Calibrating Imbalanced Classifiers with Focal Loss: An Empirical Study

Cheng Wang, Jorge Balazs, Gyuri Szarvas

Patrick Ernst, Lahari Poddar, Pavel Danchenko
Amazon
{cwngam, jabalazs, szarvasg, peernst, poddarl, danchenk}@amazon.com

Abstract

Imbalanced data distribution is a practical and common challenge in building machine learning (ML) models in industry, where data usually exhibits long-tail distributions. For instance, in virtual AI Assistants, such as Google Assistant, Amazon Alexa and Apple Siri, the *play music* or *set timer* utterance is exposed to an order of magnitude more traffic than other skills. This can easily cause trained models to overfit to the majority classes, categories or intents, leading to model miscalibration. The uncalibrated models output unreliable (mostly overconfident) predictions, which are at high risk of affecting downstream decision-making systems. In this work, we study the calibration of models in the practical application of predicting *product return reason codes* in customer service conversations of an online retail store; The returns reasons also exhibit class imbalance. To alleviate the resulting miscalibration in the trained ML model, we streamline the model development and deployment using *focal loss* (Lin et al., 2017). We empirically show the effectiveness of model training with focal loss in learning better calibrated models, as compared to standard cross-entropy loss. Better calibration, in turn, enables better control of the precision-recall trade-off for the trained models.

1 Introduction

Building and developing ML models in industry has many practical challenges. Imbalanced data distributions (He and Garcia, 2009), particularly long-tail distributions (Wang et al., 2021a), make models overfit to majority data classes and lead to miscalibration (Guo et al., 2017; Mukhoti et al., 2020), i.e. the model-predicted probability fails to estimate the likelihood of true correctness and provides over- or under-confident predictions. To address imbalanced data, some mainstream strategies are rebalancing the dataset through upsampling minority and/or downsampling majority classes.

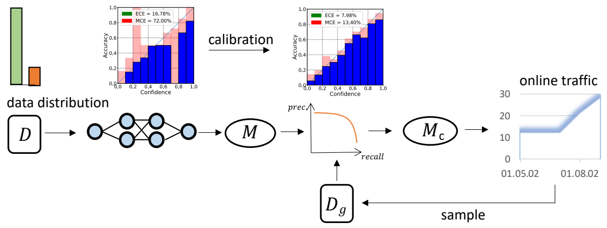


Figure 1: Our procedure to calibrate the ML model that trained with an imbalanced dataset D. With focal loss, model M is learned to be a better calibrated model M_c. In a follow-up stage, a dataset D_g is sampled from actual conversations (the inference distribution) and manually annotated as a golden dataset. It is used to further calibrate models and tune precision-recall threshold to achieve specific precision or recall to serve customer requests.

See SMOTE (Chawla et al., 2002) for an example. However, miscalibration caused by imbalanced data cannot be easily handled by these methods. Guo et al. (Guo et al., 2017) recently found that negative log likelihood (NLL) trained deep neural networks (DNNs) tend to be poorly calibrated as compared to traditional ML methods (Niculescu-Mizil and Caruana, 2005a).

Calibrating models that have been trained with imbalanced data plays an important role in industry. In applications such as medical diagnosis (Caruana et al., 2015), decisions do not only depend on a predicted class, but also on the predicted probabilities, e.g. to quantify patients' risks (Caruana et al., 2015) in order to determine appropriate treatments. Therefore, it is of particular importance that models are well calibrated in high-risk domains such as medicine, financial management (Fischer and Krauss, 2018), self-driving cars (Bojarski et al., 2016), etc., to assure that the predicted probabilities reflect the true values.

In this work, we empirically study the effectiveness of focal loss (Lin et al., 2017) in building reliable ML models for a practical text classification task – *return reason code prediction* in customer service chatbots. Formally, focal loss is:

$$\mathcal{L}_f = -\sum_{i=1}^{N}(1 - p_{i,y_i})^{\gamma} \log p_{i,y_i} \qquad (1)$$

where p_{i,y_i} is predicted probability of the i-th sample and γ is a hyper-parameter that is usually set to $\gamma = 1$.

