CONSERVATIVE PREDICTION VIA TRANSDUCTIVE CONFIDENCE MINIMIZATION

Anonymous authors
Paper under double-blind review

1 INTRODUCTION

While deep networks have demonstrated impressive performance, they often exhibit unexpected failures on high-confidence inputs Simonyan & Zisserman (2014); Zhang et al. (2017). Such errors can lead to poor performance or even catastrophic failure, especially in safety-critical applications such as healthcare, and may prevent the deployment of machine learning altogether. This motivates the need for *conservative* models that can abstain from making predictions on inputs when likely to make an error. For example, a model trained to predict a patient's risk of developing a disease may have low confidence on a rare variant of the disease. In such scenarios, it may be preferable to defer to a human expert.

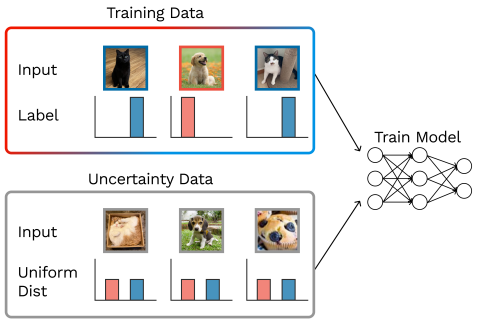

Figure 1: Visual overview of Transductive Confidence Minimization (TCM), our method for training a model to make conservative predictions. We minimize cross-entropy loss on labeled training data and minimize confidence on uncertainty data. The uncertainty dataset is instantiated in different ways for OOD detection (where we use unlabeled data) and selective classification (where we use misclassified validation data).

A common approach to addressing this issue is through out-of-distribution (OOD) detection, which aims to detect when the model is facing an OOD input. However, as noted in prior work (Tajwar et al., 2021), no existing approach can consistently detect OOD examples across different ID-OOD dataset pairs due to the ill-defined nature of the problem setting. To address this challenge, we introduce a transductive assumption in which the model has access to an unlabeled test set during training. While this assumption precludes some applications, there are real-world settings such as unannotated medical data from a new hospital (Sagawa et al., 2021) where unlabeled test examples are available in batch. Our key insight is that the effect of minimizing the model's confidence on all unlabeled datapoints can be "cancelled out" by the regular ID training objective, which maximizes confidence on ID data.

We propose Transductive Confidence Minimization (TCM), a simple method for training a conservative model that can refuse to make predictions on uncertain inputs (Figure 1). TCM minimizes confidence on all examples in the unlabeled test set while minimizing standard cross-entropy loss on the labeled training set. We empirically verify our approach through experiments on a variety of standard benchmarks for OOD detection and selective classification. TCM outperforms recent OOD detection methods, including a method that leverages a very large OOD set that is over $10,000$ times larger than ours (Hendrycks et al., 2018), and one that shares our data assumptions but learns an ensemble of multiple models (Tifrea et al., 2022). In selective classification settings, TCM consistently outperforms the best prior methods (Binary Classifier and Fine-Tuning) when testing on data from a previously unseen distribution.

2 PROBLEM SETUP

We consider two problem settings, out-of-distribution detection and selective classification, which both test the extent to which a model's predictive confidence can be used to determine if its prediction is trustworthy. We denote input and label spaces as $\mathcal{X}$, $\mathcal{Y}$, and we assume that the training dataset D_{tr} is drawn from a distribution $\mathcal{P}_{\text{ID}}$. *Out-of-distribution detection* considers situations where the model may be tested on inputs for which no corresponding label in $\mathcal{Y}$ exists. In *selective classification*, all inputs have a correct label within $\mathcal{Y}$ but the model may make errors due to e.g. overfitting. We next describe the two problem settings; in Section 3, we describe the two corresponding instantiations of our method.

Out-of-distribution detection. In this problem setting, the model may be tested on datapoints from a related but different distribution $\mathcal{P}_{\text{OOD}}$, i.e. out-of-distribution (OOD) data. The test dataset is sampled from a mixture distribution, i.e. $\alpha\mathcal{P}_{\text{ID}} + (1-\alpha)\mathcal{P}_{\text{OOD}}$, where the mixture coefficient α is not known in advance. Because the two distributions are different, a model trained solely to minimize loss on D_{tr} may be inaccurate when tested on novel inputs from $\mathcal{P}_{\text{OOD}}$. To address this difficulty, we assume access to an additional unlabeled set D_{u} that is drawn from a mixture of $\mathcal{P}_{\text{ID}}$ and $\mathcal{P}_{\text{OOD}}$, where the ratio of this mixture is unknown to the model. Using held-out test data, we evaluate the model on two aspects: (1) whether the model's confidence is a good metric for distinguishing whether an input is OOD; and in the case that datapoint is an ID input, (2) whether the model's prediction is accurate.

Selective classification. In the selective classification setting, we have labeled validation data D_{val} in addition to training data. Unless otherwise noted, the validation set is assumed to be drawn from $\mathcal{P}_{\text{ID}}$, and therefore can be constructed by randomly partitioning an original training dataset into training and validation splits. We do not necessarily assume that the validation data includes anomalies. The model is evaluated on test data, which may include examples from a different distribution. Even when some test inputs are from a different distribution, they correspond to a well-defined ground-truth label. The model is evaluated on both accuracy (the ratio of correctly classified inputs) and coverage (the ratio of inputs that the model did not reject). This problem can arise in situations where an incorrect classification attempt has a disproportionately higher cost compared to the benefit from being correct.

3 Transductive Confidence Minimization

We aim to produce a model that achieves high accuracy on training data D_{tr} while having a predictive confidence that reflects the degree to which its prediction will be reliable. The crux of the method is to introduce a regularizer that minimizes confidence on a dataset that is disjoint from the training dataset. We will refer to this dataset as the "uncertainty dataset," since it is intended to contain examples that the model should be uncertain about. The exact choice of uncertainty dataset depends on the problem setting.

We first pre-train a model $f : \mathcal{X} \to \mathbb{P}(\mathcal{Y})$ on the training set using cross-entropy loss $\mathcal{L}_{\text{xent}}(f, D_{\text{tr}}) = -\mathbb{E}_{(x,y)\sim D_{\text{tr}}} [\log f(y; x)]$. However, a model trained solely to minimize cross-entropy can suffer from overconfidence. We therefore fine-tune this model while introducing an additional regularizer to minimize predictive confidence on such inputs. More specifically, in the fine-tuning stage, we continue to minimize cross-entropy loss on a fine-tuning dataset, which the original training data is a subset of ($D_{\text{tr}} \subseteq D_{\text{ft}}$). Our additional regularizer minimizes confidence on the "uncertainty set"; we define the corresponding regularization loss as $\mathcal{L}_{\text{conf}}(f, D_{\text{conf}}) = \mathbb{E}[\log f(y_u; x')]$, where the expectation is taken over $x' \sim D_{\text{conf}}, y_u \sim U$. Our final objective is a weighted sum of the fine-tuning and confidence losses:

$$\mathcal{L}_{\text{xent}}(f, D_{\text{ft}}) + \lambda \mathcal{L}_{\text{conf}}(f, D_{\text{conf}}), \tag{1}$$

where U is the uniform distribution over labels and λ is a hyperparameter. The loss weight hyperparmeter is set to $\lambda = 0.5$ in all experiments unless otherwise specified. The two algorithm variants differ only in how they select D_{ft} and D_{conf}, as described below.

TCM for out-of-distribution detection. Our goal in this problem setting is to produce a model that achieves high accuracy on the ID distribution $\mathcal{P}_{\text{ID}}$ while having low confidence on inputs from the OOD distribution $\mathcal{P}_{\text{OOD}}$. Recall that we assume access to an unlabeled dataset D_{u} which includes a both ID and OOD inputs, and use this entire dataset as the uncertainty dataset for reducing confidence ($D_{\text{conf}} = D_{\text{u}}$). Intuitively, minimizing confidence loss on discourages the model from making overly confident predictions anywhere. We expect this regularization to have different effects on ID and OOD inputs, because of its interaction with the original cross-entropy loss. In the ID data distribution, the confidence loss is "averaged out" by the cross-entropy loss because maximizing the log likelihood of the true label entails increasing the predictive confidence for that input. However, in the OOD data distribution, the confidence loss is the only loss term, so the model is forced to output low-confidence predictions. This difference in ID and OOD examples allows us to distinguish between the two data distributions based on predictive confidence. For the out-of-distribution detection setting, the fine-tuning dataset is exactly the training dataset ($D_{\text{ft}} = D_{\text{tr}}$), and the uncertainty dataset is the unlabeled dataset ($D_{\text{conf}} = D_{\text{u}}$).

TCM for selective classification. Recall that we aim to produce a model that achieves high accuracy while having low confidence on inputs it is likely to misclassify, and that we assume a labeled

ID Dataset	Method	FPR95 (↓)	FPR99 (↓)	AUROC (↑)	AUPR-In (↑)	AUPR-Out (↑)	Rank (↓)
CIFAR-10	MSP	33.8	58.8	89.9	97.7	60.8	6.8
	Outlier Exposure	11.0	24.3	97.4	99.5	92.8	3.6
	Energy fine-tuning	**8.5**	**21.3**	**98.0**	**99.6**	**92.8**	2.6
	TCM-softmax (ours)	12.3	24.9	97.2	99.2	91.7	2.2
	TCM-energy (ours)	12.3	23.3	97.2	99.2	92.3	**1.6**
CIFAR-100	MSP	75.2	85.2	71.2	92.6	30.3	8.0
	Outlier Exposure	60.2	76.3	79.9	95.1	44.1	5.4
	Energy fine-tuning	59.3	75.6	80.7	94.9	46.5	5.0
	TCM-softmax (ours)	13.7	33.4	95.5	98.9	87.3	2.0
	TCM-energy (ours)	**13.1**	**29.4**	**95.8**	**98.9**	**88.4**	**1.0**

Table 1: OOD detection performance of a WideResNet-40-2 model, averaged across five OOD datasets. The average rank is calculated by ranking each method among the 8 methods we compare against for each (ID, OOD) pair according to the average AUROC, and then taking the average over all OOD datasets for a certain ID dataset. TCM outperforms all other methods in 9 out of 10 ID-OOD dataset pairs, as reflected in the average rank. TCM is outperformed by only one method (Energy) in one setting (C10 to C100), but the difference in that dataset dominates the average statistics.

