Characterization of a Reliability Domain for Image Classifiers

Adrien Le Coz

université PARIS-SACLAY

Characterization of a Reliability Domain for Image Classifiers

Caractérisation d'un Domaine de Fiabilité des Classifieurs d'Images

Thèse de doctorat de l'université Paris-Saclay

École doctorale n°580 : sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat : Informatique
Graduate School : Informatique et Sciences du Numérique. Référent : Faculté des sciences d'Orsay

Thèse préparée dans l'unité de recherche **Traitement de l'information et systèmes (Université Paris-Saclay, ONERA)** et à l'**IRT SystemX**,
sous la direction de **Stéphane HERBIN**, directeur de recherche à l'ONERA,
et le co-encadrement de **Faouzi ADJED**, ingénieur chercheur à l'IRT SystemX.

Thèse soutenue à Paris-Saclay, le 19 décembre 2024, par

Adrien LE COZ

Composition du jury

Membres du jury avec voix délibérative

Mathilde MOUGEOT Professeure des universités, ENSIIE et ENS Paris-Saclay	Présidente
Liming CHEN Professeur des universités, École Centrale de Lyon	Rapporteur & Examinateur
Ahmed SAMET Maître de conférences HDR, INSA Strasbourg	Rapporteur & Examinateur
Frédéric JURIE Professeur des universités, Université de Caen	Examinateur

THESE DE DOCTORAT
NNT : 2024UPASG109

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre : Caractérisation d'un Domaine de Fiabilité des Classifieurs d'Images
Mots clés : Intelligence Artificielle, Apprentissage Profond, Vision par Ordinateur, IA de Confiance

Résumé : Les réseaux de neurones profonds ont révolutionné le domaine de la vision par ordinateur. Ces modèles apprennent une tâche de prédiction à partir d'exemples. La classification d'images consiste à identifier l'objet principal présent dans l'image. Malgré de très bonnes performances des réseaux de neurones sur cette tâche, il arrive fréquemment qu'ils se trompent de façon imprévue. Cette limitation est un frein à leur utilisation pour de nombreuses applications. L'objectif de cette thèse est d'explorer des moyens de définir un domaine de fiabilité qui expliciterait les conditions pour lesquelles un modèle est fiable. Trois aspects ont été considérés. Le premier est qualitatif : générer des exemples extrêmes synthétiques permet d'illustrer les limites d'un classifieur et de mieux comprendre ce qui le fait échouer. Le second aspect est quantitatif : la classification sélective permet au modèle de s'abstenir en cas de forte incertitude, et la calibration permet de mieux quantifier l'incertitude de prédiction. Enfin, le troisième aspect est d'inclure de la sémantique : des modèles multimodaux qui associent images et texte sont utilisés pour décrire textuellement les images susceptibles de provoquer de mauvaises, ou inversement, de bonnes prédictions.

Title : Characterization of a Reliability Domain for Image Classifiers
Keywords : Artificial Intelligence, Deep Learning, Computer Vision, Trustworthy AI

Abstract : Deep neural networks have revolutionized the field of computer vision. These models learn a prediction task from examples. Image classification involves identifying the main object present in the image. Despite the very good performance of neural networks on this task, they often fail unexpectedly. This limitation prevents them from being used in many applications. The goal of this thesis is to explore methods for defining a reliability domain that would clarify the conditions under which a model is trustworthy. Three aspects have been considered. The first is qualitative : generating synthetic extreme examples helps illustrate the limits of a classifier and better understand what causes it to fail. The second aspect is quantitative : selective classification allows the model to abstain in cases of high uncertainty, and calibration helps better quantify prediction uncertainty. Finally, the third aspect involves semantics : multimodal models that associate images and text are used to provide textual descriptions of images likely to lead to incorrect or, conversely, to correct predictions.

Contents

Acronyms

ADAS Advanced Driver-Assistance System. 3, 4

AI Artificial Intelligence. 1–6, 12, 13, 78, 93, 96, 97, 99

AURC Area Under the Risk-Coverage curve. 48, 49, 53

AUROC Area Under the Receiver Operating Characteristic curve. 49, 53, 54, 63, 65, 66, 86, 88–90, 100, 110

BBQ Bayesian Binning into Quantiles. 60

Beta Beta Calibration. 60

BO Bayesian Optimization. 77, 79–81

CoSim Cosine Similarity. 52, 53, 76, 85

CT Combinatorial Testing. 74, 75, 79, 103

DA Domain Adaptation. 6, 17, 94, 100

DC Dirichlet calibration. 59, 62, 63

DDPM Denoising Diffusion Probabilistic Model. 27, 73

DG Domain Generalization. 6, 17, 94, 100

DL Deep Learning. 2

DNN Deep Neural Network. 2–6, 13, 15, 18, 20–22, 77, 97

ECE Expected Calibration Error. 23, 49, 54, 56, 60–65, 67–69, 107–109, 113

FPR False Positive Rate. 49

GA Genetic Algorithm. 77, 79–81, 103

GAN Generative Adversarial Network. 8–10, 12, 17, 24–32, 38, 40–45, 47, 66–69, 94–96

GP Gaussian Process. 77

GPU Graphics Processing Unit. 2, 15

HB Histogram Binning. 50, 60, 62, 65

ICL In-Context Learning. 63

Iso Isotonic Regression. 60

LLM Large Language Model. 1, 2, 12, 20, 28, 63, 64, 96, 97, 99, 100

ML Machine Learning. 1, 2, 5, 6, 12, 13, 103

MSP Maximum Softmax Probability. 10, 20, 21, 39–42, 47, 51, 53, 54, 70, 72, 87–91, 94, 98, 100

ODD Operational Design Domain. 5, 6, 13, 19, 74

OOD Out-of-Distribution. 6, 16, 17, 21, 22, 26, 51, 65, 70, 94, 97, 99, 100

OvA One-versus-All. 24, 50, 55, 58, 63

SAC Selective Accuracy Constraint. 49

SB Semantic Binning. 84, 86

SC Selective Classification. 6, 10–12, 20–22, 46, 50–54, 65, 66, 70, 84, 97, 98, 100

SD Stable Diffusion. 73, 78, 91, 101, 103

SSC Semantic Selective Classification. 83, 84, 86, 87, 89–91, 93, 95, 96, 99

TCP True Class Probability. 51–53

TI Textual Inversion. 85, 91, 92, 103

TPR True Positive Rate. 49

TS Temperature Scaling. 50, 59–63

TvA Top-versus-All. 24, 56–66, 98, 104, 106–108

VS Vector Scaling. 50, 59–63

XAI Explainable Artificial Intelligence. 8, 10, 12, 13, 29, 42, 71

Résumé en Français

Contexte Les réseaux de neurones profonds ont révolutionné de nombreux domaines, notamment la vision par ordinateur. Ces modèles d'apprentissage machine apprennent une tâche de prédiction à partir d'exemples. Cette thèse considère la classification d'images, qui consiste à identifier l'objet principal présent dans l'image. Malgré de très bonnes performances des réseaux de neurones sur cette tâche, il arrive fréquemment qu'ils se trompent de façon imprévue. Un modèle peut par exemple se tromper quand une image diffère trop de ce qu'il a vu pendant son entraînement. Les échecs relativement fréquents de ces modèles sont un frein à de nombreuses applications. En réponse à ce problème, des initiatives se sont crées pour développer l'intelligence artificielle de confiance. C'est le cas du programme Confiance.ai, qui vise à développer les applications de l'intelligence artificielle dans diverses industries françaises, notamment pour des applications critiques.

L'objectif de cette thèse est d'explorer des moyens de définir un domaine de fiabilité qui expliciterait les conditions pour lesquelles un classifieur d'images est fiable. Étant donné l'ampleur du sujet, différents aspects peuvent être étudiés. Le premier aspect consiste à prendre le point de vue de l'explicabilité. J'ai utilisé des modèles génératifs pour créer des images illustrant les conditions de défaillance d'un classifieur. Le deuxième aspect consiste à examiner le problème d'un point de vue plus quantitatif, en utilisant les notions de classification sélective et de calibration. J'ai développé une nouvelle approche pour améliorer la calibration des classifieurs. Enfin, le troisième aspect est l'intégration de la sémantique en décrivant les échecs des classifieurs avec du texte. J'ai amélioré une technique permettant de découvrir rapidement les défaillances des classifieurs en utilisant des images synthétiques pour un modèle texte-image et j'ai incorporé des attributs textuels dans la classification sélective.

Aspect Qualitatif : Utiliser des Modèles Génératifs pour Illustrer les Échecs Les exemples extrêmes pourraient être un moyen de définir le domaine d'un classifieur : ils donnent des indications sur ses limites. Cependant, ils sont généralement rares dans les données disponibles car ils ont une faible probabilité de se produire. Utiliser des modèles génératifs est une solution pour créer des exemples extrêmes synthétiques. Les *Generative Adversarial Networks (GANs)* peuvent transformer des vecteurs situés dans un espace latent (entrée du modèle) en images réalistes (sortie du modèle). Ces modèles sont entraînés à générer des images synthétiques indistinguables des images des données d'entraînement. Ils construisent une représentation comprimée des images dans l'espace latent, dont la dimension est réduite par rapport à l'espace des pixels. Cela permet de représenter les images en fonction de leurs caractéristiques visuelles (par exemple, une direction dans l'espace latent peut coder le contraste de l'image). Certains modèles séparent les différentes caractéristiques visuelles, ce qui permet de modifier les images une caractéristique à la fois (par exemple, changer la couleur des cheveux d'un portrait sans modifier le reste). Ceci peut être utilisé pour expliquer la décision d'un classifieur en

identifiant les attributs visuels qu'il utilise pour distinguer les classes.

J'ai développé un modèle génératif capable de créer des exemples extrêmes pour un classifieur donné afin de caractériser ses échecs. Pour ce faire, le classifieur est couplé à un GAN. Les données utilisées sont une version corrompue des données MNIST, qui sont des images de chiffres manuscrits. Le bruit et le flou sont des caractéristiques des systèmes d'imagerie qui peuvent altérer les performances de la classification. Ces caractéristiques sont ajoutées au générateur pour une génération plus réaliste. La tâche du classifieur est de reconnaître le chiffre présent dans l'image. Un modèle génératif est entraîné pour créer des images synthétiques réalistes et construit un espace latent permettant la manipulation de l'image. En particulier, nous pouvons détecter les directions de l'espace latent qui ont le plus d'impact sur la classification, c'est-à-dire celles qui corrompent fortement les images. La détection de ces directions se fait à l'aide du gradient de la prédiction du classifieur par rapport aux vecteurs de l'espace latent. Nous pouvons ensuite visualiser les attributs visuels en partant de certaines images et en les perturbant en fonction des directions les plus impactantes. Par exemple, un zéro est principalement perturbé par les attributs de forme et de contraste, et un neuf par les niveaux de bruit ou de flou. Cela nous permet de visualiser les limites du classifieur. Il n'est pas robuste à certains attributs visuels, car certains exemples extrêmes sont encore reconnaissables par l'œil humain.

J'ai ensuite essayé une approche différente en associant le classifieur à un GAN. Cette fois, la confiance du classifieur conditionne directement la génération. La confiance d'un classifieur est la probabilité qu'il attribue à sa prédiction. En effet, les classifieurs prédisent un vecteur de probabilité, un pour chaque classe, à partir duquel la classe prédite est déduite (celle qui correspond à la probabilité maximale). Souvent, les classifieurs se trompent lorsque la probabilité maximale est faible : le modèle hésite entre plusieurs classes. En conditionnant le GAN avec la confiance du classifieur, nous pouvons directement guider la génération vers les images difficiles. Cette approche permet de corrompre les images et de générer des exemples extrêmes. Cependant, les premiers résultats montrent que la confiance pour une image générée n'est pas nécessairement celle fixée comme condition pour le générateur. Les contraintes données par la condition sont en effet difficiles à apprendre pour le générateur, car le comportement du classifieur est imparfaitement résumé en un seul nombre. Néanmoins, le contrôle de cette condition permet de mieux comprendre le comportement du classifieur.

