, for instance Winner of ILSVRC 2015 (Image Classification, Localisation, Detection) contest has 152 layers.

However, in contrast to some old neural networks, the modern variants tend to be poorly calibrated and overconfident. The possible causes to recent miscalibration of neural networks seem to be related not only to the depth and width of neural networks but also to other factors such as batch normalisation and weight decay.

These calibration problems need to be addressed in error sensitive applications, such as autonomous driving and medical diagnoses, where the neural network should also be able to output its uncertainty about a given prediction.

Throughout our experiments, we studied the calibration performance of DeepLab V3+ network, with the MobileNetV2 backbone, using Temperature Scaling and Dirichlet Output Layer methods on the CamVid Semantic Segmentation dataset.

1. Introduction

With the latest advancements in the Deep Learning field, neural networks have become very accurate in complex tasks such as image segmentation and object detection [2], speech recognition [6] and medical diagnosis [1]. In such scenarios, neural networks are expected to provide not only accurate predictions but also to provide reliable information about it's uncertainty regarding the prediction that was made.

But, most of the state-of-the-art deep learning architectures don't have a reliable representation of uncertainty (predictions are agnostic of whether they are reliable or not). Even worse, predictive distributions of common classification models relying on the softmax are not well calibrated and tend to be overconfident [4].

With this work we will compare two different approaches to improve the calibration of a neural network: The first one will be based on [4] by replacing the output layer of Deeplab v3+ model with a probabilistic one. In this case we'll use a Dirichlet Output layer (in which the neural network will output the concentration parameters of a Dirichlet distribution).

In [4], the way in which the calibration of the Dirichlet output layer was compared with softmax output layer was by the assessment of predictive classification distributions, in which, the Dirichlet layer presented a much stronger correlation between its uncertainty and the empirical error than the softmax layer, while still achieving similar or better accuracy.

We'll extend on this work by analysing the Dirichlet output layer for a Semantic Segmentation dataset instead of a Classification dataset and we'll assess and compare the calibration of this Dirichlet output layer with softmax using ECE (3) and MCE (4).

The second one will be based on [5], by using a single-parameter variant of Platt Scaling - Temperature Scaling. This method is referred as simple and effective at calibrating predictions and can be done as a post-processing method, after the training, without affecting the accuracy of the neural network and the temperature parameter can be found by the minimisation of the NLL

loss (5) over the validation set.

We'll extend this work on Temperature Scaling by using, not only the NLL loss but, also, the MCE (4) and ECE (3) loss functions to find the optimal temperature, which directly measure the calibration of a model.

2. Related Work

Firstly, Guo et al. [5], conducted an in-depth analysis of several state-of-the-art deep neural networks investigating the reasons behind these calibration problems. The research highlighted several potential causes, including the depth of the network, batch normalization, layers, weight decay and filters per layer as can be seen in Figure 1.

From their findings, it seems that the presence of batch normalization and higher network width and depth or low weight decay tends to "miscalibrate" neural networks.

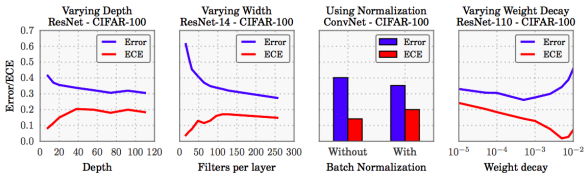


Figure 1: ECE (3) performance under different network depth, filters per layer, batch normalization and weight decay [5].

Another miscalibration reason might be the disconnection that tend to appear between NLL loss and accuracy, at some point in training, in miscalibrated neural networks. This occurs because neural networks can overfit the NLL loss without overfitting the 0/1 loss (Figure 2).

From this, it was concluded that, while overfitting, the neural network appears to improve the accuracy at the expense of well-modelled probabilities so it is expected that overfitting will manifest in a form of probabilistic error (miscalibration) and not in classification error.

This last reason seems to be the most plausible cause for the miscalibration to be happening since overfitting is more prone to happen on neural networks with higher width and depth and lower weight decay.