Task	Method	FPR95 (↓)	FPR99 (↓)	AUROC (↑)	AUPR (↑)
CIFAR-10 [0:5] → CIFAR-10 [5:10]	Binary classifier	92.84 (2.44)	97.75 (0.71)	55.00 (4.24)	19.72 (2.10)
	ERD	**72.46 (3.85)**	**92.05 (1.83)**	**79.33 (0.62)**	**47.86 (3.48)**
	TCM-softmax (ours)	**66.00 (5.68)**	**89.18 (2.27)**	**81.22 (0.67)**	**45.72 (1.32)**
	TCM-energy (ours)	**67.34 (5.98)**	**89.10 (2.02)**	**81.44 (0.60)**	**46.32 (1.61)**
CIFAR-100 [0:50] → CIFAR-100 [50:100]	Binary classifier	88.96 (11.45)	92.48 (13.17)	51.4 (2.89)	17.67 (1.82)
	ERD	**75.43 (1.93)**	**88.81 (1.04)**	**71.32 (0.63)**	**30.15 (1.18)**
	TCM-softmax (ours)	**67.28 (1.01)**	**86.34 (1.36)**	**74.26 (0.45)**	**32.08 (1.77)**
	TCM-energy (ours)	**66.70 (1.19)**	**87.56 (2.47)**	**73.88 (0.42)**	**32.08 (1.30)**

Table 2: OOD detection performance on near-OOD detection setting (architecture: ResNet18). Numbers in parenthesis represent the standard deviation over 5 seeds. TCM achieves similar performance to ERD with one third of the computational resources.

held-out validation set, which may or may not be from the same distribution as the training data D_{tr}. After training the model f on training data, we expect its errors on the (initially held-out) validation set to reflect the failure modes of the original model. We obtain the set of correct and misclassified validation examples $D^{\circ}_{\text{val}}, D^{\times}_{\text{val}}$. The misclassified example set $D^{\times}_{\text{val}}$ shows where the model's decision boundary conflicts with the true labeling function. We set the fine-tuning dataset to be the union of the training dataset and the correct validation examples ($D_{\text{ft}} = D_{\text{tr}} \cup D^{\circ}_{\text{val}}$), and use the misclassified validation examples as the uncertainty set ($D_{\text{conf}} = D^{\times}_{\text{val}}$) By minimizing confidence on only the misclassified examples, we expect the model to have lower confidence on all examples which share commonalities with samples which initially produced errors.

4 Experiments

In this section, we conduct experiments to evaluate the effectiveness of TCM for both OOD detection and selective classification on several image classification datasets. We aim to empirically answer the following questions: (1) Does TCM result in better calibration? (2) How does TCM compare to existing methods for OOD detection and selective classification? We provide experimental details and additional empirical results in the appendix.

4.1 OOD Detection

Comparisons. Our primary OOD detection experiments consider six OOD detection methods for comparison: (1) MSP (Hendrycks & Gimpel, 2016), ODIN (Liang et al., 2017a), Mahalanobis (Lee et al., 2018), Energy score Liu et al. (2020), Outlier exposure Hendrycks et al. (2018), and Energy based fine-tuning Liu et al. (2020). We additionally compare to two state-of-the-art methods in more challenging near-OOD detection settings: ERD and Binary Classifier Tifrea et al. (2022).

Results. We present results averaged across OOD datasets in Table 1, with complete results in Appendix E. We find that our method provides the strongest performance on nine out of ten ID/OOD pairs compared to six prior method. Table 2 reports the results of the near-OOD detection results. We see that TCM outperforms binary classification and achieves similar performance as ERD. However, ERD requires an ensemble of three networks, while TCM uses a single network, i.e. one third of the computational resources.

Method	ECE (↓)	AUC (↑)	Acc@90 (↑)	Acc@95 (↑)	Acc@99 (↑)	Cov@90 (↑)	Cov@95 (↑)	Cov@99 (↑)
Val = CIFAR-10, Test = CIFAR-10								
MSP	0.45	99.3	98.4	97.2	95.7	**100**	**100**	87.0
Binary Classifier	1.42	99.3	98.4	97.2	95.7	**100**	**100**	87.0
Fine-Tuning	**0.29**	**99.6**	**99.1**	**98.7**	**97.5**	**100**	**100**	**91.6**
TCM (ours)	1.02	99.2	98.0	96.5	94.8	**100**	98.6	83.9
Val = CIFAR-10, Test = CIFAR-10-C								
MSP	14.5 (5.7)	71.9 (20.0)	59.6 (19.9)	58.1 (19.3)	56.9 (18.8)	27.9 (30.0)	16.3 (24.9)	5.5 (16.2)
Binary Classifier	13.6 (5.8)	72.8 (18.2)	59.5 (19.7)	58.0 (19.2)	56.7 (18.7)	28.5 (29.5)	16.4 (24.3)	8.2 (19.1)
Fine-Tuning	12.8 (5.2)	75.4 (18.2)	61.9 (19.9)	60.3 (19.4)	59.0 (19.0)	33.8 (30.3)	22.7 (26.8)	9.8 (20.4)
TCM (ours)	**12.4 (5.0)**	**77.3 (17.1)**	**63.6 (19.0)**	**61.9 (18.4)**	**60.4 (18.0)**	**36.0 (30.2)**	**25.0 (26.4)**	**11.2 (19.3)**
Val = CIFAR-10, Test = CIFAR-10 + CIFAR-10-C								
MSP	9.3 (3.9)	92.6 (4.0)	80.4 (9.4)	78.3 (9.5)	76.4 (9.4)	72.4 (15.3)	60.6 (15.0)	27.4 (19.7)
Binary Classifier	7.9 (3.2)	92.5 (4.0)	80.3 (9.4)	78.1 (9.5)	76.2 (9.4)	72.0 (15.3)	59.9 (15.1)	30.5 (17.8)
Fine-Tuning	8.3 (3.6)	93.3 (3.8)	81.3 (9.4)	79.0 (9.5)	77.7 (9.6)	74.1 (14.8)	63.3 (14.1)	43.4 (13.2)
TCM (ours)	**8.0 (3.5)**	**93.6 (3.6)**	**82.0 (9.0)**	**79.7 (9.1)**	**77.7 (9.0)**	**75.2 (14.5)**	**63.8 (13.7)**	**43.6 (11.3)**

Table 3: Selective classification performance on distribution shift tasks constructed from the CIFAR-10 and CIFAR-10C datasets. Bold numbers represent superior results, and parentheses show the standard deviation over 15 corruptions. TCM consistently outperforms MSP and Binary Classifier, and outperforms Fine-Tuning when the validation and test sets are from different distributions.

Method	ECE (↓)	AUC (↑)	Acc@90 (↑)	Acc@95 (↑)	Acc@99 (↑)	Cov@90 (↑)	Cov@95 (↑)	Cov@99 (↑)
Val = Camelyon17 ID Val-1, Test = Camelyon17 ID Val-2)								
MSP	33.4 (4.8)	65.6 (7.6)	65.2 (6.0)	64.7 (5.8)	64.2 (5.6)	2.5 (2.6)	2.1 (2.4)	1.5 (2.3)
Binary Classifier	34.6 (6.2)	66.9 (10.4)	63.7 (7.4)	63.0 (7.1)	62.6 (6.9)	3.4 (3.0)	2.9 (2.8)	2.2 (2.8)
Fine-Tuning	**4.6 (2.8)**	**99.6 (0.1)**	**99.1 (0.3)**	**98.3 (0.3)**	**97.1 (0.4)**	**100 (0.0)**	**100 (0.0)**	**90.9 (2.5)**
TCM (ours)	12.2 (1.4)	98.9 (0.3)	97.9 (0.4)	97.1 (0.4)	95.8 (0.5)	**100 (0.0)**	**100 (0.0)**	60.1 (16.3)
Val = Camelyon17 ID Val-1, Test = Camelyon17 OOD Test								
MSP	31.8 (0.6)	64.9 (8.8)	59.8 (3.1)	59.3 (3.0)	58.9 (3.0)	5.8 (7.8)	4.0 (5.4)	1.6 (2.0)
Binary Classifier	31.3 (1.3)	72.2 (1.6)	62.8 (1.1)	62.2 (1.1)	61.7 (1.0)	12.0 (1.0)	8.8 (1.3)	3.9 (1.2)
Fine-Tuning	**19.2 (1.9)**	84.2 (4.6)	**75.2 (5.3)**	**74.1 (4.9)**	**73.2 (4.7)**	27.6 (30.4)	0.7 (0.6)	4.6 (6.3)
TCM (ours)	26.8 (1.6)	**84.6 (2.5)**	74.0 (1.3)	73.0 (1.5)	72.1 (1.5)	**29.7 (17.5)**	**20.8 (13.5)**	**6.5 (0.6)**
Val = Camelyon17 ID Val-1, Test = Camelyon17 ID Val-1 + Camelyon17 OOD								
MSP	29.3 (0.3)	68.5 (6.8)	62.9 (1.9)	62.4 (1.9)	61.9 (1.8)	5.7 (4.7)	4.0 (3.2)	1.6 (1.2)
Binary Classifier	28.2 (1.6)	73.0 (0.9)	64.8 (1.3)	64.0 (0.9)	63.0 (0.2)	9.0 (0.8)	6.1 (0.4)	2.8 (0.8)
Fine-Tuning	**15.0 (1.3)**	91.2 (1.1)	81.3 (2.0)	80.0 (1.8)	78.9 (1.7)	**62.5 (6.2)**	40.1 (5.9)	11.7 (0.1)
TCM (ours)	16.5 (1.3)	**91.8 (0.6)**	**81.6 (1.3)**	**80.2 (1.3)**	**79.1 (1.3)**	59.2 (2.2)	**43.4 (1.3)**	**23.8 (4.1)**
Val = FMoW ID Val, Test = FMoW ID Test								
MSP	1.6 (0.7)	81.8 (1.1)	63.4 (1.5)	61.1 (1.4)	59.2 (1.4)	37.5 (3.0)	20.9 (5.0)	3.3 (5.1)
Binary Classifier	1.9 (0.7)	82.3 (0.6)	64.3 (0.1)	62.0 (0.3)	60.2 (0.2)	37.6 (0.9)	20.9 (7.5)	3.7 (4.6)
Fine-Tuning	1.5 (0.8)	**83.2 (1.1)**	**65.0 (0.9)**	**62.7 (0.8)**	**60.8 (0.8)**	**41.1 (2.3)**	**26.2 (5.3)**	3.5 (5.6)
TCM (ours)	**1.3 (0.8)**	81.9 (2.5)	62.8 (2.8)	60.5 (2.8)	58.7 (2.8)	37.1 (6.2)	24.1 (9.2)	**6.4 (4.8)**
Val = FMoW ID Val, Test = FMoW OOD Test								
MSP	2.8 (0.8)	75.6 (0.9)	**56.8 (0.2)**	**54.9 (0.1)**	**53.3 (0.1)**	21.1 (6.6)	7.5 (8.6)	0.8 (1.1)
Binary Classifier	2.8 (0.8)	75.6 (0.9)	**56.8 (0.2)**	**54.9 (0.1)**	**53.3 (0.1)**	21.1 (6.6)	7.6 (8.3)	1.0 (0.7)
Fine-Tuning	2.6 (0.8)	75.7 (0.3)	56.5 (0.4)	54.5 (0.4)	52.9 (0.2)	23.3 (0.2)	8.1 (0.2)	1.0 (0.6)
TCM (ours)	**1.3 (0.1)**	**77.2 (0.4)**	56.6 (0.6)	54.4 (0.6)	52.7 (0.5)	**28.3 (0.3)**	**18.2 (0.1)**	**5.0 (0.4)**
Val = FMoW ID Val, Test = FMoW ID Test + FMoW OOD Test								
MSP	2.3 (0.6)	78.2 (0.5)	59.2 (0.3)	57.2 (0.2)	55.3 (0.4)	29.2 (0.2)	16.7 (0.3)	1.1 (1.6)
Binary Classifier	2.5 (0.8)	78.0 (0.7)	59.3 (0.1)	57.3 (0.0)	55.6 (0.0)	27.5 (2.5)	9.3 (10.7)	1.2 (1.5)
Fine-Tuning	2.2 (0.9)	78.6 (1.8)	**59.2 (1.6)**	**57.1 (1.5)**	**55.4 (1.4)**	30.1 (4.7)	16.2 (6.4)	2.7 (3.9)
TCM (ours)	**1.1 (0.0)**	**79.2 (0.4)**	58.9 (0.8)	56.7 (0.7)	55.0 (0.6)	**32.6 (0.1)**	**21.5 (0.4)**	**5.7 (1.3)**

