Posterior Calibrated Training on Sentence Classification Tasks

Taehee Jung[1] **Dongyeop Kang**[2] **Hua Cheng**[3] **Lucas Mentch**[1] **Thomas Schaaf**[3]
[1]Department of Statistics, University of Pittsburgh, Pittsburgh, PA, USA
[2]School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA
[3]3M|M*Modal, Pittsburgh, PA, USA
{taj41,lkm31}@pitt.edu dongyeok@cs.cmu.edu
{hcheng,tschaaf}@mmm.com

Abstract

Most classification models work by first predicting a posterior probability distribution over all classes and then selecting that class with the largest estimated probability. In many settings however, the quality of posterior probability itself (e.g., 65% chance having diabetes), gives more reliable information than the final predicted class alone. When these methods are shown to be poorly calibrated, most fixes to date have relied on posterior calibration, which rescales the predicted probabilities but often has little impact on final classifications. Here we propose an end-to-end training procedure called posterior calibrated (PosCal) training that directly optimizes the objective while minimizing the difference between the predicted and empirical posterior probabilities. We show that PosCal not only helps reduce the calibration error but also improve task performance by penalizing drops in performance of both objectives. Our PosCal achieves about 2.5% of task performance gain and 16.1% of calibration error reduction on GLUE (Wang et al., 2018) compared to the baseline. We achieved the comparable task performance with 13.2% calibration error reduction on xSLUE (Kang and Hovy, 2019), but not outperforming the two-stage calibration baseline. PosCal training can be easily extendable to any types of classification tasks as a form of regularization term. Also, PosCal has the advantage that it incrementally tracks needed statistics for the calibration objective during the training process, making efficient use of large training sets[1].

1 Introduction

Classification systems, from simple logistic regression to complex neural network, typically predict posterior probabilities over classes and decide the final class with the maximum probability. The model's performance is then evaluated by how accurate the predicted classes are with respect to out-of-sample, ground-truth labels. In some cases, however, the quality of posterior estimates themselves must be carefully considered as such estimates are often interpreted as a measure of confidence in the final prediction. For instance, a well-predicted posterior can help assess the fairness of a recidivism prediction instrument (Chouldechova, 2017) or select the optimal number of labels in a diagnosis code prediction (Kavuluru et al., 2015).

Guo et al. (2017) showed that a model with high classification accuracy does not guarantee good posterior estimation quality. In order to correct the poorly calibrated posterior probability, existing calibration methods (Zadrozny and Elkan, 2001; Platt et al., 1999; Guo et al., 2017; Kumar et al., 2019) generally rescale the posterior distribution predicted from the classifier after training. Such post-processing calibration methods re-learn an appropriate distribution from a held-out validation set and then apply it to an unseen test set, causing a severe discrepancy in distributions across the data splits. The fixed split of the data sets makes the post-calibration very limited and static with respect to the classifier's performance.

We propose a simple but effective training technique called Posterior Calibrated (PosCal) training that optimizes the task objective while calibrating the posterior distribution in training. Unlike the post-processing calibration methods, PosCal directly penalizes the difference between the predicted and the true (empirical) posterior probabilities dynamically over the training steps.

PosCal is not a simple substitute of the post-processing calibration methods. Our experiment shows that PosCal can not only reduce the calibration error but also increase the task performance on the classification benchmarks: compared to the baseline MLE (maximum likelihood estimation)

[1]Code is publicly available at https://github.com/THEEJUNG/PosCal/

training method, PosCal achieves 2.5% performance improvements on GLUE (Wang et al., 2018) and 0.5% on xSLUE (Kang and Hovy, 2019), and at the same time 16.1% posterior error reduction on GLUE and 13.2% on xSLUE.