1.1 Practical Considerations

(1) *Theoretical Effectiveness*. Focal loss was originally proposed to handle the issue of imbalanced data distribution which is frequently observed in industrial data (Kilkki, 2007). In learning models, the majority class samples dominate the optimization and gradient descent to update weights in the direction where models become more confident in predicting the majority class. Focal loss can be interpreted as a trade-off between minimizing Kullback–Leibler (KL) divergence and maximizing the entropy, depending on γ (Mukhoti et al., 2020)[1]:

$$\mathcal{L}_f \geq \mathrm{KL}(q \parallel p) + \underbrace{\mathbb{H}(q)}_{constant} - \gamma\mathbb{H}(p) \qquad (2)$$

The rationale behind the equation is that we learn a probability p to have a high value (confident) due to the KL term, but not too high (overconfident) due to the entropy regularization term (Pereyra et al., 2017).

(2) *Computational and Algorithmic Complexity*. Out of many popular calibration methods such as temperature scaling (Platt et al., 1999), Bayesian methods (Maddox et al., 2019), label smoothing (Müller et al., 2019) and kernel-based methods (Kumar et al., 2018), focal loss neither increases computational overhead nor requires architectural modifications. For example, the widely used temperature scaling (Platt et al., 1999) requires additional post-training calibration while focal loss offers in-training implicit calibration (by using eq. (2)).

In section 4, we will empirically show the intriguing properties of focal loss in calibrating the trained ML models. Our contributions are summarized as follows:

- We empirically examine the effectiveness of using focal loss in handling model miscalibration in a practical application setting.
- We show that good calibration is important to achieve a desired precision or recall target by tuning the classification thresholds. The standard cross-entropy loss is incapable of achieving this goal due to a skewed predicted probability distribution.
- We demonstrate the performance of calibrated models through a chatbot that serves customers' requests across three conversational bot use-cases.

[1]More theoretical findings can be found in the paper

2 Background and Preliminaries

2.1 Background

We consider the task of automatic classification of return reason codes in an online retail store, to showcase the development and deployment of ML models. Whenever a customer requests to return a purchased item, a reason code is determined to select the most appropriate resolution and process a return.

For instance, if a customer is not satisfied with the item (its size, color or material, for example) this would map to the return reason *Customer Preference*. In such a case, the appropriate resolution is to process a return and issue a refund, while replacement with the same item is not appropriate (as the customer would face the same issue). In our case study, we consider two use cases: binary reason code prediction and multi-class reason code prediction, where we consider 5 different categories.

2.2 Preliminaries

In this work, we consider the calibration of supervised binary and multi-class classifiers that are trained with imbalanced datasets.

2.2.1 Model Calibration

Calibration (Guo et al., 2017) measures and verifies how the predicted probability estimates the true likelihood of correctness. Assume a model $\mathbf{m}$ trained with dataset $\{\mathbf{x}, y\}, \mathbf{x} \in \mathcal{X}, y \in \mathcal{Y}$. $\hat{\mathbf{p}}$ is the predicted softmax probability. If $\mathbf{m}$ makes 100 independent predictions, each with confidence $p = \arg\max(\hat{\mathbf{p}}) = 0.9$, ideally, a calibrated $\mathbf{m}$ approximately gives 90 correct predictions. Formally, $\mathrm{accuracy}(\mathbf{m}(D)) = \mathrm{confidence}(\mathbf{m}(D))$ if $\mathbf{m}$ is perfectly calibrated on dataset D. It is difficult to achieve perfect calibration in practice.

