UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Luís Felipe Prates Cattelan

Post-hoc methods to enhance selective classification performance of deep neural networks

Florianópolis
2023

Luís Felipe Prates Cattelan

Post-hoc methods to enhance selective classification performance of deep neural networks

Dissertação submetida ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina para a obtenção do título de mestre em Engenharia Elétrica.
Orientador: Danilo Silva, Ph.D.

Florianópolis
2023

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Cattelan, Luís Felipe Prates
Post-hoc methods to enhance selective classification performance of deep neural networks / Luís Felipe Prates Cattelan ; orientador, Danilo Silva, 2023.
66 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós-Graduação em Engenharia Elétrica, Florianópolis, 2023.

Inclui referências.

1. Engenharia Elétrica. 2. Aprendizado Profundo. 3. Estimação de incerteza. 4. Classificação seletiva. I. Silva, Danilo. II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia Elétrica. III. Título.

Luís Felipe Prates Cattelan

Post-hoc methods to enhance selective classification performance of deep neural networks

O presente trabalho em nível de mestrado foi avaliado e aprovado por banca examinadora composta pelos seguintes membros:

Prof. Cristiano Torezzan, Dr.
UNICAMP

Prof. Aldebaro Barreto da Rocha Klautau Júnior, Ph.D.
UFPA

Prof. Eduardo Luiz Ortiz Batista, Dr.
UFSC

Certificamos que esta é a **versão original e final** do trabalho de conclusão que foi julgado adequado para obtenção do título de mestre em Engenharia Elétrica.

Documento assinado digitalmente
Telles Brunelli Lazzarin
Data: 19/10/2023 05:16:30-0300
CPF: ***.400.719-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Telles Brunelli Lazzarin, Dr.
Coordenador do Programa de
Pós-Graduação em Engenharia Elétrica

Documento assinado digitalmente
Danilo Silva
Data: 17/10/2023 16:58:35-0300
CPF: ***.557.924-**
Verifique as assinaturas em https://v.ufsc.br

Danilo Silva, Ph.D.
Orientador

Florianópolis, 2023.

RESUMO

No cenário tecnológico em constante evolução de hoje, a ampla adoção do Aprendizado Profundo tem inaugurado uma era de conquistas sem precedentes na inteligência artificial. Conforme esses poderosos algoritmos continuam a permear vários aspectos de nossas vidas, surge uma necessidade inerente de garantir a confiabilidade e segurança de suas previsões. Esta dissertação explora o problema da classificação seletiva para redes neurais profundas, permitindo que os modelos se abstenham de fazer previsões de baixa confiança para evitar erros potenciais.

Especificamente, nosso foco está na otimização do estimador de confiança de um classificador fixo para aprimorar o desempenho de detecção de erros de classificação. Essa melhoria visa aprimorar a capacidade do modelo de distinguir entre previsões corretas e incorretas, atribuindo valores de confiança mais altos às primeiras. Pesquisas anteriores indicaram que diferentes classificadores exibem níveis variados de desempenho na detecção de erros de classificação, especialmente ao usar a probabilidade máxima softmax (MSP) como medida de confiança. Argumentamos que essas disparidades são resultado de estimadores de confiança subótimos sendo utilizados para cada modelo. Para abordar esse problema, propomos um estimador de confiança *post-hoc* simples e eficiente chamado *p*-NormSoftmax. Esse estimador envolve a transformação dos logits por meio da normalização *p*-norma e do escalonamento de temperatura (i.e., da multiplicação do logits por um escalar), seguida pelo cálculo da MSP. Os valores de *p* e da temperatura são otimizados com base em um conjunto de validação, tornando o estimador prontamente aplicável a modelos já treinados. Em muitos casos, ele aprimora significativamente o desempenho de classificação seletiva dos modelos.

Por meio de avaliação empírica em 84 classificadores pré-treinados do conjunto de dados Imagenet, nosso método proposto *p*-NormSoftmax demonstra uma melhoria média de 16% na área sob a curva risco-cobertura (AURC), com alguns modelos exibindo melhorias de quase 50%. Além disso, observamos que, após a aplicação do *p*-NormSoftmax, esses modelos alcançam níveis equivalentes de desempenho na detecção de erros de classificação, sugerindo que o desempenho de classificação seletiva de um modelo é predominantemente determinado pela acurácia global em cobertura completa. Esta pesquisa contribui para avançar a compreensão da classificação seletiva em redes neurais profundas e oferece um método eficaz para aprimorar suas capacidades de detecção de erros de classificação.

Palavras-chave: Aprendizado Profundo, classificação seletiva, detecção de erros, estimação de incerteza

ABSTRACT

In today's rapidly evolving technological landscape, the widespread adoption of Deep Learning has ushered in an era of unprecedented achievements in artificial intelligence. As these powerful algorithms continue to pervade various aspects of our lives, there arises an inherent need to ensure the reliability and safety of their predictions. This dissertation explores the problem of selective classification for deep neural networks, allowing models to abstain from making low-confidence predictions to avoid potential errors. Specifically, our focus lies in optimizing the confidence estimator of a fixed classifier to enhance its misclassification detection performance. This enhancement aims to improve the model's ability to distinguish between correct and incorrect predictions by assigning higher confidence values to the former. Previous research has indicated that various classifiers exhibit differing levels of misclassification detection performance, particularly when using the maximum softmax probability (MSP) as a confidence measure. We argue that these disparities are largely a result of sub-optimal confidence estimators being employed for each model. To address this issue, we propose a straightforward and efficient post-hoc confidence estimator named p-NormSoftmax. This estimator involves transforming the logits through p-norm normalization and temperature scaling, followed by computing the MSP. The values of p and the temperature are optimized based on a hold-out set, making the estimator readily applicable to already trained models. In many cases, it significantly improves the selective classification performance of the models. Through empirical evaluation on 84 pretrained Imagenet classifiers, our proposed p-NormSoftmax method demonstrates an average improvement of 16% in the area under the risk-coverage curve (AURC), with some models exhibiting almost 50% of enhancements. Moreover, we observe that after applying p-NormSoftmax, these models attain equivalent levels of misclassification detection performance, suggesting that a model's selective classification performance is predominantly determined by its overall accuracy at full coverage. This research contributes to advancing the understanding of selective classification in deep neural networks and provides an effective method to improve their misclassification detection capabilities.

Keywords: Deep Learning. Classifiation. Selective Classification. Misclassification Detection. Uncertainty Estmation.

RESUMO EXPANDIDO

INTRODUÇÃO

Nos últimos anos, o Aprendizado Profundo emergiu como uma força revolucionária, avançando significativamente as capacidades da inteligência artificial e revolucionando diversos domínios. Desde visão computacional e processamento de linguagem natural até cuidados de saúde e sistemas autônomos, os modelos de Aprendizado Profundo têm demonstrado um notável sucesso ao realizar tarefas que antes eram consideradas fora do alcance das máquinas. À medida que esses modelos continuam a permear aplicações críticas, garantir sua confiabilidade se torna essencial para assegurar sua integração segura em cenários do mundo real.

Um desafio inerente enfrentado pelos modernos algoritmos de Aprendizado Profundo reside na capacidade de fornecer estimativas de incerteza para suas previsões. Embora excelentes na produção de resultados precisos, esses modelos frequentemente carecem de meios para quantificar e comunicar o nível de confiança associado às suas saídas. Essa limitação é uma preocupação significativa, especialmente em domínios de alto risco, como saúde, finanças e condução autônoma, onde decisões equivocadas podem levar a consequências graves.

Abordar o problema da estimativa de incerteza em Aprendizado Profundo é fundamental para construir confiança e segurança em sistemas de IA. Ao quantificar a incerteza, esses modelos podem expressar quando estão inseguros sobre um resultado, capacitando os tomadores de decisão a fazer escolhas informadas e tomar ações adequadas. Além disso, a estimativa de incerteza desempenha um papel crucial em melhorar a robustez e segurança das aplicações de IA, permitindo que os modelos se abstenham de fornecer previsões de baixa confiança, evitando assim potenciais erros.

Um resultado de destaque na literatura recente da área é que os modelos de classificadores de imagem apresentam desempenhos variados na estimativa de incerteza. Essa disparidade suscita uma importante questão: o que causa tais diferenças e se existem métodos capazes de aprimorar a capacidade de quantificar a incerteza em um modelo já treinado. Esse fenômeno ainda é um mistério a ser desvendado, e compreendê-lo pode abrir caminhos para o desenvolvimento de abordagens que aperfeiçoem a estimativa de incerteza em aplicações de Aprendizado Profundo.

Nesta dissertação, focamos no desafio da classificação seletiva para redes neurais profundas. A classificação seletiva envolve permitir que os modelos se abstenham de fazer previsões de baixa confiança, o que pode melhorar significativamente seu desempenho e confiabilidade geral. Nossa pesquisa explora a otimização de estimadores de confiança para classificadores fixos, com o objetivo de aprimorar suas capacidades de detecção de erros de classificação. Propomos um novo estimador de confiança pós-processamento, denominado p-NormSoftmax, que efetivamente transforma os logits através da normalização com a norma p e do ajuste de temperatura, resultando em um melhor desempenho na classificação seletiva.

OBJETIVOS

Neste trabalho, o objetivo geral é investigar o que torna um classificador de Aprendizado Profundo bom (ou ruim) em classificação seletiva e propor métodos para aprimorar a capacidade de modelos treinados de alcançar um bom desempenho nesse contexto.

Os objetivos específicos desta dissertação são:

1. Fornecer uma visão abrangente da importância da estimativa de incerteza no contexto de modelos de Aprendizado Profundo e suas aplicações.
2. Propor um estimador de confiança *post-hoc* simples e eficiente para classificação seletiva;
3. Investigar por que alguns modelos são bons ou ruins em classificação seletiva.

METODOLOGIA

Neste trabalho foi apresentado um método para aumentar a habilidade de uma rede neural de classificação de imagens em detectar possíveis erros. O método é construído a partir de observações empíricas, e os resultados obtidos através da utilização de redes pre-treindas disponibilizadas *online* e/ou treinando redes para alguns datasets. As comparações foram feitas utilizando métricas de classificação seletiva e detecção de erros de classificação. Ainda, são apresentadas propriedades relevantes para um método como o proposto, como a eficiência de generalização e a robustez a *datasets* de distribuições distintas ao treinamento.

RESULTADOS E DISCUSSÃO

Ao aplicar o método *p*-NormSoftmax como estimador de confiança *post-hoc*, observamos uma melhoria média de 16% na área sob a curva risco-cobertura (AURC) entre os classificadores testados. Além disso, para alguns modelos, as melhorias ultrapassaram 40%, demonstrando o potencial desse método para aprimorar substancialmente o desempenho da classificação seletiva. Esses resultados promissores destacam a importância das normas dos logits no papel de quantificar a incerteza.

Além disso, a análise do desempenho dos classificadores antes e depois da aplicação do *p*-NormSoftmax revelou insights intrigantes sobre seu comportamento. Modelos pré-treinados que apresentaram desempenhos diversos na detecção de erros de classificação ao depender da máxima probabilidade softmax (MSP) demonstraram uma significativa convergência em seus níveis de desempenho após a utilização do *p*-NormSoftmax. Isso sugere que o desempenho da classificação seletiva é amplamente determinado pela precisão do modelo em cobertura total, e nosso estimador de confiança proposto ajuda a preencher a lacuna entre a calibração dos modelos e suas habilidades de classificação seletiva.

CONCLUSÃO

A efetividade e eficiência do método *p*-NormSoftmax tornam-no uma promissora adição ao conjunto de técnicas de estimativa de incerteza para redes neurais profundas. Sua implementação direta permite uma fácil aplicação pós-processamento em modelos treinados, sem a necessidade de retreinamento. Essa praticidade é especialmente vantajosa em aplicações do mundo real, onde retreinar um modelo pode não ser viável ou prático.

Em geral, os resultados obtidos nesta tese contribuem significativamente para a compreensão da estimativa de incerteza em Aprendizado Profundo e suas implicações para a classificação seletiva. O método *p*-NormSoftmax oferece uma valiosa solução para abordar o problema de sobreconfiança exibido por alguns modelos, aprimorando a confiabilidade dos sistemas de IA e promovendo sua integração mais segura em domínios críticos. As percepções obtidas com esta pesquisa abrem caminhos para investigações futuras em aprendizado profundo consciente de incerteza e seu potencial impacto em aplicações mais amplas de IA.

LIST OF FIGURES

LIST OF TABLES

CONTENTS

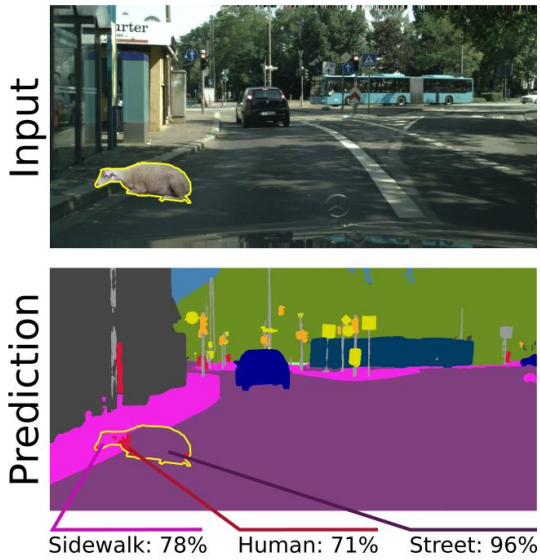
1 INTRODUCTION

Over the last few decades, there has been a significant trend of computers assuming roles that were previously exclusive to human. In recent years, a groundbreaking technology, **Deep Learning**, has emerged as a pivotal driving force in this paradigm shift. Demonstrating unprecedented capabilities, Deep Learning has proven its ability to excel in exceptionally complex and critical tasks. Its achievements span a wide array of domains, encompassing computer vision, natural language processing, healthcare, and autonomous systems, among others. This surge in Deep Learning's accomplishments has effectively paved the way for a new era where machines actively aid in accomplishing diverse tasks that were once solely reliant on human expertise.

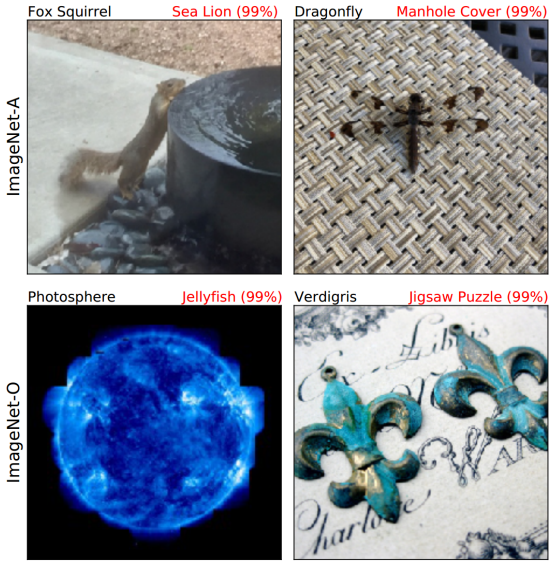
As artificial intelligence models find their way into more and more critical applications, it becomes essential to know when we can truly rely on their outputs. Consider a scenario where a patient visits a doctor seeking a diagnosis. In some cases, the doctor might provide an inconclusive answer, prompting the need for further tests or a second opinion. On the other hand, traditional machine learning algorithms tend to offer definitive responses. This disparity underscores a vital aspect of a machine learning algorithm: its capacity to express *"I don't know"*. Consequently, it becomes increasingly crucial not only to obtain accurate predictions but also to quantify the level of uncertainty associated with these predictions.

By being able to assess uncertainty, we gain valuable insights into the reliability of the machine learning (ML) model's predictions. This knowledge empowers us to make more informed decisions and take appropriate actions, especially in critical situations where erroneous or overly confident predictions could lead to detrimental consequences. Uncertainty estimation in Deep Learning serves as a powerful tool, bridging the gap between AI's capabilities and human understanding, thus fostering a safer and more dependable integration of AI technologies in real-world applications. In summary, a reliable model must be able to identify cases where it is likely to make an incorrect prediction and withhold the output to prevent a wrong decision(ZOU et al., 2023; NEUMANN; ZISSERMAN; VEDALDI, 2018).

Unfortunately, it is well-known that modern deep neural networks often exhibit overconfidence in their predictions (GUO et al., 2017; GOODFELLOW; SHLENS; SZEGEDY, 2014). In Figure 1, we can observe practical situations where real AI models demonstrate overconfident errors. Particularly noteworthy is Figure 1a, which vividly exemplifies the potential disastrous consequences of misclassifications in critical applications like autonomous driving. Furthermore, Figure 1b highlights that overconfidence is not solely restricted to cases where the model should accurately predict classes seen during training. It extends to situations where the model struggles to classify images correctly, even for classes that were not encountered during training. This distinction



(a) Semantic segmentation for autonomous driving misclassification (BLUM et al., 2021).



(b) Overconfident errors in image classification (HENDRYCKS et al., 2021). The true class is in black, while the predicted class in red (with the associated confidence).

Figure 1 – Examples of practical overconfident errors in Deep Learning classification.

becomes apparent as the top images in the figure correspond to classes seen in the training data, while the bottom images represent samples from previously unseen classes.

These findings emphasize the urgency of addressing the issue of overconfidence in Deep Learning models and has motivated a lot of recent research in the general subject of uncertainty estimation in deep learning (GAWLIKOWSKI et al., 2022; ZHANG, X.-Y. et al., 2023; ABDAR et al., 2021). Nevertheless, the inherent complexity and non-linear nature of deep neural networks make uncertainty estimation a challenging task.

The possibility of a model to abstain from likely wrong predictions is referred as *reject option* (ZHANG, X.-Y. et al., 2023; MURPHY, 2022). In the literature, this problem is commonly addressed as the task of *misclassification detection* (HENDRYCKS; GIMPEL, 2016), where the uncertainty estimator is treated as a binary classifier of mistakes, or as *selective classification* (GEIFMAN; EL-YANIV, 2017), which refer to the task of enhancing a classifier's accuracy by abstaining from low-confidence predictions. Although different metrics can be used, the problem is intrinsically the same.

In the case of neural networks with softmax outputs, which are by far the most common framework of Deep Learning classifiers, the default approach is to treat the outputs of the softmax function as probabilities and, from there, estimate the uncertainty. Indeed, the natural baseline is to take the maximum softmax probability (MSP) as a confidence estimator (GEIFMAN; EL-YANIV, 2017; HENDRYCKS; GIMPEL, 2016).