Ces deux travaux ont montré que les GANs peuvent générer des exemples extrêmes ou difficiles pour un classifieur donné. Cela permet d'illustrer les attributs visuels les plus susceptibles de perturber la classification. Toutefois, les deux approches sont limitées par le fait que les GANs ont du mal à générer de manière réaliste des images plus complexes, telles que les images naturelles de l'ensemble de données ImageNet. Un autre objectif était de définir les zones de l'espace latent qui correspondent à une bonne performance du classifieur. Un domaine pourrait alors être défini dans l'espace latent. Cependant, la structure de l'espace latent est complexe. En outre, la projection d'images dans l'espace latent (inversion de GAN) est un processus complexe qui est nécessaire pour valider le domaine de fiabilité.

Aspect Quantitatif : Classification Sélective et Calibration Une limitation des exemples extrêmes synthétiques est qu'ils ne permettent pas de quantifier la fiabilité d'une prédiction ni de définir un domaine de fiabilité. Je me suis donc intéressé aux domaines de recherche de la classification sélective et de la calibration. Ils visent tous deux à anticiper le moment où les prédictions d'un classifieur sont correcte ou non, mais avec des

objectifs différents. Ces notions sont liées au concept de domaine de fiabilité : la classification sélective classe explicitement toutes les données en deux catégories : sélectionnées ou rejetées, ce que nous pouvons considérer comme “in-domain” ou “out-domain”.

J’ai d’abord mené des expériences sur la classification sélective. L’objectif est de rejeter les données de classification pour lesquelles la prédiction est susceptible d’être erronée. L’un des principaux aspects de la classification sélective est la fonction de sélection qui détermine quelles données sont sélectionnées ou rejetées. Elle est généralement basée sur le seuillage d’un score de confiance. L’utilisation de la probabilité maximale prédite par le classifieur est le moyen standard, mais des approches plus avancées apprennent un réseau qui prédit un score de confiance. En pratique, mes expériences préliminaires montrent que le moyen standard est en fait très performant.

Ensuite, en observant les similitudes entre la classification sélective et la calibration, j’ai développé une nouvelle approche de calibration. La calibration vise à obtenir des probabilités prédites représentatives de la probabilité de faire une prédiction correcte. Lorsqu’un réseau prédit une classe avec une probabilité de 0,8, il devrait être correct dans 80 % des cas. Dans le cas contraire, il est trop ou pas assez confiant. Les méthodes de calibration post-traitement peuvent améliorer la calibration des classifieurs pré entraînés. Ces méthodes cherchent à optimiser les paramètres ou à apprendre une fonction pour recalculer la confiance des prédictions. Des données de calibration, extérieures aux données d’apprentissage, sont utilisées pour l’optimisation. Ces données sont généralement rares, ce qui limite les méthodes standard de calibration. Pour résoudre ce problème, je transforme le problème de la calibration d’un classifieur multiclasse en calibration d’un seul classifieur binaire de substitution. Cette reformulation simple permet une meilleure utilisation des méthodes de calibration existantes avec une modification minimale de leurs algorithmes d’origine. Des expériences complètes en classification d’image et de texte prouvent que l’approche améliore de manière significative les méthodes de calibration existantes.

Une autre approche pour répondre au besoin de données de calibration consiste à utiliser des données synthétiques. Un GAN conditionnel peut générer des données proches des données d’apprentissage mais différentes, ce dont les méthodes de calibration post-traitement ont besoin. Des expériences préliminaires montrent que l’utilisation de données synthétiques est au moins aussi efficace que l’utilisation de données de validation standard. Toutefois, ces résultats pourraient ne pas s’appliquer à des données plus complexes, qu’il est plus difficile de générer avec fidélité.

Aspect Sémantique : Incorporer des Descriptions Textuelles Le dernier aspect étudié est l’incorporation de sémantique : l’utilisation du texte pour décrire le domaine du comportement approprié du classifieur. Des travaux récents utilisent des modèles génératifs *texte-to-image* basés sur des modèles de diffusion. Ces modèles peuvent générer des images correspondant à des descriptions textuelles et identifier celles qui provoquent l’échec d’un classifieur. Par exemple, on peut découvrir que les mouches sont parfois classées comme des abeilles lorsqu’elles se trouvent à côté d’une fleur.

J’ai d’abord développé une méthode pour améliorer une approche existante. Cette dernière utilise des modèles de diffusion pour générer des images correspondant à certains sous-groupes d’images définis par des attributs textuels (météo, localisation, couleur, etc.). En raison du temps d’inférence élevé de ces modèles, seuls certains sous-groupes peuvent être évalués. J’ai développé une méthode basée sur l’optimisation bayésienne qui identifie rapidement les sous-groupes potentiellement problématiques. Ainsi, la découverte

des sous-domaines conduisant à des échecs de classification est beaucoup plus exhaustive et rapide.

La description des échecs à l'aide de texte permet de comprendre les limites du classifieur, mais n'aide pas à déterminer quand le classifieur est fiable. La classification sélective peut définir un domaine de fiabilité basé sur des valeurs de score de confiance, mais ne dit pas grand-chose sur le type de données qui est sélectionné ou rejeté. Le dernier travail de cette thèse consiste à incorporer des descriptions textuelles dans la classification sélective afin de décrire sémantiquement quand un classifieur fonctionne de manière fiable. J'ai développé un moyen de filtrer les données susceptibles d'être erronées en exploitant les attributs textuels des données qui sont traduits en un score de confiance. Le domaine de fiabilité peut être décrit par une liste d'attributs textuels. Cependant, l'approche nécessite de nombreuses données de validation annotées, ce qui n'est possible de manière réaliste qu'en utilisant des modèles génératifs. Cela signifie que le domaine de fiabilité reste dans le monde des images synthétiques et ne peut pas être validé avec des images réelles.

Discussion et Perspectives La définition d'un domaine de fiabilité pour un classifieur d'images (ou d'un réseau de neurones en général) reste un problème difficile à exprimer clairement ainsi qu'à résoudre. Cette thèse a pu faire progresser ces deux aspects. Aussi, elle ouvre plusieurs perspectives. Premièrement, je pense qu'il existe un fort lien théorique et pratique entre la classification sélective et la calibration. C'est en me basant sur ce lien que j'ai développé une nouvelle approche à la calibration. Il y a sans doute d'autres travaux à réaliser. Deuxièmement, il serait intéressant d'évaluer les modèles multimodaux (texte-image). Ces derniers se sont développés rapidement durant cette thèse. Malgré des résultats impressionnants, ils sont aussi sujets à des échecs et biais. Troisièmement, je trouve que beaucoup de termes liés aux échecs des modèles sont peu clairs et mal définis. En effet, des termes comme "anomalie" ou "*out-of-distribution*" sont employés différemment en fonction des articles de recherche. Les domaines de recherche actuels se focalisent souvent sur une sous-partie des causes d'échecs possibles. Il serait intéressant de clarifier tout cela, notamment en identifiant bien toutes les causes d'échecs possibles et quels travaux existants s'y intéressent. Enfin, je pense qu'un travail nécessaire serait de développer une procédure d'évaluation de domaine de fiabilité. Notamment en rassemblant les travaux existants qui considèrent des modes d'échecs bien particuliers. Les méthodes de détection d'échecs pourraient être évaluées simultanément sur plusieurs aspects comme la détection de données *out-of-distribution*, d'anomalies, ou d'attaques adverses.

Chapter 1

Introduction

1.1 The Rise of Deep Learning

In recent years, the field of Artificial Intelligence (AI) has experienced unprecedented growth, revolutionizing various sectors such as healthcare, finance, transportation, and entertainment (Deng, 2014; LeCun et al., 2015; Shinde and Shah, 2018; Christin et al., 2019; Haghighat et al., 2020; Choudhary et al., 2022). Machine Learning (ML), a subset of AI, refers to the development of algorithms that enable computers to learn from data how to make predictions or decisions. This paradigm shift, from rule-based programming to data-driven approaches, has been fueled by the exponential increase in computational power, availability of vast amounts of data, and advances in algorithmic design.

ML encompasses various techniques and methods that allow systems to learn and improve from experience without being explicitly programmed. It can be broadly categorized into three types: supervised learning, unsupervised learning, and reinforcement learning.

In supervised learning, algorithms are trained on labeled datasets, where the input data is paired with the correct output. Part of the data is used to train the predictive model, which becomes able to generalize to unseen test data. Techniques such as support vector machines (Boser et al., 1992) have been instrumental in advancing this domain. Common applications include classification and regression tasks, where the goal is to predict labels or continuous values, respectively. In image classification, the goal of the model is to identify the main content of an image. For instance, classifiers can be trained to recognize elements of everyday life, including animals, objects, and scenes. An example of a regression model is a model trained to predict wind power output in the near future, which is useful to manage the power grid.

Unlike supervised learning, unsupervised learning deals with unlabeled data. The objective is to uncover hidden patterns or structures within the data. Clustering algorithms like k-means and dimensionality reduction techniques such as principal component analysis fall under this category. A very closely linked approach is self-supervised learning. It aims to solve pretext tasks for which the labels are derived from the input data. Methods include contrastive learning or just supervised learning methods applied to the pretext task (Jaiswal et al., 2020; Jing and Tian, 2020). For instance, Large Language Models (LLMs), the base of modern chatbots like ChatGPT, are trained by masking parts of the input text and training the model to predict the masked part.

Reinforcement learning focuses on training agents to make a sequence of decisions by interacting with an environment. The agent learns to achieve its goals through trial

and error, receiving rewards or penalties based on its actions. This framework has led to significant breakthroughs in fields like game playing, robotics, and autonomous systems (Mnih et al., 2015; Silver et al., 2016; Arulkumaran et al., 2017). However, reinforcement learning has difficulties solving many real-world problems due to the large amount of interaction data necessary and the complexity of defining a good reward function.

The three learning paradigms described previously are now mostly based on Deep Learning (DL). DL is a specialized subset of ML that was originally inspired by the structure and function of the human brain (LeCun et al., 2015). It uses artificial neural networks with many layers, known as Deep Neural Networks (DNNs), to model complex patterns in data. The hierarchical structure of DNNs allows for the automatic extraction of high-level features from raw inputs, making them particularly effective for tasks involving image, audio, and natural language processing.

Arguably, the modern era of AI started in 2012, when a deep convolutional neural network (LeCun et al., 1989) called AlexNet won the ImageNet competition by a huge margin (Krizhevsky et al., 2012). The competition is about recognizing the main object in images of the ImageNet dataset (Deng et al., 2009) out of 1000 categories (animals, objects...); see Figure 1.1. The AlexNet breakthrough was achieved by training a large model on more than one million labeled images, using an efficient Graphics Processing Unit (GPU) implementation. More than a decade later, DNNs are used for many applications, including LLMs for chatbots (OpenAI, 2023), computer vision systems (Pathak et al., 2018; Liu et al., 2018), and image generators (Ramesh et al., 2022; Saharia et al., 2022).

This thesis focuses on image classification models. Image classifier applications include disease diagnosis from medical images, scene recognition for autonomous vehicles, anomaly detection in assembly lines, moderation in social networks, wildlife population monitoring from camera traps, and automatic photo organization.