Table 4: Selective classification performance on the Camelyon17 and FMoW datasets. Bold numbers represent best performance, and parentheses show the standard deviation over 3 random seeds. TCM consistently outperforms MSP and Binary Classifier on Camelyon17, and outperforms Fine-Tuning when the validation and test sets are from different distributions.

4.2 Selective Classification

Comparisons. We consider three prior methods for comparison: (1) MSP (Hendrycks & Gimpel, 2016) uses a model trained solely on training data and uses the maximum softmax probability as its confidence metric. (2) Binary Classifier (Kamath et al., 2020) trains a model on training data along with a separate binary classifier which aims to predict which inputs in the validation set the first model gets wrong. (3) Fine-Tuning, where we first train a model on training data and then fine-tune on the misclassified subset of validation data.

Results. We report our main results in Table 3 and Table 4. We see that that TCM consistently outperforms both MSP and Binary Classifier. The Fine-Tuning baseline outperforms TCM when the training and validation datasets are from the same distribution. We note that Fine-Tuning uses the ground-truth validation labels, which is strictly more information than TCM, which only observes whether or not each example was correct. In the settings where the training and validation distributions are different, TCM outperforms Fine-Tuning on most metrics. This indicates that TCM, given only labeled ID data, can learn a conservative classifier that is more effective than existing methods for selective classification in conditions of distribution shift.

REFERENCES

Peter Bandi, Oscar Geessink, Quirine Manson, Marcory Van Dijk, Maschenka Balkenhol, Meyke Hermsen, Babak Ehteshami Bejnordi, Byungjae Lee, Kyunghyun Paeng, Aoxiao Zhong, et al. From detection of individual metastases to classification of lymph node status at the patient level: the camelyon17 challenge. *IEEE transactions on medical imaging*, 38(2):550–560, 2018. [page 17]

Clément L. Canonne. A short note on an inequality between kl and tv, 2022. [page 10]

Alex Chan, Ahmed Alaa, Zhaozhi Qian, and Mihaela Van Der Schaar. Unlabelled data improves Bayesian uncertainty calibration under covariate shift. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1392–1402. PMLR, 13–18 Jul 2020. [page 9]

Charles Corbière, Nicolas Thome, Avner Bar-Hen, Matthieu Cord, and Patrick Pérez. Addressing failure prediction by learning model confidence. *Advances in Neural Information Processing Systems*, 32, 2019. [page 9]

Corinna Cortes, Giulia DeSalvo, and Mehryar Mohri. Boosting with abstention. *Advances in Neural Information Processing Systems*, 29, 2016. [page 8]

Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. Robustbench: a standardized adversarial robustness benchmark. *arXiv preprint arXiv:2010.09670*, 2020. [page 17, 18]

Jia Deng, R. Socher, Li Fei-Fei, Wei Dong, Kai Li, and Li-Jia Li. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 00, pp. 248–255, 06 2009. doi: 10.1109/CVPR.2009.5206848. [page 12]

Jean Feng, Arjun Sondhi, Jessica Perry, and Noah Simon. Selective prediction-set models with coverage guarantees. *arXiv preprint arXiv:1906.05473*, 2019. [page 9]

Adam Fisch, Tommi Jaakkola, and Regina Barzilay. Calibrated selective classification. *arXiv preprint arXiv:2208.12084*, 2022. [page 9]

Giorgio Fumera and Fabio Roli. Support vector machines with embedded reject option. In *Pattern Recognition with Support Vector Machines: First International Workshop, SVM 2002 Niagara Falls, Canada, August 10, 2002 Proceedings*, pp. 68–82. Springer, 2002. [page 8]

Aditya Gangrade, Anil Kag, and Venkatesh Saligrama. Selective classification via one-sided prediction. In *International Conference on Artificial Intelligence and Statistics*, pp. 2179–2187. PMLR, 2021. [page 9]

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. *Advances in neural information processing systems*, 30, 2017. [page 9]

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pp. 1321–1330. PMLR, 2017. [page 8]

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. [page 13]

Martin E. Hellman. The nearest neighbor classification rule with a reject option. *IEEE Transactions on Systems Science and Cybernetics*, 6(3):179–185, 1970. doi: 10.1109/TSSC.1970.300339. [page 8]

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019. [page 17]

Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016. [page 3, 4, 8, 11, 12, 17]

Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. *arXiv preprint arXiv:1812.04606*, 2018. [page 1, 3, 8, 12, 13, 18, 19]

Amita Kamath, Robin Jia, and Percy Liang. Selective question answering under domain shift. *arXiv preprint arXiv:2006.09462*, 2020. [page 4, 9]

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pp. 5637–5664. PMLR, 2021. [page 17, 18, 19]

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). *N/A*, 2009a. [page 12]

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-100 (canadian institute for advanced research). *N/A*, 2009b. [page 12]

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020. [page 9]

Ya Le and Xuan S. Yang. Tiny imagenet visual recognition challenge. 2015. [page 12]

Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks, 2018. [page 3, 8, 11]

Yoonho Lee, Huaxiu Yao, and Chelsea Finn. Diversify and disambiguate: Learning from underspecified data. *arXiv preprint arXiv:2202.03418*, 2022. [page 8]

Shiyu Liang, Yixuan Li, and R. Srikant. Principled detection of out-of-distribution examples in neural networks. *CoRR*, abs/1706.02690, 2017a. [page 3, 11, 12]

Shiyu Liang, Yixuan Li, and Rayadurgam Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv preprint arXiv:1706.02690*, 2017b. [page 8]

Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. Energy-based out-of-distribution detection. *Advances in Neural Information Processing Systems*, 33:21464–21475, 2020. [page 3, 8, 11, 12]

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011. [page 12]

Jeremy Nixon, Michael W Dusenberry, Linchuan Zhang, Ghassen Jerfel, and Dustin Tran. Measuring calibration in deep learning. In *CVPR workshops*, 2019. [page 8]

Mark S Pinsker. *Information and information stability of random variables and processes*. Holden-Day, 1964. [page 10]

Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019. [page 17, 18]

Shiori Sagawa, Pang Wei Koh, Tony Lee, Irena Gao, Sang Michael Xie, Kendrick Shen, Ananya Kumar, Weihua Hu, Michihiro Yasunaga, Henrik Marklund, et al. Extending the wilds benchmark for unsupervised adaptation. *arXiv preprint arXiv:2112.05090*, 2021. [page 1, 8]

Amrith Setlur, Benjamin Eysenbach, Virginia Smith, and Sergey Levine. Adversarial unlearning: Reducing confidence along adversarial directions. *arXiv preprint arXiv:2206.01367*, 2022. [page 9]

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. [page 1]

Fahim Tajwar, Ananya Kumar, Sang Michael Xie, and Percy Liang. No true state-of-the-art? ood detection methods are inconsistent across datasets. *arXiv preprint arXiv:2109.05554*, 2021. [page 1, 8, 12]

Alexandru Ţifrea, Eric Stavarache, and Fanny Yang. Novelty detection using ensembles with regularized disagreement. *arXiv preprint arXiv:2012.05825*, 2020. [page 8]

Alexandru Tifrea, Eric Stavarache, and Fanny Yang. Semi-supervised novelty detection using ensembles with regularized disagreement. In *Uncertainty in Artificial Intelligence*, pp. 1939–1948. PMLR, 2022. [page 1, 3, 13, 17]

Antonio Torralba, Rob Fergus, and William T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1958–1970, 2008. doi: 10.1109/TPAMI.2008.128. [page 12]

Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. Caltech-ucsd birds 200. Technical Report CNS-TR-201, Caltech, 2010. URL /se3/wp-content/uploads/2014/09/WelinderEtal10_CUB-200.pdf,http://www.vision.caltech.edu/visipedia/CUB-200.html. [page 17]

Pingmei Xu, Krista A Ehinger, Yinda Zhang, Adam Finkelstein, Sanjeev R. Kulkarni, and Jianxiong Xiao. Turkergaze: Crowdsourcing saliency with webcam based eye tracking, 2015. [page 12]

Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015. [page 12]

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks, 2016. [page 12]

C Zhang, S Bengio, M Hardt, B Recht, and O Vinyals. Understanding deep learning requires rethinking generalization. *5th International Conference on Learning Representations (ICLR)*, 2017. [page 1]

ABSTRACT

Errors of machine learning models can be prohibitively costly, especially in safety-critical settings such as healthcare. However, machine learning may be applicable to such scenarios if the learned model can abstain and defer to a human on difficult examples instead of making errors. In safety-critical settings, we prefer *conservative* models that defer to humans at the cost of some overall accuracy. Unfortunately, selective classification and out-of-distribution detection are notably difficult as it is hard to anticipate all possible examples. To mitigate this challenge, we focus on the transductive setting, where unlabeled examples from the test distribution are available during training. We propose Transductive Confidence Minimization (TCM), which minimizes prediction confidence on unlabeled test examples while simultaneously optimizing the training objective. We theoretically show that TCM learns a lower bound on the true confidence, and that this property can be leveraged to provably detect examples that are sufficiently different from training examples, regardless of what distribution they came from. In our experiments, TCM consistently shows high performance, achieving the highest OOD detection performance compared to 6 other methods on 9 out of 10 ID→OOD pairs and consistently outperforming methods for selective classification in settings where we test on data from a previously unseen distribution.