2.2.2 Reliability Diagrams

Reliability Diagrams (DeGroot and Fienberg, 1983; Niculescu-Mizil and Caruana, 2005b) visualize whether a model is over- or under-confident by grouping predictions into bins according to their prediction probability. Predictions are grouped into N interval bins (each of width $1/N$) and the accuracy of samples y_i wrt. to the ground truth label $\hat{y}_i$ in each bin b_n is computed as:

$$\text{acc}(b_n) = \frac{1}{I_n} \sum_{i}^{I_n} \mathbb{1}(\hat{y}_i = y_i) \tag{3}$$

where $I_n = |b_n|$ i.e. the number of elements in b_n. Let $\hat{p}_i$ be the probability for sample y_i, then average confidence is defined as

$$\text{conf}(b_n) = \frac{1}{I_n} \sum_{i}^{I_n} \hat{p}_i. \tag{4}$$

A model is perfectly calibrated if $\text{acc}(b_n) = \text{conf}(b_n), \forall n$ and in a diagram the bins would follow the identity function. Any deviation from this represents a miscalibration.

2.2.3 Expected Calibration Error (ECE)

ECE (Naeini et al., 2015) is a scalar summary statistic of calibration. It computes the difference between model accuracy and confidence as a weighted average across bins,

$$\text{ECE} = \frac{1}{I} \sum_{n=1}^{N} I_n |\text{acc}(b_n) - \text{conf}(b_n)|, \tag{5}$$

where I is the total number of samples.

2.2.4 Maximum Calibration Error (MCE)

MCE (Naeini et al., 2015) measures the worst-case deviation between accuracy and confidence,

$$\text{MCE} = \max_{n \in \{1,\ldots,N\}} |\text{acc}(b_n) - \text{conf}(b_n)|. \tag{6}$$

and is particularly important in high-risk applications where reliable confidence measures are absolutely necessary. For a perfectly calibrated classifier, both ECE and MCE are equal to 0.

3 Datasets and Implementation Details

We use historical logged data on return reasons for past human-customer service interactions (which is sometimes noisy), and use human annotated reliable data for model calibration before deployment.

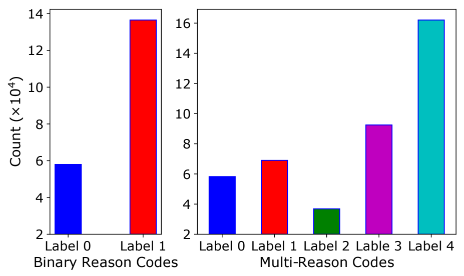


Figure 2: The distribution of randomly sampled datasets for experiments, clearly, we can see the imbalanced label distribution.

Concretely, we prepared 1013 human annotated samples as a golden dataset for the binary reason code model, and a set of 1839 annotated samples for the multi-reason code model. The numbers reported in the tables in Section 4 are based on the annotated dataset.

Figure 2 gives the statistics of randomly sampled datasets from historical logs. We considered 4 widely used return reasons in online retail stores (for details see link below[2]), subsequently referred as `“item is defective or does not work” (LABEL 0)`, `“missing parts or accessories” (LABEL 1)`, `“performance or quality not adequate” (LABEL 2)`, and `“customer preference” (LABEL 3)`. In the binary case, we aim to detect one particular return type `LABEL 0`, and discriminate it from any other return reason (referred as `OTHERS (LABEL 1)`). Similarly in the multi-class case, we consider the four broad return categories (`LABELS 0 .. 3`) and gloss all other return types under `OTHERS (LABEL 4)`. In both settings `OTHERS` is the most frequent class. As we can see the label distribution in both use-cases is imbalanced. For a full dataset D, we split it into $\{D_{train}, D_{val}, D_{test}\}$ with 8:1:1 into train/val/test splits. We train models with standard cross-entropy (CE) and focal loss with different γ values (FL_γ).