2 Related Work

Our work is primarily motivated by previous analyses of posterior calibration on modern neural networks. Guo et al. (2017) pointed out that in some cases, as the classification performance of neural networks improves, its posterior output becomes poorly calibrated. There are a few attempts to investigate the effect of posterior calibration on natural language processing (NLP) tasks: Nguyen and O'Connor (2015) empirically tested how classifiers on NLP tasks (e.g., sequence tagging) are calibrated. For instance, compared to the Naive Bayes classifier, logistic regression outputs well-calibrated posteriors in sentiment classification task. Card and Smith (2018) also mentioned the importance of calibration when generating a training corpus for NLP tasks.

As noted above, numerous post-processing calibration techniques have been developed: traditional *binning* methods (Zadrozny and Elkan, 2001, 2002) set up bins based on the predicted posterior $\hat{p}$, re-calculate calibrated posteriors $\hat{q}$ per each bin on a validation set, and then update every $\hat{p}$ with $\hat{q}$ if $\hat{p}$ falls into the certain bin. On the other hand, *scaling* methods (Platt et al., 1999; Guo et al., 2017; Kull et al., 2019) re-scale the predicted posterior $\hat{p}$ from the softmax layer trained on a validation set. Recently, Kumar et al. (2019) pointed out that such re-scaling methods do not actually produce well-calibrated probabilities as reported since the true posterior probability distribution can not be captured with the often low number of samples in the validation set[2] . To address the issue, the authors proposed a scaling-binning calibrator, but still rely on the validation set.

In a broad sense, our end-to-end training with the calibration reduction loss can be seen as sort of regularization designed to mitigate over-fitting. Just as classical explicit regularization techniques such as the lasso (Tibshirani, 1996) penalize models large weights, here we penalize models with posterior outputs that differ substantially from the estimated true posterior.

[2]§4 shows that the effectiveness of re-calibration decreases when the size of the validation set is small.

3 Posterior Calibrated Training

In general, most of existing classification models are designed to maximize the likelihood estimates (MLE). Its objective is then to minimize the cross-entropy (Xent) loss between the predicted probability and the true probability over k different classes.

During training time, PosCal minimizes the cross-entropy as well as the calibration error as a multi-task setup. While the former is a task-specific objective, the latter is a *statistical objective* to make the model to be statistically well-calibrated from its data distribution. Such data-oriented calibration makes the task-oriented model more reliable in terms of its data distribution. Compared to the prior post-calibration methods with a fixed (and often small) validation set, PosCal *dynamically* estimates the required statistics for calibration from the train set during training iterations.

Given a training set $\mathcal{D} = \{(x_1, y_1)..(x_n, y_n)\}$ where x_i is a p-dimensional vector of input features and y_i is a k-dimensional one-hot vector corresponding to its true label (with k classes), our training minimizes the following loss:

$$\mathcal{L}_{\text{PosCal}} = \mathcal{L}_{xent} + \lambda \mathcal{L}_{cal} \tag{1}$$

where $\mathcal{L}_{xent}$ is the cross-entropy loss for task objective (i.e., classification) and $\mathcal{L}_{cal}$ is the calibration loss on the cross-validation set. λ is a weighting value for a calibration loss $\mathcal{L}_{cal}$. In practice, the optimal value of λ can be chosen via cross-validation. More details are given in §4.

Each loss term can be then calculated as follows:

$$\mathcal{L}_{xent} = -\sum_{i=1}^{n}\sum_{j=1}^{k} y_i^{(j)} log(\hat{p}_i^{(j)}) \tag{2}$$

$$\mathcal{L}_{cal} = \sum_{i=1}^{n}\sum_{j=1}^{k} d(\hat{p}_i^{(j)}, q_i^{(j)}) \tag{3}$$

where $\mathcal{L}_{xent}$ is a typical cross-entropy loss with $\hat{p}$ as an updated predicted probability while training. $\mathcal{L}_{cal}$ is our proposed loss for minimizing the calibration loss: q is an true (empirical) probability and d is an function to measure the difference (e.g., mean squared error or Kullback-Leibler divergence) between the updated $\hat{p}$ and true posterior q probabilities. The empirical probability q can be calculated by measuring the ratio of true labels per each bin split by the predicted posterior $\hat{p}$ from each update. We sum up the losses from every class $j \in \{1, 2..k\}$.