A PSEUDOCODE FOR TCM

Algorithm 1 TCM for OOD Detection

Input: Training data D_{tr}, Unlabeled data D_{u}, Hyperparamter λ

Initialize weights $\theta \leftarrow \theta_0$
while Not converged **do**
  Sample mini-batch $B_{\text{tr}} \sim D_{\text{tr}}$
  Update θ using $\nabla_\theta \mathcal{L}_{\text{xent}}(f, B_{\text{tr}})$
while Not converged **do**
  Sample mini-batches $B_{\text{tr}} \sim D_{\text{tr}}$, $B_{\text{u}} \sim D_{\text{u}}$
  Update: $\nabla_\theta \mathcal{L}_{\text{xent}}(f, B_{\text{tr}}) + \lambda \mathcal{L}_{\text{conf}}(f, B_{\text{u}})$

Algorithm 2 TCM for Selective Classification

Input: Training data D_{tr}, Validation data D_{val}, Hyperparameter λ

Initialize weights $\theta \leftarrow \theta_0$
while Not converged **do**
  Sample mini-batch $B_{\text{tr}} \sim D_{\text{tr}}$
  Update: $\nabla_\theta \mathcal{L}_{\text{xent}}(B_{\text{tr}}, f)$
Get correct set $D^{\circ}_{\text{val}} \leftarrow \{(x, y) \in D_{\text{val}} \mid f_\theta(x) = y\}$
Get error set $D^{\times}_{\text{val}} \leftarrow \{(x, y) \in D_{\text{val}} \mid f_\theta(x) \neq y\}$
while Not converged **do**
  Sample mini-batches $B_{\text{tr}} \sim D_{\text{tr}} \cup D^{\circ}_{\text{val}}$, $B^{\times}_{\text{val}} \sim D^{\times}_{\text{val}}$
  Update: $\nabla_\theta \mathcal{L}_{\text{xent}}(B_{\text{tr}}, f) + \lambda \mathcal{L}_{\text{conf}}(B^{\times}_{\text{val}}, f)$

B RELATED WORK

Out-of-distribution detection and unlabeled data. Many existing methods for OOD detection use a criterion based on the activations or predictions of a model trained on ID data (Hendrycks & Gimpel, 2016; Liang et al., 2017b; Lee et al., 2018; Liu et al., 2020). However, as noted in Tajwar et al. (2021), the performance of these methods are not consistent across different ID-OOD dataset pairs, which suggests that the OOD detection problem may be too challenging in the absence of additional information. Our method leverages an unlabeled dataset which contains a mix of ID and OOD data, similarly to Ţifrea et al. (2020). However, this method requires an ensemble of models to measure disagreement, while TCM only uses a single model. Similarly to our method, Hendrycks et al. (2018) minimizes confidence on an unlabeled set, but they do so on one big dataset regardless of the OOD data, and the support of this dataset is disjoint with the support of most OOD distributions. We additionally present theoretical results showing the benefit of minimizing confidence on an unlabeled set that includes inputs from the OOD distribution. Our experiments provide further support for this claim, showing that this transductive setting results in substantial performance gains, even if the unlabeled set is a mixture of ID and OOD data. More generally, unlabeled data has been shown to be beneficial for performance, especially in conditions of distribution shift (Sagawa et al., 2021; Lee et al., 2022).

Conservative prediction and selective classification. Prior works have studied selective classification, also known as reject option, for many model classes including SVM, boosting, and nearest neighbors (Hellman, 1970; Fumera & Roli, 2002; Cortes et al., 2016). Because deep neural networks generalize well but are often overconfident (Guo et al., 2017; Nixon et al., 2019), mitigating such overconfidence using selective classification while preserving its generalization properties is

a promising problem setting (Geifman & El-Yaniv, 2017; Corbière et al., 2019; Feng et al., 2019; Kamath et al., 2020; Fisch et al., 2022). Existing methods for learning conservative neural networks rely on additional assumptions such as pseudo-labeling (Chan et al., 2020), multiple distinct validation sets (Gangrade et al., 2021), or adversarial OOD examples (Setlur et al., 2022). Uniformly minimizing the confidence of a set that includes OOD inputs has been shown to result in a more conservative model in the offline reinforcement learning setting (Kumar et al., 2020), but this approach has not been validated in supervised learning settings. TCM only requires a small validation set, and our experiments in Section 4 demonstrate that its performance is competitive with state-of-the-art methods for selective classification, especially when tested on images from a distribution not seen during training.

C Discussion and Future Work

We presented an approach, transductive confidence minimization, which minimizes confidence on an uncertainty dataset and can be used for both out-of-distribution detection and selective classification. In the selective classification setting, TCM leverages an in-distribution validation dataset to identify misclassified examples for the uncertainty dataset, whereas for OOD detection, it assumes an unlabeled dataset that is representative of the target distribution. While we believe that access to unlabeled target data in the OOD detection setting is reasonable in some settings, it is not be feasible in all settings and TCM is only applicable to the settings where such data is available in the fine-tuning stage. Fortunately, this limitation does not apply in the selective classification setting, since ID validation data can be easily acquired by using a portion of the original training dataset. Overall, the theoretical guarantees and strong empirical performance of TCM represents a promising step towards building more robust and reliable machine learning systems.

D Theoretical Analysis

In this section we provide a simple theoretical setup for our algorithm. First, we show our algorithm achieves perfect OOD detection performance when the ID examples in the test set also appears in this training set. Next, we show that under the assumptions of function smoothness and closeness of ID train and test examples in the input space, this also holds for unseen ID and OOD examples.

D.1 Problem Setting

Let $\mathcal{X}$ be the input space and $\mathcal{Y}$ the label space. Let $\mathcal{P}_{\text{ID}}$ be a distribution over $\mathcal{X} \times \{1, \ldots, C\} \subseteq \mathcal{X} \times \mathcal{Y}$ i.e., there are C classes, and let D_{tr} be a training dataset consisting of n datapoints sampled from $\mathcal{P}_{\text{ID}}$. We train a classifier $f_\theta : \mathcal{X} \to [0,1]^C$ on the training data. We also consider a different distribution $\mathcal{P}_{\text{OOD}}$ over $\mathcal{X} \times \mathcal{Y}$ that is different from $\mathcal{P}_{\text{ID}}$ (the OOD distribution). Let D_{u} be an unlabeled test set where half the examples are sampled from $\mathcal{P}_{\text{ID}}$, the other half are sampled from $\mathcal{P}_{\text{OOD}}$. Our objective is to minimize the following loss function:

$$\mathcal{L}(\theta) = \mathop{\mathbb{E}}_{(x,y)\in D_{\text{tr}}} \left[\mathcal{L}_{\text{xent}}(f_\theta(x), y)\right] + \lambda \mathop{\mathbb{E}}_{x' \in D_{\text{u}}} \left[\mathcal{L}_{\text{con}}(f_\theta(x'))\right], \tag{2}$$

where $\lambda > 0$, $\mathcal{L}_{\text{xent}}$ is the standard cross-entropy loss, and $\mathcal{L}_{\text{con}}$ is a confidence loss which is calculated as the cross-entropy with respect to the uniform distribution over the C classes. We focus on the maximum softmax probability $\text{MSP}(p) \triangleq \max_i p_i$ as a measure of confidence in a given categorical distribution p.

D.2 Simplified Setting: ID Examples Shared Between Train and Unlabeled Sets

We start with the following lemma which characterizes the interaction of our loss function (1) with a single datapoint.

Proposition D.1 (Lower bound on true confidence)**.** *Let p be the true label distribution of input x. The minimum of the objective function (1) is achieved when the predicted distribution is $p_\lambda \triangleq \frac{p + \lambda \frac{\mathbf{1}}{C}}{1+\lambda}$. This optimal distribution p_λ satisfies MSP$(p_\lambda) \leq$ MSP(p), with equality iff $\lambda = 0$.*

Proof. Denote the predicted logits for input x as $z \in \mathbb{R}^C$, and softmax probabilities as $s = e^z / \sum_i e^{z_i} \in [0,1]^C$. The derivative of the logits with respect to the two loss terms have the closed-form expressions $\frac{\partial}{\partial z}\mathcal{L}_{\text{xent}} = s - p$, $\frac{\partial}{\partial z}\mathcal{L}_{\text{con}} = s - \frac{1}{C}\mathbf{1}$. Setting the derivative of the overall objective to zero, we have

$$\frac{\partial}{\partial z}\left(\mathcal{L}_{\text{xent}} + \lambda \mathcal{L}_{\text{con}}\right) = s - p + \lambda\left(s - \frac{\mathbf{1}}{C}\right) = 0 \implies s = \frac{p + \lambda \frac{\mathbf{1}}{C}}{1+\lambda} = p_\lambda. \tag{3}$$

To check the lower bound property, note that p_λ is a combination of p and the uniform distribution U, where U is the uniform distribution over the C classes and has the lowest possible MSP among all categorical distributions over C classes. □

The resulting predictive distribution p_λ can alternatively be seen as Laplace smoothing with pseudo-count λ applied to the true label distribution p. This new distribution can be seen as "conservative" in that it (1) has lower MSP than that of p, and (2) has an entropy greater than that of p.

Lemma D.2 (Pinsker's inequality)**.** *If P and Q are two probability distributions, then*

$$\delta_{TV}(P, Q) \leq \sqrt{\frac{1}{2} D_{KL}(P \parallel Q)}, \tag{4}$$

where $\delta_{TV}(P, Q)$ is the total variation distance between P and Q.