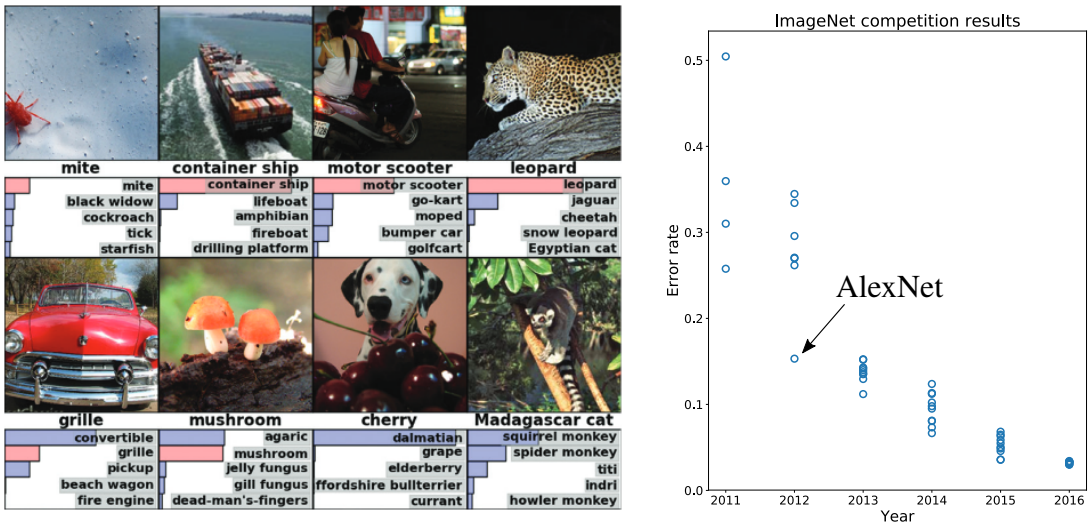


Figure 1.1: (Left) Examples of ImageNet test images with the top 5 most probable labels and associated probabilities predicted by AlexNet. From (Krizhevsky et al., 2012).
(Right) Evolution of ImageNet competition error rates. In 2012, AlexNet outperformed all other approaches, which did not improve much from the year before. After 2012, DL-based approaches further reduced the error rate. From (Wikipedia, 2024c).

1.2 Deep Learning Models Failures

Figure 1.2: Examples of failures of text-to-image generative models. Despite an impressive ability to generate realistic images corresponding to a text prompt, these models struggle with bias and concepts like quantity and negation. From (Tong et al., 2023).

Despite the rapid progress of DNNs, one major issue still prevents them from being used for many applications: their lack of reliability. Indeed, they are prone to making mistakes, and it is difficult to understand why or when they do so. While neural networks are conceptually simple, they require a huge scale to tackle high-dimensional data such as images or text, typically requiring millions or billions of internal parameters. This makes the decision process opaque and hard to explain or interpret. The learning phase requires data containing up to millions or billions of examples, and many companies keep details of the training process and data private. Because the models learn to reproduce what is in the training data, biases in the data are replicated by the model.

In critical systems such as medical or Advanced Driver-Assistance System (ADAS), prediction errors can be life-threatening. Even in less critical systems, these errors are still undesirable as they can lead to financial losses or damage a company's reputation.

A first example is the case of facial recognition. Failures due to biased predictions were one of the main criticisms of this technology, ultimately prompting companies like Amazon, IBM, Microsoft, and Google to cease selling or developing it (Wen and Holweg, 2024). In particular, Amazon's Rekognition software incorrectly matched 28 members of the United States Congress as other people who have been arrested for a crime (Snow, 2018). Facial recognition software from Amazon, IBM, and Microsoft had higher error rates for people of color (Singer, 2019). Google Photos' image classification model categorized a black woman as a gorilla (BBC, 2015). Years later, the issue still is not solved; only a temporary fix was developed: Google and Apple appear to have disabled the models' ability to label images as primates (Grant and Hill, 2023).

Another example of failure due to bias is Amazon's AI recruitment tool. It showed bias against women and was eventually shut down (Dastin, 2018).

IBM Watson for Oncology is a tool that helps oncologists make treatment decisions for cancer patients. The tool's concordance with physicians in China is not as high as previously reported in other countries (Zhou et al., 2019). Developed in the United States,

its performance decreases when applied to a different situation.

Autonomous driving has been hyped for years, but current systems are not reliable enough for wide deployment. Truly driverless cars, such as Waymo or Baidu's taxis, only operate in selected cities. Tesla's "Full Self-Driving (Supervised)" is an optional paid feature meant to work on any type of roadway, including residential and city streets (Tesla, 2024). However, it is only available in the United States and Canada and remains at SAE Level 2, meaning that the driver has to remain ready to take control at all times and is responsible in case of an accident. Even Tesla's base ADAS system, which includes lane-centering and traffic-aware cruise control, is prone to failures. See Wikipedia (2024b) for a list. Some of the failures are due to perception issues of computer vision systems.

More recently, AI capabilities are some of the most prominently advertised features from smartphone manufacturers like Google and Apple. However, model failures are frequent enough for users to lose trust in the product (Schoon, 2024). Examples of failures from recent generative models are shown in Figure 1.2.

All the examples described in this section show that unpredictable failures are one of the main barriers to deploying AI systems.

1.3 Trustworthy AI Initiatives

Figure 1.3: Confiance.ai Partners

Because DNNs fail in unpredictable ways, as described above, deploying them in real-world systems is challenging. Many industrial critical systems require safety guarantees, which current DNNs lack. Initiatives were developed around the world to promote

AI safety and trustworthiness, such as the European Union's AI Act, OECD AI Principles, the Partnership on AI, and the European Union Aviation Safety Agency's Machine Learning Application Approval (MLEAP) project (MLEAP Consortium, 2024).

Confiance.ai[1] is a French technological research program aiming to help industrials integrate trustworthy AI in their critical systems. Launched in 2020 for a duration of 4 years, it is part of the "Securing, Certifying, and Enhancing the Reliability of Systems Based on Artificial Intelligence" initiative, which is backed by the French government. The project gathers methods and tools into a *trustworthy environment* development platform. The program brings together an ecosystem of small, medium, and big companies, start-ups, and research labs; see Figure 1.3.

Confiance.ai targets, in particular, critical applications in industry, mobility, energy, environment, defense, and security. All these sectors belong to what the European Union's AI Act identifies as "high-risk" applications. Such applications will be subject to specific demands, such as adequate risk assessment and a high level of robustness, security, and accuracy.

Technological contributions of the program include an end-to-end method based on Systems Engineering and Software Engineering for the engineering of a critical ML-based system[2], and a taxonomy proposing definitions for terms related to trustworthy AI, accessible among other resources in an online catalog[3]. Scientific results include dozens of publications[4], including those from PhD theses financed by the program, such as this one.

1.4 The Initial Goal: Defining a Domain

In Systems Engineering: Operational Design Domain In autonomous driving and trustworthy AI programs, such as Confiance.ai and MLEAP, a key objective is to express high-level requirements of the "intended domain of use". This is the goal of the Operational Design Domain (ODD), a notion initially from autonomous driving and used in systems engineering. It is defined as "the operating conditions under which a given driving automation system or feature thereof is specifically designed to function, including, but not limited to, environmental, geographical, and time-of-day restrictions, and/or the requisite presence or absence of certain traffic or roadway characteristic" (On-Road Automated Driving (ORAD) Committee, 2018). For example, the Mercedes-Benz automated driving system, DRIVE PILOT, has an explicit ODD. It includes use only on the motorway, a speed limit of 60 km/h, use only during the day and on dry roads, lane markings recognized by the system, and a vehicle ahead, which is registered as an indicator of a traffic jam (Rocco, 2022). However, there is no agreement on the definition of ODD. Many variations exist, including extensions beyond autonomous driving (Wikipedia, 2024a; Mehlhorn et al., 2023; Adedjouma et al., 2024). Further discussion on this notion and how to use it in practice are beyond the thesis's scope. This is why in this thesis a simplified definition of an ODD is considered: "the conditions for which we *want/hope* model predictions to be reliable". Reliable means consistently good in quality or performance and able to be trusted (from Google's English dictionary, provided by Oxford Languages). A limitation of the notion of ODD is that it considers the conditions set during the design phase, which might not hold in practice. DNNs' behavior is

[1] https://www.confiance.ai
[2] https://bok.confiance.ai
[3] https://catalog.confiance.ai/
[4] https://hal.science/CONFIANCEAI

largely opaque, and design goals are not necessarily verified because many failures are unpredictable, as described in section 1.2.

In Machine Learning: Notions of Domain As discussed in section 1.2, DNNs can encounter various failure modes when deployed in the real world. In response to these challenges, the research community has devoted considerable effort to developing methods to prevent or mitigate such failures. However, this effort is divided into subtopics with specific hypotheses, goals, and methods. Below are some brief descriptions of topics closely related to the notion of *domain*. More details are included in section 2.3.

In ML, data is viewed as a joint probability distribution $P(X, Y)$ that represents the likelihood of observing particular combinations of input features X and corresponding output labels Y. After a model has been trained, it is deployed in the real world, in the so-called testing phase, where it might experience distribution shifts.

Covariate shift is when the distribution of the input features $P(X)$ changes. For instance, at test time, images might have different lighting conditions or background variations. In this setting, the domain is the ensemble of examples having input features similar to the ones seen during training. Domain Adaptation (DA) (Wang and Deng, 2018) and Domain Generalization (DG) (Zhou et al., 2022) aim to robustify classifiers to such shifts.

Semantic shift is when the distribution of the output labels $P(Y)$ changes. For instance, at test time, a dog/cat classifier might have to process giraffe images. In this setting, the domain contains all examples with the same labels as the ones seen during training. Out-of-Distribution (OOD) detection, among other approaches, aims to detect examples that do not belong to this domain (Yang et al., 2021).

Under the hypothesis that no shift happens, Selective Classification (SC) allows a more flexible notion of domain based on a parametric selection function (Geifman and El-Yaniv, 2017). It allows controlling the compromise between domain extension and accuracy: a stricter selection rejects more data to increase the accuracy for selected data.

None of these domain-related notions seem comprehensive enough to ensure truly reliable and trustworthy AI. None of the proposed solutions permit a reliable deployment of models, as each only solves subproblems. For instance, detecting semantic shift does not prevent errors coming from covariate shift or simply hard in-distribution data.

A Reliability Domain? This thesis considers the practical problem of identifying the conditions for which a given image classifier *actually* works reliably. The objective is to express a model's *Reliability Domain*, defined as "the conditions for which model predictions *are* reliable". Contrary to the ODD, this is the evaluation phase rather than the design phase. A given image classifier might fail for some conditions set during the design phase (e.g., Tesla's phantom braking where the vehicle brakes for no apparent reason (Siddiqui, 2022)), or succeed for unforeseen conditions (e.g., DRIVE PILOT's vision system might still work for higher speed limits). This is illustrated in Figure 1.4. Contrary to other related notions in ML (OOD, DA, or DG) where a domain is *implicitely* defined, the aim here is to make it more *explicit*.

The considered setting is that the image classifier has already been developed, and the goal is to assess in which conditions it can be used reliably. Focusing on evaluating fixed pre-trained image classifiers ensures the generality of the approach: no specific changes to the classifier are needed.