3. Technical Details

3.1. Definitions

Reliability Diagram. In order to measure calibration, we relied on the benchmark presented in [5], that

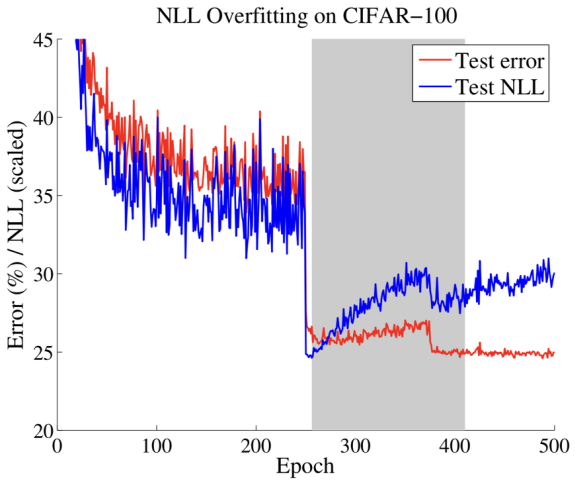


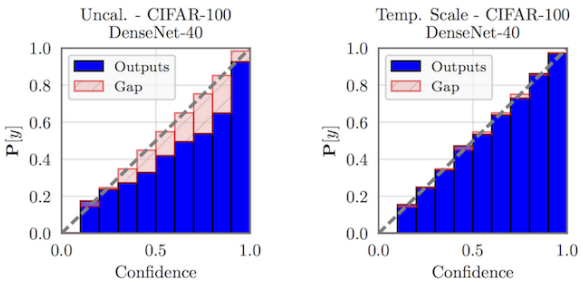
Figure 2: NLL Overfitting on CIFAR-100 when learning rate is decreased [5].

makes a partition of the inference results of a given neural network into bins of a given confidence interval. After this, for each confidence interval we also compute the accuracy of the predictions in that interval:

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

$$acc(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} \mathbf{1}(\hat{y}_i = y_i) \tag{2}$$

With (1) and (2), we can plot the reliability diagram (Figure 5) to visualise how calibrated is a given neural network. In a perfect calibrated model we'd have that the confidence and accuracy are equal for every bin in the reliability diagram.



(a) Uncalibrated (b) Calibrated

Figure 3: Example of Reliability Diagrams for a Uncalibrated (a) and a Calibrated (b) Dense-40 Neural Network on CIFAR-100 [5].

ECE. In order to have also numerical values to measure how calibrated our model is, we'll use the Expected Calibration Error (ECE), that makes a L1-norm of the gap between (1) and (2) over all bins weighted by the fraction of samples that a given bin has in comparison with the total number of samples in the bins:

$$ECE = \sum_{m=1}^{M} \frac{|B_m|}{n} |acc(B_m) - conf(B_m)| \quad (3)$$

MCE. In high-risk applications, where reliable confidence measures are absolutely necessary, we may also use the Maximum Calibration Error (MCE) value which measures the absolute worst-case deviation between (1) and (2) over all bins:

$$MCE = \max_{m \in \{1,...,M\}} |acc(B_m) - conf(B_m)| \quad (4)$$

NLL. Another way to indirectly measure the calibration of a given model is the Negative Log Likelihood (NLL) loss, which is a very common standard measure of a probabilistic model quality. This loss is also referred to as the cross entropy loss in the context of Deep Learning. Given a probabilistic model $\hat{\pi}(Y|X)$ and n samples, NLL is defined as:

$$NLL = -\sum_{i=1}^{n} log(\hat{\pi}(y_i|x_i)) \quad (5)$$

3.2. Dirichlet Output Layer

One of the ideas that is present in the current literature to estimate uncertainty is by combining Bayesian methods with Deep Learning [9]. Bayesian neural networks are gaining momentum because they incorporate directly uncertainty measures into their structure and can output not only a prediction but also the uncertainty of that prediction. Such this uncertainty information is very important particularly in high-risk applications.

In the scope of this work we will extend our baseline neural network with an probabilistic output layer, such as described in [4], where class uncertainties are obtained from the Dirichlet distribution that the neural network outputs.