Algorithm 1 Posterior Calibrated Training

```
    Inputs :
      Train set $\mathcal{D}$, Bin $B$, Number of Classes $K$
      Number of epochs $e$, Learning rate $\eta$
      Number of updating empirical probabilities $u$
    Output $\Theta$: Model Parameters
 1: Let $\mathcal{Q}$ : Empirical Probability Matrix $\in \mathbb{R}^{B \times K}$
 2: Random initialization of $\Theta$
 3: for $i \in \{1, 2, 3, ...e\}$ do
 4:     Break $\mathcal{D}$ into random mini-batches b
 5:     Find a set of steps $\mathcal{S}$ for updating $\mathcal{Q}$ by
        dividing total number of steps into $u$ equal
        parts
 6:     for b from $\mathcal{D}$ do
 7:         $\Theta \leftarrow \Theta - \eta \nabla_{\Theta} \mathcal{L}_{PosCal}(\Theta, \mathcal{Q})$
 8:         if current step $\in \mathcal{S}$ then
 9:             $\hat{p}$ = softmax$(\Theta, \mathcal{D})$
10:             $\mathcal{Q} \leftarrow CalEmpProb(\hat{p}, B)$
11:         end if
12:     end for
13: end for
```

We show a detailed training procedure of PosCal in Algorithm 1. While training, we update the model parameters (i.e., weight matrices in the classifier) as well as the empirical posterior probabilities by calculating the predicted posterior with the recently updated parameters. For $\mathcal{Q}$, we exactly calculate a label frequency per bin B. Since it is time-consuming to update $\mathcal{Q}$ at every step, we set up the number of $\mathcal{Q}$ updates per each epoch so as to only update $\mathcal{Q}$ at each batch.

4 Experiment

We investigate how our end-to-end calibration training produces better calibrated posterior estimates without sacrificing task performance.

Task: NLP classification benchmarks. We test our models on two different benchmarks on NLP classification tasks: GLUE (Wang et al., 2018) and xSLUE (Kang and Hovy, 2019). GLUE contains different types of general-purpose natural language understanding tasks such as question-answering, sentiment analysis and text entailment. Since true labels on the test set are not given from the GLUE benchmark, we use the validation set as the test set, and randomly sample 1% of train set as a validation set. xSLUE (Kang and Hovy, 2019) is yet another classification benchmark but on different types of styles such as a level of humor, formality and even demographics of authors. For the details of each dataset, refer to the original papers.

Metrics. In order to measure the task performance, we use different evaluation metrics for each task. For GLUE tasks, we report `F1` for MRPC, `Matthews` correlation for CoLA, and `accuracy` for other tasks followed by Wang et al. (2018). For xSLUE, we use `F1` score.

To measure the calibration error, we follow the metric used in the previous work (Guo et al., 2017); Expected Calibration Error (`ECE`) by measuring how the predicted posterior probability is different from the empirical posterior probability: $\texttt{ECE} = \frac{1}{K}\sum_{k=1}^{K}\sum_{b=1}^{B}\frac{|B_{kb}|}{n}|q_{kb} - \hat{p}_{kb}|$, where $\hat{p_{kb}}$ is an averaged predicted posterior probability for label k in bin b, q_{kb} is a calculated empirical probability for label k in bin b, B_{kb} is a size of bin b in label k, and n is a total sample size. The lower `ECE`, the better the calibration quality.

Models. We train the classifiers with three different training methods: **MLE**, **L1**, and **PosCal**. **MLE** is a basic maximum likelihood estimation training by minimizing the cross-entropy loss, **L1** is MLE training with L_1 regularizer, and **PosCal** is our proposed training by minimizing $\mathcal{L}_{\text{PosCal}}$ (Eq 1). For PosCal training, we use Kullback-Leibler divergence to measure $\mathcal{L}_{cal}$. We also report `ECE` with a temperature scaling (Guo et al., 2017) (**tScal**), which is considered the state-of-the-art post-calibration method.