The initial intuition was to use extreme examples to define a domain. They would describe the domain's limits, and only data "inside" the limits would be considered in-

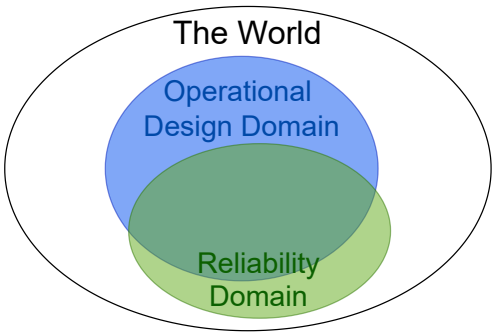


Figure 1.4: When deployed in the world, a computer vision model works reliably in some conditions (Reliability Domain), which might differ from what was expected during the design phase (Operational Design Domain).

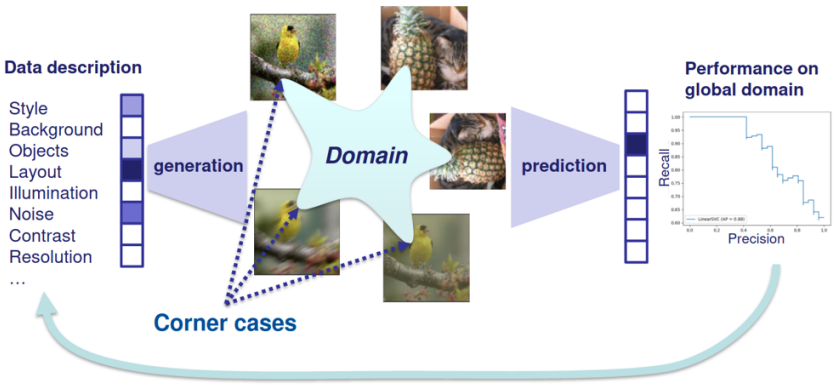


Figure 1.5: Illustration of the initial intuition: define a reliability domain using extreme examples delimiting its boundaries and linking classification performance to data description.

domain. This is a geometric view, which is represented in Figure 1.5. Because extreme examples are supposedly rare, it might be necessary to generate them using recent image-generation models. It is important to describe them by interpretable attributes, to make the domain more understandable. As written in the initial thesis subject, defining a reliability domain brings out the following questions (Q):

Q1 How can a domain be expressed? Does it have boundaries?

Q2 What are good examples to describe these limits? Can they be identified from reference data or generated artificially? Can they be described by interpretable characteristics or attributes?

Q3 Can extreme examples be characterized based on their contribution to the different sources of error in learning? (Estimation, approximation, optimization, bias / variance trade-off)

One main objective is to ensure the domain definition has practical applications and is not limited to an abstract notion. Linking data descriptions to classification performance supposedly allows better evaluations, explainability, and insights to improve the model during development. The following uses (U) should thus be addressed:

U1 As a performance indicator of a given algorithm by providing more detailed information than classical global metrics, and additional information on the local behavior of the algorithms.

U2 As an explainability tool to intuitively analyze the overall behavior of the algorithm.

U3 As a development tool to control learning and the trade-off between domain extension/performance.

U4 As a means of specifying the data to be collected.

Section 6 discusses how the thesis results relate to the initial intuition and goals.

1.5 Summary of Contributions

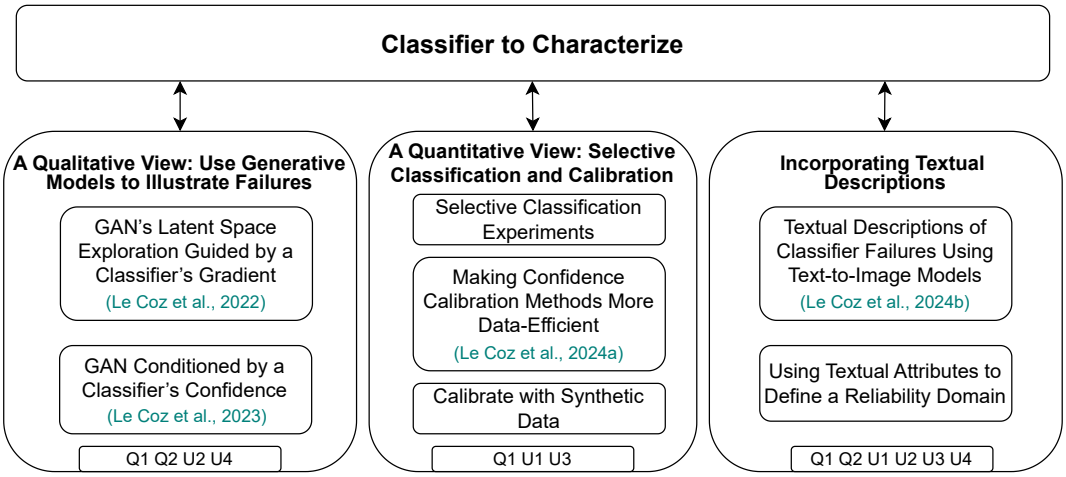


Figure 1.6: Overview of the different contributions divided in three chapters and which Questions and Uses of a reliability domain definition they address.

Given the large scope of the topic, different aspects can be studied. The first aspect is to follow the Explainable Artificial Intelligence (XAI) point of view. I leveraged generative models to create images illustrating the failure conditions of a classifier. The second aspect is to look at the problem following a more quantitative view, using the notions of selective classification and calibration. I developed a new approach to improve classifier calibration. Finally, the third aspect is the integration of semantics by describing classifier failures with text. I enhanced a technique to quickly discover classifier failures using synthetic images for a text-to-image model and incorporated textual attributes into selective classification.

1.5.1 Use Generative Models to Illustrate Failures

Extreme examples could be a way to define a classifier's domain: they give hints about its boundaries. However, they are generally rare in the available data because they have a low probability of occurring. Chapter 3 studies generative image models to create synthetic extreme examples. Generative Adversarial Networks (GANs) can transform

vectors located in a latent space (model input) into realistic images (model output). These models are trained to generate synthetic images that are indistinguishable from the images in the training data. They build a compressed representation of the images in the latent space, whose dimension is reduced compared to the pixel space. This allows the images to be represented according to visual characteristics (e.g., a direction in the latent space can encode the image's contrast). Some models separate different visual characteristics, allowing for editing images one characteristic at a time (e.g., changing the hair color of a portrait without modifying the rest). This can be used to explain the decision of a classifier by identifying which visual attributes it uses to distinguish between classes.

I developed a generative model capable of creating extreme examples for a given classifier to characterize its failures. For this, the classifier is coupled with a GAN. The data used is a corrupted version of MNIST data, which are images of handwritten digits. Noise and blur are features of imaging systems that can alter classification performance. Those features are added to the generator for a more realistic generation. The task of the classifier is to recognize which digit is present in the image. A generative model is trained to create realistic synthetic images and builds a latent space allowing image manipulation. In particular, one can detect the latent space directions that impact the classification the most, i.e., those that strongly corrupt the images. The detection of these directions is done with the gradient of the classifier's prediction with respect to the latent space vectors. One can then visualize the visual attributes by starting from some images and perturbing them according to the most impactful directions. For instance, a zero is mainly perturbed by shape and contrast attributes, and a nine by noise or blur levels, as seen in section 3.3. This allows visualizing the limits of the classifier. It is not robust to certain visual attributes because some extreme examples are still recognizable to the human eye. This work has been published (Le Coz et al., 2022).

I then tried a different approach to couple the classifier with a GAN. This time, the classifier's confidence directly conditions the generation. The confidence of a classifier is the probability it assigns to its prediction. Indeed, classifiers predict a probability vector, one for each class, from which the predicted class is inferred (the one corresponding to the maximum probability). Often, classifiers are wrong when the maximum probability is low: the model hesitates between several classes. Conditioning the GAN with the classifier's confidence can directly guide the generation towards difficult images. This approach allows corrupting images and generating extreme examples. However, the initial results show that the confidence for a generated image is not necessarily the one set as a condition for the generator. The constraints given by the condition are indeed difficult for the generator to learn, as the classifier's behavior is imperfectly summarized into one single number. Still, controlling this condition provides insights into the classifier's behavior. This work has been published (Le Coz et al., 2023).

These two works have shown that GANs can generate extreme or difficult examples for a given classifier. This allows for illustrating the visual attributes most likely to disrupt classification. However, the two approaches are limited by the fact that GANs struggle to realistically generate more complex images, such as natural images from the ImageNet dataset. Another goal was to define areas of the latent space that correspond to good classifier performance. A domain could then be defined in the latent space. However, the structure of the latent space is complex. Also, projecting images into the latent space (GAN inversion) is a complex process that is necessary to validate the reliability domain.

1.5.2 Selective Classification and Calibration

In chapter 3, as described in the previous subsection, synthetic extreme examples were used to better understand the classifier's limits. However, they do not help quantify the reliability of a prediction nor provide a way to define a reliability domain. Chapter 4 considers the research fields of selective classification and calibration. They both aim to anticipate when classifier predictions are accurate or not but with different goals. These fields are aligned with the notion of reliability domain: SC explicitly categorizes all data in two categories: selected or rejected, which can be seen as "in-domain" or "out-domain".

I first conducted experiments on SC. The goal is to reject classifying data for which the prediction is likely to be wrong. A main aspect of SC is the selection function which determines which data is selected or rejected. It is usually based on thresholding a confidence score. Using the classifier's Maximum Softmax Probability (MSP) is the standard baseline, but more advanced approaches train a network to predict a better score. In practice, my preliminary experiments show that the standard baseline actually has a strong performance.

Then, based on observing the similarities between SC and calibration, I developed a new approach to calibration. Calibration aims to obtain predicted probabilities representative of the probability of making a correct prediction. When a network predicts a class with a probability of 0.8, it should be correct 80% of the time. Otherwise, it is overconfident or underconfident. Post-processing calibration methods can improve the calibration of pre-trained classifiers. These methods seek to optimize parameters or learn a function to recalculate the confidence of predictions. Calibration data, external to the training data, are used for optimization. This data is usually scarce, which limits standard calibration methods. To address this issue, I transform the problem of calibrating a multiclass classifier into calibrating a single surrogate binary classifier. This straightforward reformulation allows a better use of existing calibration methods with minimal change to their original algorithms. Comprehensive image and text classification experiments prove the approach significantly improves existing calibration methods. This work has been published (Le Coz et al., 2025).

Another approach to tackle the need for calibration data is to use synthetic data. A conditional GAN can generate data close to the training data but different, which is what post-processing calibration methods need. Preliminary experiments show that using synthetic data is at least as good as using standard validation data. However, these results might not scale to more complex data, which is harder to generate with fidelity.

1.5.3 Incorporating Textual Descriptions

So far, we have studied the XAI aspect in chapter 3, and looked into SC and calibration, which allow for better quantification of prediction uncertainties in chapter 4. Another aspect considered in chapter 5 is to include semantics: using text to describe the domain of proper classifier behavior. Recent works use text-to-image generative models based on diffusion models. These models can generate images corresponding to textual descriptions and identify which ones cause a classifier to fail. For example, it can be discovered that flies are sometimes classified as bees when they are next to a flower.

I first developed a method to improve an existing approach. This approach uses diffusion models to generate images corresponding to certain subgroups of a domain defined by textual attributes (weather, location, color, etc.). Due to the high inference time of these models, only certain subgroups can be evaluated. I developed a method based on

Bayesian optimization that quickly identifies potentially problematic subgroups. Thus, discovering sub-domains leading to classification failures is much more exhaustive and faster. This work has been published (Le Coz et al., 2024).