Dirichlet Distribution. The Dirichlet density function as a function of its concentration parameters is defined as (6):

$$f(x_1, ..., x_k; a_1, ..., a_K) = \frac{1}{B(\alpha)} \prod_{i=1}^{K} x_i^{a_1 - 1} \quad (6)$$

As it can be seen in (6) the Dirichlet distribution is defined with the concentration parameters (α) and the support (x). The concentration parameters is the component that is learned by the neural network whilst the support is supposed to be the ground truth, which needs to have an additional additive smoothing step to prevent numerical problems (due to the fact that the Dirichlet distribution only is defined for continuous support vectors in the unit simplex) and must fulfil the constraint to add up to one.

In Figure 4, it is possible to see different Dirichlet distributions defined by different concentration parameters, thus resulting in different distributions. When training is done with an Dirichlet Output Layer, each pixel does not only have an point estimate prediction, but a Dirichlet probabilistic distribution over all possible classes.

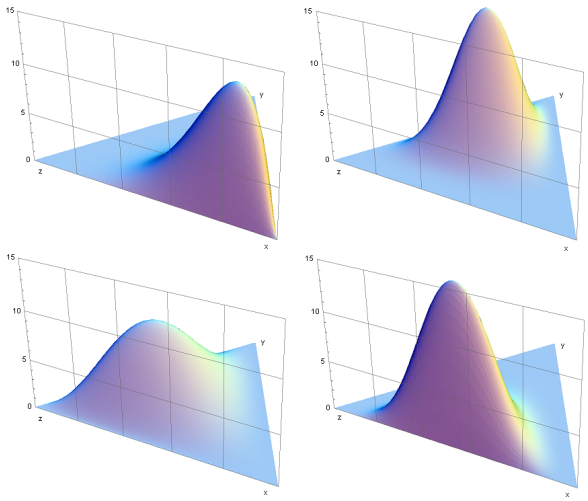


Figure 4: 3-Dimensional Dirichlet Distribution [14].

If we want to extract a point estimate from the Dirichlet distribution, it is possible, for example, to compute the mean of the density function given its concentration parameters (7).

$$E(X_i) = \frac{\alpha_i}{\sum_{i=1}^{K} \alpha_i} \quad (7)$$

It is important to also note that, if the concentration parameters are below one, the Dirichlet distribution can also have multiple density centres. We suppose this actually should be a desirable setup for a multiclass problem like image segmentation.

Implementation details. As a first approach, we included the Dirichlet output layer by simply removing the softmax output layer from the model and using the negative logarithmic probability density of it as loss, but this lead to exploding concentration parameters during training.

Hence, we have chosen to include the softmax again to bound the concentration parameters to be in a range from 0 to 1. In order to not impose directly that every distribution that the neural network outputs should have multiple density centres, we re-scaled the neural network output by adding a infinitesimal constant (since all α values should be greater than zero) and multiplied the result by a factor of 1000 (8), so that the output of the neural network gives concentration parameters greater than zero and lower than 1000.

$$\alpha_i = 1000 \times \sigma(z_i + \epsilon) \tag{8}$$

After, we computed the NLL loss by first computing the Dirichlet distribution using the neural network output with Dirichlet class of Tensorflow module *tf.distributions* and then we computed the loss that used is the negative output from the *log_prob* method of the resulting Tensorflow distribution given the Laplace smoothed ground truth (smooth of ground truth is needed for numerical reasons).

3.3. Temperature Scaling

In [5], different calibration methods were tested, namely Histogram Binning [15], Isotonic Regression [16], Bayesian Binning into quantiles [11], Platt Scaling [12] along with other methods and the conclusion was that, in overall, Temperature Scaling was the method that obtained the best performance.

Temperature Scaling. The proposed procedure is a simpler extension of Platt Scaling. The later used the output of the model as features to a simple linear regression model, which learns the parameter a and b in the output function (9) using NLL loss function.

$$q_i = \sigma(a * z_i + b) \tag{9}$$

The training is performed on the validation set. Along the line the authors proposed an alternative approach, namely a single scalar parameter $T > 0$ for all classes. With the logits vector as input the output is as following (10):

$$q_i = \max_k \sigma_{SM}(z_i/T)^{(k)} \tag{10}$$

T is called temperature, and it softens the softmax function (raising the entropy) if $T > 1$. As T increases and gets larger the probability q_i approaches $1/k$ (the maximum uncertainty). With $T \to 0$ the probability will be reduced to a point mass.