The vast majority of papers in the area attempt to improve upon this baseline by

either modifying the training procedure or designing a specific architecture to provide better uncertainty quantification. While such an approach is potentially optimal, the fact that it requires retraining a model is a significant practical drawback. An alternative, less explored approach is that of post-hoc learning a confidence estimator, which consists in determining the confidence estimation method after the training and without changing the prediction of the classifier. Papers that follow this approach typically construct a *meta-model* that feeds on intermediate features of the base model and is trained to predict whether or not the base model is correct on hold-out samples (CORBIÈRE et al., 2022; SHEN et al., 2022). However, depending on the size of such a meta-model, its training may still be computationally demanding.

Another very common approach is to make the use of ensembles (GAWLIKOWSKI et al., 2022). Originally proposed to simply improve the general performance, ensemble models have gained popularity in the field of uncertainty estimation as the variability of predictions among different models can be used as an sign of uncertainty. Moreover, some ensemble techniques can be considered as post-hoc, such as Monte-Carlo Dropout (GAL; GHAHRAMANI, 2016). However, the fact that multiple inference passes need to be performed significantly increases the computational burden at test time.

In this work, we focus on simple post-hoc methods for confidence estimation that can be computed directly from the network unnormalized *logits* (pre-softmax output). This approach is practically appealing as it can be directly applied to any pre-trained model with a Softmax activation function on the outputs. Such post-hoc methods are common in the related (but fundamentally different) task of probability calibration, which aims to provide probability estimates representative of the true likelihood of correctness. The most prominent example is temperature scaling (TS) (GUO et al., 2017), a simple and efficient logit-based method that requires tuning a single parameter and is shown to be remarkably effective for calibration. The usefulness of TS for selective classification has been investigated by (GALIL; DABBAH; EL-YANIV, 2023), who observed that, depending on the model, TS may improve or harm selective classification performance.

1.1 OBJECTIVES

In this work, the general objective is to investigate what makes a Deep Learning classifier good (or bad) at selective classification and propose methods to enhance the ability of trained models to achieve good performance in this framework.

The specific objectives of this thesis are:

1. Provide a comprehensive overview of the importance of uncertainty estimation in the context of deep learning models and their applications;
2. Propose a simple and efficient post-hoc confidence estimator optimized for selective classification;

3. Investigate why some models are good or bad at selective classification.

By addressing these objectives, this thesis aims to contribute to the existing body of knowledge on uncertainty estimation in deep learning, providing insights and practical guidelines for improving the reliability and interpretability of deep learning models in various domains.

1.2 CONTRIBUTIONS

Inspired by (GALIL; DABBAH; EL-YANIV, 2023), we propose a post-hoc confidence estimator that combines three previous ideas: temperature scaling, logit normalization (WEI et al., 2022a) and logit centralization (JIANG; GU; PAN, 2023), the latter two originally proposed as training techniques. We further extend these ideas by considering a more general p-norm normalization and by optimizing the temperature (as well as p) directly to improve selective classification performance. In addition, we propose a simple heuristic to choose the temperature so that only p needs to be optimized. Our approach, named p-NormSoftmax, is practically appealing as it can be applied to any existing model without retraining, requires tuning a single parameter, is straightforward to implement, and is very data-efficient. Moreover, it can provide significant gains in selective classification performance for a variety of existing models, with the only considerable cost of having sufficient samples not used during training.

Our method apparently solves an intriguing problem reported in (GALIL; DABBAH; EL-YANIV, 2023): some state-of-the-art ImageNet classifiers, despite attaining excellent predictive performance, nevertheless exhibit appallingly poor performance at misclassification detection. After applying our method, this issue completely disappears, suggesting that such pathologies are fixable.

In summary, the contributions of this work are:

- We investigate the trade-off between calibration and selective classification metrics in the context of temperature scaling and show that selective classification performance can be improved by disregarding calibration and directly optimizing a selective classification metric;
- We propose a simple and efficient post-hoc confidence estimator optimized for selective classification;
- An experimental study of the selective classification performance of 84 ImageNet classifiers, showing an average gain of 16% in AURC after the application of our method;
- A comparison of these models showing that, after p-NormSoftmax, all models exhibit approximately the same level of misclassification detection performance.

1.3 RELATED WORKS

Selective prediction is also known as learning with a reject option (see (ZHANG, X.-Y. et al., 2023; HENDRICKX et al., 2021) and references therein), where the rejector is usually a thresholded confidence estimator. Essentially the same problem is studied under the equivalent terms misclassification detection (HENDRYCKS; GIMPEL, 2016), failure prediction (CORBIÈRE et al., 2022; ZHU et al., 2022), and (ordinal) ranking (MOON et al., 2020; GALIL; DABBAH; EL-YANIV, 2023). Uncertainty estimation is a more general term that encompasses these tasks (where confidence may be taken as negative uncertainty) as well as other tasks where uncertainty might be useful, such as calibration and out-of-distribution (OOD) detection, among others (GAWLIKOWSKI et al., 2022; ABDAR et al., 2021). These tasks are generally not aligned: for instance, optimizing for calibration may harm selective classification performance (DING et al., 2020; ZHU et al., 2022; GALIL; DABBAH; EL-YANIV, 2023). Our focus here is on in-distribution selective classification, although we also study robustness to distribution shift. While most approaches consider the base model as part of the learning problem (GEIFMAN; EL-YANIV, 2019; HUANG; ZHANG, C.; ZHANG, H., 2020; LIU, Z. et al., 2019), we focus on simple post-hoc estimators that can be computed from the logits[1]. Note that, from a post-hoc perspective, other tasks can be treated as independent problems.

A popular tool in the uncertainty literature is the use of ensembles (LAKSHMINARAYANAN; PRITZEL; BLUNDELL, 2017; GAL; GHAHRAMANI, 2016; TEYE; AZIZPOUR; SMITH, 2018; AYHAN; BERENS, 2018). While constructing a confidence estimator from ensemble component outputs may be considered post-hoc if the ensemble is already trained, recent work has found evidence that ensembles may not be fundamental for uncertainty but simply better predictive models (ABE et al., 2022; CATTELAN; SILVA, 2022; XIA; BOUGANIS, 2022). Thus, we do not consider ensembles here.

Applying TS to improve calibration (of the MSP confidence estimator) was proposed in (GUO et al., 2017) based on the negative log-likelihood. Optimizing TS for other metrics has been explored in (MUKHOTI et al., 2020; KARANDIKAR et al., 2021; CLARTÉ et al., 2023) for calibration and in (LIANG; LI; SRIKANT, 2023) for OOD detection, but had not been proposed for selective classification. A generalization of TS is adaptive TS (ATS) (A. BALANYA; RAMOS; MAROÑAS, 2023), which uses an input-dependent temperature based on logits. Our approach can be seen as a special case of ATS, as logit norms may be seen as an input-dependent temperature; however (A. BALANYA; RAMOS; MAROÑAS, 2023) investigate a different temperature function than ours and focuses on calibration. Other logit-based confidence estimators proposed

[1] Interestingly, (FENG, Leo et al., 2023) has found that, for some of these approaches, MSP is still the best selective mechanism after the base model is trained.

for calibration and OOD detection include (LIU, W. et al., 2020; TOMANI; CREMERS; BUETTNER, 2022; RAHIMI et al., 2022; NEUMANN; ZISSERMAN; VEDALDI, 2018; GONSIOR et al., 2022).

Normalizing the logits with the L_2 norm before applying the softmax function was used in (KORNBLITH et al., 2021) and later proposed and studied in (WEI et al., 2022a) as a training technique (combined with TS) to improve OOD detection and calibration. A variation where the logits are normalized to unit variance was proposed in (JIANG; GU; PAN, 2023) to accelerate training.

Benchmarking of models in their performance at selective classification (and/or misclassification detection) has been done in (GALIL; DABBAH; EL-YANIV, 2023; DING et al., 2020), however these works mostly consider the MSP as the confidence estimator. In the context of calibration, (WANG; FENG, Lei; ZHANG, M.-L., 2021) and (ASHUKHA et al., 2020) have argued that models should be compared after simple post-hoc optimizations, since models that appear worse than others can sometimes easily be improved by methods such as TS. Here we advocate and provide further evidence for this approach in the context of selective classification.

1.4 THESIS STRUCTURE

The organization of this thesis is the follow:

- In Chapter 2, a background together with a literature review on the important topics to understand the proposed methods are presented;
- In Chapter 3, the proposed methods are introduced and discussed;
- Chapter 4 brings the results and discussion;
- Chapter 5 concludes the thesis.

2 BACKGROUND

This chapter aims to revisit and introduce the theory and practice of deep learning and classification modern approaches.

2.1 CLASSIFICATION

The goal of classification is to take an input $\boldsymbol{x}$ and to assign it to one of the C discrete possible classes. Let P be a distribution over $\mathcal{X}\times\mathcal{Y}$, where $\mathcal{X}$ is the inputs space and $\mathcal{Y}$ = {1,2,..,C} is the label space. A *classifier* is a prediction function $h : \mathcal{X} \to \mathcal{Y}$. A very common risk function associated to a classifier is the *zero-one loss*, defined as:

$$\ell_{01}(y,h) = \mathbb{I}(y \neq h) \tag{1}$$

where $\mathbb{I}$ denotes the indicator function and $y \in \mathcal{Y}$ is the true label. Note that the mean zero-one loss is equivalent to 1 minus the accuracy. This risk can also be referred as the Top-1 Error, as it measures the error when considering only one class predicted by the classifier.

The bayesian optimal decision rule for choosing $h(\boldsymbol{x})$ based on ℓ_{01} is given by (MURPHY, 2022):

$$h(\boldsymbol{x}) = \arg\max_{c\in\mathcal{Y}} p(c|\boldsymbol{x}). \tag{2}$$

Unfortunately, in general we do not know the distribution P and, consequently, the posterior distribution $p(y|\boldsymbol{x}) = P(\boldsymbol{x},y)/p(\boldsymbol{x})$. For that reason, our goal to construct a parametric function $p_\theta : \mathcal{X} \to [0,1]^C$ that tries to emulate the real posterior distribution and, consequently, achieves good classification performance.

In summary, we want to select the parameters $\boldsymbol{\theta}$ that makes p_θ (most as possible) equal to p. Therefore, one approach is to minimize the Kullback-Leibler (KL) divergence between both probabilities distributions:

$$KL(p||q_\theta) \triangleq \int_{\boldsymbol{x},y\in\mathcal{X}\times\mathcal{Y}} p(y|\boldsymbol{x}) log\left(\frac{p(y|\boldsymbol{x})}{p_\theta(y|\boldsymbol{x},\boldsymbol{\theta})}\right), \tag{3}$$

$$\theta^* = \arg\min_\theta KL(p||p_\theta). \tag{4}$$

Generally, instead of the entire space $\mathcal{X}$, one has access to N samples, forming the dataset $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)_{i=1}^N\} \sim P(\boldsymbol{x},y)$. Moreover, in supervised learning, the input is generally associated with the truth label with probability 1, i.e., the vector of probabilities is defined with *one-hot encoding*. From that, we can rewrite equation (3) and define $\mathcal{L}(p_\theta(y|\boldsymbol{x},\boldsymbol{\theta}),\mathcal{D})$, known as Negative Log Likelihood (NLL) or Cross Entropy Loss:

$$\mathcal{L}_{NLL}(p_\theta,\mathcal{D}) \triangleq -\sum_{i=1}^{N} log(p_\theta(y|\boldsymbol{x},\boldsymbol{\theta})). \tag{5}$$

Thus, minimizing $\mathcal{L}$ is equivalent to minimizing KL divergence in $\mathcal{D}$. Furthermore, we can rewrite Equation (5) as:

$$\mathcal{L}_{NLL}(p_\theta, \mathcal{D}) = -log\left(\prod_{i=1}^{N} p_\theta(y|\boldsymbol{x},\boldsymbol{\theta})\right). \tag{6}$$

Being the logarithm a monotonic function, minimizing $\mathcal{L}_{NLL}$ is equivalent to the method of Maximum Likelihood Estimation (MLE), i.e., choosing the parameters that maximizes the probabilities of the correct labels or each input.

Note that, since p_θ maps the input to a probability distribution over the C possible labels, we must require that $0 \leq p_\theta(c) \leq 1$ for each c and $\sum_{c=1}^{C} p_\theta(c) = 1$. To ensure these constraints, it is common in Machine Learning to construct a function $f_\theta : \mathcal{X} \to \mathbb{R}^C$ and convert it outputs to probabilities using the ***Softmax*** function, defined as:

$$\sigma : \mathbb{R}^C \to [0,1]^C, \qquad \sigma_k(\boldsymbol{z}) = \frac{e^{z_k}}{\sum_{j=1}^{C} e^{z_j}}, \quad k \in \{1, \ldots, C\} \tag{7}$$

where $\sigma_k(\boldsymbol{z})$ denotes the kth element of the vector $\sigma(\boldsymbol{z})$.

Thus:

$$p_\theta(c|\boldsymbol{x}) = \sigma_c(f_\theta(\boldsymbol{x})). \tag{8}$$

The input of the Softmax function i.e., the "raw" outputs of the classifier, is referred as the *logits* vector.

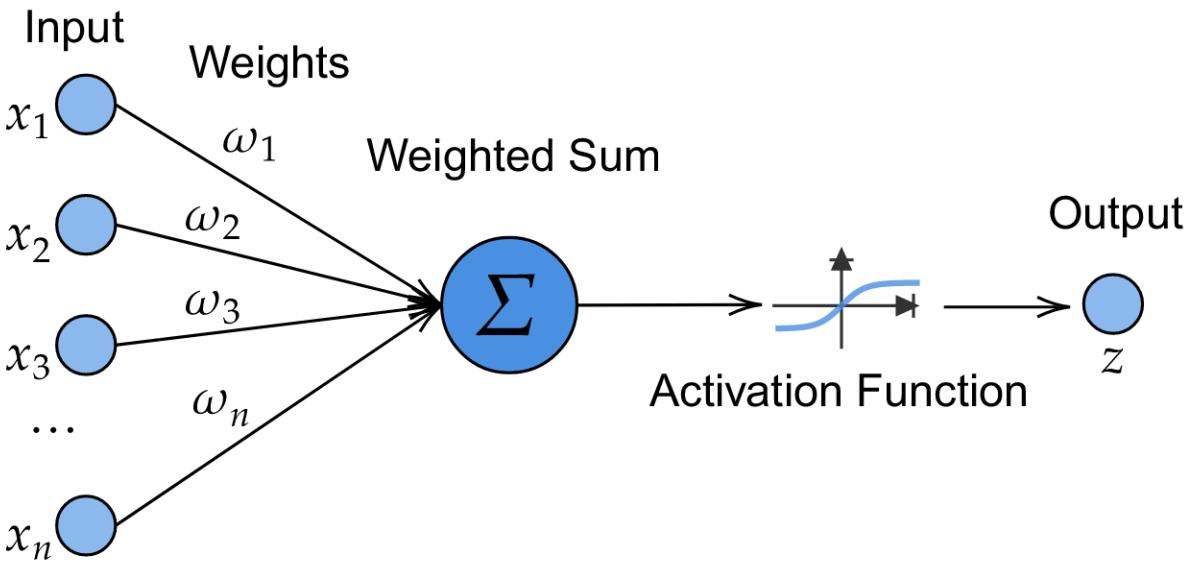
2.2 DEEP NEURAL NETWORKS

The primordial of neural networks goes back to the invention of the computational neuron, a.k.a. perceptron (ROSENBLATT, 1958), defined as $f(\boldsymbol{x}; \boldsymbol{w}, b) = H(\boldsymbol{w}^T\boldsymbol{x} + b)$, where H is the Heaviside step function and $\boldsymbol{w}$ and b are learnable parameters. The perceptron, which is represented in Figure 2a, was first introduced as a deterministic binary classifier, attached to a derivative-free learning algorithm. However, (MINSKY; PAPERT, 1969) showed some limitations of this algorithm; specially, they presented the XOR problem as an impossible problem to be solved with the perceptron.

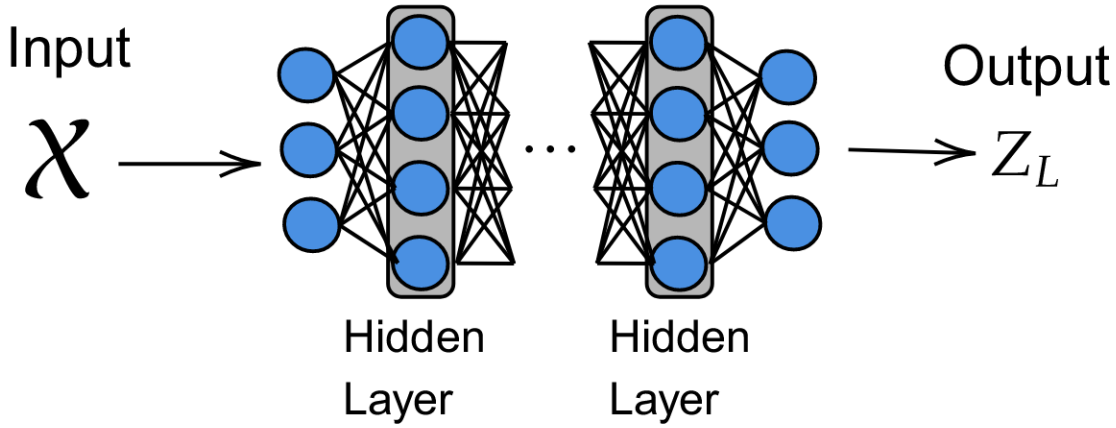
To approach more complex problems, the main idea is to stack multiple neurons on top of each other. Moreover, in order to make the model differentiable, the Heaviside function must be replaced by an differentiable *activation function* $\varphi : \mathbb{R} \to \mathbb{R}$. Thus, the **multilayer perceptron** is defined as a system with L layer such that:

$$\boldsymbol{z}_l = \varphi_l(\boldsymbol{b}_l + \boldsymbol{W}_l\boldsymbol{z}_{l-1}) \tag{9}$$

where $\boldsymbol{z}_l$ is the output and $\boldsymbol{W}_l$ and $\boldsymbol{b}_l$ are learnable parameters of the layer l. Being z_0 the input $\boldsymbol{x}$, the multiplayer perceptron is thus defined as a model $f(\boldsymbol{x};\boldsymbol{\theta}) = \boldsymbol{z}_L$, where



(a) The Perceptron



(b) A Multi-Layer Perceptron

Figure 2 – Neural Networks schematics

$\boldsymbol{\theta} = \{\boldsymbol{W}, \boldsymbol{b}\}$ represents the learnable parameters. The multilayer perceptron, also known as Deep Neural Network, is schematized in Figure 2b.