Describing failures with text is similar to the work of chapter 3: it helps understand classifier limits but does not help identify when the classifier is reliable. SC can define a reliability domain based on confidence score values but does not express much about which *kind* of data is selected or rejected. The final work in this thesis is to incorporate textual descriptions into SC to describe semantically when a classifier works reliably. I developed *Semantic Selective Classification*, a way to filter out data likely to be wrong by leveraging textual attributes of the data that are translated into a confidence score. The reliability domain can be described by a list of textual attributes. However, the approach requires massive annotated validation data, which is realistically possible only when using generative models. This means that the reliability domain remains in the world of synthetic images and cannot be validated with real images.

1.6 Publications and Code

Publications

(Le Coz et al., 2022) Adrien Le Coz, Stéphane Herbin, Faouzi Adjed. Leveraging generative models to characterize the failure conditions of image classifiers. *The IJCAI-ECAI-22 Workshop on Artificial Intelligence Safety (AISafety 2022)*, Jul 2022, Vienna, Austria. ⟨hal-03797490⟩

(Le Coz et al., 2023) Adrien Le Coz, Stéphane Herbin, Faouzi Adjed. Explaining an image classifier with a generative model conditioned by uncertainty. *Uncertainty meets Explainability — Workshop and Tutorial @ ECML-PKDD 2023*, Sep 2023, Torino, Italy. ⟨hal-04194943⟩

(Le Coz et al., 2024) Adrien Le Coz, Houssem Ouertatani, Stéphane Herbin, Faouzi Adjed. Efficient Exploration of Image Classifier Failures with Bayesian Optimization and Text-to-Image Models. *Generative Models for Computer Vision - CVPR 2024 Workshop*, Jun 2024, Seattle, United States. ⟨hal-04549384v2⟩

(Le Coz et al., 2025) Adrien Le Coz, Stéphane Herbin, Faouzi Adjed. Confidence Calibration of Classifiers with Many Classes. NeurIPS 2024, Dec 2024, Vancouver, Canada. ⟨hal-04767144⟩

Code

Most code written and used during the thesis can be found in the following GitHub repository: `https://github.com/allglc/phd/`

Chapter 2

Related Work

This thesis's subject is not a clearly defined field in existing literature. However, it is related to several existing fields, which I detail in this chapter. Defining a reliability domain would ensure the safe deployment of AI systems, so it is related to AI Safety, which is described in section 2.1. One of the thesis' goals is also to understand what makes a classifier fail, which provides insights into its behavior. This is also what the field of Explainable Artificial Intelligence aims to do, as mentioned in section 2.2. We have seen some high-level examples of failures in section 1.2 of the introduction; section 2.3 below provides technical details on failure sources. Related to identifying what makes a classifier work reliably, recent works referenced in section 2.4 aim to discover what makes a classifier fail. Selective Classification and calibration are research fields addressing the prediction of failures and uncertainty quantification; they are described in sections 2.5 and 2.6, respectively.

Section 2.7 also introduces the literature on generative models, which are used as a tool in this thesis. In particular, Generative Adversarial Networks are used to generate extreme examples in chapter 3, and text-to-image diffusion models are used to textually describe failures and the domain in chapter 5.

2.1 AI Safety

AI Safety is a field focused on ensuring that AI systems operate in safe, reliable ways and are aligned with human values. As AI systems become more powerful and integrated into various aspects of society, the risks associated with their deployment also increase. AI Safety aims to mitigate the risks of causing harm to humans. This thesis is closely related to AI Safety as it aims to identify the conditions for which given image classifiers are reliable. Reliable predictions ensure a safe and trustworthy ML system.

AI Safety has many concrete problems. (Hendrycks et al., 2021) identify four categories of problems. Robustness research aims to build systems that endure extreme, unusual, or adversarial events. Monitoring research aims to identify hazards, inspect models, and help human ML system operators for deployed models. Alignment research aims to safely optimize ML system objectives. Systemic safety research aims to address broader contextual risks to how ML systems are handled, e.g., cybersecurity. For reinforcement learning agents, safety problems include avoiding negative side effects and reward hacking, ensuring scalable oversight, safe exploration, and robustness to distribution shift (Amodei et al., 2016). Recent LLM based chatbots pose new risks, such as hallucinations, harmful content, disinformation, economic impacts, or overreliance, as

mentioned in GPT-4's system card (OpenAI, 2023). Such risks require specific mitigations during pre-training, training, and production. Discussions on AI safety also deviate towards speculative catastrophic and existential risks caused by superintelligent systems, which are inexistent today (Hendrycks et al., 2023).

Safety is of primary importance in some industries. Autonomous driving systems aim to provide a safe transportation method that can operate autonomously. Six levels of driving automation exist, from driver support to full autonomy (On-Road Automated Driving (ORAD) Committee, 2018). Notably, there is a need to define the conditions in which a system can safely operate within its given capabilities. A vehicle must detect when its capabilities do not guarantee a reliable operation and return to a safe state. This brings the notion of ODD, which comes with challenges such as creating a clear definition, updating, monitoring, or evaluating (Mehlhorn et al., 2023; Adedjouma et al., 2024; Czarnecki, 2018). Another industry with extremely demanding safety standards is the aviation industry. The typical software development assurance V-cycle has to be adapted to ML particularities (MLEAP Consortium, 2024). Challenges include data completeness and representativeness, model reliability, evaluation, robustness, and stability. Critical use cases include transcribing spoken instructions by air traffic controllers, automated visual inspection of aircraft, or an airborne collision avoidance system.

2.2 Explainability

DNNs are black boxes, meaning that their inner workings are largely unknown. They are large networks containing many layers of many parameters computing abstract representations of the data used to make a prediction. The field of Explainable Artificial Intelligence (XAI) provides methods to understand these objects better (Gunning et al., 2019; Arrieta et al., 2020). XAI is usually associated with high-level notions such as transparency, trustworthiness, fairness, etc. The work presented in chapter 3 is related to XAI. However, its goal is slightly different: it explains what makes the classifier fail rather than explaining how it makes its decision.

For computer vision, visual explanation techniques include saliency maps indicating the image regions most relevant to the model's decision (Selvaraju et al., 2020). However, they do not explain how non-localized visual attributes such as color or texture participate in the decision. This is what counterfactual explanations aim to do. They are alternative inputs for which the predicted class changes when minimal modifications are applied to the input visual features. An example is "if the animal's ears become pointy, the classifier increases the probability of predicting a cat rather than a dog". As illustrated in Figure 2.1 below, controllable generative models are well-suited for creating visual counterfactual explanations (Lang et al., 2021; Jeanneret et al., 2023; Augustin et al., 2023).

One limitation of XAI is to define what is a *good* explanation and whether it serves the intended user or not (Miller, 2019).

2.3 Uncertainty and Failure Sources

As mentioned in section 1.2, many failures can occur when DNNs are deployed in the real world. This section describes different possible sources of failures. Understanding them is essential in the context of the thesis, which addresses the notion of reliability domain. Many works from the research community aim to stop failures from occurring

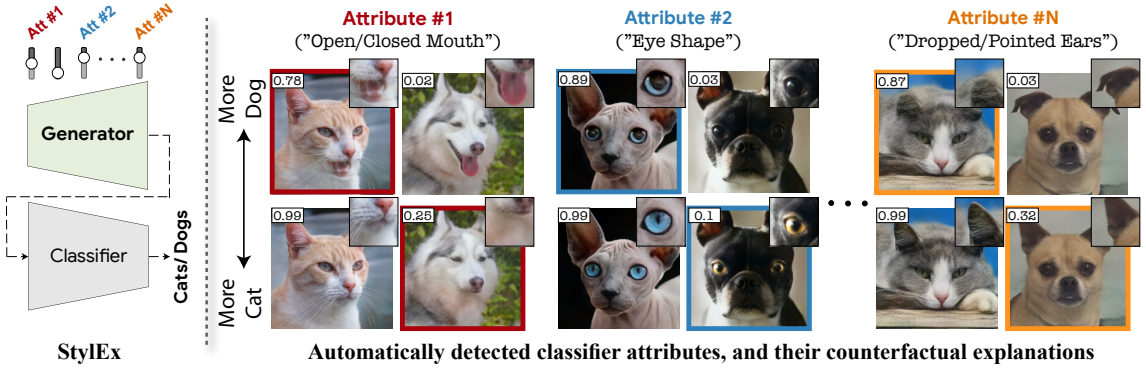


Figure 2.1: A controllable generative model permits the highlighting of the visual attributes impacting the classifier predicted probability for given classes. From (Lang et al., 2021).

by detecting them or robustifying the models. Unfortunately, several variations of the problem exist, each with its own goals, hypothesis, and set of methods.

Errors and prediction uncertainty come from either the data (aleatoric), the model (epistemic), or distribution shifts (Hüllermeier and Waegeman, 2021). Figure 2.2 represents graphically these uncertainty sources. Data uncertainty comes from the inherent randomness in the data distribution and cannot be reduced. For image classification, an example of data uncertainty is images that do not have a clear ground truth label because they contain multiple objects or the object category is ambiguous (some dogs look like cats). Model uncertainty is due to a lack of knowledge of a model and can be reduced by improving the model and training data. It can be decomposed into bias and variance. An example of model uncertainty due to bias is using a small neural network that does not have enough capacity to solve the task correctly. An overparametrized neural network can easily overfit its training data, and thus have a high uncertainty due to variance.

In practice, however, it is difficult to distinguish aleatoric and epistemic uncertainty. It is thus not clear what level of performance a perfect model can achieve. For ImageNet image classification, recent models outperform humans. See (Russakovsky et al., 2015) for quantization of human performance. At this level of performance, uncertainty is probably mostly aleatoric and thus difficult to reduce.

For clarity, failures coming from the model and the data are distinguished in the subsections below. However, in practice, failures often come from the combination of a model with the data. For instance, different models trained on the same data are more or less susceptible to covariate shifts or adversarial attacks.

2.3.1 Failure Sources in the Model

Failures caused by the model can arise from a wrong model architecture choice or suboptimal training. They can thus be reduced by improving model architecture and training procedures.

Architecture The lack of knowledge of a model can come from a wrong choice of architecture. For instance, convolutional neural networks are better adapted to process images than fully connected networks. Another key factor is model capacity: the model needs enough capacity to solve the task. Complex tasks require big networks with up

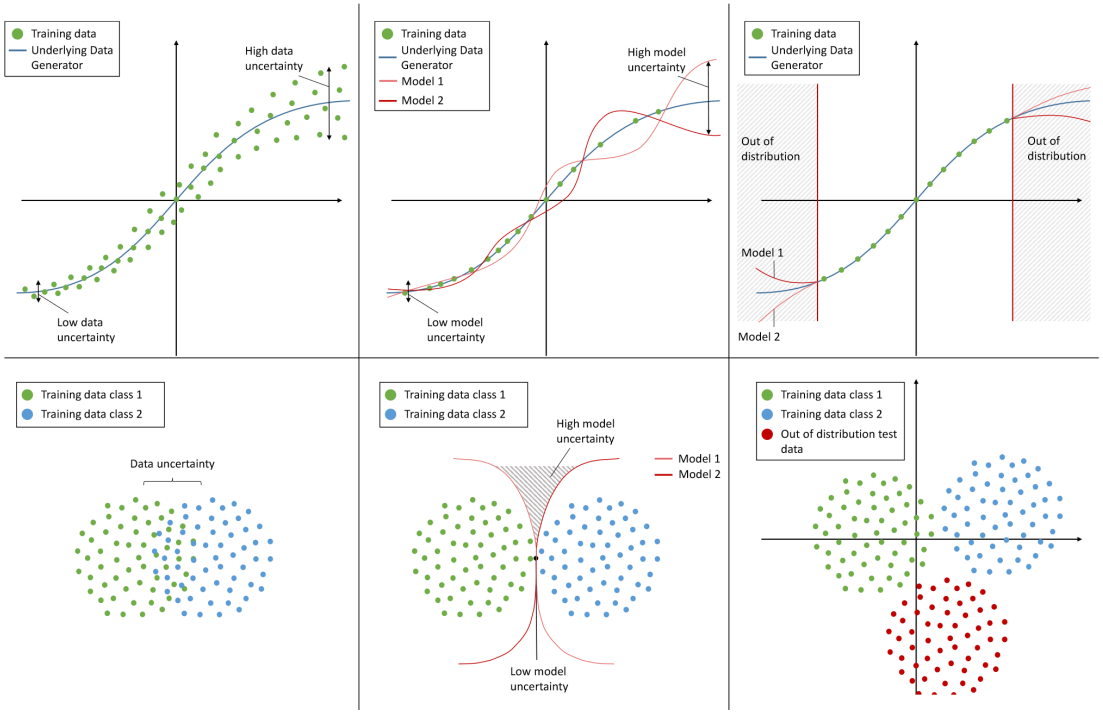


Figure 2.2: Illustration of different uncertainty sources for regression at the top and classification at the bottom. From left to right: data uncertainty, model uncertainty, and distribution shift. From (Gawlikowski et al., 2023)

to billions of learnable parameters (Kaplan et al., 2020). A good rule of thumb is that the network should have enough capacity to be able to overfit the data, and overfitting can be solved with regularization methods. For image classification, modern high-end GPUs easily handle state-of-the-art image classification models. However, for embedded devices such as smartphones, model size might be restricted (Howard, 2017).