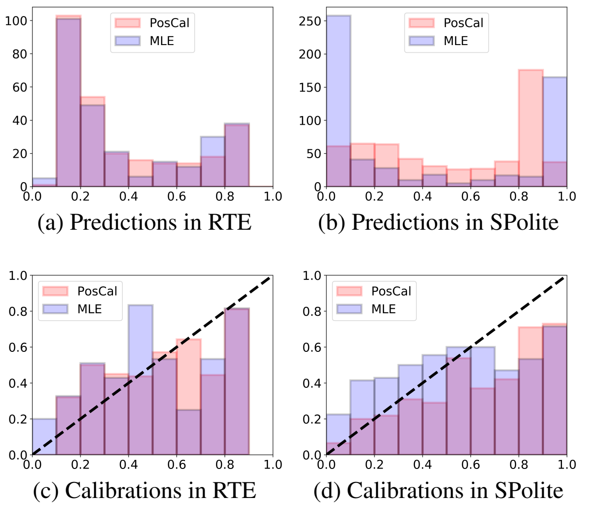
For our classifiers, we fine-tuned the pre-trained BERT classifier (Devlin et al., 2019). Details on the hyper-parameters used are given in Appendix A.

	`Task Perf.` (↑)			`Calib. ECE` (↓)			
Dataset	**MLE**	**L_1**	**PosCal**	**MLE**	**L_1**	**tScal**	**PosCal**
`CoLA`	56.7	55.3	**58.0**	.242	**.234**	.565	.231
`SST-2`	92.1	91.4	**92.4**	.144	.155	.143	**.106**
`MRPC`	88.2	88.2	**88.9**	.228	.229	.400	**.177**
`QQP`	88.8	88.9	**89.1**	.121	.122	**.054**	.107
`MNLI`	**84.0**	83.7	83.5	.158	.160	**.080**	.165
`MNLI`$_{mm}$	83.7	84.0	**84.2**	.153	.153	**.062**	.149
`QNLI`	89.9	89.7	**90.0**	.138	.124	.159	.176
`RTE`	61.7	62.4	**62.8**	.422	.441	**.175**	.394
`WNLI`	38.0	38.0	**56.9**	.287	.287	.269	**.083**
total	75.9	75.6	**78.4**	.210	.212	.252	**.176**

Table 1: Task performance (left; higher better) and calibration error (right; lower better) on GLUE. We do not include STS-B; a regression task. Note that **tScal** is only applicable for calibration reduction, because the post-calibration does not change the task performance, while **PosCal** can do both.

	Task Perf.(↑)			Calib. ECE(↓)			
Dataset	MLE	L_1	PosCal	MLE	L_1	tScal	PosCal
GYAFC	89.1	89.4	**89.5**	.178	.170	.783	**.118**
SPolite	68.7	70.0	**70.9**	.451	.431	**.133**	.238
SHumor	97.4	**97.6**	**97.6**	.050	.047	**.037**	.044
SJoke	**98.4**	98.1	98.3	.032	.037	**.019**	.029
SarcGhosh	42.5	42.5	**42.6**	.912	.912	**.898**	.910
SARC	71.3	71.5	**71.4**	.372	.375	**.079**	.186
SARC_pol	72.7	72.8	**73.8**	.434	.435	**.070**	.383
VUA	80.9	80.8	**81.4**	.268	.276	.687	**.238**
TroFi	76.7	**78.8**	77.4	.278	**.239**	.345	.265
CrowdFlower	22.0	**22.7**	22.6	.404	.413	**.261**	.418
DailyDialog	48.3	47.8	**48.7**	.225	.227	**.117**	.222
HateOffens	93.0	**93.6**	93.5	.064	.059	.100	**.055**
SRomance	99.0	99.0	**100.0**	.020	.020	.023	**.010**
SentiBank	96.7	**97.0**	96.6	.061	.057	**.037**	.054
PASTEL_gender	47.9	**48.1**	47.9	.336	.305	.185	**.143**
PASTEL_age	**23.5**	23.4	22.9	.354	.365	**.222**	.369
PASTEL_count	56.1	56.6	**58.3**	.054	.055	**.019**	.046
PASTEL_polit	46.6	**47.0**	46.8	.394	.379	**.160**	.413
PASTEL_educ	24.4	**25.2**	24.7	.314	.332	**.209**	.323
PASTEL_ethn	**25.3**	24.8	24.8	.245	.243	**.163**	.250
total	64.0	64.3	**64.5**	.272	.269	**.227**	.236