In general, the parameters θ must be selected such that the model outputs the desired prediction. The general framework for optimizing the parameters is, thus, to define a loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$, where $\mathcal{Y}$ is the output space, and select the parameters that minimize it:

$$\boldsymbol{\theta} = \arg\min_{\boldsymbol{\theta}} \ell(\hat{y}, y) \tag{10}$$

where y is the ground truth and $\hat{y} = f(\mathbf{x}; \boldsymbol{\theta})$ is the model's output. Usually, the training of a neural network is made minimizing the loss function over a dataset $\mathcal{D} = \{(x_i, y_i)_{i=1}^N\}$, defining the loss $\mathcal{L}$ as:

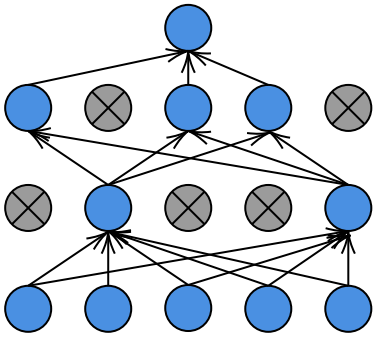


Figure 3 – Dropout method. For each iteration, a random group of parameters is disabled.

$$\mathcal{L}(f,\mathcal{D}) = \frac{1}{N}\sum_{i=1}^{N} \ell(\hat{y}(x_i),y_i). \tag{11}$$

Commonly, Deep Learning systems can have millions parameters. Therefore, their optimization must be made with efficient procedures. The most common algorithm is the Stochastic Gradient Descent (SGD), an iterative method that uses the gradient information to minimize the loss function:

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^{k} - \eta\vec{\nabla}_{\theta}\mathcal{L}(f_{\theta^k},\mathcal{D}) \tag{12}$$

where k is the current iteration and η is the learning rate, a hyper-parameter. One simple heuristic to improve SGD's convergence is to add *momentum*, implemented as follows:

$$\boldsymbol{m}^{k} = \beta\boldsymbol{m}^{k-1} + \vec{\nabla}_{\theta}\mathcal{L}(f_{\theta^k},\mathcal{D}) \tag{13}$$

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^{k} - \eta\boldsymbol{m}^{k} \tag{14}$$

where β is a hyperparameter. The gradients are calculated using the well known *back-propagation* algorithm.

2.2.1 Dropout

Often, DNN's have millions of parameters, possibly resulting in overfitting. One popular heuristic for decreasing overfitting in *dropout*, in which parameters are radomly omitted each time an inference is made during training time. More precisely, if $\theta_{l,i,j}$ is an element of $\boldsymbol{\theta}_l$ positioned in row i and column j (i.e., that connects node i in layer $l-1$ to node j in layer j), we replace it by $\tilde{\theta}_{l,i,j} = \theta_{l,i,j}\varepsilon l,i$, where $\varepsilon_{l,i} \sim Ber(1-p)$, where Ber is the Bernoulli distribution and p is the drop probability. During test time, p is set to 0, and thus all parameters are considered.

2.2.2 Batch Normalization

In order to turn training faster and more stable, it is common to add *normalization layers*, which force the empirical mean and variance of groups of activations. The most popular operator in this family is *batch normalization*, that ensures the distribution of the activations within a layer has zero mean and unit variance, when averaged across the samples in a mini-batch. Given a batch $\mathcal{B}$, it first computes the mean $\boldsymbol{\mu}_{\mathcal{B}}$ and variance $\boldsymbol{\sigma}^2_{\mathcal{B}}$ of activation $\boldsymbol{z}$ across the batch:

$$\boldsymbol{\mu}_{\mathcal{B}} = \frac{1}{|\mathcal{B}|} \sum_{\boldsymbol{z} \in \mathcal{B}} \boldsymbol{z} \tag{15}$$

$$\boldsymbol{\sigma}^2_{\mathcal{B}} = \frac{1}{|\mathcal{B}|} \sum_{\boldsymbol{z} \in \mathcal{B}} (\boldsymbol{z} - \mu_{\mathcal{B}})^2. \tag{16}$$

From it, the activation layer $\boldsymbol{z}$ is replaced by $\tilde{\boldsymbol{z}}$, which is computed as follow:

$$\tilde{\boldsymbol{z}} = \boldsymbol{\gamma} \odot \frac{\boldsymbol{z} - \mu_{\mathcal{B}}}{\sqrt{\boldsymbol{\sigma}^2_{\mathcal{B}} + \varepsilon}} + \boldsymbol{\beta} \tag{17}$$

where $\varepsilon > 0$ is a small constant and $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ are learnable parameters for this layer. Note that this normalization is defined across a batch, and thus it can be done only during training. During test time inferences, the operation is made using fixed means and variances. These values are estimated with a moving average over the training set.

2.2.3 Convolutional Neural Networks

When dealing with images, MLPs generally do not scale well. For an image $\boldsymbol{x} \in \mathcal{R}^{W,H,C}$, where W is the width, H is the height and C is the number of input channels (generally 3 for RBG color), we would need $(W \times H \times C + 1)$ parameters for each hidden unit in the first layer $\boldsymbol{\theta}_1$. This is very inefficient. Moreover, one key important property of common image applications is the *translation equivariance*, i.e., patterns must be recognized wherever they occur in the input. However, as the weights of MLPs are not shared across locations, a pattern that occurs in one location may not be recognized when it occurs in a different location. To solve these problems, (LECUN; BENGIO, et al., 1995) proposed Convolutional Neural Networks (CNNs)[1], in which the matrix multiplications are replaced by convolution operations.

Convolution layers consist in applying the same weight matrix (called kernel or filter) to each local patch of the image. This weight matrix is convoluted with the image. For the l'th layer, D_l kernel's are independently convolved with the image. Thus, $\boldsymbol{W}_l$ is usually a 4d matrix (one dimension for each dimension of the image and one dimension

[1] The core idea can be attributed to the previous work by (FUKUSHIMA, 1980)

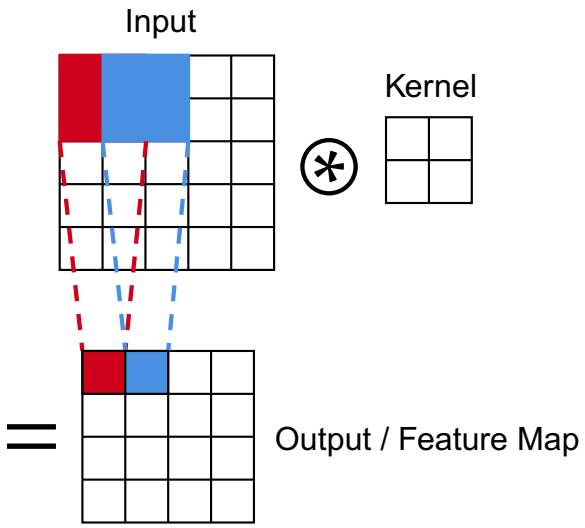


Figure 4 – Convolution Process

corresponding d corresponding to the number of filters). The output $\boldsymbol{z_l}$ is calculated as $\boldsymbol{z_l} = \varphi_l(\boldsymbol{z_{l-1}} \circledast \boldsymbol{W}_l + \boldsymbol{b}_l)$, where the element $z_{l,i,j,d}$ is calculated as:

$$\boldsymbol{z}_{i,j,d} = \varphi_l(b_{l,d} + \sum_{a=0}^{H_{l-1}} \sum_{b=0}^{W_{l-1}} \sum_{c=0}^{D_{l-1}} z_{l-1,si+a,sj+b,c} w_{l,a,b,c,d}) \tag{18}$$

where s is called the *stride*.

In order to reduce the dimension of some feature maps and to provide location invariance, a *pooling layer* (e.g., Max pooling) is generally applied.

2.2.4 Visual Transformers

The Transformer architecture (VASWANI et al., 2017) is based on the *attention* operation. The main idea of Attention is to make the weights dependent on the input, i.e., $\boldsymbol{z} = \varphi(\boldsymbol{W}(\boldsymbol{x})\boldsymbol{x})$. Generally, this is made by defining three variables: the Query $\boldsymbol{Q} \in \mathbb{R}^{N_q \times D_k}$, the key $\boldsymbol{K} \in \mathbb{R}^{N_v \times D_k}$ and the value $\boldsymbol{V} \in \mathbb{R}^{N_v \times D_v}$, calculated respectively as $\boldsymbol{W}_q\boldsymbol{x}$, $\boldsymbol{W}_k\boldsymbol{x}$ and $\boldsymbol{W}_v\boldsymbol{x}$, where $\boldsymbol{W}_q$, $\boldsymbol{W}_k$ and $\boldsymbol{W}_v$ are trainable matrices. The attention is then calculated as:

$$\text{Attention}(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \sigma\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{D_k}}\right) \cdot \boldsymbol{V} \tag{19}$$

where the softmax function is applied row-wise.

After years of success and dominance of convolutional neural networks, (DOSOVITSKIY et al., 2020) proposed to apply the idea of Transformers, which was already widely used for natural language processing, to images recognition. Since then, Visual Transformers (ViTs) have achieved state of the art performance in these tasks. The main idea of the ViT is to chop the input image into flattened patches, map it into a

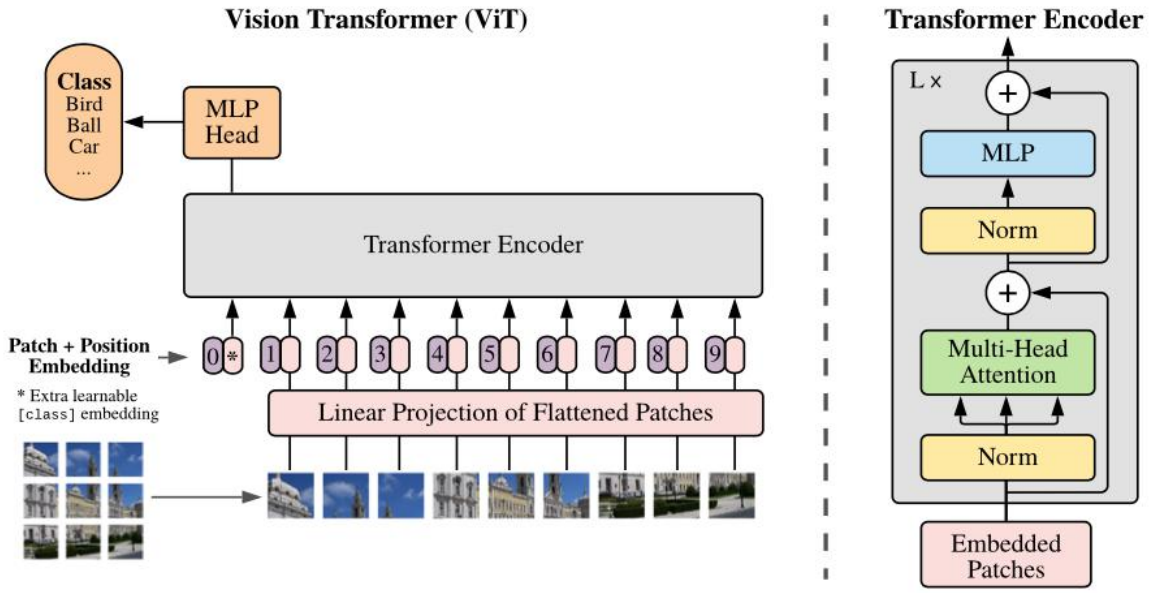
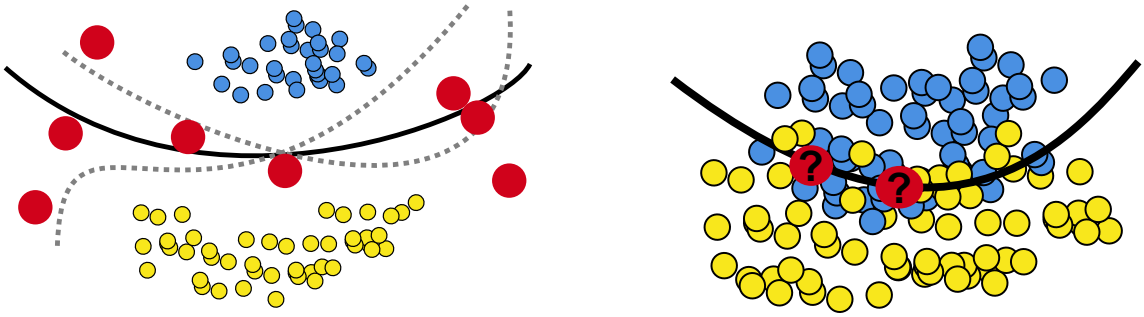


Figure 5 – Visual Transformers architecture (DOSOVITSKIY et al., 2020)



(a) Epistemic Uncertainty (b) Data Uncertainty

Figure 6 – Types of Uncertainty

embedding space and then pass these set of embeddings to a transformer. Figure 5 show the basic architecture of a ViT.

2.3 UNCERTAINTY IN DEEP LEARNING

As discussed in chapter 1, there are many reasons for a model to be uncertain about a sample during inference. Conventionally, uncertainty is divided in two forms: *epistemic uncertainty* (a.k.a model uncertainty), when the model has limited knowledge and would benefit of more training data, and *aleatoric uncertainty* (a.k.a data uncertainty), the uncertainty intrinsically related to the data. Figure 6 illustrates these kinds of uncertainty in a binary classification framework.

Although separating the origin of these uncertainties can be useful for some applications (such as active learning), if our intention is just to infer how much we should trust a prediction, this division is irrelevant. In other words, we are interested in

knowing *when* a model is uncertain, and not *why*.

As described in Section 2.1, our intention in classification is to estimate $p(y|\boldsymbol{x})$, called *predictive uncertainty*. Ideally, $p_\theta(y|\boldsymbol{x})$, should reflect the model's total uncertainty.

2.3.1 Confidence estimation

As defined previously, Deep Learning Classifiers will have as output a layer $\boldsymbol{z}_L$ (from now one referred only as $\boldsymbol{z}$ or as the *logits* vector), corresponding to a vector where each component correspond to a class, and from which the prediction is obtained as the class with the maximum attributed value, i.e., $h(x) = \arg\max_{k\in\mathcal{Y}} z_k$. Besides the prediction, we want to estimate how confident is the model on this classification based on the values outputed. Based on the network construction and the training procedure (where the minimization of the Cross Entropy Loss is equivalent to maximizing the maximum value after the Softmax function), the most popular confidence estimator is arguably the *maximum softmax probability* (MSP) (DING et al., 2020), also known as *maximum class probability* (CORBIÈRE et al., 2022) or *softmax response* (GEIFMAN; EL-YANIV, 2017)

$$g(x) = \text{MSP}(\boldsymbol{z}) \triangleq \max_{k\in\mathcal{Y}} \sigma_k(\boldsymbol{z}) = \sigma_{\hat{y}}(\boldsymbol{z}) \tag{20}$$

where $\hat{y} = \arg\max_{k\in\mathcal{Y}} z_k$.

However, other functions of the logits can be considered. Some examples are the *softmax margin* (SM)(BELGHAZI; LOPEZ-PAZ, 2021; LUBRANO et al., 2023), the *max logit* (HENDRYCKS et al., 2022), the *logits margin* (LM) (STREETER, 2018; LEBOVITZ et al., 2023), the *negative entropy* (NE)[2] (BELGHAZI; LOPEZ-PAZ, 2021), defined, respectively, as

$$\text{SM}(\boldsymbol{z}) \triangleq \sigma_{\hat{y}}(\boldsymbol{z}) - \max_{k\in\mathcal{Y}:k\neq\hat{y}} \sigma_k(\boldsymbol{z}) \tag{21}$$

$$\text{MaxLogit}(\boldsymbol{z}) \triangleq z_{\hat{y}} \tag{22}$$

$$\text{LM}(\boldsymbol{z}) \triangleq z_{\hat{y}} - \max_{k\in\mathcal{Y}:k\neq\hat{y}} z_k \tag{23}$$

$$\text{NE}(\boldsymbol{z}) \triangleq \sum_{k\in\mathcal{Y}} \sigma_k(\boldsymbol{z}) \log \sigma_k(\boldsymbol{z}). \tag{24}$$

2.3.2 Calibration

All the presented training framework for classification was considering the classifier's outputs as probabilities distributions. Hence, we would like that these outputs indeed reflect the true probability of a class.

[2] Note that any uncertainty estimator can be used as a confidence estimator by taking its negative.

Consider a classifier $h\colon \mathcal{X} \to \mathcal{Y}$ and a confidence estimator $\pi\colon \mathcal{X} \to [0,1]$. We say that π is *calibrated* (GUO et al., 2017; GAWLIKOWSKI et al., 2022) if

$$P[h(x) = y \mid \pi(x) = p] = p, \quad \forall p \in [0,1], \quad (x,y) \sim P. \tag{25}$$

In practice, empirical measures of calibration are used, based on a test dataset $\{(x_i, y_i)\}_{i=1}^{N}$ drawn i.i.d. from P. The most popular one is arguably the *expected calibration error* (ECE) (NAEINI; COOPER; HAUSKRECHT, 2015), which is computed by grouping predictions into M equal-sized interval bins $B_m = \{i \in \{1,\dots,N\} : \pi(x_i) \in (\frac{m-1}{M}, \frac{m}{M}]\}$, $m = 1,\dots,M$, and then taking a weighted average of the difference between accuracy and confidence in each bin:

$$\mathrm{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{N} \left|\mathrm{acc}(B_m) - \mathrm{conf}(B_m)\right| \tag{26}$$

where $\mathrm{acc}(B_m) = \frac{1}{|B_m|}\sum_{i\in B_m} \mathbb{I}[h(x_i) = y_i]$ and $\mathrm{conf}(B_m) = \frac{1}{|B_m|}\sum_{i\in B_m} \pi(x_i)$.

2.3.3 Selective Classification

Let P be an unknown distribution over $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is the input space and $\mathcal{Y} = \{1,\dots,C\}$ is the label space, and C is the number of classes. A *classifier* is a prediction function $h : \mathcal{X} \to \mathcal{Y}$. The classifier's (true) *risk* is $R(h) = E_P[\ell(h(x), y)]$, where $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ is a given loss function, for instance, the 0/1 loss $\ell(\hat{y}, y) = \mathbb{I}[\hat{y} \neq y]$, where $\mathbb{I}[\cdot]$ denotes the indicator function.