We implemented our models with PyTorch (Paszke et al., 2019), each model consists of 2 bidirectional LSTM layers and 2 dense layers. The embedding dimension is 1024. The hidden layer dimension is set to 128 and 512 for the binary and multi-reason code models respectively. We apply dropout with rates of 0.1 and 0.2 to the embedding and dense layers respectively. The models are trained with Adam (Kingma and Ba,

[2]`https://www.amazon.de/-/en/gp/help/customer/display.html?nodeId=G6E3B2E8QPHQ88KF`

2014) as the optimization algorithm

4 Model Selection and Calibration

This section describes how we build a model which is well calibrated while retaining its original predictive performance, e.g., 85% precision or 90% recall. We adapt the focal loss (Lin et al., 2017) and empirically evaluate its calibration effectiveness (Mukhoti et al., 2020) in both binary and multi-class text classification tasks in a practical setting.

We observed the two types of trade-offs which are crucial for using models in practice: (1) discrimination-calibration trade-off and (2) precision-recall trade-off. We also found that the value of γ in focal loss plays an important role in learning better calibrated models. We denote the models trained with cross-entropy as CE and with focal loss, for a given value of γ, as FL_γ.

4.1 Discrimination-Calibration Trade-Off

A calibration method should be predictive (discriminative in our case) performance preserving (Zhang et al., 2020).

4.1.1 Binary Reason Code Prediction

Table 1 presents the results of models with different loss functions. Note that the CE based model achieves the best predictive performance while FL-based models give slightly lower scores. However, FL performs significantly better than CE in terms of calibration related metrics (i.e., NLL, ECE and MCE). We can also observe that the higher the γ value the better calibration performance on human annotated samples.

Figure 3 presents the predicted softmax probability distribution (top) as well as the reliability diagram (bottom). From (a)-(e), we can clearly see that probabilities change from a spiking distribution (overconfident: p is close to either 1 or 0) to a flatter distribution, for instance, $p = \{0.6, 0.4\}$. Figure 3 (f)-(j) show that a miscalibrated model exhibits high ECE and MCE scores. We can also observe this miscalibration through the gaps between confidence and true likelihood of correctness. In this binary imbalance classification case, we find that calibration slightly hurts predictive performance but it is modest. The best model can be obtained by selecting the candidate that achieves the best discrimination-calibration trade-off. Here it should be FL_5 according to Table 1.

4.1.2 Multi-Reason Code Prediction

We further conducted multi-reason code prediction experiments covering 5 reason codes. Table 2 and Figure 4 demonstrate the effectiveness of focal loss in learning well calibrated models. It is important to note that preparing a small set of human labeled samples as a golden dataset is crucial to measure whether a model is calibrated or not. The golden dataset reflects the online data distribution which our model will predict after deployment. The difference in calibration effects can be observed by comparing Figure 4 (top row, (a)-(c)).

4.2 Precision-Recall Tradeoff

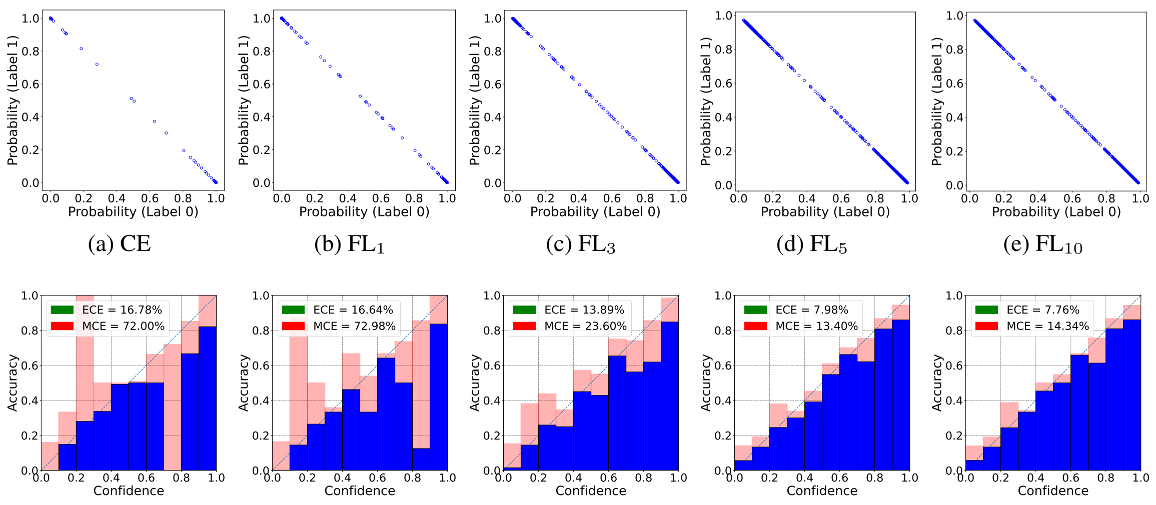
The trade-off between model precision and recall (Buckland and Gey, 1994) is an important aspect in deploying trained models. For instance, if a certain prediction task requires high model precision (recall), it means recall (precision) needs to be sacrificed to some extent. The trade-off can be achieved through classification threshold tuning. In this subsection, we empirically demonstrate that a better calibrated model can help to accomplish this goal.