Table 2: Task performance (left; higher better) and calibration error (ECE; lower better) on xSLUE. We do not include EmoBank; a regression task.



(a) Predictions in RTE (b) Predictions in SPolite

(c) Calibrations in RTE (d) Calibrations in SPolite

Figure 1: Histogram of predicted probabilities (top) and their calibration histograms (bottom) between **MLE** (blue-shaded) and **PosCal** (red-shaded) on RTE in GLUE and SPoliteness in xSLUE. The overlap is purple-shaded . X-axis is the predicted posterior, and Y-axis is its frequencies (top) and empirical posterior probabilities (bottom). The diagonal, linear line in (c,d) means the expected (or perfectly calibrated) case. We observe that PosCal alleviate the posterior probabilities with the small predictions toward the expected calibration . Best viewed in color.

Results. Table 1 and 2 show task performance and calibration error on two benchmarks: GLUE and xSLUE, respectively. In general, PosCal outperforms the MLE training and MLE with L_1 regularization in GLUE for both task performance and calibration, though not in xSLUE. Compared to the tScal, PosCal shows a stable improvement over different tasks on calibration reduction, while tScal sometimes produces a poorly calibrated result (e.g., CoLA, MRPC).

Analysis. We visually check the statistical effect of PosCal with respect to calibration. Figure 1 shows how predicted posterior distribution of **PosCal** is different from **MLE**. We choose two datasets where PosCal improves both accuracy and calibration quality compared with the basic MLE: RTE from GLUE and Stanford's politeness dataset from xSLUE. We then draw two different histograms: a histogram of $\hat{p}$ frequencies (top) and a calibration histogram, $\hat{p}$ versus the empirical posterior probability q (bottom). Figure 1(c,d) show that PosCal spreads out the extremely predicted posterior probabilities (0 or 1) from MLE to be more well calibrated over different bins. The well-calibrated posteriors also help correct the skewed predictions in Figure 1(a,b).

Data	MLE → PosCal predictions	Size (%)	MLE avg($\hat{p}$)	PosCal avg($\hat{p}$)	label dist. 0	1
RTE	COR → COR	164(59.2)	79.2	78.6	42.8	47.2
	COR → INCOR	3(1.1)	59.7	39.0	0	100
	INCOR → COR	9(3.3)	40.6	56.7	100	0
	INCOR → INCOR	101(36.4)	23.6	24.9	27.7	72.3
SPolite.	COR → COR	342(60.3)	95.0	82.6	58.8	41.2
	COR → INCOR	54(9.5)	82.1	26.8	96.3	3.7
	INCOR → COR	60(10.6)	16.9	73.9	15.0	85.0
	INCOR → INCOR	111(19.6)	9.8	21.7	54.0	46.0

Table 3: Size of correct (**COR**) and incorrect (**INCOR**) prediction labels with their averaged $\hat{p}(\%)$ of true labels for **MLE** and **PosCal** on RTE and Stanford's politeness (**SPolite**) dataset. Each has two labels : entail(0) / not entail(1) for RTE, and polite(0) / impolite(1) for SPolite. PosCal improves 2.2%/1.1% accuracy than MLE for RTE/SPolite.