A *selective classifier* (GEIFMAN; EL-YANIV, 2017) is a pair (h, g), where h is a classifier and $g : \mathcal{X} \to \mathbb{R}$ is a *confidence estimator* (also known as *confidence score function* or *confidence-rate function*), which quantifies the model's confidence on its prediction for a given input. For some fixed threshold t, given an input x, the selective model makes a prediction $h(x)$ if $g(x) \geq t$, otherwise it abstains from making a prediction. We say that x is *selected* in the former case and *rejected* in the latter. A selective model's *coverage* $\varphi(h, g) = P[g(x) \geq t]$ is the probability mass of the selected samples in $\mathcal{X}$, while its *selective risk* $R(h,g) = E_P[\ell(h(x), y) \mid g(x) \geq t]$ is its risk restricted to the selected samples. In particular, a model's risk equals its selective risk at *full coverage* (i.e., for t such that $\varphi(h, g) = 1$). These quantities can be evaluated empirically given a given a test dataset $\{(x_i, y_i)\}_{i=1}^{N}$ drawn i.i.d. from P, yielding the *empirical coverage* $\hat{\varphi}(h, g) = (1/N)\sum_{i=1}^{N} \mathbb{I}[g(x_i) \geq t]$ and the *empirical selective risk*

$$\hat{R}(h,g) = \frac{\sum_{i=1}^{N} \ell(h(x_i), y_i)\mathbb{I}[g(x_i) \geq t]}{\sum_{i=1}^{N} \mathbb{I}[g(x_i) \geq t]}. \tag{27}$$

Note that, by varying t, it is generally possible to trade off coverage for selective risk, i.e., a lower selective risk can usually (but not necessarily always) be achieved if

more samples are rejected. This tradeoff is captured by the *risk-coverage (RC) curve* (GEIFMAN; EL-YANIV, 2017), a plot of $\hat{R}(h,g)$ as a function of $\hat{\varphi}(h,g)$.

While the RC curve provides a full picture of the performance of a selective classifier, it is convenient to have a scalar metric that summarizes this curve. A commonly used metric is the *area under the RC curve* (AURC) (DING et al., 2020; GEIFMAN; UZIEL; EL-YANIV, 2019). However, when comparing selective models, if two RC curves cross, then each model may have a better selective performance than the other depending on the operating point chosen, which cannot be captured by the AURC. Another interesting metric, which forces the choice of an operating point, is the *selective accuracy constraint* (SAC) (GALIL; DABBAH; EL-YANIV, 2023), defined as the minimum coverage required for a model to achieve a specified accuracy.

2.3.4 Misclassification Detection

Misclassification detection (HENDRYCKS; GIMPEL, 2016), which refers to the problem of discriminating between correct and incorrect predictions made by a classifier, is closely related to selective classification. Both tasks rely on ranking predictions according to their confidence estimates, where correct predictions should be ideally separated from incorrect ones. More precisely, if $(x_1, y_1), (x_2, y_2) \in \mathcal{X} \times \mathcal{Y}$ are such that $\ell(h(x_1), y_1) > \ell(h(x_2), y_2)$, then we would like to have $g(x_1) < g(x_2)$, i.e., an optimal g orders samples in decreasing order of their losses. In the case of the 0/1 loss, a natural metric of ranking performance (GALIL; DABBAH; EL-YANIV, 2023) is the area under the ROC curve (AUROC) (FAWCETT, 2006) for misclassification detection. The ROC curve is the plot of the true positive rate (TPR) against the false positive rate (FPR), at various threshold settings. It can be shown that the AUROC can be calculated through the following formula:

$$AUROC = \frac{\sum_{i=1}^{n^+} \sum_{j=1}^{n^-} \mathbb{I}[g_i^+ > g_j^-]}{n^+ n^-} \tag{28}$$

where g^+ is the confidence of the samples where the prediction is correct, g^+ the confidence for the misclassification and n^+ and n^- are the number of samples for each case. From this equation, we can write AUROC as the probability that the model give a higher confidence for the correct classification than for the wrong ones:

$$AUROC = \mathrm{P}[g_1 > g_2 | \ell_1 < \ell_2]. \tag{29}$$

Note that this metric is blind to the classifier performance and focuses exclusively on the quality of the confidence estimates, i.e., given a fixed classifier h, different confidence estimators g can be compared in their ranking performance. Thus, misclassification detection can also be seen as a proxy problem on which to evaluate confidence estimators for selective classification. Indeed, (DING et al., 2020) propose the following

theorem: *For any two models A and B of the same accuracy and their uncertainties measured by arbitrary methods (which can be different for A and B), the curve of A dominates that of B in the ROC space, if and only if the curve of A dominates that of B in the Risk-Coverage space.* Thus, comparing in AUROC or AURC domain lead to similar results.

2.4 DO DNNS KNOW WHAT THEY DO NOT KNOW?

The Bayesian framework discussed in Section 2.3 reveals that conventional training of Deep Learning classifiers generally lacks explicit information regarding epistemic/model's uncertainty. Consequently, there remains theoretical indeterminancy as to whether and when $p(y|\mathrm{x}, \boldsymbol{\theta}^*)$ alone is sufficient or informative for uncertainty estimation.

When employing Maximum Likelihood Estimation for training, neural networks are typically trained to produce binary probabilities (1's or 0's) for the classes, often resulting in near-zero loss values. As a consequence, such training procedures can lead to an overconfident behavior in these systems. Additionally, theoretical findings indicate that architectures employing ReLU as an activation function are prone to becoming overconfident (HEIN; ANDRIUSHCHENKO; BITTERWOLF, 2019) for some regions out of training distribution. Moreover, certain inputs can be intentionally modified to achieve confident misclassifications, a technique known as adversarial attack (CHAKRABORTY et al., 2018).

To investigate whether Deep Learning classifiers can quantify their uncertainty in a proper way, multiple empirical results have been made over the last years. In (GUO et al., 2017), the authors show that modern architectures are poorly calibrated. Figure 7 shows the binned reliability diagram (as defined in Section 2.3.2) of a ResNet, considered a modern CNN, and a LeNet, considered a primitive CNN. As it can bee seen, while the probabilities predicted by the LeNet are calibrated, the ResNet returns values higher than the real rate of correct predictions.

Following (GUO et al., 2017) analysis, (MINDERER et al., 2021) investigated the calibration of *newer* Deep Learning classifier - more specifically, Visual Transformers. The conclusions are that these models are actually well calibrated. Furthermore, several recent works have pointed to Visual Transformers as being more robust and better at uncertainty estimation than Convolutional Neural Networks (BHOJANAPALLI et al., 2021).

The comparisons between different classification models were extended by (GALIL; DABBAH; EL-YANIV, 2023), where the authors compared 523 classifiers of ImageNet in the context of calibration and selective classification. The main conclusions are:

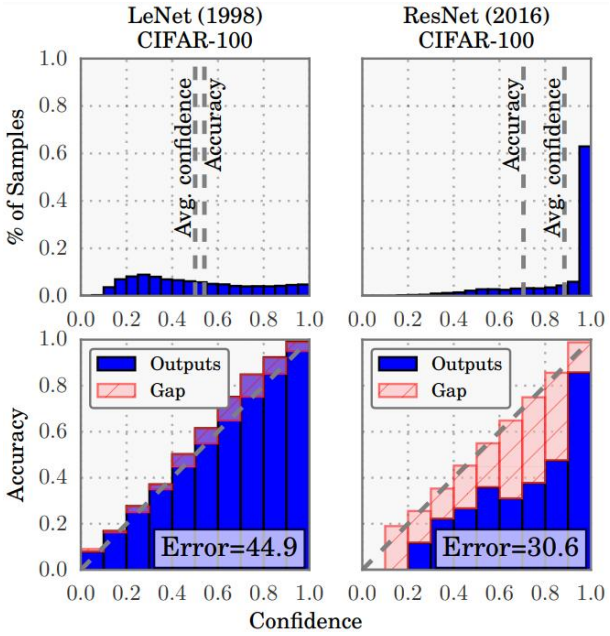
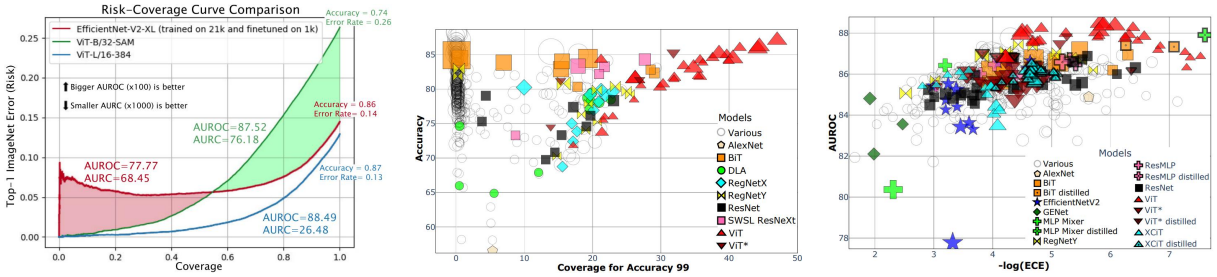


Figure 7 – Calibration plots of a LeNet and a ResNet (GUO et al., 2017)



(a) Comparison between the RC curve of different 3 models.

(b) Relation between total accuracy and SAC for accuracy 99%

(c) Relation between selective classification and calibration.

Figure 8 – Figures from (GALIL; DABBAH; EL-YANIV, 2023) comparing different ImageNet classifiers in terms of uncertainty estimation.

- Some models are innately best at selective classification;
- The correlations between ECE and the accuracy or the number of model parameters are nearly zero;
- The best model in terms of AUROC or SAC is not always the best in terms of calibration;
- Certain architectures are more inclined to perform better or worse at uncertainty estimation, and the correlations between AUROC, ECE, accuracy and the number of parameters are dependent on the architecture analyzed.

The first item is well exemplified by Figures 8a and 8b. As it can be seen, some models can have best accuracy but worst uncertainty estimations, resulting in worst performance for low coverages. Figure 8c is directly related to the last item. As it can be seen, there is some negative correlation between AUROC and ECE.

The trade-off between the model structure and size with the uncertainty estimation capacity have been reported for other works as well (DING et al., 2020). Furthermore, it remains unknown the exact reasons of why some models are better or worse in these tasks than others.

2.5 IMPROVING UNCERTAINTY ESTIMATION

As noticed, many Deep Learning models are bad in estimating uncertainty, both for selective classification and calibration. Hence, multiple methods to mitigate these issues have been proposed.

Maybe the most popular way to attack uncertainty estimation is with the use of ensemble. Although these methods have shown good results in increasing robustness, recent results indicates that the best selective mechanism in ensembles is to use their probabilities response (CATTELAN; SILVA, 2022). Indeed, Deep Ensemble (LAKSHMINARAYANAN; PRITZEL; BLUNDELL, 2017), usually considered as the state of the art of uncertainty estimation, can be shown to achieve the same performance in estimating confidence than simply a larger network (ABE et al., 2022; XIA; BOUGANIS, 2022).

In order to mitigate possible overconfidence due to the Cross Entropy Loss, some works aim to develop alternative training recipes (ZHU et al., 2022) and/or loss functions (MUKHOTI et al., 2020; WANG; FENG, Lei; ZHANG, M.-L., 2021). In fact, some of these losses achieve success in calibrating the neural network; however, (WANG; FENG, Lei; ZHANG, M.-L., 2021) show that these approaches can reduce the *potential for improvement* of the models , i.e., when comparing the models with simple post-hoc optimizations (such as Temperature Scaling), using alternative losses can lead to worst results. Thus, the authors propose the concept of *calibratable*, i.e., instead of comparing models by their calibration, one should compare it by *how calibrated* can it be after simple post-hoc techniques. In practice, it follows (ASHUKHA et al., 2020) methodology: “Comparison [...] should only be performed at the optimal temperature.”

For selective classification, one possible approach is to train a reject option in parallel with the classifier (ZHANG, X.-Y. et al., 2023). Some works attempt to train from scratch both *heads* together (GEIFMAN; EL-YANIV, 2019; DEVRIES; TAYLOR, 2018). However, recent results have shown that, for these networks, the best selective mechanism is to use the classifier outputs and discard the trained g (FENG, Leo et al., 2023). Thus, this approach becomes simply a new training loss approach.

As in general a model is trained to maximize its prediction performance, we would like to, after training, train separately the uncertainty estimator (or enhance the performance of the model’s one) without harming the classifier’s accuracy. This can be done using what we call *post-hoc* methods. One line of work is to develop a MLP exclusively for the selective mechanism (CORBIÈRE et al., 2022). Other approaches apply simple and efficient optimizations in order to enhance the selective classification

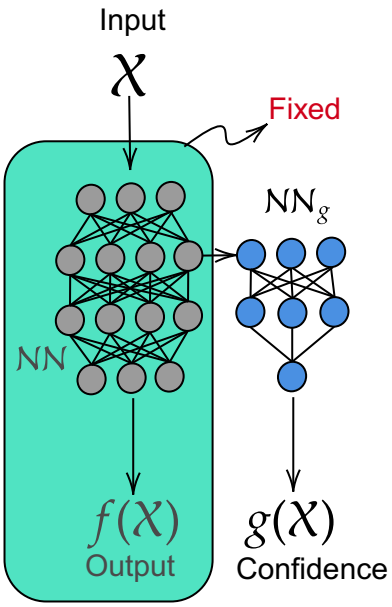


Figure 9 – Training an uncertainty estimator as a post-processing

performance. The most popular post-hoc method is called *Temperature Scaling*.

Figure 9 shows an illustration of this approach. Note that, although the figure shows a MLP confidence estimator connected to a hidden layer of the classifier, the connection and the architecture are arbitrary; indeed, in this work the only approach is to consider methods with few parameters using only the logits vector.

2.5.1 Temperature Scaling

Temperature scaling (TS) (GUO et al., 2017) is a post-processing method that consists in, for a fixed trained classifier, transforming the logits as $\mathbf{z}' = \mathbf{z}/T$, before applying the softmax function. The parameter T, called the temperature, is then optimized over a hold-out dataset $\{(x_i, y_i)\}_{i=1}^{N}$ (not used during training of the classifier). An important property of this method is that it does not change the model's predictions. The conventional way of applying TS, as proposed in (GUO et al., 2017) for calibration and referred to here as *standard* TS, consists in optimizing T with respect to the negative log-likelihood (NLL) (MURPHY, 2022)

$$\mathcal{L} = -\sum_{i=1}^{N} \log\left((\sigma(\boldsymbol{z}_i/T))_{\hat{y}}\right) \tag{30}$$

where $\boldsymbol{z}_i = f(x_i)$.

2.5.2 Sample Dependent Temperature Scaling

The Adaptive Temperature Scaling (ATS) method consists in, instead of a constant temperature T, use a different temperature for each sample:

$$\boldsymbol{z}' = \frac{\boldsymbol{z}}{T(\boldsymbol{z})}. \tag{31}$$

With that, $T(z)$ can be defined as a MLP $T_\theta(z)$ and trained as a post-hoc method (JOY et al., 2023). Alternatively, (A. BALANYA; RAMOS; MAROÑAS, 2023) propose to use analytical functions in terms of the logits directly.

2.5.3 Logits Normalization

Normalizing logits for training was first seen in (KORNBLITH et al., 2021), but was proposed as a method to mitigate overconfidence in (WEI et al., 2022a). (WEI et al., 2022a) argued that, as training progresses, a model will tend to become overconfident on correctly classified training samples by increasing $\|\boldsymbol{z}\|_2$, a phenomenon that they confirmed experimentally. The authors, thus, propose to mitigate the issue of overconfidence by normalizing (using L_2 norm) the logits during the training, before applying the Softmax function (and calculating the Cross Entropy Loss).

The reasoning in (WEI et al., 2022a) is that the predicted class depends only on $\tilde{\boldsymbol{z}} = \boldsymbol{z}/\|\boldsymbol{z}\|_2$, but the training loss on correctly classified training samples can still be decreased by increasing $\|\boldsymbol{z}\|_2$ while keeping $\tilde{\boldsymbol{z}}$ fixed. Thus, the model would become overconfident on those samples, since increasing $\|\boldsymbol{z}\|_2$ also increases the confidence (as measured by MSP) of the predicted class. Indeed, (WEI et al., 2022a) verified experimentally that the average magnitude of logits (and therefore also the average 2-norm) tends to increase during training. Thus, the authors propose to redefine the Cross Entropy Loss as:

$$\ell_{LN} = -log\frac{e^{z_y/(\tau\|\boldsymbol{z}\|_2)}}{\sum_{i=1}^{C} e^{z_i/(\tau\|\boldsymbol{z}\|_2)}}$$

where τ is an hyperparameter equivalent to the temperature. With this, the results indicates considerable gain in OOD detection, which refers to the task of identifying samples for neither of the possible classes (ZOU et al., 2023). It is worth noting that, in inference time, the authors do not normalize the logits - i.e., the traditional MSP is used. Additionally, the method allows better calibration after Temperature Scaling.

(JIANG; GU; PAN, 2023) proposed a similar idea to (WEI et al., 2022a), although the goals of their work had no relation to uncertainty estimation. They propose a normalization to the Softmax function in order to get better training:

$$NormSoftmax(\boldsymbol{z}) = \sigma\left(\frac{\sqrt{C}\boldsymbol{z}}{\|\boldsymbol{z} - \mu(\boldsymbol{z})\|_2}\right)$$

where $\mu(\boldsymbol{z})$ is the logits mean over all classes. In (WEI et al., 2022b), the authors propose to clip the Lp-norm of the logits by adding a constraint in the optimization

function. Although the proposed idea use the norm order as a hyperparameter, the authors just present results for the L1 and L2 norms.

3 PROPOSED METHODS

Based on the relative success of post-hoc methods for calibration and OOD detection, in this chapter we propose to apply post-hoc optimization seeking selective classification performance directly.

3.1 TEMPERATURE SCALING FOR SELECTIVE CLASSIFICATION

As presented in Section 2.5.1, the standard TS consists in optimizing the temperature T with respect to the NLL loss in a hold-out dataset. However, although this method can improve misclassification detection for most cases, it can be harmful (GALIL; DABBAH; EL-YANIV, 2023; ZHU et al., 2022). Thus, we propose to optimize selective classification metrics directly, instead of NLL. Since a single parameter needs to be tuned, this can easily be done via grid search. From now on, we will refer to the standard TS as TS-NLL, while TS-AURC indicates optimizing the temperature with respect to the AURC and TS-AUROC to the AUROC.

Figure 10 shows how the behavior of different metrics as a function of the temperature T for a ViT-H-4 (DOSOVITSKIY et al., 2021) model evaluated in ImageNet. In this case, optimizing NLL can lead to better, but not optimal, selective classification performance, measured in terms of AURC and AUROC. Also, it can be seen that optimizing ECE does not necessarily help, illustrating our point that these two problems should be treated independently. In Figure 11, it can be seen an example where the Standard Temperature Scaling can harm selective prediction performance. Finally, it can be seen that AURC and AUROC exhibit practically identical behavior with temperature, suggesting that they are equally good to be used as the objective.

3.2 LOGITS NORMALIZATION

First, we note that, following (WEI et al., 2022a) argument that the norms are responsible for overconfidence, normalizing the logits can be used as a post-hoc process, although being proposed originally as a training routine adaptation.