Proof. Refer to (Pinsker, 1964; Canonne, 2022) for a detailed proof. □

Lemma D.3 (Low loss implies separation, transductive case)**.** *Assume that all ID examples in D_u are also in D_{tr}, and that $\mathcal{D}_{in} \cap \mathcal{D}_{out} = \varnothing$. Let $D_{in}^{test} = \{x \in D^{test} : x \sim \mathcal{D}_{in}\} (= D^{train})$ and $D_{out}^{test} = \{x \in D^{test} : x \sim \mathcal{D}_{out}\} = D^{test} \backslash D^{train}$. Let $\mathcal{L}_0$ be the lowest achievable loss for the objective (1) with $\lambda > 0$. Then there exists $\epsilon > 0$ such that $\mathcal{L}(\theta) - \mathcal{L}_0 < \epsilon$ implies the following relationship between the max probabilities holds:*

$$\min_{x \in D_{in}^{test}} \mathit{MSP}(f_\theta^i(x)) > \max_{x \in D_{out}^{test}} \mathit{MSP}(f_\theta^i(x)) \tag{5}$$

Proof. Since the training set is a subset of the unlabeled set, we can rearrange the objective (1) as

$$\mathcal{L}(\theta) = \mathop{\mathbb{E}}_{(x,y) \in D_{in}^{test}} \left[\mathcal{L}_{\text{xent}}(f_\theta(x), y) + \lambda \mathcal{L}_{\text{con}}(f_\theta(x))\right] + \mathop{\mathbb{E}}_{x \in D_{out}^{test}} \left[\lambda \mathcal{L}_{\text{con}}(f_\theta(x'))\right]. \tag{6}$$

Note that the first term is the cross-entropy between $f_\theta(x)$ and $p_\lambda \triangleq \frac{p + \lambda \frac{1}{C}}{1+\lambda}$, and the second term is the cross-entropy between $f_\theta(x)$ and the uniform distribution U. We now rearrange to see that

$$\mathcal{L}(\theta) - \mathcal{L}_0 = \mathop{\mathbb{E}}_{(x,y) \in D_{in}^{test}} \left[D_{KL}(p_\lambda \parallel f_\theta(x))\right] + \mathop{\mathbb{E}}_{x \in D_{out}^{test}} \left[D_{KL}(U \parallel f_\theta(x))\right], \tag{7}$$

where the lowest achievable loss $\mathcal{L}_0$ is obtained by setting $f_\theta(x) = p_\lambda$ for ID inputs and $f_\theta(x) = U$ for OOD inputs. Because $\mathcal{L} - \mathcal{L}_0 < \epsilon$, we know that $D_{KL}(p_\lambda \parallel f_\theta(x)) < N\epsilon$ for all ID inputs and $D_{KL}(U \parallel f_\theta(x)) < N\epsilon$ for all OOD inputs.

By Lemma D.2, we have for ID input x

$$\delta_{\text{TV}}(p_\lambda, f_\theta(x)) \leq \sqrt{\frac{1}{2} D_{KL}(p_\lambda \parallel f_\theta(x))} = \sqrt{\frac{N\epsilon}{2}}. \tag{8}$$

By the triangle inequality and because MSP is 1-Lipschitz with respect to output probabilities, we have for all ID inputs

$$\text{MSP}(f_\theta(x)) \geq \text{MSP}(p_\lambda) - \sqrt{\frac{N\epsilon}{2}} = \frac{1}{1+\lambda} + \frac{\lambda}{1+\lambda}\frac{1}{M} - \sqrt{\frac{N\epsilon}{2}}. \tag{9}$$

Similarly, by Lemma D.2, we have for OOD input x

$$\delta_{\text{TV}}(U, f_\theta(x)) \leq \sqrt{\frac{1}{2} D_{KL}(U \parallel f_\theta(x))} = \sqrt{\frac{N\epsilon}{2}}. \tag{10}$$

By the triangle inequality and because MSP is 1-Lipschitz with respect to output probabilities, we have for all OOD inputs

$$\text{MSP}(f_\theta(x)) \leq \text{MSP}(U) + \sqrt{\frac{N\epsilon}{2}} = \frac{1}{M} + \sqrt{\frac{N\epsilon}{2}}. \tag{11}$$

Letting $\epsilon < \frac{1}{2N}\left(\frac{M-1}{(1+\lambda)M}\right)^2$, we have

$$\min_{x \in D_{in}^{test}} \text{MSP}(f_\theta^i(x)) \geq \frac{1}{1+\lambda} + \frac{\lambda}{1+\lambda}\frac{1}{M} - \sqrt{\frac{N\epsilon}{2}} > \frac{1}{M} + \sqrt{\frac{N\epsilon}{2}} \geq \max_{x \in D_{out}^{test}} \text{MSP}(f_\theta^i(x)). \tag{12}$$

□

Lemma D.3 shows that in the transductive setting, minimizing our objective $L(\theta)$ (1) below some threshold provably leads to a separation between ID and OOD examples in terms of the maximum predicted probability for each example.

D.3 More general setting

We prove a more general version of the claim in Lemma D.3 which applies to datapoints outside of the given dataset D^{test}. Our theorem below depends only on a mild smoothness assumption on the learned function.

Proposition D.4 (Low loss implies separation)**.** *Assume that all ID examples in D_u are also in D_{tr}, and that $\mathcal{D}_{in} \cap \mathcal{D}_{out} = \varnothing$. Let $D_{in}^{test} = \{x \in D^{test} : x \sim \mathcal{D}_{in}\} (= D^{train})$ and $D_{out}^{test} = \{x \in D^{test} : x \sim \mathcal{D}_{out}\} = D^{test} \backslash D^{train}$. Assume that the classifier $f_\theta : \mathcal{X} \to [0,1]^C$ is K-Lipschitz continuous for all θ, i.e., for all $x, x' \in \mathcal{X}$, $||f_\theta(x) - f_\theta(x')||_\infty \leq K d(x, x')$ for some constant $K > 0$. Let $\mathcal{L}_0$ be the lowest achievable loss for the objective (1) with $\lambda > 0$. For $\delta > 0$, denote the union of δ-balls around the ID and OOD datapoints as*

$$D_{in}^{\delta} \triangleq \{x | \exists x' \in D_{in}^{test} \; s.t. \; d(x, x') < \delta\}, \quad D_{out}^{\delta} \triangleq \{x | \exists x' \in D_{out}^{test} \; s.t. \; d(x, x') < \delta\}. \tag{13}$$

Then there exists $\epsilon, \delta > 0$ such that $\mathcal{L}(\theta) - \mathcal{L}_0 < \epsilon$ implies the following relationship between the max probabilities holds:

$$\inf_{x \in D_{in}^{\delta}} MSP(f_\theta^i(x)) > \sup_{x \in D_{out}^{\delta}} MSP(f_\theta^i(x)) \tag{14}$$

Proof. By Lemma D.3, we have for some ϵ, $\min_{x \in D_{in}^{test}} \text{MSP}(f_\theta^i(x)) > \max_{x \in D_{out}^{test}} \text{MSP}(f_\theta^i(x))$. Fix ϵ and denote the difference of these two terms as

$$\min_{x \in D_{in}^{test}} \text{MSP}(f_\theta^i(x)) - \max_{x \in D_{out}^{test}} \text{MSP}(f_\theta^i(x)) = \Delta. \tag{15}$$

For any $x_{\text{in}}^{\delta} \in D_{in}^{\delta}$ and $x_{\text{out}}^{\delta} \in D_{out}^{\delta}$, let $x_{\text{in}} \in D_{in}^{test}, x_{\text{out}} \in D_{out}^{test}$ satisfy $d(x_{\text{in}}^{\delta}, x_{\text{in}}) < \delta$ and $d(x_{\text{out}}^{\delta}, x_{\text{out}}) < \delta$. By the K-Lipschitz property, we have

$$\text{MSP}(f_\theta^i(x_{\text{in}}^{\delta})) \geq \text{MSP}(f_\theta^i(x_{\text{in}})) - K\delta, \quad \text{MSP}(f_\theta^i(x_{\text{out}}^{\delta})) \leq \text{MSP}(f_\theta^i(x_{\text{out}})) + K\delta. \tag{16}$$

Setting $\delta < \frac{\Delta}{2K}$, we have

$$\text{MSP}(f_\theta^i(x_{\text{in}}^{\delta})) \geq \text{MSP}(f_\theta^i(x_{\text{in}})) - K\delta > \text{MSP}(f_\theta^i(x_{\text{out}})) + K\delta \geq \text{MSP}(f_\theta^i(x_{\text{out}}^{\delta})). \tag{17}$$

Since the choice of x_{in}^{δ} and x_{out}^{δ} was arbitrary, the equation above holds for all datapoints inside each δ-ball. Therefore, we have

$$\inf_{x \in D_{in}^{\delta}} \text{MSP}(f_\theta^i(x)) > \sup_{x \in D_{out}^{\delta}} \text{MSP}(f_\theta^i(x)). \tag{18}$$

□

E Detailed OOD Detection Results in the Regular Setting

E.1 Comparisons

We compare our algorithm's performance against several popular OOD detection methods:

- **MSP** (Hendrycks & Gimpel, 2016): A simple baseline for OOD detection, where we take a network trained on ID samples and threshold on the network's maximum softmax probability prediction on a test example to separate ID and OOD examples.
- **ODIN** (Liang et al., 2017a): This method uses temperature scaling and adding small noise perturbations to the inputs to increase the separation of softmax probability between ID and OOD examples.
- **Mahalanobis** (Lee et al., 2018): This method takes a pretrained softmax classifier and uses the mahalanobis distance in the embedding space to separate ID examples from OOD examples.
- **Energy score** Liu et al. (2020): Instead of the softmax probability, this method uses energy scores to separate ID and OOD examples.

- **Outlier exposure** Hendrycks et al. (2018): Since traditional neural networks can produce high probabilities on anomalous examples, this method leverages examples from a pseudo-OOD distribution, i.e., a distribution different from the in-distribution but maybe not the same OOD distribution one would see during test-time, and fine-tunes a pre-trained network to minimize confidence on this pseudo-OOD examples.
- **Energy based fine-tuning** Liu et al. (2020): Similar to outlier exposure, but minimizes energy-based confidence score instead of softmax-based confidence score on the psuedo-OOD examples.

E.2 ID Datasets

We use the following ID datasets from common benchmarks:

- **CIFAR-10** (Krizhevsky et al., 2009a): CIFAR-10 contains 50,000 train and 10,000 test images, separated into 10 disjoint classes. The images have 3 channels and are of size 32 x 32. The classes are similar but disjoint from CIFAR-100.
- **CIFAR-100** (Krizhevsky et al., 2009b): Similar to CIFAR-10 and contains 50,000 train and 10,000 test images. However, the images are now separated into 100 fine-grained and 20 coarse (super) classes. Each super-class contains 5 fine-grained classes.