To better understand in which case PosCal helps correct the wrong predictions from MLE, we analyze how prediction $\hat{p}$ is different between MLE and PosCal in test set. Table 3 shows the number of correct/incorrect predictions and its correspond-

Data	Sentence	True label	MLE $\hat{p}$	PosCal $\hat{p}$
RTE	**(S1)** Researchers at the Harvard School of Public Health say that people who drink coffee may be doing a lot more than keeping themselves awake - this kind of consumption apparently also can help reduce the risk of diseases. **(S2)** Coffee drinking has health benefits.	entail	49.7	51.3 INCOR $\rightarrow$ COR
RTE	**(S1)** The biggest newspaper in Norway, Verdens Gang, prints a letter to the editor written by Joe Harrington and myself. **(S2)** Verdens Gang is a Norwegian newspaper.	entail	43.9	61.9 INCOR $\rightarrow$ COR
SPolite.	Not at all clear what you want to do. What is the full expected output?	impolite	10.5	74.9 INCOR $\rightarrow$ COR
SPolite.	Are you sure that it isn't due to the error that the compiler is thrown off, and generating multiple errors due to that one error? Could you give some example of this?	polite	6.9	57.9 INCOR $\rightarrow$ COR

Table 4: Predicted $\hat{p}(\%)$ of true label from **MLE** and **PosCal** with corresponding sentences in RTE and SPolite dataset. True label is either entail or not entail for RTE, and polite or impolite for SPolite. Provided examples are the cases only PosCal predicts correctly, which correspond to INCOR $\rightarrow$ COR in table 3.

ing label distributions grouped by the two models. For example, COR by MLE and INCOR by PosCal in the fourth row of Table 3 means that there are three test samples that MLE correctly predicts while PosCal not.

We find that in most of cases, PosCal corrects the wrong predictions from MLE by re-scaling $\hat{p}$ in a certain direction. In RTE, most inconsistent predictions between MLE and PosCal have their posterior predictions near to the decision boundary (i.e., 50% for binary classification) with an averaged predicted probability about 40%. This is mainly because PosCal does not change the majority of the predictions but helps correct the controversial predictions near to the decision boundary. PosCal improves 3.3% of accuracy but only sacrifices 1.1% by correctly predicting the samples predicted as 'not entailment' by MLE to 'entailment'.

On the other hand, SPolite has more extreme distribution of $\hat{p}$ from MLE than RTE. We find a fair trade-off between two models (-9.5%, +10.6%) but still PosCal outperforms MLE.

Table 4 shows examples that only PosCal predicts correctly, with corresponding $\hat{p}$ of true label from MLE and PosCal (INCOR $\rightarrow$ COR cases in Table 3). The predicted probability $\hat{p}$ should be greater than 50% if models predict the true label.

In the first example of RTE dataset, two expressions from S1 and S2 (e.g, "reduce the risk of disease" in S1 and "health benefits" in S2) make MLE confusing to predict, so $\hat{p}$ of true label becomes slightly less than the borderline probability (e.g., $\hat{p} = 49.7\% < 50\%$), making incorrect prediction. Another example of RTE shows how the MLE fails to predict the true label since the model cannot learn the connection between the location of newspaper (e.g., "Norway") and its name (e.g., "Verden Gang"). In the two cases from SPolite dataset, the level of politeness indicated on phrases (e.g., "Not at all" in the first case and "Could you" in the second case) is not captured well by MLE, so the model predicts the incorrect label.

From our manual investigation above, we find that statistical knowledge about posterior probability helps correct $\hat{p}$ while training PosCal, so making $\hat{p}$ switch its prediction. For further analysis, we provide more examples in Appendix C.