Figure 5 presents the precision-recall curves for binary reason code models. The corresponding softmax probability distribution is shown in Figure 3 (top). For CE model, Figure 5 (a) and Figure 3 (a), we found that it gives more polarized probability, i.e., the predicted probability is more spiky. Given this skewed distribution, it is difficult to tune a precision based on a given recall, or a certain recall based on an expected precision, e.g., 85%. On the other hand, FL models in Figure 5 (b-d) learn a flattened probability distribution, which is better distributed across the $[0, 1]$ interval and is therefore more amenable to thresholding for a particular precision (or recall) target. This behavior can be also observed in Figure 6 when we compare the computed thresholds in CE and FL models.

5 Experimental Results

To serve customers' requests, we use the binary reason code model for three conversational use-cases[3]. The model detects whether a product return

[3]This empirical study was conducted using the model trained with focal loss, $\gamma = 1$ (FL_1 in Table 1)). Thus it was not the best model variant according to our findings. Note that to minimize risk and maintain customer experience, we don't deploy and compare multiple models online, at same time. However, the superiority of FL-based model as compared to CE-based model is clearly observed from offline results in previous sections.



(a) CE (b) FL_1 (c) FL_3 (d) FL_5 (e) FL_{10}

(f) CE (acc.=83.8%) (g) FL_1(acc.=82.4%) (h) FL_3 (acc.=83.1%) (i) FL_5 (acc.=83.4%) (j) FL_{10} (acc.=81.2%)

Figure 3: The reliability diagram plots for binary-reason code models with 10 bins. The diagonal dash line presents perfect calibration (on a specific bin, confidence is equal to accuracy.)

Metric	CE	FL_1	FL_3	FL_5	FL_{10}
Accuracy	**0.836**	0.824	0.831	0.834	0.816
Precision	**0.838**	0.822	0.830	0.834	0.807
Recall	**0.823**	0.814	0.821	**0.823**	0.805
F1	**0.828**	0.817	0.824	0.827	0.806
NLL	2.159	1.438	0.608	0.258	**0.178**
ECE	0.168	0.166	0.139	0.080	**0.078**
MCE	0.720	0.730	0.236	0.134	**0.143**

Table 1: The performance of binary reason code models with CE and Focal loss (with different γ values) on 1013 human annotated samples. For predictive performance (top rows e.g., accuracy) the differences across models are negligible. However, on the calibration related metrics (bottom rows), FL-based models show significantly better performance. The best scores are marked **bold** and the second best scores are underlined, same as Table 2

Metric	CE	FL_1	FL_5
Accuracy	0.751	**0.760**	0.751
Precision	**0.814**	0.807	**0.814**
Recall	**0.757**	0.755	**0.757**
F1	**0.764**	0.760	**0.764**
NLL	0.599	0.429	**0.309**
ECE	0.037	**0.023**	0.037
MCE	0.296	**0.197**	0.299