Implementation details. The Optimal Temperature (T) is usually found by minimising the NLL loss function of the neural network output before softmax, divided by the given T, with respect to the one-hot encoded ground truth labels.

Diving deeper in the literature, we found a paper [10] that extends the Temperature Scaling method by proposing an alternative loss function to find the optimal temperature. According to this paper, the need for an alternative to the NLL loss to find the optimal temperature comes from the fact that when the validation set does not contain enough correctly and misclassified samples, Temperature Scaling finds the sub-optimal temperature value.

This case happens frequently when calibrating the highly accurate neural networks where the number of misclassified samples for them is few or when the size of validation set is small.

With these results in mind, we'll evaluate the temperature found by NLL loss function in Temperature Scaling by comparing it with the optimal temperature that minimises MCE and/or ECE.

To find the optimal temperature, we used the L-BFGS-B optimizer and at each step of the optimizer, we computed the whole calibration calculations in (3) and (4) to find the MCE and ECE values, respectively.

4. Results

4.1. Training Setup

Dataset. The CamVid dataset [3] is a road/driving scene understanding database which was originally captured as five video sequences with a 960 $\times$ 720 resolution camera mounted on the dashboard of a car.

Those sequences were sampled (four of them at 1 fps and one at 15 fps) adding up to 701 frames. Those stills were manually annotated with 32 classes.

It is important to remark the partition that we'll use in this work, introduced by Sturgess et al. [13] which divided the dataset into 367/100/233 training, validation, and testing images respectively. That partition makes use of a subset group of the 32 class labels: building, tree, sky, car, sign, road, pedestrian, fence, pole, sidewalk, and bicyclist.

Training. To perform the training on the CamVid dataset, we inspired ourselves from the training setup

of [7] in which the training is divided into 2 parts: The Data Augmentation part for 150 epochs and the Finetuning part for 50 epochs.

In the Data Augmentation part, we used a 12× data augmentation with 240×240 random crops and vertical flips and, as optimizer, we used Adam with an learning rate starting at 0,001 and with an exponential decay of 0,995 per epoch.

In the Finetuning part, we used the full-size 960×720 images and also the Adam optimizer with a constant learning rate of 0,0001.

Neural Networks variants. During the training, we always used the same neural network already mentioned (Deeplab v3+ with the MobileNetV2 backbone) but, for the purpose of referring to the different neural networks configurations we used, we'll define 4 configurations: *Baseline*, *Dirichlet*, *Baseline w/Annealing* and *Dirichlet w/Annealing*.

Baseline refers to the neural network trained with a dropout rate of 0.1 and a weight decay of 0,0001.

Dirichlet refers to the neural network with a Dirichlet output layer but without dropout and weight decay and a constant laplace smooth parameter of 10^{-6}.

Baseline w/Annealing refers to the baseline network, without dropout and weight decay, using an annealing of the laplace smooth parameter during training from 0,3 to 10^{-6}.

Dirichlet w/Annealing refers for the same configuration as the *Baseline w/Annealing* but using an Dirichlet Output Layer.

4.2. Dirichlet Output Layer Results

Here in this section we'll discuss the results that we obtained from the performance and calibration analysis of the *Baseline*, *Dirichlet*, *Baseline w/Annealing* and *Dirichlet w/Annealing* network configurations.

Models	ECE [%]	MCE [%]
Baseline	**3.43**	**12.54**
Dirichlet	5.33	28.47
Baseline w/Annealing	5.6	20.8
Dirichlet w/Annealing	6.9	31.87

Table 1: Comparison of ECE and MCE between the different neural network configurations.

As it can be clearly seen on Figure 5, the network configurations that used a Dirichlet Output Layer are more miscalibrated than the Baseline configuration with and without annealing.