5 Conclusion and Future Directions

We propose a simple yet effective training technique called PosCal for better posterior calibration. Our experiments empirically show that PosCal can improve both the performance of classifiers and the quality of predicted posterior output compared to MLE-based classifiers. The theoretical underpinnings of our PosCal idea are not explored in detail here, but developing formal statistical support for these ideas constitutes interesting future work. Currently, we fix the bin size at 10 and then estimate q by calculating accuracy of p per bin. Estimating q with adaptive binning can be a potential alternative for the fixed binning.

Acknowledgements

We thank Matt Gormley and the anonymous reviewers for their helpful comments and discussion.

References

Dallas Card and Noah A Smith. 2018. The importance of calibration for estimating proportions from annotations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1636–1646.

Alexandra Chouldechova. 2017. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data*, 5(2):153–163.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330, International Convention Centre, Sydney, Australia. PMLR.

Dongyeop Kang and Eduard H. Hovy. 2019. xslue: A benchmark and analysis platform for cross-style language understanding and evaluation. *ArXiv*, abs/1911.03663.

Ramakanth Kavuluru, Anthony Rios, and Yuan Lu. 2015. An empirical evaluation of supervised learning approaches in assigning diagnosis codes to electronic medical records. *Artificial intelligence in medicine*, 65(2):155–166.

Meelis Kull, Miquel Perello Nieto, Markus Kängsepp, Telmo Silva Filho, Hao Song, and Peter Flach. 2019. Beyond temperature scaling: Obtaining well-calibrated multi-class probabilities with dirichlet calibration. In *Advances in Neural Information Processing Systems*, pages 12295–12305.

Ananya Kumar, Percy S Liang, and Tengyu Ma. 2019. Verified uncertainty calibration. In *Advances in Neural Information Processing Systems*, pages 3787–3798.

Khanh Nguyen and Brendan O'Connor. 2015. Posterior calibration and exploratory analysis for natural language processing models. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1587–1598, Lisbon, Portugal. Association for Computational Linguistics.

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.

Robert Tibshirani. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Bianca Zadrozny and Charles Elkan. 2001. 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. Morgan Kaufmann Publishers Inc.

Bianca Zadrozny and Charles Elkan. 2002. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 694–699. ACM.

A Details on Hyper-Parameters

All models are trained with equal hyper-parameters:learning rate 2e-5, and BERT model size BERT$_{BASE}$. Also, we set up an early stopping rule for train: we track the validation loss for every 50 steps and then halt to train if current validation loss is bigger than the averaged 10 prior validation losses (i.e., patience 10). For **L1**, we use the regularization weight value 1-e8. For **PosCal**, we set up another weight value λ for $\mathcal{L}_{Cal}$, and the number of updating empirical probability per epoch (u). We tune these two hyper-parameters per each task. For more details, see Table 5. As a baseline of post-calibration method, we also report ECE with a temperature scaling (Guo et al., 2017), which is current state-of-the-art method.

xSLUE	u	λ	**GLUE**	u	λ
GYAFC	5	0.6	CoLA	5	0.2
SPolite	5	0.6	SST-2	10	1.0
SHumor	5	1.0	MRPC	10	1.0
SJoke	5	1.0	QQP	10	1.0
SarcGhosh	5	0.6	MNLI	2	0.2
SARC	5	0.6	MNLI$_{mm}$	2	0.2
SARC_pol	5	1.0	QNLI	1	0.6
VUA	2	1.0	RTE	10	1.0
TroFi	5	1.0	WNLI	2	0.2
CrowdFlower	5	0.6			
DailyDialog	5	1.0			
HateOffens	5	1.0			
SRomance	5	1.0			
SentiBank	5	1.0			
PASTEL_gender	5	1.0			
PASTEL_age	5	1.0			
PASTEL_count	5	1.0			
PASTEL_polit	5	1.0			
PASTEL_educ	5	1.0			
PASTEL_ethn	5	1.0			

Table 5: Hyper-parameters for PosCal training across tasks : the number of updating empirical probabilities per epoch u and weight value λ for $\mathcal{L}_{Cal}$. We tune them using the validation set.