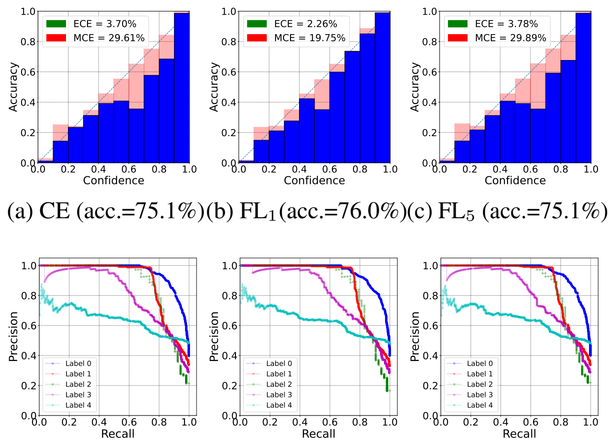
Table 2: The performance of multi-reason code models with CE and focal losses (with different γ values).

is a special case `LABEL 0` according customers' free-text input. We analyze the performance of the model through both intrinsic evaluation of model's accuracy of predicted reason code and extrinsic evaluation on a downstream conversational chatbot system. Before deployment, we tuned model to achieve expected 85% precision with threshold=0.512 for `LABEL 0`.

5.1 Intrinsic Evaluation

We conduct a human evaluation of the model's prediction on a sample of contacts from actual conversations that the model has served. First, we randomly sample 485 contacts where the reason code model predicted the `LABEL 0` code. By manually annotating those contacts, we found 384/485 to be correctly predicted, i.e. the model achieves a precision of 83.8% after deployment. This aligns well with the precision (81.4%) computed on the offline test set.

Although we are primarily interested in the precision metric, we also analyze the negative predictive value of the model for opportunity analysis of future improvements. We randomly sampled 200 contacts where the model *did not predict* the `LABEL 0` code. Out of 200 contacts, we find 194 predic-



(a) CE (acc.=75.1%) (b) FL$_1$ (acc.=76.0%) (c) FL$_5$ (acc.=75.1%)

(d) CE (f1=76.4%) (e) FL$_1$ (f1=76.0%) (f) FL$_5$ (f1=76.4%)

Figure 4: The reliability diagram plots for multi-reason code models with 10 bins. The diagonal dash line presents perfect calibration (on a specific bin, confidence is equal to accuracy.)

tions are true, i.e. the model demonstrates a high precision of $194/200 = 97\%$ for `OTHERS` .

5.2 Extrinsic Evaluation

To evaluate the impact of our model in a downstream application, we run an online A/B experiment, with and without the presented model, for a conversational bot in three use-cases: (1) *control branch*: conversational bot with customer selected reason code; (2) *treatment branch*: conversational bot with additional check for return reason code (`LABEL 0`) in customer free text input.

To evaluate a conversational bot, we measure the following three key metrics: (1) *Automation Rate* (AR): The % of contacts that were resolved by the conversational bot without requiring any human involvement; (2) *Positive Response Rate* (PRR): The rate measures the % of times customers responded positively to the resolution provided by chatbot; (3) *Repeat Rate* (24RR): The rate of customer contact us again for the same issue in 24 hours.

Table 3 presents the results on these key metrics over a period of two weeks. As can be observed from the results, the reason code model improves the performance of the conversational system on both automation rate and customer experience related metrics. This is achieved through enabling the bot to provide appropriate solutions based on the specific customer situations.

Applications	Metrics	C(%)	T (%)
use-case A	AR	–	+ 2.13
	PRR	–	+ 3.18
	24RR	–	- 0.65
use-case B	AR	–	+ 2.10
	PRR	–	+ 0.97
	24RR	–	- 0.68
use-case C	AR	–	+ 3.98
	PRR	–	+12.85
	24RR	–	- 1.02