Additionally, we remark that (WEI et al., 2022a) argument holds unchanged for any L_p normalization, as nothing in their analysis requires $p = 2$. On the other hand, note that the argument in (WEI et al., 2022a) discussed above becomes more compelling when the logit vector $\boldsymbol{z}$ has zero mean across components. After all, adding an (input-dependent) constant to $\boldsymbol{z}$ has no impact on the training loss but affects the p-norm. This fact has been recognized by (JIANG; GU; PAN, 2023) in their discussion of LogitNorm. Thus, we apply centralization to the logits.

In conclusion, (WEI et al., 2022a) and (JIANG; GU; PAN, 2023) logits normalization can be adapted to using the following sequence of transformations on the logits:

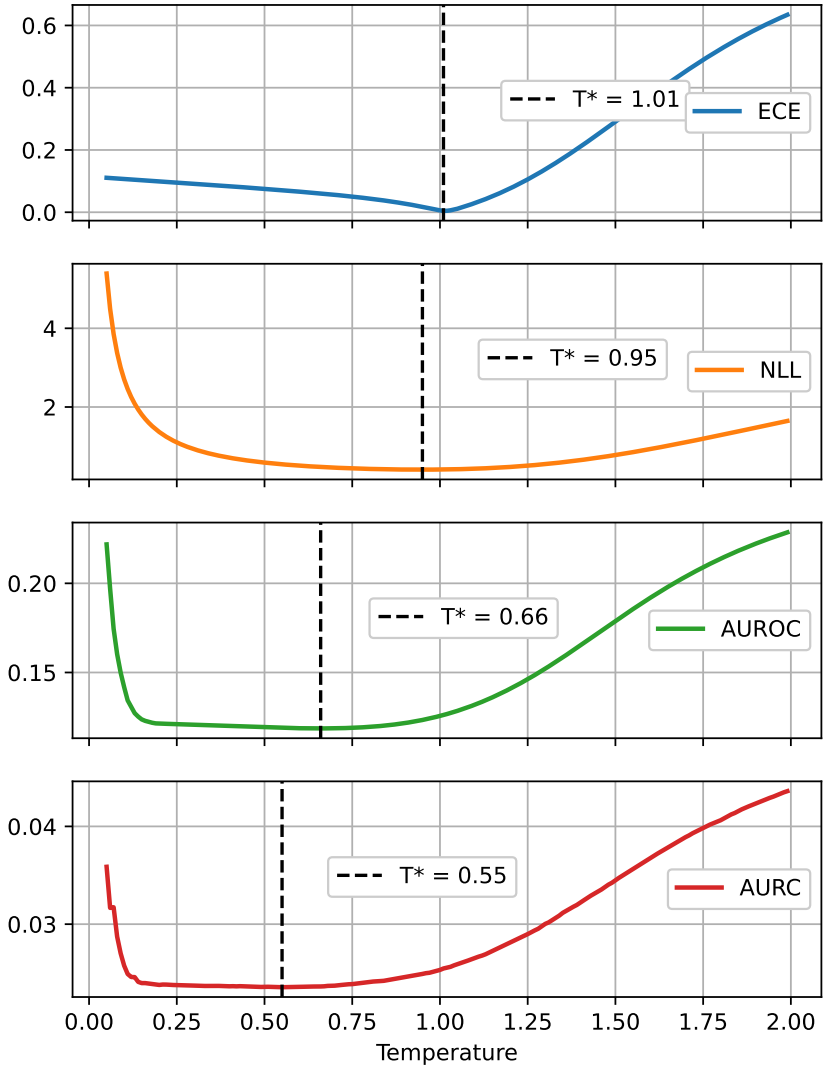


Figure 10 – Variation of metrics with the Temperature).

centralization ($\boldsymbol{z} \leftarrow \boldsymbol{z} - \mu(\boldsymbol{z})$), p-normalization ($\boldsymbol{z} \leftarrow \boldsymbol{z}/\|\boldsymbol{z}\|_p$) and temperature scaling ($\boldsymbol{z} \leftarrow \beta\boldsymbol{z}$). From that, we propose to use the MSP as a confidence estimator:

$$\boldsymbol{z}' = \beta \frac{(\boldsymbol{z} - \mu(\boldsymbol{z}))}{\|\boldsymbol{z} - \mu(\boldsymbol{z})\|_p}, \quad g(\mathrm{x}) = MSP(\boldsymbol{z}') \tag{32}$$

where

$$\mu(\boldsymbol{z}) \triangleq (z_1 + \cdots + z_C)/C \tag{33}$$

and

$$\|\boldsymbol{z}\|_p \triangleq (|z_1|^p + \cdots + |z_C|^p)^{1/p} \tag{34}$$

and, thus, p can be optimized in a hold-out set similarly to temperature scaling method. Note that, in (WEI et al., 2022a), β (in their work defined through $\tau = \beta^{-1}$) is defined as a hyperparameter. Gathering with the idea of temperature scaling, we propose to optimize β and p together.

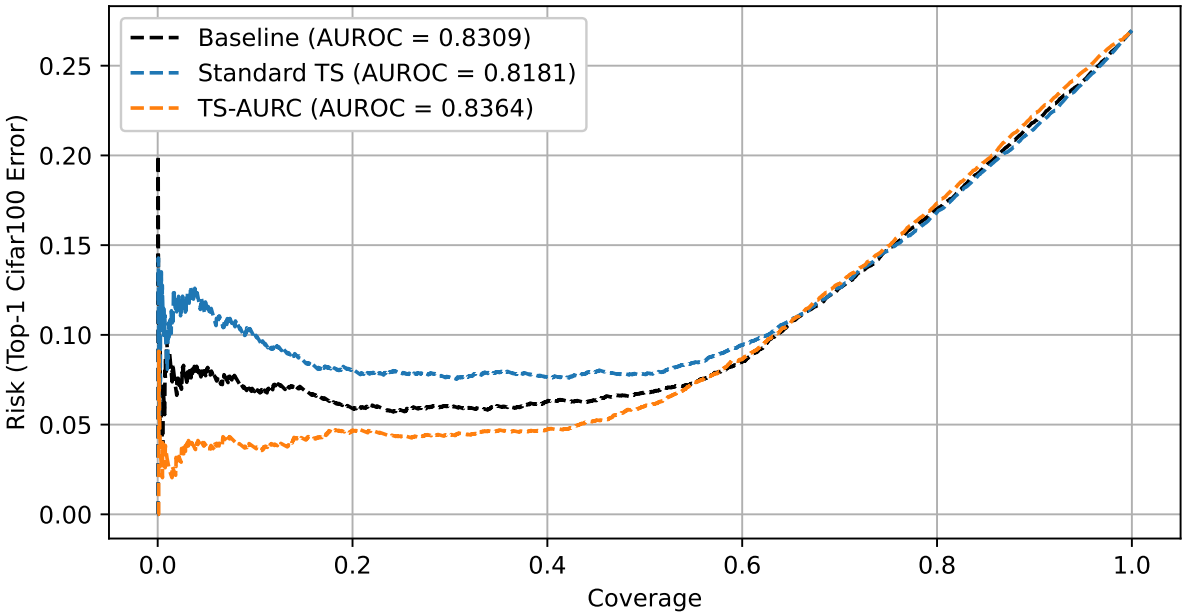


Figure 11 – RC curves for Temperature Scaling - VGG-19 for Cifar100.

As will be more detailed in the following section, some models appear to achieve considerable gains in selective prediction when this operation is applied. However, for other models, this operation can harm selective classification performance. Thus, p must be determined for each model and, moreover, to ensure that our method can never cause harm, we augment the definition of p-norm with

$$\|\boldsymbol{z}\|_{\emptyset} = 1 \tag{35}$$

so that the allowed range for p is $\mathbb{R} \cup \{\emptyset\}$.

The hyperparameters p and β are optimized (e.g., via grid search) based on a hold-out set $\{(x_i, y_i)\}_{i=1}^{N}$, using directly the AURC (or the AUROC) as the objective. In practice, the logits $\boldsymbol{z}_i = f(x_i)$ of all hold-out samples can be pre-computed and stored, so that any metric based on them can be computed very quickly.

Note that, as p approaches infinity, $||\boldsymbol{z}||_p \rightarrow max(\boldsymbol{z})$. Indeed, it tends to converge reasonable quickly. Thus, the grid search on p can be made only for small p. In our experiments, we noticed that it suffices to evaluate a few values of p, such as $p \in \{\emptyset, 2, 3, 4, 5, 6\}$.

3.2.1 An heuristic for β

One relevant fact about the L_p norms of the logits is that they in general assume values considerably higher than 1 - Figure 12 shows a histogram for the L_p norms of a ResNet-50 for ImageNet validation set. However, as a sample dependent temperature, we would expect it to increase the confidence on correct predictions and to decrease it for misclassifications. With that in mind, we propose to use as a reference point for β the expected value of the norms (given p):

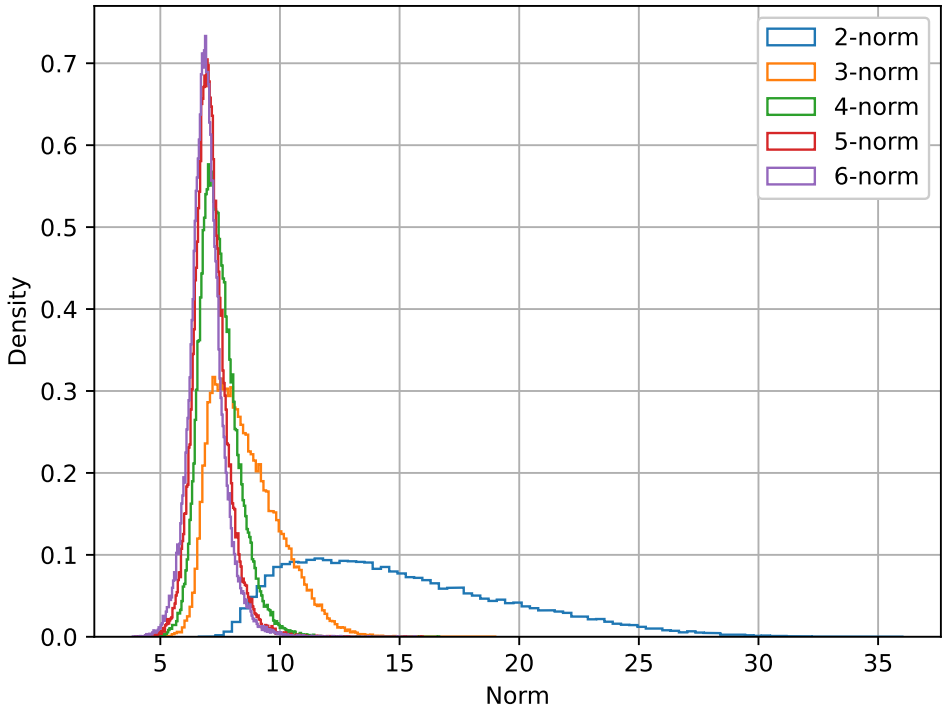


Figure 12 – Histogram of the p-norms of logits for different p.

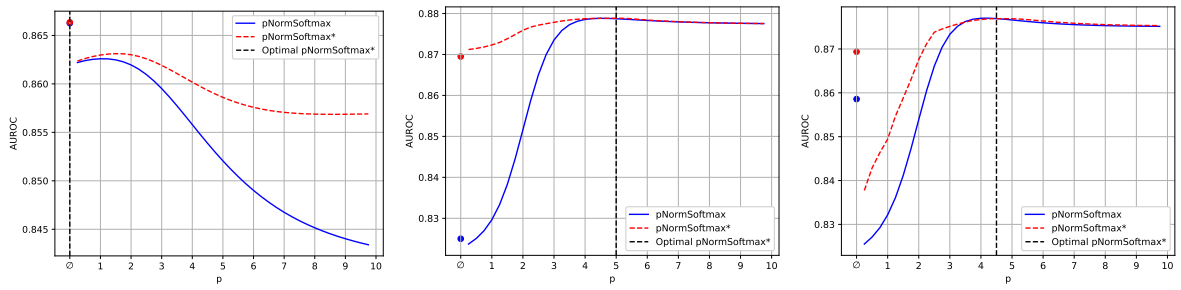
$$\beta = \frac{1}{N} \sum_{i=1}^{N} \|\boldsymbol{z}_i\|_p. \tag{36}$$

Note that this implies $E[T(\boldsymbol{z})] = 1$ and $\|\boldsymbol{z}'\|_p = \beta = E[\|\boldsymbol{z}\|_p]$. This can be interpreted as trying to change the logits as little as possible, since most of the logits will have their temperature approximately unchanged.

Fortunately, we note that, when utilizing this methodology, the selective performance is already improved, reaching higher gains than temperature scaling. Indeed, as will be shown and discussed in the next chapter, optimizing p and choosing β as in Equation (36) leads to results *almost as good* as optimizing p and β jointly. Moreover, optimizing p solely leads to a considerable faster method, since we can consider only a few values in a grid search.

Our proposed method with the heuristic choice of β is named p-NormSoftmax, while the method with full optimization of β is denoted p-NormSoftmax*.

Figure 13 brings examples of the efficacy of the heuristic. Note that, although optimizing β can significantly improve the results for a considerable part of p, the heuristic is just nearly optimal for the optimal p (note that, as we will be optimizing p, this is the only value that will interest us).



(a) VGG16. (b) ConvNext Large. (c) MaxVit-T.

Figure 13 – Comparison of AUROC between p-NormSoftmax and p-NormSoftmax* for different values of p. Note that, for part (a), the optimum is obtained for $p = \emptyset$. In this case, p-NormSoftmax reduces to the baseline, while p-NormSoftmax* reduces to TS.

4 RESULTS AND DISCUSSION

All the experiments regarding the proposed method and the subsequent investigations were conducted using the open-source library PyTorch (PASZKE et al., 2019) and all of its provided pre-trained classifiers on ImageNet (DENG et al., 2009). Additionally, some models of the (WIGHTMAN, 2019) repository were utilized, particularly the ones highlighted by (GALIL; DABBAH; EL-YANIV, 2023). The list of all models and results is presented in Appendix A. In total, 84 ImageNet models were used for experiments. The validation set of ImageNet was randomly split into 5000 hold-out images for post-hoc optimization and 45000 for tests and comparisons. Investigations on the stability of this split are presented in Section 4.5. All codes and experiments can be found in `https://github.com/lfpc/pNormSoftmax`.

Additionally, experiments were conducted on CIFAR-100 dataset in order to testify the efficiency of the method for more datasets. These results are presented separately in the last section.

4.1 COMPARISON OF METHODS

In Figure 14, we shown an example of the RC curves of the proposed methods for a ResNext101-32x8d (XIE et al., 2017) for ImageNet. It can be seen that, while standard TS (TS optimizing NLL) outperforms the baseline, optimizing the AURC directly (TS-AURC) achieves better results. Moreover, it can be noted that *p*-NormSoftmax leads to even better performance which practically identical to that of *p*-NormSoftmax*.

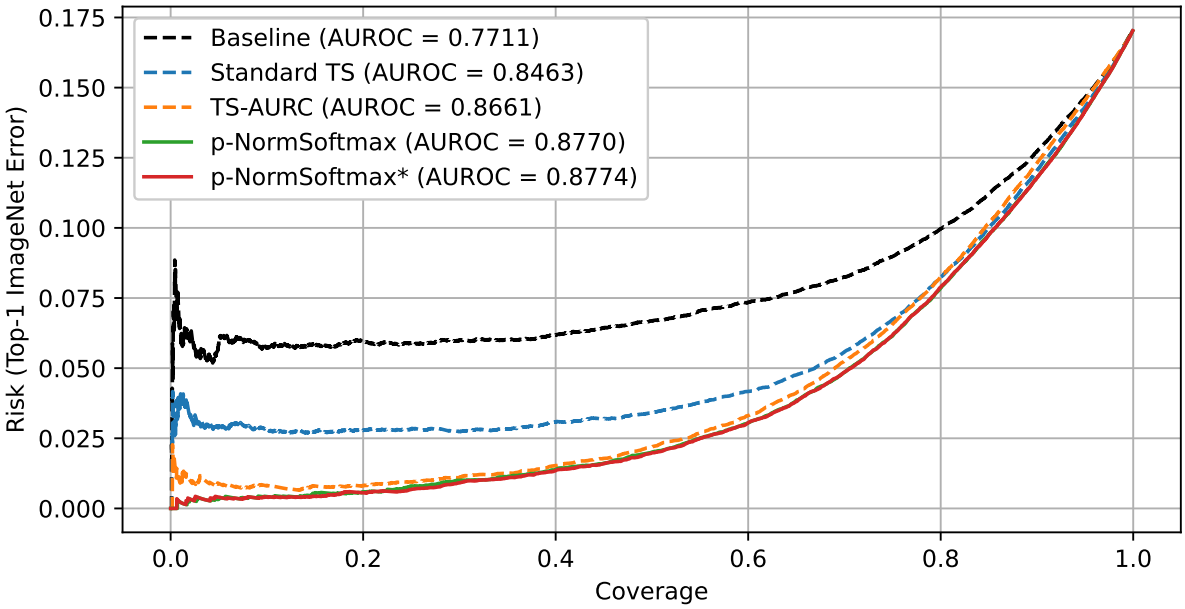


Figure 14 – RC curves of the proposed methods, the baseline and standard TS for a ResNext101-32x8d

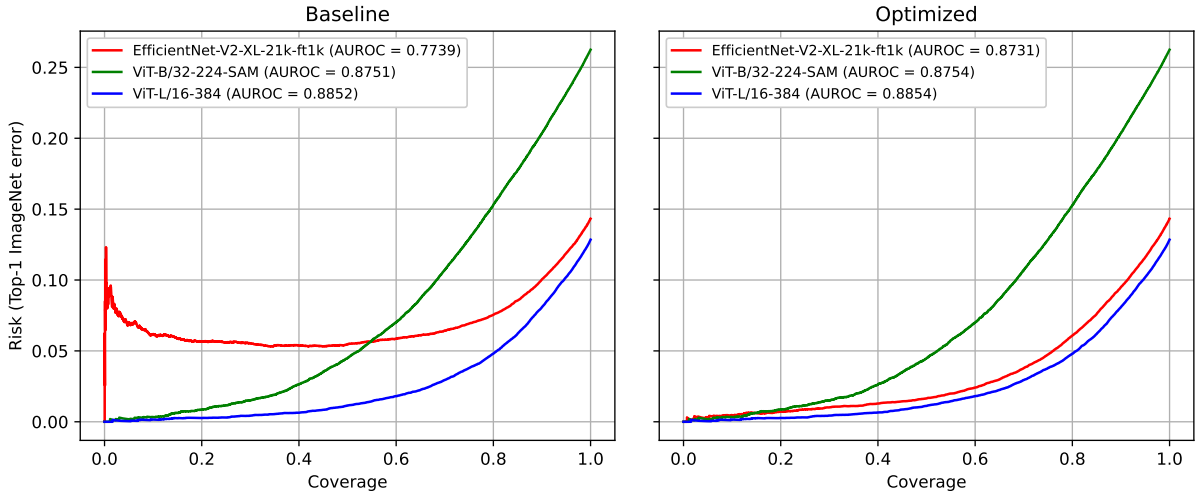


Figure 15 – A comparison of RC curves made by three models selected in (GALIL; DABBAH; EL-YANIV, 2023), including examples of highest (ViT-L/16-384) and lowest (EfficientNet-V2-XL) AUROC. After the application of our post-hoc method, the apparent pathology in EfficientNet-V2-XL completely disappears, resulting in significantly improved selective classification performance.