B Examples When MLE and PosCal Predicts Different Label

Table 6 shows some examples in RTE and Stanford-Politeness datasets with their predicted $\hat{p}$ of true label from **MLE** and **PosCal**.

Data	Sentence	True label	MLE $\hat{p}$	PosCal $\hat{p}$
RTE	(S1) Charles de Gaulle died in 1970 at the age of eighty. He was thus fifty years old when, as an unknown officer recently promoted to the (temporary) rank of brigadier general, he made his famous broadcast from London rejecting the capitulation of France to the Nazis after the debacle of May-June 1940. (S2) Charles de Gaulle died in 1970.	entail	34.9 INCOR $\rightarrow$	58.9 COR
RTE	(S1) Police in the Lower Austrian town of Amstetten have arrested a 73 year old man who is alleged to have kept his daughter, now aged 42, locked in the cellar of his house in Amstetten since 29th August 1984. The man, identified by police as Josef Fritzl, is alleged to have started sexually abusing his daughter, named as Elisabeth Fritzl, when she was eleven years old, and to have subsequently fathered seven children by her. One of the children, one of a set of twins born in 1996, died of neglect shortly after birth and the body was burned by the father. (S2) Amstetten is located in Austria.	entail	45.5 INCOR $\rightarrow$	57.3 COR
RTE	(S1) Blair has sympathy for anyone who has lost their lives in Iraq. (S2) Blair is sorry for anyone who has lost their lives in Iraq.	entail	31.3 INCOR $\rightarrow$	50.1 COR
RTE	(S1) Capital punishment acts as a deterrent. (S2) Capital punishment is a deterrent to crime.	entail	41.6 INCOR $\rightarrow$	64.5 COR
RTE	(S1) The U.S. handed power on June 30 to Iraqâs interim government chosen by the United Nations and Paul Bremer, former governor of Iraq. (S2) The United Nations officially transferred power to Iraq.	not entail	59.2 COR $\rightarrow$	44.9 INCOR
SPolite.	I don't know what page you are talking about, as this is your only edit. Did you perhaps have another account?	impolite	47.3 INCOR $\rightarrow$	65.4 COR
SPolite.	Hi. Not complaining, but why did you remove the category "high schools in california" from this article?	impolite	1.2 INCOR $\rightarrow$	91.7 COR
SPolite.	Hi, sorry I think I'm missing something here. Why are you adding a red link to the vandalism page?	impolite	5.6 INCOR $\rightarrow$	61.9 COR
SPolite.	Huh, looks fine to me. Maybe this computer just lies to me to get me to shut up and stop complaining?	impolite	3.3 INCOR $\rightarrow$	58.1 COR
SPolite.	Can you put an NSLog to make sure it's being called only once? Also, can you show us where you are declaring your int?	polite	16.5 INCOR $\rightarrow$	76.5 COR
SPolite.	I don't understand the reason for <url>. Would you please explain it to me?	polite	91.5 COR $\rightarrow$	37.1 INCOR
SPolite.	Another question: Does "Senn" exist in Japanese? If it does, is it possible to render Sennin as Senn-in?	polite	88.8 COR $\rightarrow$	45.5 INCOR
SPolite.	@Smjg, thanks. But why did you also remove the categories I added?	impolite	78.3 COR $\rightarrow$	45.7 INCOR
SPolite.	You can place islands so there is no path between points. What should happen then?	impolite	91.7 COR $\rightarrow$	35.8 INCOR

Table 6: Predicted $\hat{p}$(%) of true label from **MLE** and **PosCal** with corresponding sentences in RTE (top) and Stanford's politeness (bottom) dataset. True label is either entail or not entail for RTE, and polite or impolite for SPolite. We show the cases where two methods predict the label differently. The case with INCOR $\rightarrow$ COR means only PosCal predicts the true label correctly, while the case with COR $\rightarrow$ INCOR means only MLE predicts the true label correctly.