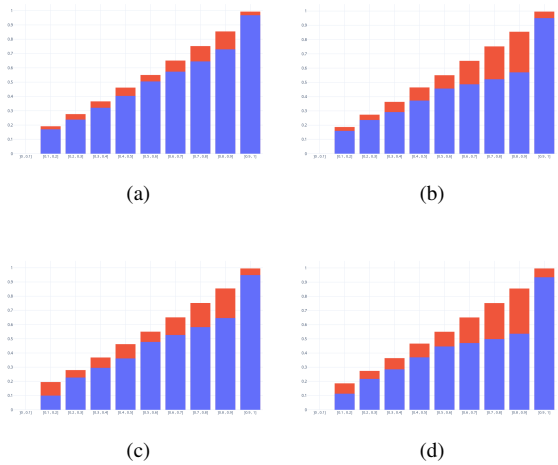


Figure 5: Reliability Diagrams for (a) *Baseline*, (b) *Dirichlet*, (c) *Baseline w/Annealing* and (d) *Dirichlet w/Annealing* networks. (Confidence is in red and Accuracy in blue)

Analysing the Table 1 it is also possible to note that the presence of annealing of the laplace smooth parameter during training presented even worse results than the network configurations without annealing.

By seeing the performance of the network configurations on CamVid test set (Table 2), it is possible to see that while the presence of annealing improves the mean IoU it seems to decrease the global accuracy of the network configuration.

It is also possible to see that, the presence of a Dirichlet Output Layer appears to increase the global accuracy.

The reasons for the better calibration of the *Baseline* over the other network configurations appear to be related with the fact that the *Baseline* configuration has the presence of weight decay, which significantly reduces the NLL loss overfitting of this model (section 2), thus reducing the miscalibration of this configuration over the others.

Another interesting fact is the worse calibration of the configurations with a Dirichlet Output Layer compared to the configurations without it. The reason behind this fact appears to be that, by using a softmax layer (8), we are imposing the constraint that the concentration parameters should all sum up to 1 and that may be miscalibrating the probabilistic output of this configuration. One possible way to exclude this hypothesis would be to use a softplus layer instead of a softmax layer before the scaling.

Models	Pretrained	# of Parameters (M)	Building	Tree	Sky	Car	Sign	Road	Pedestrian	Fence	Pole	Sidewalk	Mean IoU	Global Accuracy
SegNet	✓	29.5	68.7	52.0	87.0	58.5	13.4	86.2	25.3	17.9	16.0	24.8	46.4	62.5
FCN8	✓	134.5	77.8	71.0	88.7	76.1	32.7	91.2	41.7	24.4	19.9	72.7	57.0	88.0
DeepLab-LFOV	✓	37.3	81.5	74.6	89.0	82.2	42.3	92.2	48.4	27.2	14.3	75.4	61.6	-
Dilation8	✓	140.8	82.6	76.2	89.0	84.0	46.9	92.2	56.3	35.8	23.4	75.3	65.3	79.0
Dilation8 + FSO	✓	140.8	**84.0**	77.2	91.3	**85.6**	**49.9**	92.5	59.1	**37.6**	16.9	76.0	66.1	88.3
FC-DenseNet103	X	9.4	83.0	**77.3**	**93.0**	77.3	43.9	**94.5**	**59.6**	37.1	**37.8**	**82.2**	**66.9**	91.5
Baseline	X	2.1	77.6	68.4	88.6	69.6	37.3	88.1	44.1	30.2	34.0	75.2	57.9	92.1
Dirichlet	X	2.1	77.6	69.2	90.1	73.1	44.1	89.3	46.9	30.5	34.3	78.5	59.8	**92.9**
Baseline w/Annealing	X	2.1	81.6	71.5	90.5	63.8	39.5	93.3	48.7	44.2	30.8	75.3	60.6	91.2
Dirichlet w/Annealing	X	2.1	82.1	70.3	90.4	65.1	41.9	93.9	48.9	35.1	29.9	77.2	60.2	91.5

Table 2: Results on CamVid test set. (Just as in the paper from Jegou et al.[8] the values for each class are the mean-iou of that class)

4.3. Temperature Scaling Results

Here in this section we'll make the calibration analysis of Temperature Scaling method performance on the *Baseline* network configurations.

	Uncalibrated	Optimal NLL	Optimal ECE	Optimal MCE
T	1.00	1.34	1.33	1.21
ECE [%]	1.22	0.23	**0.23**	0.51
MCE [%]	7.22	5.87	5.90	**4.98**

Table 3: *Baseline* Temperature Calculation on CamVid Validation Set using L-BFGS-B.