Table 1 – AUROC and AURC average gains for all considered ImageNet classifiers. AURC gains are calculated as the reduction of AURC in relation to the baseline. For both, the higher the better.

	AURC [%]		AUROC [x100]	
Method	Mean	Max	Mean	Max
TS-AURC	12.65	44.32	1.7	9.32
p-NormSoftmax	15.9	48.46	2.61	10.60
p-NormSoftmax*	16.02	48.52	2.63	10.65

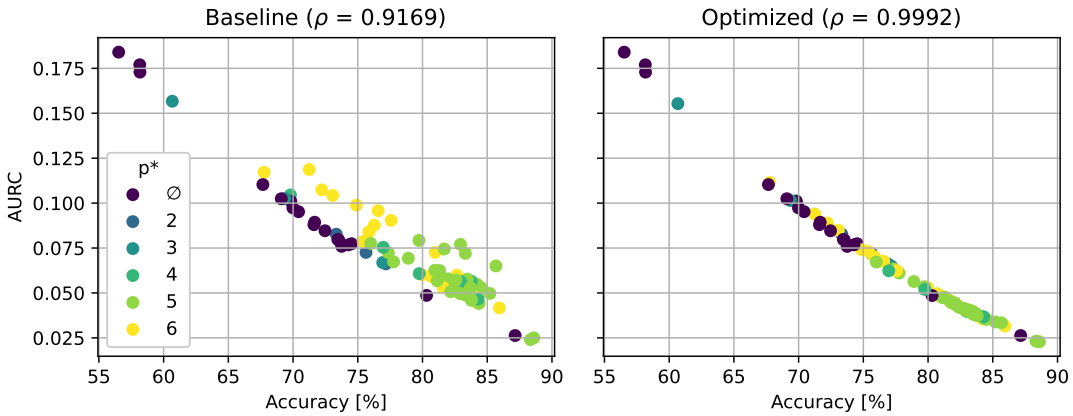
Indeed, the conclusion that full optimization of β is unnecessary when using the proposed heuristic (while always optimizing p) was observed for all considered models. The average AUROC gain of β optimization with respect to the heuristic is 0.0002, while the maximum value across all the analyzed models is 0.0019. These gains are imperceptible in the RC curve and may be considered negligible. The results for all the evaluated models are summarized in Table 1 and presented with more details in Appendix A.

4.2 COMPARISON OF MODELS

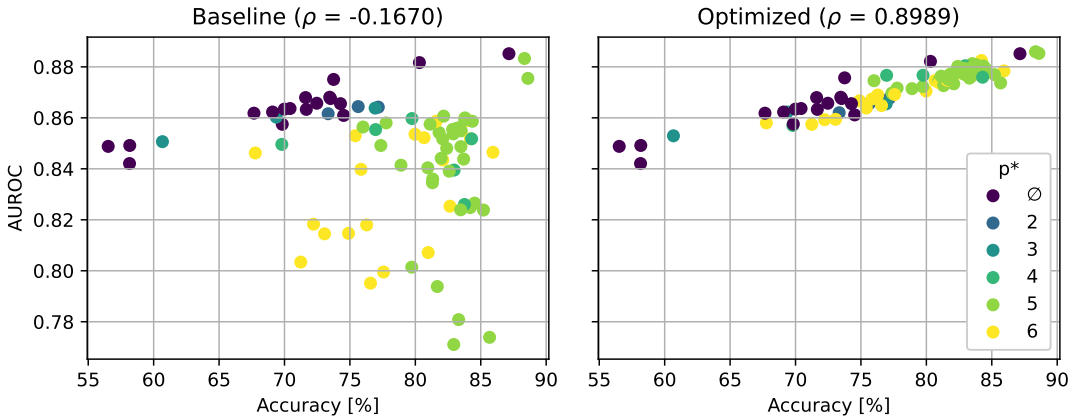
(GALIL; DABBAH; EL-YANIV, 2023) showed that some models are much better than others in selective classification. An example is shown in the left figure of Figure 15. Although the EfficientNet v2 XL (TAN; LE, 2021) has better accuracy than the ViTB/32 SAM (CHEN; HSIEH; GONG, 2022), the latter is better in identifying misclassification

and, thus, in most of the RC curve. However, the figure in the right shows that, after *p*-NormSoftmax optimization, the ViTs have negligible gain, while the EfficientNet has a huge one, hence becoming better in the RC curve than the ViTB/32 SAM.

In Figures 16a and 16b, the AURC and AUROC are presented with respect to the accuracy for each model. It can be seen that, while for the baseline there are models with higher accuracy but worse (higher) AURC than others, this does not happen after the models are optimized with *p*-NormSoftmax. The Spearman's correlation between the AURC and the accuracy goes from 0.9169 to 0.9992, indicating that, while this is not the case for the baseline, the selective classification performance of the optimized models is almost entirely determined by its accuracy at full coverage.



(a) AURC



(b) AUROC

Figure 16 – AURC and AUROC of all ImageNet models with respect to their accuracy. ρ is the Spearman's correlation between the metric and the corresponding accuracy and the color indicates the valued of p that optimizes each model.

We can also observe that, after the optimization, the AUROC for all models lies within the range $[0.8421, 0.8859]$. This small range suggests that all models are at roughly the same level of misclassification detection, although we can still see some dependency on accuracy (better predictive models are slightly better at predicting their own failures).

4.3 COMPARING UNCERTAINTY MEASURES

The *p*-NormSoftmax method is proposed as the MSP of the transformed logits $\boldsymbol{z}' = \beta \frac{\boldsymbol{z}-\mu(\boldsymbol{z})}{\|\boldsymbol{z}-\mu(\boldsymbol{z})\|_p}$. However, instead of the MSP, this transformed logit vector can in principle be combined with any confidence estimator that takes logits as input, mainly the one proposed in Section 4.3. Then, the hyperparameters of the resulting estimator (p and, if necessary, β) can be optimized for the desired metric. When our (optimized) logit transformation is combined with the confidence estimator X, the resulting method is denoted as *p*-Norm-X, e.g., *p*-Norm-MSP is the *p*-NormSoftmax.

Tables 2 and 3 show that, while the MSP can be surpassed by other methods over the baseline, it still provides the best results after our logit transformation and optimization. These results are obtained by averaging the AURC or AUROC across the 84 ImageNet classifiers evaluated.

Table 2 – Mean AURC (x1000) values for different confidence estimators (lower is better; **bold** indicates the best result of each row)

	Confidence estimator (X)				
Method	MSP	NE	SM	MaxLogit	LM
Baseline	73.46	87.74	70.50	107.27	**66.85**
TS-AURC	64.73	**64.47**	65.36	107.27	66.85
p-Norm-X	**63.22**	64.89	63.86	64.70	65.65
p-Norm-X*	**63.09**	63.77	63.79	64.70	65.65

Table 3 – Mean AUROC (x100) values for different uncertainty measures (higher is better; **bold** indicates the best result of each row)

	Confidence estimator (X)				
Method	MSP	NE	SM	MaxLogit	LM
Baseline	84.52	79.86	85.26	76.61	**85.68**
TS-AURC	86.63	**86.75**	86.31	76.61	85.68
p-Norm-X	**87.13**	86.57	86.82	86.87	86.15
p-Norm-X	**87.16**	86.93	86.85	86.87	86.15

4.4 ABLATION FOR THE CENTRALIZATION STEP

The first step of *p*-NormSoftmax method involves centralization of the logits. In Table 4 we present numerical results justifying this choice. Temperature scaling is not considered, since centralization does not change the confidence value in this case. It is important to emphasize that, for most of the models evaluated, the logits have virtually zero means, i.e., the logits are already almost centralized, in which case centralization

Table 4 – AUROC and AURC gains for ImageNet. AURC gains are calculated as the reduction of AURC in relation to the baseline. For both, the higher the better.

	AURC [%]		AUROC [x100]	
Method	Mean	Max	Mean	Max
p-NormSoftmax (no centralization)	15.73	48.73	2.59	10.60
p-NormSoftmax* (no centralization)	15.74	48.98	2.60	10.65
p-NormSoftmax	15.90	48.46	2.61	10.60
p-NormSoftmax*	16.02	48.52	2.63	10.65
p-NormSoftmax (optional centralization)	15.90	48.73	2.61	10.60
p-NormSoftmax* (optional centralization)	16.02	48.98	2.63	10.65

cannot help. However, some models have their logits with comparatively large means. For these cases, centralization can lead to better results (the most significant is MaxVit-T, where it reaches 3.64 percentage points in additional AURC gain for *p*-NormSoftmax). We also noticed that, for a few models, centralization slightly degraded performance (the highest degradation was observed in EfficientNet-V2-XL-21k-ft1k, exactly the model for which our method was most beneficial, corresponding to the Max column in Table 4).

On average, centralization appears to be positive and consequently it is used as part of the proposed method. Using optional centralization as a hyperparameter (i.e., only when it improves performance) provided only negligible gains so we did not include this possibility.

4.5 DATA EFFICIENCY

As mentioned, the experiments conducted in ImageNet used a hold-out dataset of 5,000 images randomly sampled from the validation dataset, resulting in 45,000 images reserved for the test phase.

One important aspect of post-hoc methods is its data efficiency (ZHANG, J.; KAILKHURA; HAN, 2020), i.e., the efficiency of the method in learning with few data. The primary aim was to investigate the data efficiency of the methods, which indicates their capacity to learn and generalize from limited data. To accomplish this, the optimization process was executed multiple times, utilizing different fractions of the hold-out set while keeping the test set fixed at 45,000 samples. Consequently, two distinct types of random splits were implemented using the validation dataset. The first involved dividing the validation set into hold-out and test sets, while the second involved sampling fractions from the hold-out set. To ensure the findings were generalizable and robust, both of these random split procedures were repeated five times each, culminating in a total of 25 experiments for each analyzed fraction of the hold-out set.

Figure 17 displays the outcomes of these studies for an WideResNet50-2 trained on ImageNet. As observed, *p*-NormSoftmax demonstrates exceptional data efficiency,

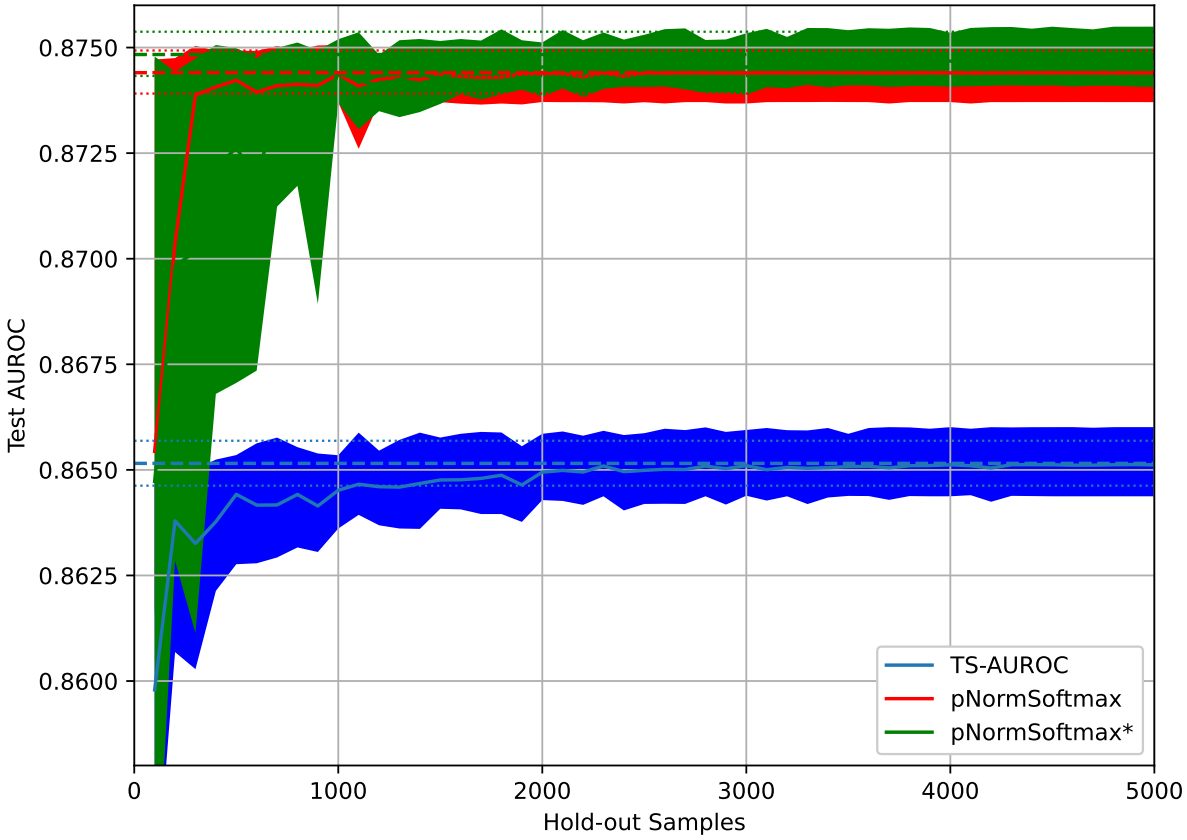


Figure 17 – Sample complexity curves: Average AUROC variation with number of hold-out samples used, for a WideResNet50-2. Dashed lines represent the optimal AUROC for each method, i.e., the achieved value when the optimization is made directly on the test set. Highlighted regions (as well as the dotted lines) for each curve correspond to percentiles 10 and 90.

reaching its maximum value with fewer than 2,000 samples.

4.6 ROBUSTNESS TO DISTRIBUTION SHIFT

Up to this point, the presented results have been evaluated utilizing the validation set of ImageNet. Generally, this set is considered to have a data distribution similar to that of the training set. However, a reliable model must also be robust for dataset shifts (OVADIA et al., 2019). For evaluating a model's performance under data shift, we evaluate our methods on ImageNet-C (HENDRYCKS; DIETTERICH, 2018), which consists in 15 different corruptions of the ImageNet's validation set. We follow the standard approach for evaluating robustness with this dataset, which is to use it only for inference; thus, the post-hoc methods are optimized using only the 5000 hold-out images from uncorrupted ImageNet validation dataset.

Generally, classifiers lose accuracy in the presence of data shift. Hence, we use SAC as performance metric, with the target accuracy chosen as the accuracy of the model on ImageNet validation data at full coverage. Table 5 shows these results

when *p*-NormSoftmax is applied to a ResNet-50 (HE et al., 2016). We can see that *p*-NormSoftmax enhances the model's performance in selective classification under data shift at all corruption levels.

Table 5 – *p*-NormSoftmax applied to a ResNet-50 under dataset shift. The target accuracy is the one achieved for corruption level 0 (i.e., 80.86%).

		Corruption level					
	Method	0	1	2	3	4	5
Accuracy [%]	-	80.86	68.56	60.03	51.85	39.44	27.09
Coverage (SAC) [%]	Baseline	100	75.97	56.79	41.43	21.65	9.09
	TS-AURC	100	77.13	60.51	45.49	27.41	13.32
	p-NormSoftmax	100	78.49	62.35	47.63	29.59	15.62
	p-NormSoftmax*	100	78.52	62.39	47.76	29.67	15.66

4.7 RESULTS ON CIFAR-100

The hold-out set for CIFAR-100, consisting of 5000 samples, was taken from the training set before training. All models were forked from `github.com/kuangliu/pytorch-cifar`, and adapted for CIFAR-100 (KRIZHEVSKY, 2009). All of them were trained for 200 epochs with Cross Entropy Loss, using a SGD optimizer with initial learning rate of 0.1 and a Cosine Annealing learning rate schedule with period 200. Moreover, a weight decay of 0.0005 and a Nesterov's momentum of 0.9 were used. Data transformations were applied, specifically standardization, random crop (for size 32x32 with padding 4) and random horizontal flip.

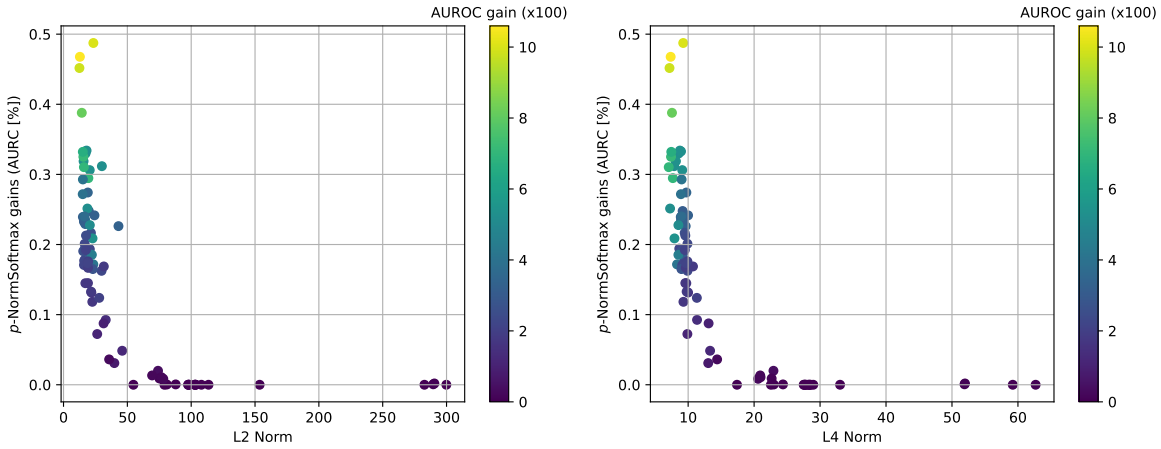
Table 6 summarizes the gains of the proposed methods on CIFAR-100 for all the evaluated models, and Appendix A brings the results for all of them. It can be seen that, while for fewer models, the same conclusions are taken in this dataset, with *p*-NormSoftmax achieving high gains for some models.