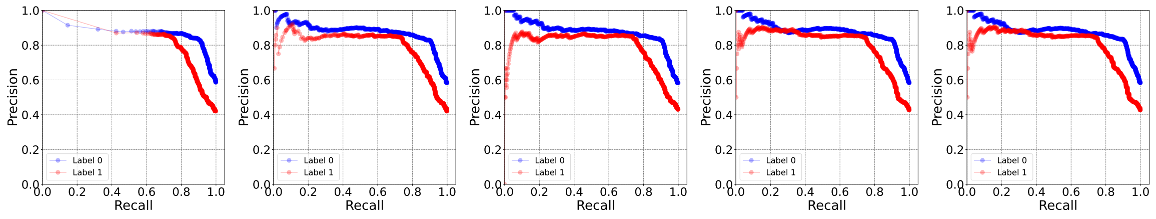
Table 3: Online evaluation of our model for three applications (top, middle, bottom). The relative numbers are reported. AR and PPR, the higher the better. 24RR, the lower the better. We only report performance relative numbers due to confidential issues. C: Control branch; T: Treatment branch.

6 Related Work

Guo *et al.* (Guo et al., 2017) and Mukhoti *et al.* (Mukhoti et al., 2020) showed that the miscalibration of larger, modern networks is related to the over-fitting on the likelihood of the training dataset. Conventional NNs are trained to minimize the negative log likelihood (NLL) which can be positive even if the accuracy is already perfect. Modern networks continue to minimize NLL during training after accuracy is optimal, overfitting to the training dataset and becoming increasingly confident in incorrect predictions as a result. To tackle this, researchers propose a variety of regularizers for model predicted probability including the focal loss (Lin et al., 2017), acting as a maximum entropy regularizer (Mukhoti et al., 2020), and temperature scaling (Guo et al., 2017). Muller *et al.* (Müller et al., 2019) suggest using label smoothing to regularize network outputs. Besides, some recent methods, such as Bayesian method (Maddox et al., 2019), meta-learning (Bohdal et al., 2021), Gumbel-softmax Trick (Jang et al., 2017; Wang et al., 2021b) and kernel-based method (Kumar et al., 2018) are proposed to learn better calibrated model directly from training.

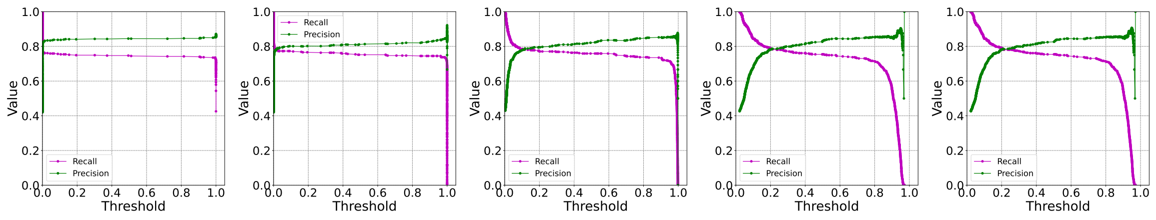
7 Discussion

When calibrating models in practical scenarios, the complexity of calibration is an issue to consider. Focal loss (Lin et al., 2017; Mukhoti et al., 2020) offers in-training calibration via entropy regularization (Pereyra et al., 2017). In this work we have shown analytically (Section 4) that it provides a



(a) CE (f1= 82.8%) (b) FL$_1$ (f1= 81.7 %) (c) FL$_3$ (f1= 82.4%) (d) FL$_5$ (f1= 82.7%) (e) FL$_{10}$ (f1= 80.6%)

Figure 5: The precision-recall curves for binary reason code models. (a) CE model gives more polarized probability and makes it difficult to tune a precision based on a given recall or vice versa. (b-d) FL learns to give better distributed probability, precision or recall can be tuned more easily.



(a) CE (nll= 2.159) (b) FL$_1$ (nll=1.438) (c) FL$_3$ (nll= 0.608) (d) FL$_5$ (nll= 0.258) (e) FL$_{10}$ (nll=0.178)

Figure 6: The curves of precision and recall against threshold in binary reason code models.

simple and effective way to calibrate a trained ML model. On the other hand, in some cases we need to tune γ value for datasets. Experimentally, we found that setting a high γ value would not significantly hurt predictive performance while providing good calibration performance.