On the Table 3, we calculated the optimal temperature for each one of the 3 loss functions considered and then we computed the ECE and MCE to compare it with the uncalibrated *Baseline* configuration. It is interesting to note that the ECE and NLL loss functions had similar optimal temperatures.

	Uncalibrated	Optimal NLL	Optimal ECE	Optimal MCE
T	1.00	1.34	1.33	1.21
ECE [%]	3.43	**1.64**	1.70	2.35
MCE [%]	12.54	**4.97**	5.18	7.74

Table 4: *Baseline* Calibration Analysis on CamVid Test Set.

Looking at the Table 4, we can see that in the test set the NLL loss was the one with the best performance on the *Baseline* configuration.

5. Conclusion

In this paper we have a proposed a during training deep neural network calibration method, a Bayesian alternative, in which the non-probabilistic output of the neural network serves as the concentration parameters of a Dirichlet distribution. We later compared this approach performance against the best performing one according to our research: *temperature scaling*. The results are as following, on the one hand the Dirichlet output layer made the calibration performance even worse than the baseline and the reasons behind it are detailed in section 1, on the other hand *temperature scaling* as expected performed well and improved over baseline.

Many different tests, and experiments have been left for the future due to the lack of time (i.e. the training of neural networks is usually very time consuming), these include testing using other neural networks and or other data sets. Although, temperature scaling, has the best performance for different neural networks and data sets including deeplabV3 plus on camvid, the urge for another alternative that provides an out of the box calibrated network after training, is still strong due the inconvenience of post processing techniques.

References

[1] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 1721–1730, New York, NY, USA, 2015. ACM.

[2] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. *arXiv e-prints*, page arXiv:1802.02611, Feb 2018.

[3] Alberto Garcia-Garcia, Sergio Orts-Escolano, Sergiu Oprea, Victor Villena-Martinez, and José García Rodríguez. A review on deep learning techniques applied to semantic segmentation. *CoRR*, abs/1704.06857, 2017.

[4] Jochen Gast and Stefan Roth. Lightweight Probabilistic Deep Networks. *arXiv e-prints*, page arXiv:1805.11327, May 2018.

[5] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On Calibration of Modern Neural Networks. *arXiv e-prints*, page arXiv:1706.04599, Jun 2017.

[6] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. Deep Speech: Scaling up end-to-end speech recognition. *arXiv e-prints*, page arXiv:1412.5567, Dec 2014.

[7] Simon Jégou, Michal Drozdzal, David Vázquez, Adriana Romero, and Yoshua Bengio. The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation. *CoRR*, abs/1611.09326, 2016.

[8] Simon Jégou, Michal Drozdzal, David Vazquez, Adriana Romero, and Yoshua Bengio. The One Hundred Layers Tiramisu: Fully Convolutional DenseNets for Semantic Segmentation. *arXiv e-prints*, page arXiv:1611.09326, Nov 2016.

[9] Alex Kendall, Vijay Badrinarayanan, and Roberto Cipolla. Bayesian SegNet: Model Uncertainty in Deep Convolutional Encoder-Decoder Architectures for Scene Understanding. *arXiv e-prints*, page arXiv:1511.02680, Nov 2015.

[10] Azadeh Sadat Mozafari, Hugo Siqueira Gomes, Steeven Janny, and Christian Gagné. A new loss function for temperature scaling to have better calibrated deep networks. *CoRR*, abs/1810.11586, 2018.

[11] Mahdi Pakdaman Naeini, Gregory F. Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *AAAI*, pages 2901–2907. AAAI Press, 2015.

[12] John C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *ADVANCES IN LARGE MARGIN CLASSIFIERS*, pages 61–74. MIT Press, 1999.

[13] Alahari K. Ladicky L. Torr P.H.S. Sturgess, P. Combining appearance and structure from motion features for road scene understanding. In *BMVC*, 2009.

[14] Wikipedia. Dirichlet-verteilung — wikipedia, die freie enzyklopdie, 2018. [Online; Stand 19. Mrz 2019].

[15] Bianca Zadrozny and Charles Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 609–616, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[16] Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 694–699, New York, NY, USA, 2002. ACM.