Training The training process of a neural network is easy in principle but complex to get right. Hyperparameters such as the number of epochs, the batch size, the optimizer choice, or the learning rate have a big impact on the resulting model performance. Many machine learning practitioners are familiar with tricks to optimize the training process, using intuition and random or grid searches to find good hyperparameters. Also, DNNs need a lot of data to learn a task. For image classification, at least tens of thousands of labeled images are usually required before model accuracy saturation. Training a model with insufficient data results in a high model uncertainty as the model struggles to generalize to new data. Data augmentation techniques and synthetic data can help but hardly replace real data (Perez and Wang, 2017).

This thesis considers already trained classifiers and do not aim to improve them. Whatever architecture and training procedure, the goal is to evaluate model performance using a data-centric view.

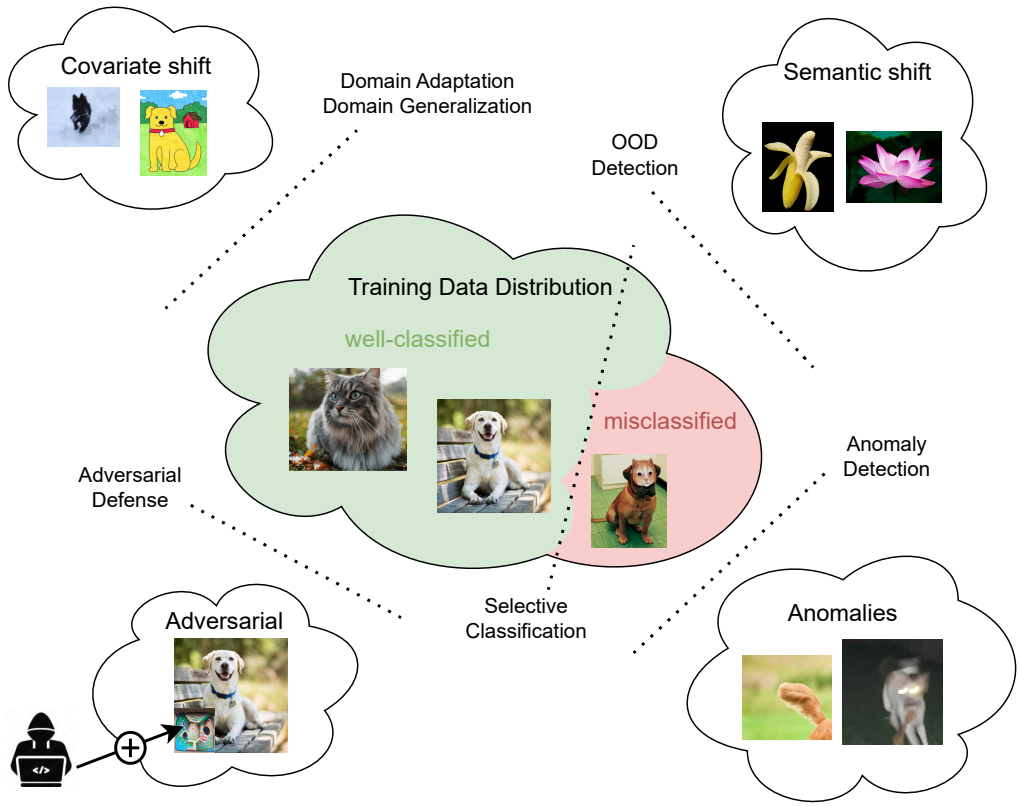


Figure 2.3: Non-exhaustive illustration of different failure sources described in this section and the research fields addressing them.

2.3.2 Failure Sources in the Data

Many failures are due to data characteristics, from anomalies to adversarial attacks. Many approaches were developed to detect problematic data to avoid making wrong predictions, as detailed below. See Figure 2.3 for an illustration.

Semantic Shift In general, training data comes from a given distribution, and test data might come from a different distribution. Training data might not be representative enough of the deployment environment. Semantic shift is when the distribution of the output labels $P(Y)$ changes, typically the model encounters new classes not seen during training. A few research fields tackle this problem, such as semantic anomaly detection (Ruff et al., 2021), novelty detection (Markou and Singh, 2003), open set recognition (Geng et al., 2020), and OOD detection (Hendrycks and Gimpel, 2017). These fields are quite related, and Yang et al. (2021) proposes to unify them into the notion of generalized OOD. The notion of OOD is widely used and sometimes with different meanings. In some cases, it refers to anomalies or rare samples. In most cases, contrary to the generality suggested by the expression, OOD refers exclusively to semantic shift. This is the definition used here.

A few baselines for OOD detection apply to pre-trained networks without needing access to OOD data. A straightforward way is to use a threshold on classifier confidence (Hendrycks and Gimpel, 2017). For large-scale multi-class settings, using the maximum

logits performs better (Hendrycks et al., 2022). Using an energy score is another option (Liu et al., 2020). Assuming that classifier features follow a class-conditional Gaussian distribution, a confidence score can be derived from the Mahalanobis distance between features of a test sample and the closest class-conditional Gaussian distribution. This allows detecting both OOD and adversarial samples (Lee et al., 2018b).

While many approaches only use in-distribution data, some explicitly use OOD data. ODIN follows this strategy and tunes coefficients, scaling the classifier probabilities to better detect OOD data (Liang et al., 2018). Outlier Exposure makes an even better use of available OOD data (Hendrycks et al., 2019).

Instead of improving an already trained classifier's ability to detect OOD data, one can train from scratch a classifier with the explicit goal of improving OOD detection. For instance, using specific loss functions and synthetic OOD samples from a GAN (Lee et al., 2018a). Assuming again that classifier features follow a class-conditional Gaussian distribution, synthetic outlier features can be sampled directly from the feature space and used to build native OOD into the model (Du et al., 2022). This work also extends the approach to object detection. (Du et al., 2024) leverage a generative model to improve OOD detection. A network learns how to represent training images in the text-conditioned latent space of the generating model. Sampling embeddings in the low-probability regions to condition the generation results in outlier images. These synthetic images are then used to develop an OOD detection tool.

The lack of a unified benchmark is an issue causing many evaluations of approaches to be unfair. A comprehensive comparison of OOD methods shows progress over the last few years, but also that DeepEnsemble (Lakshminarayanan et al., 2017) remains a top performer despite its age (Yang et al., 2022). Creating model-specific benchmarks allows better comparisons and insights (Galil et al., 2023a).

OOD detectors might not be well-suited as runtime monitors as they aim to detect the samples' data sources instead of detecting samples leading to errors (Guérin et al., 2023).

Covariate Shift Covariate shift is when the distribution of the input features $P(X)$ changes at test time; typically, the image quality might be different.

Domain Adaptation (DA) is a specific form of transfer learning that leverages labeled data from one or more related source domains to perform tasks in a target domain (Wang and Deng, 2018). It generally assumes that a source domain has sufficient labeled data to train a model, and that a target domain has limited labeled data (supervised), unlabeled data (unsupervised), or both (semi-supervised). For the unsupervised setting, the hardest one, some methods directly modify classifier training to align the model features across domains. For instance, a loss should be added to minimize the difference in learned feature covariances across domains during training (Sun and Saenko, 2016). Another option is to incorporate a domain classifier whose performance is minimized to ensure that features from the two domains are aligned and undistinguishable (Ganin et al., 2016).

Domain Generalization (DG) learn a model using data from a single or multiple related but distinct source domains in such a way that the model can generalize well to any target domain (Blanchard et al., 2011; Zhou et al., 2022). It does not require the strong assumption of DA: access to target domain data. One way is to minimize the dissimilarity across source domains, which improves the generalization on target domains (Muandet et al., 2013). Aligning the feature representation distributions across domains can be done adversarially (Li et al., 2018).

Besides semantic and covariate shifts, concept drift can also be a problem (Lu et al.,

2018). It happens when the relation between labels and features $P(Y|X)$ changes over time. For instance, consumer behavior changes over time, and recommendation systems should be updated accordingly. However, this is not a major issue in computer vision, especially for natural image classification. Visual features that define a dog will not define a cat in the future.

Adversarial Attacks DNNs are not robust to some specific small perturbations of their inputs. Adversaries can add some optimized invisible noise to an image to completely change the classifier prediction (Szegedy, 2013; Goodfellow et al., 2014b). This poses security issues for models operating on digital images, such as social network moderation tools. Fortunately, methods can detect if an attacker added invisible noise, e.g., (Lee et al., 2018b), and specific training regimes can make networks robust to such attacks (Madry et al., 2018; Cohen et al., 2019). Also, these attacks are not a threat in the physical world, as the attacker does not have access to the digital images captured by a camera (and if they do, adding an invisible noise is probably not the best attack method).

Adversarial patch attacks are more realizable in the physical world: adding an optimized motif in the scene can fool classifiers (Brown et al., 2017). For instance, optimized stickers applied on a stop sign in the physical world cause a classifier to misclassify it (Eykholt et al., 2018). Adversarial patches can also be used as a black-box attack, meaning that the attacker can build a patch without explicit knowledge of the attacked model (Labarbarie et al., 2024). Transforming images at test time, e.g., smoothing salient regions in the image (Naseer et al., 2019), is a possible way to defend against patches.

Anomalies and Corner Cases An anomaly is an observation that deviates considerably from some concept of normality (Ruff et al., 2021). However, what constitutes normality is unclear and often specific to an application or a method. Anomalies can be due to covariate or semantic shifts, thus overlapping with other categories of failures described above.

Errors can be introduced during the data acquisition process. For image classification, it can be that the image quality does not permit the clear identification of the object or that the label is ambiguous, e.g., if the image contains multiple objects. Labels can also be noisy (Northcutt et al., 2021).