8 Conclusion

In this paper, we empirically showed the effectiveness of using focal loss in learning better calibrated models and finding the precision-recall trade-off in practical application of deep neural network models. We conducted an in-depth analysis of miscalibration caused by imbalanced data distribution and the existing issues of using cross-entropy trained models in practical settings. We also showed that the hyperparameter γ, which theoretically controls the entropy regularization term, is important to model calibration. We studied the deployment of an ML model in practical use cases and demonstrated that better calibration helps to control the precision-recall trade-off through posterior thresholding and improves post-deployment metrics (in an online A/B test).

Ethical Considerations

Development and Experiments. We used anonymized text dialogue snippets to train the models. The particular model described in this work has no way to reveal customer information. We do not release the datasets used in the experiments.

Failure Modes. Regarding risks related to system errors, incorrect predictions of the models described in this work may result in wrong return reason assignment. However, the practical risk related to such misclassification is limited, because the customers interacting with the chatbot have an option to talk to a human associate if they consider the system doesn't work as expected.

References

Ondrej Bohdal, Yongxin Yang, and Timothy Hospedales. 2021. Meta-calibration: Meta-learning of model calibration using differentiable expected calibration error. *arXiv preprint arXiv:2106.09613*.

Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. 2016. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*.

Michael Buckland and Fredric Gey. 1994. The relationship between recall and precision. *Journal of the American society for information science*, 45(1):12–19.

Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. 2015. 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*, pages 1721–1730.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357.

Morris DeGroot and Stephen Fienberg. 1983. The comparison and evaluation of forecasters. *The Statistician*.

Thomas Fischer and Christopher Krauss. 2018. Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270(2):654–669.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. 2017. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR.

Haibo He and Edwardo A. Garcia. 2009. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparametrization with gumble-softmax. In *International Conference on Learning Representations (ICLR 2017)*. OpenReview. net.

Kalevi Kilkki. 2007. A practical model for analyzing long tails. *First Monday*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Aviral Kumar, Sunita Sarawagi, and Ujjwal Jain. 2018. Trainable calibration measures for neural networks from kernel mean embeddings. In *International Conference on Machine Learning*, pages 2805–2814. PMLR.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. 2019. A simple baseline for bayesian uncertainty in deep learning. *Advances in Neural Information Processing Systems*, 32.

Jishnu Mukhoti, Viveka Kulharia, Amartya Sanyal, Stuart Golodetz, Philip Torr, and Puneet Dokania. 2020. Calibrating deep neural networks using focal loss. *Advances in Neural Information Processing Systems*, 33:15288–15299.

Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. 2019. When does label smoothing help? *Advances in neural information processing systems*, 32.

Mahdi Pakdaman Naeini, Gregory F. Cooper, and Milos Hauskrecht. 2015. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI)*.

Alexandru Niculescu-Mizil and Rich Caruana. 2005a. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd international conference on Machine learning*, pages 625–632.

Alexandru Niculescu-Mizil and Rich Caruana. 2005b. Predicting Good Probabilities with Supervised Learning. In *Proceedings of the 22nd International Conference on Machine Learning (ICML)*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *NeuIPS'19*, pages 8024–8035.

Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*.

John Platt et al. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74.

Cheng Wang, Sun Kim, Taiwoo Park, Sajal Choudhary, Sunghyun Park, Young-Bum Kim, Ruhi Sarikaya, and Sungjin Lee. 2021a. Handling long-tail queries with slice-aware conversational systems. In *ICLR 2021 Workshop on Weakly Supervised Learning*.

Cheng Wang, Carolin Lawrence, and Mathias Niepert. 2021b. Uncertainty estimation and calibration with finite-state probabilistic rnns. In *ICLR*.

Jize Zhang, Bhavya Kailkhura, and T Yong-Jin Han. 2020. Mix-n-match: Ensemble and compositional methods for uncertainty calibration in deep learning. In *International conference on machine learning*, pages 11117–11128. PMLR.