Table 6 – Average AUROC and AURC gains for CIFAR-100. AURC gains are calculated as the reduction of AURC in relation to the baseline. For both, the higher the better.

	AURC [%]		AUROC [x100]	
Method	Mean	Max	Mean	Max
TS-AURC	2.34	9.62	0.25	1.16
p-NormSoftmax	4.55	24.87	0.56	2.92
p-NormSoftmax*	5.10	25.26	0.67	3.06

4.8 WHEN—AND WHY—IS *p*-NORMSOFTMAX BENEFICIAL?

In this section we investigate in which circumstances *p*-NormSoftmax yields high gains. This is analogous to ask when a model's baseline is already optimal (within the *p*-normalization framework). In Figure 18 it is possible to see a relation between the gains and the average norms (L2 and L4) of the logits of each models. It is straightforward to see the relation; models with high norms tend to have already a good baseline, while models with low norms have poor baselines and tend to achieve high gains when normalized. Indeed, this relation between high and low norms appear to have clear threshold for both norms.



(a) L2 Norm (b) L4 Norm

Figure 18 – Gains of *p*-NormSoftmax (with optimal *p*) versus the mean of the logit norms for each model. Colors represent the AUROC gain ($\times$100).

5 CONCLUSION

In this thesis, we considered the problem of selective classification for deep neural networks. In order to improve the selective mechanism for a given trained model, we proposed *p*-NormSoftmax, a post-hoc method for enhancing misclassification detection of neural network classifiers. Our method achieves an improvement in AURC of 16% on average when compared to the baseline for the evaluated classifiers trained on ImageNet, reaching almost 50% for some specific models.

Furthermore, our analysis revealed that, after implementing *p*-NormSoftmax, the models exhibited similar levels of misclassification performance. This finding results in a model's selective classification performance being almost completely determined by its accuracy at full coverage, and suggests that the previous observations regarding different performance between models' selective performance are mostly due to the use of sub-optimal confidence estimators. Additionally, *p*-NormSoftmax exhibit impressive data efficiency, due to the fact that a single parameter needs to be tuned. Moreover, the method achieves satisfactory gains for selective classification under data shift. It is also worth mentioning that our method is compatible with classifiers constructed directly for improving confidence estimation, including ensembles, specific architectures and models with specific training routines.

Finally, we point out some possible reasons and initial investigations on why and in which circumstances *p*-NormSoftmax achieves gains. For future work, we intend to explore more deeply why post-hoc normalization can lead to improved selective mechanisms and to evaluate our method on different tasks.

REFERÊNCIAS

A. BALANYA, Sergio; RAMOS, Daniel; MAROÑAS, Juan. **Adaptive Temperature Scaling for Robust Calibration of Deep Neural Networks**. en. Rochester, NY: [s.n.], Mar. 2023. Available from: https://papers.ssrn.com/abstract=4379258. Visited on: 17 May 2023.

ABDAR, Moloud et al. A Review of Uncertainty Quantification in Deep Learning: Techniques, Applications and Challenges. **Information Fusion**, v. 76, p. 243–297, Dec. 2021. arXiv:2011.06225 [cs]. ISSN 15662535.

ABE, Taiga; BUCHANAN, Estefany Kelly; PLEISS, Geoff; ZEMEL, Richard; CUNNINGHAM, John P. Deep Ensembles Work, But Are They Necessary? en. **Advances in Neural Information Processing Systems**, v. 35, p. 33646–33660, Dec. 2022.

ASHUKHA, Arsenii; MOLCHANOV, Dmitry; LYZHOV, Alexander; VETROV, Dmitry. PITFALLS OF IN-DOMAIN UNCERTAINTY ESTIMATION AND ENSEMBLING IN DEEP LEARNING. en, 2020.

AYHAN, M.; BERENS, Philipp. Test-time Data Augmentation for Estimation of Heteroscedastic Aleatoric Uncertainty in Deep Neural Networks. In.

BELGHAZI, Mohamed Ishmael; LOPEZ-PAZ, David. **What classifiers know what they don't?** [S.l.]: arXiv, July 2021. arXiv:2107.06217 [cs]. Available from: http://arxiv.org/abs/2107.06217. Visited on: 17 May 2023.

BHOJANAPALLI, Srinadh; CHAKRABARTI, Ayan; GLASNER, Daniel; LI, Daliang; UNTERTHINER, Thomas; VEIT, Andreas. Understanding robustness of transformers for image classification. In: PROCEEDINGS of the IEEE/CVF international conference on computer vision. [S.l.: s.n.], 2021. P. 10231–10241.

BLUM, Hermann; SARLIN, Paul-Edouard; NIETO, Juan; SIEGWART, Roland; CADENA, Cesar. The fishyscapes benchmark: Measuring blind spots in semantic segmentation. **International Journal of Computer Vision**, Springer, v. 129, p. 3119–3135, 2021.

CATTELAN, Luís Felipe P.; SILVA, Danilo. On the performance of uncertainty estimation methods for deep-learning based image classification models. pt. In: ANAIS

do Encontro Nacional de Inteligência Artificial e Computacional (ENIAC). [S.l.]: SBC, Nov. 2022. P. 532–543. ISSN: 2763-9061.

CHAKRABORTY, Anirban; ALAM, Manaar; DEY, Vishal; CHATTOPADHYAY, Anupam; MUKHOPADHYAY, Debdeep. Adversarial attacks and defences: A survey. **arXiv preprint arXiv:1810.00069**, 2018.

CHEN, Xiangning; HSIEH, Cho-Jui; GONG, Boqing. When Vision Transformers Outperform ResNets without Pre-training or Strong Data Augmentations. en. In.

CLARTÉ, Lucas; LOUREIRO, Bruno; KRZAKALA, Florent; ZDEBOROVÁ, Lenka. **Expectation consistency for calibration of neural networks**. [S.l.]: arXiv, Mar. 2023. arXiv:2303.02644 [cs, stat]. Available from: `http://arxiv.org/abs/2303.02644`. Visited on: 8 Mar. 2023.

CORBIÈRE, Charles; THOME, Nicolas; SAPORTA, Antoine; VU, Tuan-Hung; CORD, Matthieu; PÉREZ, Patrick. Confidence Estimation via Auxiliary Models. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 44, n. 10, p. 6043–6055, Oct. 2022. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence. ISSN 1939-3539.

DENG, Jia; DONG, Wei; SOCHER, Richard; LI, Li-Jia; LI, Kai; FEI-FEI, Li. ImageNet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. [S.l.: s.n.], June 2009. P. 248–255. ISSN: 1063-6919.

DEVRIES, Terrance; TAYLOR, Graham W. Learning confidence for out-of-distribution detection in neural networks. **arXiv preprint arXiv:1802.04865**, 2018.

DING, Yukun; LIU, Jinglan; XIONG, Jinjun; SHI, Yiyu. Revisiting the Evaluation of Uncertainty Estimation and Its Application to Explore Model Complexity-Uncertainty Trade-Off. en. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). Seattle, WA, USA: IEEE, June 2020. P. 22–31.

DOSOVITSKIY, Alexey et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. en. In.

DOSOVITSKIY, Alexey et al. An image is worth 16x16 words: Transformers for image recognition at scale. **arXiv preprint arXiv:2010.11929**, 2020.

FAWCETT, Tom. An introduction to ROC analysis. en. **Pattern Recognition Letters**, v. 27, n. 8, p. 861–874, June 2006. ISSN 0167-8655.

FENG, Leo; AHMED, Mohamed Osama; HAJIMIRSADEGHI, Hossein; ABDI, Amir H. Towards Better Selective Classification. en. In.

FUKUSHIMA, Kunihiko. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. **Biological cybernetics**, Springer, v. 36, n. 4, p. 193–202, 1980.

GAL, Yarin; GHAHRAMANI, Zoubin. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. en. In: PROCEEDINGS of The 33rd International Conference on Machine Learning. [S.l.]: PMLR, June 2016. P. 1050–1059. ISSN: 1938-7228.

GALIL, Ido; DABBAH, Mohammed; EL-YANIV, Ran. What Can we Learn From The Selective Prediction And Uncertainty Estimation Performance Of 523 Imagenet Classifiers? en. In.

GAWLIKOWSKI, Jakob et al. **A Survey of Uncertainty in Deep Neural Networks**. [S.l.]: arXiv, Jan. 2022. arXiv:2107.03342 [cs, stat]. Available from: http://arxiv.org/abs/2107.03342. Visited on: 17 May 2023.

GEIFMAN, Yonatan; UZIEL, Guy; EL-YANIV, Ran. **Bias-Reduced Uncertainty Estimation for Deep Neural Classifiers**. [S.l.]: arXiv, Apr. 2019. arXiv:1805.08206 [cs, stat]. Available from: http://arxiv.org/abs/1805.08206. Visited on: 10 Mar. 2023.

GEIFMAN, Yonatan; EL-YANIV, Ran. **Selective Classification for Deep Neural Networks**. [S.l.]: arXiv, June 2017. arXiv:1705.08500 [cs]. Available from: http://arxiv.org/abs/1705.08500. Visited on: 17 May 2023.

GEIFMAN, Yonatan; EL-YANIV, Ran. SelectiveNet: A Deep Neural Network with an Integrated Reject Option. en. In: PROCEEDINGS of the 36th International Conference on Machine Learning. [S.l.]: PMLR, May 2019. P. 2151–2159. ISSN: 2640-3498.

GONSIOR, Julius; FALKENBERG, Christian; MAGINO, Silvio; REUSCH, Anja; THIELE, Maik; LEHNER, Wolfgang. **To Softmax, or not to Softmax: that is the question when applying Active Learning for Transformer Models**. [S.l.]: arXiv, Oct.

2022. arXiv:2210.03005 [cs]. Available from: http://arxiv.org/abs/2210.03005. Visited on: 17 May 2023.

GOODFELLOW, Ian J.; SHLENS, Jonathon; SZEGEDY, Christian. Explaining and Harnessing Adversarial Examples. **CoRR**, Dec. 2014.

GUO, Chuan; PLEISS, Geoff; SUN, Yu; WEINBERGER, Kilian Q. On Calibration of Modern Neural Networks. en. In: PROCEEDINGS of the 34th International Conference on Machine Learning. [S.l.]: PMLR, July 2017. P. 1321–1330. ISSN: 2640-3498.

HE, Kaiming; ZHANG, Xiangyu; REN, Shaoqing; SUN, Jian. Deep Residual Learning for Image Recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). [S.l.: s.n.], June 2016. P. 770–778. ISSN: 1063-6919.

HEIN, Matthias; ANDRIUSHCHENKO, Maksym; BITTERWOLF, Julian. Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem. In: PROCEEDINGS of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. [S.l.: s.n.], 2019. P. 41–50.

HENDRICKX, Kilian; PERINI, Lorenzo; PLAS, Dries Van der; MEERT, Wannes; DAVIS, Jesse. Machine Learning with a Reject Option: A survey. **ArXiv**, abs/2107.11277, 2021.

HENDRYCKS, Dan; BASART, Steven; MAZEIKA, Mantas; ZOU, Andy; KWON, Joseph; MOSTAJABI, Mohammadreza; STEINHARDT, Jacob; SONG, Dawn. Scaling Out-of-Distribution Detection for Real-World Settings. In: PROCEEDINGS of the 39th International Conference on Machine Learning. [S.l.]: PMLR, June 2022. P. 8759–8773.

HENDRYCKS, Dan; DIETTERICH, Thomas. Benchmarking Neural Network Robustness to Common Corruptions and Perturbations. en. In.

HENDRYCKS, Dan; GIMPEL, Kevin. A baseline for detecting misclassified and out-of-distribution examples in neural networks. **arXiv preprint arXiv:1610.02136**, 2016.

HENDRYCKS, Dan; ZHAO, Kevin; BASART, Steven; STEINHARDT, Jacob; SONG, Dawn. Natural Adversarial Examples. **CVPR**, 2021.

HUANG, Lang; ZHANG, Chao; ZHANG, Hongyang. Self-Adaptive Training: beyond Empirical Risk Minimization. In: ADVANCES in Neural Information Processing Systems. [S.l.]: Curran Associates, Inc., 2020. v. 33, p. 19365–19376.

JIANG, Zixuan; GU, Jiaqi; PAN, David Z. NormSoftmax: Normalize the Input of Softmax to Accelerate and Stabilize Training. en, Feb. 2023.

JOY, Tom; PINTO, Francesco; LIM, Ser-Nam; TORR, Philip HS; DOKANIA, Puneet K. Sample-dependent adaptive temperature scaling for improved calibration. In: 12. PROCEEDINGS of the AAAI Conference on Artificial Intelligence. [S.l.: s.n.], 2023. v. 37, p. 14919–14926.

KARANDIKAR, Archit; CAIN, Nicholas; TRAN, Dustin; LAKSHMINARAYANAN, Balaji; SHLENS, Jonathon; MOZER, Michael Curtis; ROELOFS, Rebecca. Soft Calibration Objectives for Neural Networks. en. In.

KORNBLITH, Simon; CHEN, Ting; LEE, Honglak; NOROUZI, Mohammad. Why Do Better Loss Functions Lead to Less Transferable Features? In: ADVANCES in Neural Information Processing Systems. [S.l.]: Curran Associates, Inc., 2021. v. 34, p. 28648–28662.

KRIZHEVSKY, Alex. Learning Multiple Layers of Features from Tiny Images. en, 2009.

LAKSHMINARAYANAN, Balaji; PRITZEL, Alexander; BLUNDELL, Charles. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. In: ADVANCES in Neural Information Processing Systems. [S.l.]: Curran Associates, Inc., 2017. v. 30.

LEBOVITZ, Luzian; CAVIGELLI, Lukas; MAGNO, Michele; MULLER, Lorenz K. Efficient Inference With Model Cascades. **Transactions on Machine Learning Research**, May 2023. ISSN 2835-8856.

LECUN, Yann; BENGIO, Yoshua, et al. Convolutional networks for images, speech, and time series. **The handbook of brain theory and neural networks**, Citeseer, v. 3361, n. 10, p. 1995, 1995.

LIANG, Shiyu; LI, Yixuan; SRIKANT, R. Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks. en. In.

LIU, Weitang; WANG, Xiaoyun; OWENS, John; LI, Yixuan. Energy-based out-of-distribution detection. **Advances in neural information processing systems**, v. 33, p. 21464–21475, 2020.

LIU, Ziyin; WANG, Zhikang; LIANG, Paul Pu; SALAKHUTDINOV, Russ R; MORENCY, Louis-Philippe; UEDA, Masahito. Deep gamblers: Learning to abstain with portfolio theory. **Advances in Neural Information Processing Systems**, v. 32, 2019.

LUBRANO, Mélanie; BELLAHSEN-HARRAR, Yaëlle; FICK, Rutger; BADOUAL, Cécile; WALTER, Thomas. Simple and Efficient Confidence Score for Grading Whole Slide Images. **arXiv preprint arXiv:2303.04604**, 2023.

MINDERER, Matthias; DJOLONGA, Josip; ROMIJNDERS, Rob; HUBIS, Frances; ZHAI, Xiaohua; HOULSBY, Neil; TRAN, Dustin; LUCIC, Mario. Revisiting the calibration of modern neural networks. **Advances in Neural Information Processing Systems**, v. 34, p. 15682–15694, 2021.

MINSKY, Marvin; PAPERT, Seymour. An introduction to computational geometry. **Cambridge tiass., HIT**, v. 479, n. 480, p. 104, 1969.

MOON, Jooyoung; KIM, Jihyo; SHIN, Younghak; HWANG, Sangheum. Confidence-Aware Learning for Deep Neural Networks. en. In: PROCEEDINGS of the 37th International Conference on Machine Learning. [S.l.]: PMLR, Nov. 2020. P. 7034–7044. ISSN: 2640-3498.

MUKHOTI, Jishnu; KULHARIA, Viveka; SANYAL, Amartya; GOLODETZ, Stuart; TORR, Philip; DOKANIA, Puneet. Calibrating Deep Neural Networks using Focal Loss. In: ADVANCES in Neural Information Processing Systems. [S.l.]: Curran Associates, Inc., 2020. v. 33, p. 15288–15299.

MURPHY, Kevin P. **Probabilistic Machine Learning: An Introduction**. [S.l.]: MIT Press, 2022.

NAEINI, Mahdi Pakdaman; COOPER, Gregory F.; HAUSKRECHT, Milos. Obtaining Well Calibrated Probabilities Using Bayesian Binning. **Proceedings of the ... AAAI Conference on Artificial Intelligence. AAAI Conference on Artificial Intelligence**, v. 2015, p. 2901–2907, Jan. 2015. ISSN 2159-5399.

NEUMANN, Lukas; ZISSERMAN, Andrew; VEDALDI, Andrea. Relaxed softmax: Efficient confidence auto-calibration for safe pedestrian detection. OpenReview, 2018.

OVADIA, Yaniv; FERTIG, Emily; REN, Jie; NADO, Zachary; SCULLEY, D.; NOWOZIN, Sebastian; DILLON, Joshua; LAKSHMINARAYANAN, Balaji; SNOEK, Jasper. Can you trust your model' s uncertainty? Evaluating predictive uncertainty under dataset shift. In: ADVANCES in Neural Information Processing Systems. [S.l.]: Curran Associates, Inc., 2019. v. 32.

PASZKE, Adam et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In: ADVANCES in Neural Information Processing Systems. [S.l.]: Curran Associates, Inc., 2019. v. 32.

RAHIMI, Amir; MENSINK, Thomas; GUPTA, Kartik; AJANTHAN, Thalaiyasingam; SMINCHISESCU, Cristian; HARTLEY, Richard. **Post-hoc Calibration of Neural Networks by g-Layers**. [S.l.]: arXiv, Feb. 2022. arXiv:2006.12807 [cs, stat]. Available from: `http://arxiv.org/abs/2006.12807`. Visited on: 17 May 2023.

ROSENBLATT, Frank. The perceptron: a probabilistic model for information storage and organization in the brain. **Psychological review**, American Psychological Association, v. 65, n. 6, p. 386, 1958.

SHEN, Maohao; BU, Yuheng; SATTIGERI, Prasanna; GHOSH, Soumya; DAS, Subhro; WORNELL, Gregory. **Post-hoc Uncertainty Learning using a Dirichlet Meta-Model**. [S.l.]: arXiv, Dec. 2022. arXiv:2212.07359 [cs]. Available from: `http://arxiv.org/abs/2212.07359`. Visited on: 17 May 2023.

STREETER, Matthew. Approximation Algorithms for Cascading Prediction Models. In: PROCEEDINGS of the 35th International Conference on Machine Learning. [S.l.]: PMLR, July 2018. P. 4752–4760.