In the field of autonomous driving, corner cases are a related notion. They usually denote situations occurring outside of the limits of normal operating parameters - specifically, when multiple conditions happening together lead to system failure. For instance, an autonomous car might function correctly under high rain or low luminosity but not when those two conditions happen together. In section 3.3, corner cases are viewed as examples at the boundary of the classifier's decision. Some works study corner cases in the context of Machine Learning. Many come from autonomous driving, where the notion of corner cases is widely discussed. However, a big issue is the lack of a clear definition of such a notion. Many works propose their own definition or use the term to denote related concepts (e.g., anomaly). (Pei et al., 2017) and (Tian et al., 2018) consider corner cases as "error-inducing inputs". (Ouyang et al., 2021a,b) see them as "rare conditions", "misclassified data", "analogous to bugs in traditional software", "related to the classifier boundary". They also propose a mathematical definition expressing that corner cases are samples susceptible to making the classifier prediction shift when a small perturbation is added. (Bolte et al., 2019) propose a specific definition for visual perception in autonomous driving: "A corner case is given, if there is a non-predictable relevant ob-

ject/class in relevant location". (Breitenstein et al., 2020, 2021) propose a systemization of corner cases for automated driving. They divide corner cases into multiple levels from low to high abstraction: from pixel level to scenario level. (Heidecker et al., 2019) extend the systemization to LiDAR and RADAR sensor modalities. (Heidecker et al., 2024) propose a definition associated with a mathematical formulation.

From the various studies, two main goals emerged. The first goal is to generate synthetic corner cases to evaluate a given model or augment the training data to improve robustness. (Pei et al., 2017; Tian et al., 2018) generate synthetic data that maximize neuron coverage and model behavior change. This data is generated by modifying existing images using a set of transformations such as rotation, blur, brightness, rain, fog, etc. Works in sections 3.3 and 3.4 are related to this first goal. The second goal is to detect corner cases to improve safety during deployment (warning system) and characterize datasets for better training and testing (corner cases should be seen during training, and evaluation data should include corner cases). (Ouyang et al., 2021a,b) develop a metric to detect corner cases. It can capture testing data corner cases (abnormal images) or adversarial examples. (Bolte et al., 2019) propose their own definition of corner cases and a method to detect them. (Breitenstein et al., 2020, 2021; Heidecker et al., 2021) aim to clarify the detection methods. A corner case dataset has been created to facilitate their detection for autonomous driving (Li et al., 2022b).

In chapter 3, synthetic extreme examples are generated. They might be considered as corner cases because they represent images close to the classifier decision boundary.

2.4 Classifier Failures Discovery

Recently, there has been a growing interest in detecting and describing failures or "bugs" in image classification models. Especially since recent tools allow for the description of these failures with text. Such tools usually leverage large multimodal models that allow linking images and text. A standard model for this task is CLIP (Radford et al., 2021). The works described in this section are closely related to the goals of the thesis, except that the point of view is the opposite: they study failures, while the thesis aims to study successes. Identifying some failures is useful, but it does not imply that all other cases lead to success.

One can use large labeled datasets and human verification to identify bugs (Gao et al., 2023). To avoid these requirements, other approaches are based on generative models. In particular, leveraging recent text-to-image generative models allows linking textual attributes to classification performance. It is possible to identify bugs in a given classifier by generating many images and then clustering and captioning the ones leading to classification failure (Wiles et al., 2022). For instance, the presence of a flower in the images increases the likelihood of misclassification of flies into bees. However, the required computing resources are enormous. (Vendrow et al., 2023) personalizes the generation to a specific dataset to create distribution-shifted versions of the dataset. They can be used to test classification models' robustness to shifts.

(Metzen et al., 2023) identifies subgroups of data leading to degraded performance. Starting from an ODD defined by domain experts and consisting of several semantic dimensions. An image classifier is tested on selected subgroups of this domain. Section 5.3 improves this work by deriving a guided and efficient exploration of the attributes.

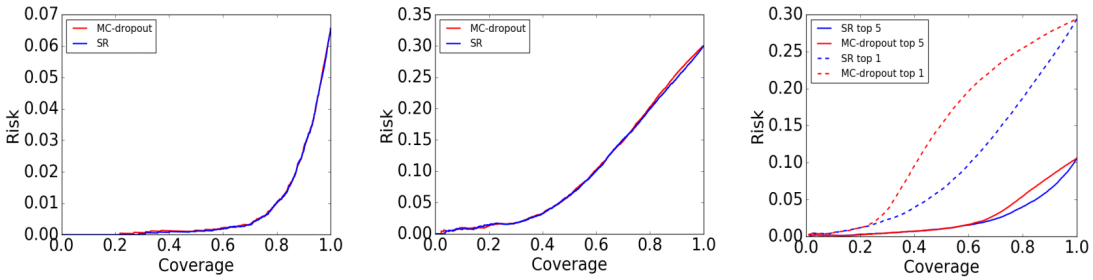
To identify systematic failures of multimodal models, (Tong et al., 2023) propose to first identify individual failures by finding sentences encoded similarly by the model

CLIP while they contain different information. An LLM categorizes these failures into groups of systematic failures. These failures are found in many multimodal models that use CLIP embeddings, such as text-to-image generators. Text representing absence or presence, negations, or quantifiers is not properly encoded and often leads to failures.

An approach by (Jain et al., 2023) uses linear classifiers to automatically represent model failure modes as directions in CLIP latent space. It enables the discovery and captioning of challenging subpopulations for targeted model improvement. (Chen et al., 2023a) leverages large pre-trained models to identify the important visual attributes of the task and then predict the attribute values. For instance, for lipstick classification, attributes include gender or age. Underperforming data slices can thus be detected and associated with natural language descriptions. Work by (Rezaei et al., 2024) emphasized the interpretability of the failures' text descriptions. The Recognize Anything Model (Huang et al., 2024; Zhang et al., 2024) is applied to tag images, followed by a brute-force approach to evaluate combinations of tags leading to failures.

Slice discovery algorithms are automated methods that partition the data into high-error and coherent (sharing attributes or characteristics) subsets. For image data, slices can be discovered with clustering in the classifier's feature space (d'Eon et al., 2022). Using cross-modal embeddings, e.g., from CLIP, helps identify and describe the error slices (Eyuboglu et al., 2022). However, the practical use of these slice discovery methods is still unclear: coherence of the output slices does not imply that users can form correct behavioral hypotheses, and different users form different hypotheses when shown the same slice (Johnson et al., 2023).

2.5 Selective Classification



(a) CIFAR-10 (b) CIFAR-100 (c) Image-Net

Figure 2.4: Risk-coverage curves for 3 image classification datasets (risk = 1-accuracy). Data samples are selected according to the softmax response (SR), i.e. MSP values. Filtering out, e.g., 50% of the data, significantly reduces errors in the selected data.

Selective Classification (SC), or classification with a reject option, aims to improve a model's prediction performance by trading off data coverage (Chow, 1957, 1970; El-Yaniv and Wiener, 2010; Geifman and El-Yaniv, 2017; Hendrickx et al., 2024). It can be evaluated with risk-coverage curves that plot this trade-off (see Figure 2.4). SC is done with a reject option to filter out data that might result in wrong predictions. In the case of a given DNN-based pre-trained classifier, Geifman and El-Yaniv (2017) studied how to learn a rejection function that filters out data such that the classification error rate is guaranteed to be below a desired value with a high probability. The rejection function

is based on a confidence-rate function, such as the softmax response (the maximum predicted probability output of a classifier, also called MSP or confidence). Geifman and El-Yaniv (2017) showed that the classifier confidence outperforms MC-dropout Gal and Ghahramani (2016) and Feng et al. (2023) showed that it also outperforms entropy. Using an auxiliary model that predicts the true class probability of the classifier can improve selective classification (Corbière et al., 2019, 2021).

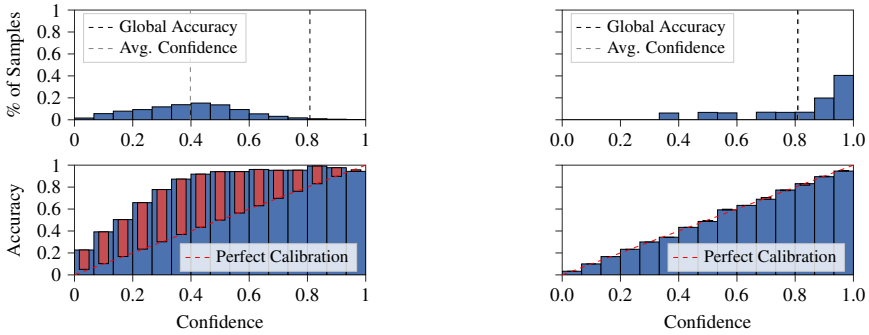
The SC performance of a model increases and then decreases during training (Geifman et al., 2019). The phenomenon seems similar to DNNs' overfitting. Averaging confidence score values from several earlier snapshots of the model thus increases SC performance, compared to only using the final trained version of the model.

Improving SC performance can be done by directly optimizing both classification and rejection during training to learn a deep neural network whose performance is maximized over the covered domain (for fixed target coverage, e.g., 80% of the dataset) (Geifman and El-Yaniv, 2019). However, the improved performance of methods involving a customized training might be due to training a more generalizable classifier rather than the proposed selection mechanism (Feng et al., 2023).

(Fisch et al., 2022) introduce a SC variation where the goal is to reject examples with "uncertain" uncertainties, i.e., that are miscalibrated. (Narasimhan et al., 2024) propose a combination of SC and OOD detection methods in one framework.

A large empirical study by Galil et al. (2023b) provides insights regarding the relation of SC and uncertainty quantification. Even though correlations between SC and calibration metrics can be positive or negative depending on the model architecture, the calibration method of Temperature Scaling (Guo et al., 2017) usually improves SC. The SC performance also highly depends on the model architecture (ViT (Dosovitskiy et al., 2021) being the best) and training regime (incorporating knowledge distillation (Hinton, 2015) has a positive impact).

2.6 Calibration



(a) Before post hoc calibration.

(b) After post hoc calibration.

Figure 2.5: Reliability diagrams for one ResNet-50 model trained on ImageNet. The model was originally particularly underconfident, and a post hoc calibration step solved the issue. Samples are binned according to classifier confidence values, and bin accuracies are computed. A post hoc calibration step improves the calibration, with bin accuracies closer to the perfect calibration target (plots at the bottom) and average confidence closer to global accuracy (plots at the top).

Confidence Calibration To improve DNNs's reliability, a key concern is to estimate the probability of wrong decisions correctly. When models are expected to be embedded in safety-critical systems (e.g., medical or transportation), estimating this probability is crucial to mitigate catastrophic behavior. One way to address this question is to treat it as an uncertainty quantification problem (Abdar et al., 2021; Gawlikowski et al., 2023), where the uncertainty value computed for each prediction is considered as a confidence. This confidence can be used to reject uncertain decisions proposed by the DNN (Geifman and El-Yaniv, 2017), for out-of-distribution detection (Hendrycks and Gimpel, 2017), or to control active learning (Li and Sethi, 2006) or reinforcement learning based systems (Zhao et al., 2019). When confidence values reliably reflect the true probability of correct decisions, i.e., their accuracy, a predictive system is said to be *calibrated*. For all samples predicted with a confidence of 0.8, a calibrated model makes correct predictions 80% of the time. In this case, confidence values can be used to reliably control decision-making.

DNNs outputs can be used to provide a confidence score at no cost, i.e., without necessitating heavy estimation such as Bayesian sampling (Goan and Fookes, 2020) or ensemble methods (Lakshminarayanan et al., 2017). Indeed, most neural architectures for classification instantiate their decision as a softmax layer, where the maximum value can be interpreted as the maximum of the posterior probability and, therefore, as a confidence. Unfortunately, the uncertainty values computed in this way are often miscalibrated. DNNs have been shown to be over-confident (Guo et al., 2017), meaning their confidence is higher than their accuracy: predictions with 90% confidence might be correct only 80% of the time. A later study (Minderer et al., 2021) suggests that model architecture impacts calibration more than model size, pre-training, and accuracy. For ImageNet classifiers, model families are correlated to calibration (Galil et al., 2023b).