E.3 OOD Datasets

In addition to CIFAR-10 and CIFAR-100, we follow prior work (Tajwar et al., 2021; Hendrycks & Gimpel, 2016; Liu et al., 2020) and use the following benchmark OOD detection dataset:

- **SVHN** (Netzer et al., 2011): SVHN contains images of the 10 digits in English which represent the 10 classes in the dataset. The dataset contains 73,257 train and 26,032 test images. The original dataset also contains extra training images that we do not use for our experiments. Each image in the dataset has 3 channels and has shape 32 x 32.
- **TinyImageNet (resized)** (Le & Yang, 2015; Deng et al., 2009; Liang et al., 2017a): TinyImageNet contains 10,000 test images divided into 200 classes and is a subset of the larger ImageNet (Deng et al., 2009) dataset. The original dataset contains images of shape 64 x 64 and Liang et al. (2017a) further creates a dataset by randomly cropping and resizing the images to shape 32 x 32. We use the resized dataset here for our experiments.
- **LSUN (resized)** (Yu et al., 2015; Liang et al., 2017a): The **L**arge-scale **S**cene **UN**derstanding dataset (LSUN) contains 10,000 test images divided into 10 classes. Similar to the TinyImageNet dataset above, Liang et al. (2017a) creates a dataset by randomly cropping and resizing the images to shape 32 x 32. We use the resized dataset here for our experiments.
- **iSUN** (Xu et al., 2015; Liang et al., 2017a): iSUN contains 6,000 training, 926 validation and 2,000 test images. We use the same dataset used by Liang et al. (2017a).

Instructions on how to download the TinyImageNet, LSUN and iSUN datasets can be found here: https://github.com/ShiyuLiang/odin-pytorch

E.4 Architecture and training details

- **Architecture**: For all experiments in this section, we use a WideResNet-40-2 (Zagoruyko & Komodakis, 2016) network.
- **Hyper-parameters**: Outlier exposure and energy based fine-tuning uses 80 million tiny images (Torralba et al., 2008) as the pseudo-OOD dataset which has been withdrawn due to containing derogatory terms as categories. Since it is no longer available, for fair comparison, we just use the pre-trained weights provided by these papers' authors for our experiments. For MSP, ODIN, Mahalanobis and energy score, we train our networks for 110 epochs with an initial learning rate of 0.1, weight decay of 5×10^{-4}, dropout 0.3 and batch size 128. ODIN and Mahalanobis require a small OOD validation set to tune hyper-parameters. Instead, we tune the hyper-parameters over the entire test set and report the best numbers, since we only want an upper bound on the performance of these methods. For ODIN, we try $T \in \{1, 10, 100, 1000\}$ and $\epsilon \in \{0.0, 0.0005, 0.001, 0.0014, 0.002, 0.0024, 0.005, 0.01, 0.05, 0.1, 0.2\}$ as our hyper-parameter search grid, and for Mahalanobis, we use the same hyper-parameter grid for

ϵ. For our method, we pre-train our network for 100 epochs with the same setup, and fine-tune the network with our modified loss objective for 10 epochs using the same setting, except we use a initial learning rate of 0.001, batch size 32 for ID train set and 64 for the unlabeled set. During fine-tuning, we use 27,000 images per epoch, 9,000 of which are labeled ID train examples and the rest are from the unlabeled set. Finally, we use $\lambda = 0.5$ for all experiments, similar to Hendrycks et al. (2018), without any additional hyper-parameter tuning.

- **Dataset train/val split**: For all methods except outlier exposure and energy based fine-tuning, we use 40,000 out of the 50,000 train examples for training and 10,000 train examples for validation. Note that outlier exposure and energy based fine-tuning uses weights pre-trained with all 50,000 training examples, which puts our method in disadvantage.
- **Unlabeled and test set construction**: For our method, we use two disjoint sets of 6,000 images as the unlabeled set and test set. Each set contains 5,000 ID examples and 1,000 OOD examples.
- **Augmentations**: For all methods, we use the same standard random flip and random crop augmentations during training/fine-tuning.

E.5 Detailed results

Table 5 contains results on various OOD datasets when CIFAR-100 is used as the ID dataset, and Table 6 corresponds to the similar table for CIFAR-10.

F Semi-supervised novelty detection setting

For the sake of fair comparison, we also compare our algorithm's performance to binary classifier and ERD (Tifrea et al., 2022). These methods leverage an unlabeled dataset that contains both ID and OOD examples drawn from the distribution that we will see during test-time.

- **ERD**: Generates an ensemble by fine-tuning an ID pre-trained network on a combined ID + unlabeled set (which is a mixture of ID and OOD examples and given one label for all examples). Uses an ID validation set to early stop, and then uses the disagreement score between the networks on the ensemble to separate ID and OOD examples.
- **Binary classifier**: The approach learns to discriminate between labeled ID set and unlabeled ID-OOD mixture set, with regularizations to prevent the entire unlabeled set to be classified as OOD.

We use the same datasets as Appendix E.

F.1 Architecture and training details

- **Architecture**: For all experiments in this section, we use a ResNet-18 (He et al., 2015) network, same as Tifrea et al. (2022).
- **Hyper-parameters**: For ERD and binary classifier, we use the hyper-parameters and learning rate schedule used by Tifrea et al. (2022). For ERD, we standardize the experiments by using ensemble size $= 3$ for all experiments. The ensemble models are initialized with weights pre-trained solely on the ID training set for 100 epochs, and then each is further trained for 10 epochs. For binary classifier, we train all the networks from scratch for 100 epochs with a learning rate schedule described by Tifrea et al. (2022). For our method, we use the same hyper-parameters as Appendix E.

We use the same dataset splits, augmentations, unlabeled and test sets as Appendix E.

F.2 Detailed results

Table 8 contains OOD detection performance metrics when for various commonly used OOD datasets when CIFAR-100 is used as the ID set, and Table 7 contains the same when CIFAR-10 is used as the ID set.

OOD dataset		FPR95 ↓	FPR99 ↓	AUROC ↑	AUPR-In ↑	AUPR-Out ↑
CIFAR-10	MSP	64.4 (1.4)	80.5 (0.7)	74.6 (0.7)	93.9 (0.2)	32.8 (1.7)
	ODIN	67.6 (2.8)	85.8 (2.2)	75.8 (0.8)	93.9 (0.3)	34.7 (1.3)
	Mahalanobis	86.7 (1.6)	96.3 (0.8)	62.9 (1.1)	89.2 (0.6)	21.8 (0.6)
	Energy score	67.2 (3.2)	86.6 (1.6)	75.7 (0.9)	93.8 (0.3)	34.4 (1.0)
	Outlier exposure	63.5	77.9	75.2	94.0	32.7
	Energy fine-tuning	**57.8**	**74.6**	77.3	94.7	34.3
	TCM-softmax (ours)	**58.0 (1.7)**	79.3 (2.4)	**80.8 (1.2)**	**95.3 (0.3)**	**44.3 (2.1)**
	TCM-energy (ours)	**60.3 (2.8)**	80.4 (1.6)	**81.0 (1.5)**	**95.3 (0.4)**	**47.6 (2.5)**
SVHN	MSP	58.0 (4.9)	73.6 (4.2)	77.7 (1.4)	94.8 (0.4)	37.1 (2.4)
	ODIN	42.7 (8.4)	64.2 (5.9)	85.7 (6.5)	96.8 (1.5)	52.9 (15.1)
	Mahalanobis	36.4 (4.3)	58.0 (5.8)	91.5 (2.1)	98.1 (0.5)	70.5 (8.1)
	Energy score	51.3 (5.8)	70.4 (6.8)	81.7 (2.4)	95.9 (0.7)	38.9 (4.7)
	Outlier exposure	40.4	53.1	88.2	97.5	54.4
	Energy fine-tuning	12.6	27.6	96.8	99.4	70.7
	TCM-softmax (ours)	**0.6 (0.7)**	**12.6 (4.0)**	**99.6 (0.1)**	**99.9 (0.0)**	**98.8 (0.3)**
	TCM-energy (ours)	**0.3 (0.3)**	**8.0 (1.7)**	**99.7 (0.1)**	**99.9 (0.0)**	**99.1 (0.2)**
TinyImageNet	MSP	77.0 (5.7)	90.9 (3.9)	68.0 (3.2)	91.4 (1.1)	27.4 (2.2)
	ODIN	61.2 (11.9)	80.4 (7.7)	81.9 (3.7)	95.4 (1.2)	49.8 (5.4)
	Mahalanobis	45.7 (9.2)	65.3 (8.7)	88.3 (4.0)	97.2 (0.9)	65.0 (12.2)
	Energy score	73.5 (10.5)	88.5 (6.5)	73.1 (4.3)	92.7 (1.7)	34.3 (4.6)
	Outlier exposure	71.6	86.9	75.7	93.7	38.5
	Energy fine-tuning	85.2	100.0	70.9	91.4	35.6
	TCM-softmax (ours)	**5.9 (2.9)**	**35.2 (6.0)**	**98.7 (0.3)**	**99.7 (0.1)**	**96.9 (0.8)**
	TCM-energy (ours)	**3.5 (2.5)**	**30.8 (7.5)**	**99.0 (0.3)**	**99.7 (0.1)**	**97.6 (0.8)**
LSUN	MSP	75.6 (3.7)	89.1 (5.4)	68.5 (1.5)	91.7 (0.6)	27.7 (1.4)
	ODIN	57.0 (11.5)	74.8 (8.2)	83.3 (3.9)	95.9 (1.2)	51.2 (5.7)
	Mahalanobis	41.0 (7.7)	62.6 (9.3)	89.2 (3.7)	97.5 (0.9)	65.0 (11.6)
	Energy score	70.0 (8.7)	85.8 (8.6)	75.2 (4.9)	93.5 (1.8)	35.9 (5.1)
	Outlier exposure	59.1	79.9	81.4	95.4	49.4
	Energy fine-tuning	65.6	86.8	80.9	95.0	47.9
	TCM-softmax (ours)	**1.1 (1.0)**	**14.8 (6.7)**	**99.5 (0.2)**	**99.9 (0.0)**	**98.6 (0.5)**
	TCM-energy (ours)	**0.5 (0.6)**	**7.0 (3.4)**	**99.7 (0.1)**	**99.9 (0.0)**	**99.1 (0.3)**
iSUN	MSP	77.6 (3.7)	91.7 (1.4)	67.1 (2.4)	91.2 (0.8)	26.4 (2.1)
	ODIN	59.8 (9.1)	78.9 (5.9)	82.3 (3.0)	95.6 (1.0)	48.8 (4.2)
	Mahalanobis	48.7 (5.7)	68.2 (5.4)	87.3 (2.8)	97.0 (0.7)	61.7 (8.9)
	Energy score	72.0 (8.2)	88.1 (4.7)	73.8 (3.8)	93.2 (1.4)	33.3 (4.0)
	Outlier exposure	66.4	83.6	79.2	94.7	45.3
	Energy fine-tuning	75.1	89.0	77.4	93.9	43.9
	TCM-softmax (ours)	**2.74 (1.9)**	**25.1 (7.0)**	**99.1 (0.2)**	**99.8 (0.0)**	**97.8 (0.5)**
	TCM-energy (ours)	**0.9 (0.6)**	**20.7 (7.1)**	**99.4 (0.2)**	**99.8 (0.0)**	**98.6 (0.3)**

Table 5: OOD detection performance on common out-of-distribution datasets when the ID dataset is CIFAR-100 (architecture: WideResNet-40-2). Bold numbers represent superior results. Numbers in parenthesis represent the standard deviation over 5 seeds. ↓: lower is better, ↑: higher is better.