TAN, Mingxing; LE, Quoc. EfficientNetV2: Smaller Models and Faster Training. en. In: PROCEEDINGS of the 38th International Conference on Machine Learning. [S.l.]: PMLR, July 2021. P. 10096–10106. ISSN: 2640-3498.

TEYE, Mattias; AZIZPOUR, Hossein; SMITH, Kevin. Bayesian Uncertainty Estimation for Batch Normalized Deep Networks. en. In: PROCEEDINGS of the 35th International Conference on Machine Learning. [S.l.]: PMLR, July 2018. P. 4907–4916. ISSN: 2640-3498.

TOMANI, Christian; CREMERS, Daniel; BUETTNER, Florian. **Parameterized Temperature Scaling for Boosting the Expressive Power in Post-Hoc Uncertainty Calibration**. [S.l.]: arXiv, Sept. 2022. arXiv:2102.12182 [cs]. Available from: http://arxiv.org/abs/2102.12182. Visited on: 16 Mar. 2023.

VASWANI, Ashish; SHAZEER, Noam; PARMAR, Niki; USZKOREIT, Jakob; JONES, Llion; GOMEZ, Aidan N; KAISER, Łukasz; POLOSUKHIN, Illia. Attention is all you need. **Advances in neural information processing systems**, v. 30, 2017.

WANG, Deng-Bao; FENG, Lei; ZHANG, Min-Ling. Rethinking Calibration of Deep Neural Networks: Do Not Be Afraid of Overconfidence. In: ADVANCES in Neural Information Processing Systems. [S.l.]: Curran Associates, Inc., 2021. v. 34, p. 11809–11820.

WEI, Hongxin; XIE, Renchunzi; CHENG, Hao; FENG, Lei; AN, Bo; LI, Yixuan. Mitigating Neural Network Overconfidence with Logit Normalization. en. In: PROCEEDINGS of the 39th International Conference on Machine Learning. [S.l.]: PMLR, June 2022. P. 23631–23644. ISSN: 2640-3498.

WEI, Hongxin; ZHUANG, Huiping; XIE, Renchunzi; FENG, Lei; NIU, Gang; AN, Bo; LI, Yixuan. Logit clipping for robust learning against label noise. **arXiv preprint arXiv:2212.04055**, 2022.

WIGHTMAN, Ross. **Pytorch Image Model**. en. [S.l.: s.n.], 2019. Available from: https://github.com/huggingface/pytorch-image-models. Visited on: 30 Apr. 2023.

XIA, Guoxuan; BOUGANIS, Christos-Savvas. **On the Usefulness of Deep Ensemble Diversity for Out-of-Distribution Detection**. [S.l.]: arXiv, Sept. 2022. arXiv:2207.07517 [cs]. Available from: http://arxiv.org/abs/2207.07517. Visited on: 28 Apr. 2023.

XIE, Saining; GIRSHICK, Ross; DOLLAR, Piotr; TU, Zhuowen; HE, Kaiming. Aggregated Residual Transformations for Deep Neural Networks. en. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Honolulu, HI: IEEE, July 2017. P. 5987–5995.

ZHANG, Jize; KAILKHURA, Bhavya; HAN, T. Yong-Jin. Mix-n-Match : Ensemble and Compositional Methods for Uncertainty Calibration in Deep Learning. en. In:

PROCEEDINGS of the 37th International Conference on Machine Learning. [S.l.]: PMLR, Nov. 2020. P. 11117–11128. ISSN: 2640-3498.

ZHANG, Xu-Yao; XIE, Guo-Sen; LI, Xiuli; MEI, Tao; LIU, Cheng-Lin. A Survey on Learning to Reject. **Proceedings of the IEEE**, v. 111, n. 2, p. 185–215, Feb. 2023. Conference Name: Proceedings of the IEEE. ISSN 1558-2256.

ZHU, Fei; CHENG, Zhen; ZHANG, Xu-Yao; LIU, Cheng-Lin. Rethinking Confidence Calibration for Failure Prediction. In: AVIDAN, Shai; BROSTOW, Gabriel; CISSÉ, Moustapha; FARINELLA, Giovanni Maria; HASSNER, Tal (Eds.). **Computer Vision – ECCV 2022**. Cham: Springer Nature Switzerland, 2022. v. 13685. Series Title: Lecture Notes in Computer Science. P. 518–536. ISBN 978-3-031-19805-2 978-3-031-19806-9.

ZOU, Ke; CHEN, Zhihao; YUAN, Xuedong; SHEN, Xiaojing; WANG, Meng; FU, Huazhu. **A Review of Uncertainty Estimation and its Application in Medical Imaging**. [S.l.]: arXiv, Feb. 2023. arXiv:2302.08119 [cs, eess]. Available from: `http://arxiv.org/abs/2302.08119`. Visited on: 14 Mar. 2023.

Appendix

APPENDIX A – RESULTS

Table 7 – Results for all models evaluated on ImageNet

			Baseline		Temperature Scaling		*p*-NormSoftmax		*p*-NormSoftmax*	
Model	Accuracy	Optimal *p*	AURC	AUROC	AURC	AUROC	AURC	AUROC	AURC	AUROC
vit_h_14	88.60	5	25.02	87.55	23.17	88.36	22.83	88.53	22.83	88.53
regnet_y_128gf	88.34	5	24.05	88.33	23.26	88.54	23.18	88.59	23.18	88.59
vit_l_16_384	87.15	0	26.30	88.52	26.27	88.55	26.30	88.52	26.27	88.55
efficientnet_v2_l	85.93	6	41.64	84.65	32.93	87.35	31.58	87.84	31.56	87.85
efficientnetv2_xl.in21k_ft_in1k†	85.68	5	65.04	77.39	35.82	86.10	33.34	87.37	33.21	87.43
efficientnet_v2_m	85.21	5	49.72	82.38	35.46	86.98	33.87	87.69	33.78	87.72
convnext_large	84.53	5	52.94	82.65	37.15	87.12	35.32	87.92	35.24	87.96
efficientnet_v2_s	84.36	5	44.14	85.87	36.85	87.61	35.57	88.03	35.51	88.04
swin_v2_b	84.30	4	46.62	85.18	38.58	86.92	36.70	87.59	36.67	87.62
efficientnet_b7	84.21	6	47.22	85.18	36.76	87.82	35.52	88.25	35.49	88.27
convnext_base	84.20	5	54.93	82.48	38.93	86.85	36.81	87.66	36.75	87.67
efficientnet_b6	84.08	6	45.13	85.89	37.52	87.75	36.40	88.09	36.34	88.11
swin_v2_s	83.79	5	46.26	86.00	38.62	87.60	37.37	88.09	37.27	88.12
maxvit_t	83.77	5	45.93	85.92	39.76	87.16	38.18	87.73	38.10	87.76
convnext_small	83.75	4	56.59	82.60	39.79	87.02	37.69	87.87	37.62	87.89
swin_b	83.69	5	52.96	84.38	40.22	86.96	38.44	87.78	38.25	87.83
efficientnet_b5	83.51	5	48.74	85.49	39.89	87.58	38.18	88.13	38.08	88.13
regnet_y_32gf	83.48	5	49.84	84.87	39.83	87.30	38.18	87.99	38.16	88.00
efficientnet_b4	83.46	5	57.68	82.39	40.62	87.30	39.70	87.63	39.43	87.74
resnext101_64x4d	83.31	5	72.04	78.08	41.23	87.15	39.50	87.88	39.26	87.95

swin_s	83.30	5	48.83	85.61	42.10	86.97	40.24	87.56	40.03	87.61
regnet_x_32gf	83.08	5	50.04	85.57	41.68	87.42	39.98	87.89	39.85	87.92
regnet_y_16gf	82.96	4	56.25	83.95	41.83	87.29	39.78	88.04	39.78	88.05
resnext101_32x8d	82.94	5	77.04	77.11	43.32	86.75	41.01	87.70	40.75	87.76
regnet_y_8gf	82.90	5	49.72	85.37	41.72	87.48	40.25	87.94	40.20	87.96
regnet_x_16gf	82.85	5	49.92	85.54	42.28	87.41	41.11	87.73	41.02	87.77
convnext_tiny	82.65	6	59.88	82.53	43.77	86.97	41.55	87.82	41.36	87.89
wide_resnet101_2	82.59	5	57.25	83.91	43.93	87.13	41.69	87.87	41.54	87.91
resnet152	82.40	5	55.21	84.80	43.62	87.53	42.00	88.02	41.81	88.06
swin_v2_t	82.17	5	50.70	86.07	44.57	87.39	43.35	87.76	43.30	87.77
regnet_y_3_2gf	82.09	5	53.52	85.17	45.73	87.02	44.38	87.33	44.37	87.33
efficientnet_b3	82.09	6	57.11	84.35	45.68	87.17	44.19	87.64	43.99	87.67
resnet101	81.98	5	57.85	84.41	46.26	87.08	44.41	87.61	44.33	87.64
regnet_x_8gf	81.82	5	54.16	85.42	46.05	87.33	44.52	87.71	44.44	87.74
wide_resnet50_2	81.68	5	74.46	79.38	47.59	86.83	45.58	87.51	45.38	87.57
swin_t	81.59	6	53.07	85.86	47.14	87.13	46.09	87.48	46.03	87.48
resnext50_32x4d	81.32	5	62.44	83.60	49.21	86.75	47.45	87.30	47.17	87.34
regnet_x_3_2gf	81.31	5	62.20	83.45	49.72	86.73	47.55	87.26	47.43	87.28
vit_b_16	81.13	5	56.80	85.75	49.44	87.20	47.33	87.63	47.16	87.69
resnet50	80.98	6	72.56	80.71	49.85	86.95	48.46	87.38	48.12	87.47
regnet_y_1_6gf	80.95	5	62.61	84.04	50.39	86.94	48.26	87.45	48.19	87.45
efficientnet_b2	80.67	6	59.27	85.22	51.93	86.95	49.64	87.46	49.54	87.47
vit_base_patch16_224.sam†	80.31	0	48.64	88.17	48.34	88.24	48.55	88.22	48.34	88.24
efficientnet_b1	79.97	6	60.11	85.36	53.71	86.91	53.00	87.05	52.81	87.09
vit_l_16	79.74	4	60.77	85.97	54.73	86.85	51.97	87.67	51.87	87.71

regnet_x_1_6gf	79.72	5	79.24	80.14	55.38	86.76	53.48	87.22	53.31	87.32
regnet_y_800mf	78.90	5	69.29	84.14	58.68	86.62	56.36	87.14	56.36	87.14
efficientnet_b0	77.77	5	67.31	85.80	62.73	86.80	61.08	87.17	61.08	87.17
regnet_x_800mf	77.57	6	90.39	79.95	64.84	86.33	62.34	86.91	62.23	86.93
inception_v3	77.36	5	72.38	84.91	66.06	86.34	63.41	86.97	63.35	86.99
densenet161	77.17	2	66.19	86.42	66.08	86.44	64.87	86.77	64.87	86.77
vit_l_32	76.96	4	75.40	85.54	66.22	86.81	62.32	87.67	62.32	87.67
densenet201	76.93	3	66.92	86.38	66.77	86.40	66.03	86.55	66.03	86.55
mnasnet1_3	76.57	6	95.80	79.51	69.30	86.11	67.57	86.48	67.26	86.57
shufflenet_v2_x2_0	76.28	6	87.71	81.80	69.72	86.48	67.74	86.90	67.36	87.00
vit_b_32	76.00	5	77.54	85.64	69.85	86.92	67.24	87.46	67.07	87.46
regnet_y_400mf	75.85	6	84.00	83.98	72.80	86.26	70.14	86.74	70.08	86.74
densenet169	75.63	2	72.55	86.44	72.55	86.44	71.90	86.55	71.90	86.55
mobilenet_v3_large	75.41	6	78.66	85.30	73.23	86.41	72.97	86.38	72.85	86.41
regnet_x_400mf	74.89	6	98.88	81.47	76.04	86.27	74.03	86.67	73.91	86.69
densenet121	74.52	0	77.40	86.09	77.40	86.09	77.40	86.11	77.40	86.11
vgg19_bn	74.28	0	76.79	86.56	76.79	86.56	76.79	86.56	76.79	86.56
vit_base_patch32_224.sam†	73.75	0	75.95	87.51	75.80	87.57	75.90	87.57	75.80	87.57
mnasnet1_0	73.52	0	79.53	86.76	79.51	86.76	79.53	86.76	79.51	86.76
vgg16_bn	73.46	0	79.78	86.81	79.74	86.81	79.78	86.81	79.74	86.81
resnet34	73.34	2	82.70	86.16	82.68	86.16	82.67	86.20	82.67	86.20
shufflenet_v2_x1_5	73.06	6	104.24	81.45	86.59	85.60	84.92	85.94	84.63	86.00
vgg19	72.47	0	84.60	86.57	84.60	86.58	84.60	86.57	84.60	86.58
mobilenet_v2	72.22	6	107.35	81.83	89.04	85.86	88.92	85.93	88.19	86.04
vgg13_bn	71.67	0	89.36	86.33	89.36	86.34	89.36	86.33	89.36	86.34

vgg16	71.61	0	87.99	86.80	87.93	86.80	87.99	86.80	87.93	86.80
mnasnet0_75	71.24	6	118.72	80.34	95.28	85.43	93.94	85.75	93.32	85.87
vgg11_bn	70.42	0	95.21	86.37	95.16	86.37	95.21	86.37	95.16	86.37
vgg13	69.99	0	97.33	86.34	97.25	86.36	97.33	86.34	97.25	86.36
resnet18	69.83	0	101.00	85.75	100.85	85.78	100.95	85.76	100.85	85.78
googlenet	69.80	4	104.56	84.96	101.93	85.57	101.33	85.70	101.33	85.70
shufflenet_v2_x1_0	69.37	3	102.53	86.03	102.10	86.09	101.43	86.23	101.43	86.23
vgg11	69.09	0	102.42	86.23	102.23	86.25	102.42	86.23	102.23	86.25
mnasnet0_5	67.77	6	117.14	84.62	112.99	85.48	111.46	85.80	111.46	85.80
mobilenet_v3_small	67.67	0	110.32	86.18	110.15	86.19	110.32	86.18	110.15	86.19
shufflenet_v2_x0_5	60.67	3	156.69	85.06	156.04	85.16	155.36	85.30	155.36	85.30
squeezenet1_1	58.17	0	172.90	84.92	172.47	84.99	172.90	84.92	172.47	84.99
squeezenet1_0	58.16	0	176.97	84.21	175.91	84.36	176.97	84.21	175.91	84.36
alexnet	56.53	0	184.01	84.88	183.79	84.91	184.01	84.88	183.79	84.91

† Models from Timm repository

Table 8 – Results for all models evaluated on CIFAR-100

			Baseline		Temperature Scaling		*p*-NormSoftmax		*p*-NormSoftmax*	
Model	Accuracy	Optimal *p*	AURC	AUROC	AURC	AUROC	AURC	AUROC	AURC	AUROC
ResNeXt29_2x64d	0.79	2	0.05	0.88	0.05	0.88	0.05	0.88	0.05	0.89
DenseNet121	0.80	2	0.05	0.87	0.05	0.88	0.05	0.88	0.05	0.88
VGG_19	0.73	2	0.11	0.84	0.10	0.85	0.08	0.87	0.08	0.87
MobileNetV2	0.72	0	0.09	0.86	0.09	0.86	0.09	0.86	0.09	0.86
PNASNetB	0.72	0	0.09	0.85	0.09	0.86	0.09	0.85	0.09	0.86
PreActResNet50	0.79	2	0.06	0.87	0.06	0.87	0.05	0.88	0.05	0.88
DPN26	0.79	2	0.05	0.88	0.05	0.88	0.05	0.88	0.05	0.88
RegNetY_400MF	0.80	2	0.05	0.87	0.05	0.87	0.05	0.88	0.05	0.88
VGG_16	0.75	3	0.09	0.86	0.08	0.86	0.07	0.87	0.07	0.87
ResNeXt29_8x64d	0.81	2	0.05	0.88	0.04	0.89	0.04	0.89	0.04	0.89
VGG_13	0.75	2	0.07	0.87	0.07	0.87	0.07	0.88	0.07	0.88
ResNeXt29_4x64d	0.80	2	0.05	0.88	0.05	0.88	0.05	0.89	0.05	0.89
PNASNetA	0.56	0	0.21	0.81	0.21	0.81	0.21	0.81	0.21	0.81
RegNetX_200MF	0.77	2	0.06	0.87	0.06	0.87	0.06	0.87	0.06	0.88
ResNeXt29_32x4d	0.80	2	0.05	0.87	0.05	0.88	0.05	0.88	0.05	0.88
WideResNet28_10	0.82	2	0.04	0.88	0.04	0.89	0.04	0.89	0.04	0.89
SENet18	0.78	0	0.06	0.87	0.06	0.87	0.06	0.87	0.06	0.87
VGG_11	0.60	0	0.17	0.83	0.16	0.83	0.17	0.83	0.16	0.83
GoogLeNet	0.80	2	0.05	0.87	0.05	0.88	0.05	0.88	0.05	0.88
ResNet18	0.78	2	0.06	0.88	0.06	0.88	0.05	0.89	0.05	0.89

PreActResNet101	0.80	3	0.05	0.88	0.05	0.88	0.05	0.88	0.05	0.88
DenseNet161	0.80	2	0.05	0.88	0.05	0.88	0.05	0.88	0.05	0.88
EfficientNetB0	0.70	1	0.10	0.86	0.10	0.86	0.10	0.86	0.10	0.86
RegNetX_400MF	0.79	2	0.05	0.88	0.05	0.88	0.05	0.88	0.05	0.88
DenseNet201	0.80	2	0.05	0.86	0.05	0.87	0.05	0.87	0.05	0.87
DLA	0.79	2	0.06	0.88	0.05	0.88	0.05	0.88	0.05	0.88
DenseNet169	0.80	2	0.05	0.88	0.05	0.88	0.05	0.88	0.05	0.88
ResNet50	0.79	2	0.06	0.87	0.06	0.88	0.05	0.88	0.05	0.88
LeNet	0.42	0	0.33	0.80	0.33	0.80	0.33	0.80	0.33	0.80
ResNet34	0.79	2	0.06	0.87	0.06	0.88	0.05	0.88	0.05	0.88
ResNet152	0.81	2	0.05	0.88	0.05	0.88	0.05	0.88	0.05	0.89
ResNet101	0.80	2	0.05	0.87	0.05	0.87	0.05	0.88	0.05	0.88
DPN92	0.81	2	0.05	0.88	0.05	0.88	0.05	0.88	0.04	0.89
PreActResNet152	0.80	3	0.05	0.87	0.05	0.88	0.05	0.88	0.05	0.88