OOD dataset		FPR95 ↓	FPR99 ↓	AUROC ↑	AUPR-In ↑	AUPR-Out ↑
CIFAR-100	MSP	45.7 (2.5)	81.0 (5.6)	86.8 (0.3)	96.8 (0.1)	53.4 (1.0)
	ODIN	59.6 (2.6)	89.0 (1.4)	86.1 (0.4)	96.2 (0.2)	58.9 (0.6)
	Mahalanobis	65.7 (1.9)	85.2 (2.5)	80.3 (0.6)	94.9 (0.2)	46.0 (0.7)
	Energy score	59.6 (2.6)	89.0 (1.4)	86.2 (0.4)	96.2 (0.2)	59.2 (0.5)
	Outlier exposure	**28.3**	**57.9**	93.1	98.5	76.5
	Energy fine-tuning	29.0	63.4	**94.0**	**98.6**	**81.6**
	TCM-softmax (ours)	57.5 (6.1)	90.0 (2.8)	87.6 (0.7)	96.5 (0.4)	63.1 (0.7)
	TCM-energy (ours)	60.4 (5.3)	90.5 (2.6)	87.0 (0.9)	96.3 (0.4)	64.3 (1.2)
SVHN	MSP	43.4 (23.3)	71.4 (21.2)	87.2 (5.6)	96.7 (2.2)	54.0 (8.6)
	ODIN	53.0 (13.9)	73.8 (5.8)	78.8 (10.0)	95.0 (2.7)	39.2 (17.1)
	Mahalanobis	16.5 (4.5)	37.9 (5.7)	97.1 (0.8)	99.3 (0.2)	90.1 (2.4)
	Energy score	59.7 (22.7)	86.8 (12.3)	82.8 (10.5)	94.5 (5.1)	50.4 (12.4)
	Outlier exposure	4.8	15.6	98.5	99.7	90.3
	Energy fine-tuning	2.1	13.3	99.3	99.8	96.2
	TCM-softmax (ours)	**0.4 (0.3)**	**8.4 (2.7)**	**99.7 (0.1)**	**99.9 (0.0)**	**99.1 (0.2)**
	TCM-energy (ours)	**0.1 (0.1)**	**5.1 (2.7)**	**99.8 (0.1)**	**100.0 (0.1)**	**99.4 (0.2)**
TinyImageNet	MSP	32.8 (6.0)	57.1 (8.6)	90.3 (1.4)	97.9 (0.4)	60.7 (3.0)
	ODIN	34.4 (10.9)	57.1 (11.8)	92.8 (2.4)	98.3 (0.6)	76.3 (7.1)
	Mahalanobis	35.9 (6.4)	59.6 (4.8)	91.6 (2.7)	98.1 (0.6)	71.5 (10.0)
	Energy score	34.0 (10.9)	61.1 (13.8)	92.0 (2.8)	98.1 (0.8)	72.0 (7.4)
	Outlier exposure	13.0	25.3	97.4	99.5	88.7
	Energy fine-tuning	7.0	18.8	98.2	99.6	92.1
	TCM-softmax (ours)	**2.6 (1.6)**	**15.1 (4.0)**	**99.3 (0.3)**	**99.8 (0.1)**	**98.1 (0.7)**
	TCM-energy (ours)	**1.0 (0.8)**	**14.7 (3.2)**	**99.4 (0.2)**	**99.8 (0.1)**	**98.7 (0.5)**
LSUN	MSP	21.3 (2.6)	39.8 (5.9)	93.3 (0.9)	98.6 (0.2)	69.7 (3.1)
	ODIN	15.2 (5.5)	35.4 (7.4)	96.8 (1.0)	99.3 (0.3)	88.1 (3.2)
	Mahalanobis	28.2 (5.5)	48.8 (7.7)	93.0 (1.9)	98.5 (0.4)	72.1 (7.5)
	Energy score	16.0 (5.0)	39.2 (10.2)	96.2 (1.2)	99.2 (0.3)	84.8 (3.9)
	Outlier exposure	3.7	11.5	99.1	99.8	95.7
	Energy fine-tuning	1.9	**4.2**	99.3	**99.9**	97.4
	TCM-softmax (ours)	**0.5 (0.4)**	**5 (2.8)**	**99.8 (0.1)**	**99.9 (0.1)**	**99.3 (0.4)**
	TCM-energy (ours)	**0.1 (0.1)**	**2.1 (1.8)**	**99.9 (0.1)**	**100.0 (0.1)**	**99.7 (0.2)**
iSUN	MSP	25.9 (4.1)	44.8 (8.4)	92.0 (1.3)	98.3 (0.3)	66.0 (3.3)
	ODIN	21.1 (7.7)	42.5 (11.8)	95.6 (1.6)	99.0 (0.4)	83.5 (5.0)
	Mahalanobis	28.7 (4.8)	50.9 (7.3)	92.6 (1.9)	98.4 (0.4)	71.5 (7.4)
	Energy score	23.2 (8.6)	44.7 (13.5)	94.9 (1.9)	98.9 (0.5)	80.6 (5.6)
	Outlier exposure	5.0	11.3	99.1	99.8	95.2
	Energy fine-tuning	2.6	6.8	99.4	**99.9**	96.9
	TCM-softmax (ours)	0.6 (0.2)	6.2 (0.9)	**99.7 (0.1)**	**99.9 (0.0)**	99.1 (0.1)
	TCM-energy (ours)	**0.1 (0.1)**	**4.2 (0.8)**	**99.8 (0.1)**	**100.0 (0.1)**	**99.5 (0.1)**

Table 6: OOD detection performance on common out-of-distribution datasets when the ID dataset is CIFAR-10 (architecture: WideResNet-40-2). Bold numbers represent superior results. Numbers in parenthesis represent the standard deviation over 5 seeds. ↓: lower is better, ↑: higher is better.

OOD dataset		FPR95 ↓	FPR99 ↓	AUROC ↑	AUPR ↑
SVHN	ERD	2.33 (0.88)	31.18 (3.18)	99.01 (0.14)	97.78 (0.26)
	Binary Classifier	25.76 (40.78)	56.41 (50.93)	95.10 (6.81)	93.26 (6.83)
	TCM-softmax (ours)	2.48 (1.35)	21.10 (4.11)	99.28 (0.13)	98.04 (0.36)
	TCM-energy (ours)	1.02 (0.64)	19.02 (3.77)	99.46 (0.13)	98.56 (0.27)
LSUN	ERD	0.82 (0.38)	12.21 (3.45)	99.50 (0.13)	98.76 (0.28)
	Binary Classifier	0.04 (0.05)	24.18 (28.89)	99.19 (0.40)	98.87 (0.61)
	TCM-softmax (ours)	1.58 (1.46)	16.16 (5.85)	99.44 (0.24)	98.40 (0.72)
	TCM-energy (ours)	0.78 (0.71)	10.96 (5.34)	99.64 (0.17)	98.88 (0.50)
TinyImageNet	ERD	5.42 (1.85)	32.37 (9.44)	98.75 (0.25)	96.82 (0.70)
	Binary Classifier	0.72 (0.57)	34.58 (32.64)	98.98 (0.67)	98.35 (0.83)
	TCM-softmax (ours)	7.84 (2.47)	29.96 (3.64)	98.72 (0.26)	96.40 (0.80)
	TCM-energy (ours)	4.90 (2.33)	27.78 (6.29)	98.96 (0.25)	97.22 (0.70)
iSUN	ERD	1.66 (0.88)	21.77 (6.56)	99.22 (0.14)	98.17 (0.36)
	Binary Classifier	3.98 (5.33)	59.98 (10.66)	98.26 (0.51)	97.24 (0.95)
	TCM-softmax (ours)	6.26 (3.28)	26.84 (9.31)	98.82 (0.44)	96.84 (1.12)
	TCM-energy (ours)	2.90 (1.65)	21.76 (6.98)	99.16 (0.28)	97.86 (0.67)

Table 7: OOD detection performance (Semi-supervised novelty detection setting) on common out-of-distribution datasets when the ID dataset is CIFAR-100 (architecture: ResNet18). Numbers in parenthesis represent the standard deviation over 5 seeds. ↓: lower is better, ↑: higher is better.

OOD dataset		FPR95 ↓	FPR99 ↓	AUROC ↑	AUPR ↑
SVHN	ERD	1.66 (1.24)	20.64 (6.70)	99.28 (0.20)	98.38 (0.45)
	Binary Classifier	1.32 (0.97)	33.35 (5.76)	98.93 (0.23)	96.99 (59.69)
	TCM-softmax (ours)	1.04 (0.55)	13.90 (5.77)	99.48 (0.16)	98.62 (0.36)
	TCM-energy (ours)	0.60 (0.47)	14.09 (5.97)	99.50 (0.20)	98.80 (0.34)
LSUN	ERD	0.18 (0.18)	8.93 (5.44)	99.71 (0.12)	99.23 (0.26)
	Binary Classifier	0.31 (0.58)	37.29 (28.79)	99.01 (0.34)	98.53 (0.24)
	TCM-softmax (ours)	0.90 (0.34)	7.50 (2.94)	99.72 (0.08)	99.02 (0.30)
	TCM-energy (ours)	0.46 (0.21)	5.90 (2.86)	99.80 (0.07)	99.32 (0.24)
TinyImageNet	ERD	1.65 (0.59)	17.71 (2.70)	99.34 (0.09)	98.32 (0.23)
	Binary Classifier	1.83 (3.80)	49.83 (25.68)	98.68 (0.63)	97.91 (1.17)
	TCM-softmax (ours)	4.14 (1.25)	23.26 (12.27)	99.10 (0.32)	97.52 (0.58)
	TCM-energy (ours)	3.20 (0.96)	28.44 (13.94)	99.10 (0.32)	97.76 (0.50)
iSUN	ERD	0.48 (0.35)	9.03 (4.40)	99.65 (0.17)	99.06 (0.34)
	Binary Classifier	1.63 (2.50)	39.78 (22.81)	98.75 (0.83)	97.96 (1.05)
	TCM-softmax (ours)	1.72 (0.36)	12.44 (2.56)	99.48 (0.08)	98.44 (0.22)
	TCM-energy (ours)	1.20 (0.29)	13.44 (2.93)	99.48 (0.08)	98.60 (0.20)

Table 8: OOD detection performance (Semi-supervised novelty detection setting) on common out-of-distribution datasets when the ID dataset is CIFAR-10 (architecture: ResNet18). Numbers in parenthesis represent the standard deviation over 5 seeds. ↓: lower is better, ↑: higher is better.

G NEAR-OOD DETECTION SETTING

G.1 ARCHITECTURE AND TRAINING DETAILS

- **Datasets**: Similar to Tifrea et al. (2022), we try two settings: (1) ID = first 5 classes of CIFAR-10, OOD = last 5 classes of CIFAR-100, (2) ID = first 50 classes of CIFAR-100, OOD = last 50 classes of CIFAR-100.
- **Dataset splits**: We use 20,000 train and 5,000 validation label-balanced images during training.
- **Unlabeled and test split construction**: We use two disjoint datasets of size 3,000 as unlabeled and test sets. Each dataset contains 2,500 ID and 500 OOD examples.

We use the same architecture, hyper-parameters and augmentations, as Appendix F.

H SELECTIVE CLASSIFICATION EXPERIMENT DETAILS

H.1 BASELINES

- **MSP** (Hendrycks & Gimpel, 2016): A simple and strong baseline for selective classification, which directly uses the probability assigned by the base model as an estimate of confidence. MSP has been shown to distinguish in-distribution test examples that the model gets correct from the ones that it gets incorrect.
- **Binary Classifier**: Trains a classifier on the labeled training set and validation set to predict when the base model is correct. The classifier takes as input the softmax probabilities outputted by base model. For the Binary Classifier, we experimented with a random forest classifier, MLP, softmax probabilities, and last-layer features, and found the MLP with softmax probabilities to work best. In all experiments, we use a 2-layer MLP with hidden layer size of 512, SGD and cosine learning rate scheduler with an initial learning rate of 1e-3 and weight decay 5×10^{-4}.
- **Fine-tuning**: Fine-tune the pretrained network on the validation set.

H.2 DATASETS

- **CIFAR-10 $\rightarrow$ CIFAR-10-C** Hendrycks & Dietterich (2019): The task is to classify images into 10 classes, where the target distribution contains severely corrupted images. We run experiments over 15 of the corruptions (brightness, contrast, defocus blur, elastic transform, fog, frost, gaussian noise, glass blur, impulse noise, jpeg compression, motion blur, pixelate, shot noise, snow, zoom blur) and use the data loading code from Croce et al. (2020).
- **Waterbirds** Welinder et al. (2010); Sagawa et al. (2019): The Waterbirds dataset consists of images of landbirds and waterbirds on land or water backgrounds from the Places dataset . The train set consists of 4,795 images, of which 3,498 are of waterbirds on water backgrounds, and 1,057 are of landbirds on land backgrounds. There are 184 images of waterbirds on land and 56 images of landbirds on water, which are the minority groups.
- **Camelyon17** Koh et al. (2021); Bandi et al. (2018): The Camelyon17 dataset is a medical image classification task from the WILDS benchmark Koh et al. (2021). The dataset consists of $450,000$ whole-slide images of breast cancer metastases in lymph node from 5 hospitals. The input is a 96×96 image, and the label y indicates whether there is a tumor in the image. The train set consists of lymph-node scans from 3 of the 5 hospitals, while the OOD validation set and OOD test datasets consists of lymph-node scans from the 4th and 5th hospitals, respectively.
- **FMoW** Koh et al. (2021): The FMoW dataset is a satellite image classification task from the WILDS benchmark Koh et al. (2021). The dataset consists of satellite images in various geographic locations from $2002 - 2018$. The input is a 224×224 RGB satellite image, and the label y is one of 62 building or land use categories. The train, validation, and test splits are based on the year that the images were taken: the train, ID validation, and ID test sets consist of images from $2002 - 2013$, the OOD validation set consists of images from $2013 - 2016$, and the OOD test set consists of images from $2016 - 2018$.

H.3 CIFAR-10 → CIFAR-10-C TRAINING DETAILS

- **Architecture:** We use the Standard model from Croce et al. (2020) (WideResNet-28-10), which is trained on the source CIFAR-10 distribution and attains 94.78% source accuracy.
- **Hyper-parameters**: For TCM, Fine-tuning, and Binary Classifier, we fine-tune on the validation set for 10 epochs. For TCM and Fine-tuning, we use an initial learning rate of 0.001 and a cosine learning rate schedule, weight decay of 5×10^{-4}, and batch sizes of 128 and 256 for the fine-tuning set and misclassified validation sets, respectively. We tune all baselines over the 3 learning rates {1e-3, 1e-4, 1e-5}. For TCM, we use a default confidence weight of $\lambda = 0.5$ for all corruptions, as in Hendrycks et al. (2018).
- **Validation and test set construction:** We use the CIFAR-10 test set, and split it into a validation set of 5000 images, a test set of 4000 images, and set aside 1000 images for hyperparameter tuning. Similarly, for CIFAR-10-C, we use a validation set of 5000 images, a test set of 4000 images, and set aside 1000 images for hyperparameter tuning.

  Each of our settings merges the train/val/test splits from the corresponding datasets. For example, Val = CIFAR-10, Test = CIFAR-10 + CIFAR-10-C uses a validation set of 5000 CIFAR-10 images for fine-tuning and a test set of 4000 CIFAR-10 and 4000 CIFAR-10-C images. Note that our combined CIFAR-10 + CIFAR-10-C test sets have a 1:1 clean-to-corrupted ratio.
- **Augmentations:** For TCM and fine-tuning, we use the same standard random horizontal flip and random crop (32×32).

H.4 WATERBIRDS TRAINING DETAILS

- **Architecture:** For our base model, we train a pretrained ResNet50 from `torchvision` on a subset of the Waterbirds train set (details of the split are described below). We follow the training details for the ERM baseline used by Sagawa et al. (2019), and use SGD with a momentum term of 0.9, batch normalization, and no dropout. We use a fixed learning rate of 0.001, a ℓ_2 penalty of $\lambda = 0.0001$ and train for 300 epochs.
- **Hyper-parameters**: For TCM, Fine-tuning, and Binary Classifier, we fine-tune on the validation set for 10 epochs. For TCM and Fine-tuning, we fine-tune on the validation set for 10 epochs with an initial learning rate of 0.001 and a cosine learning rate schedule, weight decay of 5×10^{-4}, and batch sizes of 64 for the fine-tuning and misclassified validation sets. We tune all baselines over the 3 learning rates {1e-3, 1e-4, 1e-5}. For TCM, we use a confidence weight of $\lambda = 0.01$ and tune over $\lambda \in \{0.005, 0.01, 0.5, 1.5\}$.
- **Validation and test set construction:** We split the Waterbirds train set from Sagawa et al. (2019) into two sets, one which we use to pretrain a base ERM model, and the other which we use as our ID validation set. We maintain group ratios: the ID train and validation sets each contain 2,397 images, of which 1749 are of waterbirds on water, 528 are of landbirds on land, 92 are of waterbirds on land, and 28 are of landbirds on water. Our test set is the same test set from Sagawa et al. (2019).
- **Augmentations:** For TCM and fine-tuning, we use the same standard random horizontal flip and random center crop (224×224).

H.5 CAMELYON17 TRAINING DETAILS

- **Architecture:** We use a DenseNet121 pre-trained on the Camelyon17 train set from Koh et al. (2021) as our base model. These models use a learning rate of 0.001, ℓ_2 regularization strength of 0.01, batch size of 32, and SGD with momentum set to 0.9.
- **Hyper-parameters**: For TCM, Fine-tuning, and Binary Classifier, we fine-tune on the validation set for 1 epoch. For TCM and Fine-tuning, we use an initial learning rate of 1e-5 with a cosine learning rate schedule, weight decay of 5×10^{-4}, and batch sizes of 64 for the fine-tuning and misclassified validation sets. For the binary classifier MLP, we use an initial learning rate of 0.001. For TCM, we use a default confidence weight of $\lambda = 0.5$ for all corruptions, as in Hendrycks et al. (2018).
- **Validation and test set construction:** We use the train / ID val / OOD val / OOD test splits from the WILDS benchmark to construct our validation and test sets. For our ID validation set and ID test set, we split the Camelyon17 ID validation set into two equally-sized subsets and maintain group ratios. The Camelyon17 ID validation consists of samples

from the same 3 hospitals as the train set. We use the OOD test set as our target distribution, which contains samples from the 5th hospital.

- **Augmentations:** Following Koh et al. (2021), we normalize and resize images to 224×224, but use no random augmentations.

H.6 FMoW TRAINING DETAILS

- **Architecture:** We use FMoW ERM models from the WILDS benchmark Koh et al. (2021) as our base model. These models use DenseNet121 pretrained on ImageNet with no ℓ_2 regularization, Adam optimizer with an initial learning rate of 1e-4 that decays by 0.96 per epoch, and train for 50 epochs with early stopping and batch size of 64.
- **Hyper-parameters**: For TCM, Fine-tuning, and Binary Classifier, we fine-tune on the validation set for 10 epochs. For TCM and Fine-tuning, we use an initial learning rate of 1e-6 and a cosine learning rate schedule, weight decay of 5×10^{-4}, and batch sizes of 64 for the fine-tuning and misclassified validation sets. For the binary classifier MLP, we use an initial learning rate of 0.001. For TCM, we use a default confidence weight of $\lambda = 0.5$ for all corruptions, as in Hendrycks et al. (2018).
- **Validation and test set construction:** We use the train / ID val / OOD val / OOD test splits from the WILDS benchmark as our validation and test sets. Specifically, we use the ID validation set, ID test set, and OOD test sets. For example, the task Val = FmoW ID, Test = FMoW ID + FMoW OOD uses the WILDS ID val set for validation, and the WILDS ID and OOD test sets for testing.
- **Augmentations:** Following Koh et al. (2021), we use no random augmentations.