These studies show that it is difficult to anticipate the calibration level of confidence values computed directly from DNNs. Improving calibration can be done with a specific training regime or a complementary post-processing calibration step. This calibration process can be seen as a learning step that exploits data from a calibration set, distinct from the training set, and is used to learn a function that maps classifier outputs into better-calibrated values. This process is typically lightweight and decoupled from the issue of improving model performance.

Calibration shares similarities with SC: they both aim to better estimate the prediction uncertainties. While SC only requires a confidence score that provides a ranking of predictions related to their correctness, calibration requires that the absolute value of the confidence score represents the prediction uncertainty. Calibration does not directly help define an reliability domain contrary to SC, but insights gained working on SC led to a new approach for calibration described in section 4.4.

Calibration Notions There are various notions of multiclass calibration. One can consider confidence (Guo et al., 2017), class-wise (Kull et al., 2017), top-r (Gupta et al., 2021), top-label (Gupta and Ramdas, 2022), decision (Zhao et al., 2021a), projection smooth (Gopalan et al., 2024), or strong (Vaicenavicius et al., 2019; Widmann et al., 2019) calibration. For recent surveys, see (Filho et al., 2023) and (Wang, 2023). Section 4.4 focuses on confidence calibration and not on the calibration of the full probability vector. Indeed, confidence calibration is useful for many applications: SC (Geifman and El-Yaniv, 2017), OOD detection (Hendrycks and Gimpel, 2017), or active learning (Li and Sethi, 2006). For these applications, stronger notions of calibration are both difficult and even useless as only a single confidence value is required.

Metrics Calibration can be visualized with reliability diagrams. Figure 2.5 shows an example. It plots the accuracy as a function of confidence for two different situations. Predictions are grouped into equal-size bins according to the confidence values, and accuracies are calculated for each bin. For a perfectly calibrated model, the diagram shows a function close to the identity: the bin accuracy equals the average bin confidence.

Several metrics have been proposed to quantify calibration error. The most common is the Expected Calibration Error (ECE) (Naeini et al., 2015) (see Equation 4.2). ECE has flaws: the estimation quality is influenced by the binning scheme, and it is not a proper scoring rule (Gneiting and Raftery, 2007; Vaicenavicius et al., 2019; Nixon et al., 2019). This means that the ECE can be minimized even for a predicted uncertainty far from the true uncertainty, e.g., for a network that always predicts a probability equal to the accuracy. Despite its flaws, ECE remains the standard comparison metric for confidence calibration. Variants of ECE have also been developed: classwise-ECE (Kull et al., 2019), ECE with equal mass bins (Nixon et al., 2019; Minderer et al., 2021), or top-label-ECE, which adds a conditioning on the predicted class (Gupta and Ramdas, 2022). The Brier score (Brier, 1950) is also used to measure calibration. This work mainly uses the standard ECE, but more metrics are included in the Appendix.

Training Calibrated Networks Several solutions have been proposed in the literature to improve calibration by training neural networks in specific ways, generally by making use of a new loss term (Kumar et al., 2018; Thulasidasan et al., 2019; Karandikar et al., 2021; Cheng and Vasconcelos, 2022; Chen et al., 2023b). While these methods directly optimize calibration during the training phase of the networks, they require a high development time, often compromise accuracy, and are not adapted to pre-trained foundation models. That is why the thesis focuses on calibrating already-trained models.

Post-Processing (or Post Hoc) Calibration Another family of methods aims to calibrate already-trained models. This lowers the development time because it decouples the accuracy optimization and the calibration. Note that the term "calibration" designates both the model property defined above and the action of calibrating a model, i.e., improving its calibration. In this thesis, calibration methods are divided into two categories: scaling and binary.

Scaling methods are derived from Platt scaling (Platt, 1999) and optimize some parameters to scale the logits. Temperature Scaling is a popular simple post-processing calibration method. The logits vector is scaled by a coefficient, which modifies the probability vector in a non-linear way. Vector Scaling is more expressive and has good performance in many cases (Guo et al., 2017; Nixon et al., 2019; Kull et al., 2019). Matrix Scaling can also be considered for more expressiveness but is difficult to apply without overfitting (Guo et al., 2017). Dirichlet Calibration (Kull et al., 2019) proposes a regularization strategy for Matrix Scaling. (Zhang et al., 2020) developed Ensemble Temperature Scaling. Scaling can be combined with binning (Kumar et al., 2019a). Instead of using the network logits or probabilities, feature layers can also be exploited (Lin et al., 2022).

Another family of methods tackles binary classification. They are denoted as binary methods. Histogram Binning (Zadrozny and Elkan, 2001) divides the prediction into B bins according to the predicted probability. For each bin, a calibrated probability is computed from the calibration data. The probability becomes discrete: it can only take B values. With some modifications, it outperforms scaling methods (Gupta and Ramdas, 2022; Patel et al., 2020). Isotonic Regression (Zadrozny and Elkan, 2002) learns a piece-

wise constant function to remap probabilities. Bayesian Binning into Quantiles (Naeini et al., 2015) brings Bayesian model averaging to Histogram Binning. Beta Calibration (Kull et al., 2017) uses a beta distribution to obtain a calibration mapping.

In chapter 4, the Top-versus-All (TvA) approach reformulates the multiclass calibration problem to more efficiently use all these calibration methods, with little to no change in their algorithms.

Multiclass to Binary Using binary calibration methods for a multiclass classifier requires adapting the multiclass setting. This is usually done with a One-versus-All (OvA) approach (Zadrozny and Elkan, 2002; Guo et al., 2017). The multiclass setting is decomposed into L OvA independent problems: one binary problem for each class. (Gupta and Ramdas, 2022) introduce the notion of top-label calibration, i.e., confidence calibration with additional conditioning on the predicted class (top-label). They describe a general multiclass-to-binary framework to develop top-label calibrators. (Cheng and Vasconcelos, 2022) derive $L(L-1)/2$ pairwise binary problems. The approach requires training the classifier from scratch, and its performance decreases with the number of classes.

The methods I-Max (Patel et al., 2020) and IRM (Zhang et al., 2020) use a shared class-wise strategy to compute a single calibrator. The calibrator is applied to all class probabilities separately, which means that the class probabilities ranking might change, and so does the prediction.

The TvA approach developed in chapter 4 tackles this multiclass-to-binary problem.

2.7 Tools

Generative models are an essential tool for most of the thesis' contributions. By generating images not present in the original data, new insights on a classifier can be gained.

Chapter 3 leverages Generative Adversarial Networks to generate extreme examples for a given classifier. These models are introduced in subsection 2.7.1 below and described more technically in section 3.2.

Text-to-image models based on diffusion models allow for describing classifier failures and a domain with text in chapter 5. They are introduced in subsection 2.7.2 below and discussed in greater technical detail in section 5.2.

2.7.1 Generative Adversarial Networks (GANs)

GANs Generative Adversarial Networks (GANs), introduced by (Goodfellow et al., 2014a), have rapidly become a cornerstone of deep learning research due to their ability to generate realistic data in various domains, particularly in image synthesis. The original GAN framework involves a minimax game between a generator, which creates synthetic data, and a discriminator, which attempts to distinguish between real and generated data. This adversarial process is described in more detail in section 3.2

Building on the original framework, numerous variants have been proposed to address specific limitations and expand the capabilities of GANs. For instance, the generation can be conditioned by a class label (Mirza and Osindero, 2014). Deep Convolutional GANs (DCGANs) leverage convolutional layers to improve stability during training and produce higher-quality images (Radford, 2015). The Wasserstein GAN (WGAN) addresses the instability and mode collapse issues inherent in traditional GAN training (Arjovsky

et al., 2017). BigGAN scales-up GANs to improve the fidelity of class-conditional image synthesis for ImageNet at 128 $\times$ 128 resolution.

The StyleGAN family of models also pushed the boundaries of synthetic image quality. The original StyleGAN architecture, introduced by (Karras et al., 2019), offers a novel approach to controlling the style and content of generated images by disentangling different levels of detail. StyleGAN's architecture allows for scale-specific control, resulting in high-quality images with unprecedented levels of detail. In particular, synthetic faces can be indistinguishable from real photos; see Figure 2.6. The original architecture was then further improved (Karras et al., 2020b), made more data-efficient (Karras et al., 2020a), and alias-free for better video and animation capabilities (Karras et al., 2021). However, despite high-quality results for structured images such as faces, the performance is degraded for large unstructured datasets such as ImageNet. StyleGAN-XL aims to solve this issue by scaling up the model (Sauer et al., 2022). Another variant introduced text conditioning to make a GAN-based model competitive with recent text-to-image diffusion models that became the new trend for image synthesis (Sauer et al., 2023).

Chapter 3 uses the StyleGAN2 architecture for its image quality and generation controllability through latent space manipulation, as described in section 3.2.

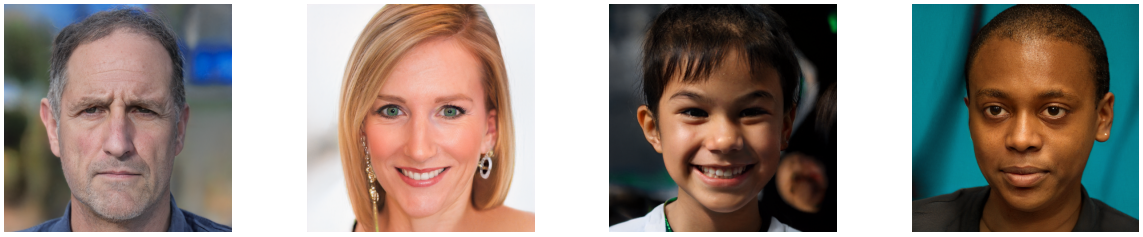


Figure 2.6: Curated examples of synthetic faces generated by StyleGAN2 trained on FFHQ dataset (Karras et al., 2019). From GitHub.

Image Editing Besides generating high-quality images, GANs offer the possibility of image editing, thanks to latent space manipulation. The latent spaces of GAN models frequently contain semantically meaningful directions. Navigating along these directions allows for human-interpretable image transformations, such as zooming or recoloring. Such directions can be discovered with various methods (Plumerault et al., 2020; Jahanian et al., 2019; Voynov and Babenko, 2020; Härkönen et al., 2020). Many methods specially adapted for the StyleGAN architecture, such as (Shen et al., 2020; Wu et al., 2021; Patashnik et al., 2021; Ling et al., 2021; Alaluf et al., 2022a; Cui et al., 2022). The main application is image editing, i.e., the ability to control the generated images semantically as shown in Figure 2.7.

Chapter 3 takes inspiration from the field of image editing to illustrate a classifier's failures.

GAN Inversion As seen above, image editing with GANs is typically performed by manipulating latent codes. When the starting point is sampled in the latent space, a synthetic image is edited. To edit a real image, its latent code needs to be known, and GAN inversion is about finding this code. This is a way to bridge the gap between real and synthetic data. For instance, to edit a picture of my face, I first need to find the latent code that would faithfully reconstruct the picture and then edit this code. See (Xia et al., 2022) for a survey.

Figure 2.7: Image editing with latent space manipulation. For each group, two independent manipulations are combined. It demonstrates the latent space disentanglement, as visual attributes are represented along specific latent dimensions. From (Wu et al., 2021).

Because training GANs is often complex and time-consuming, GAN inversion uses pre-trained models. Some approaches train an encoder on pairs (images – latent codes) obtained by sampling latent codes and generating the associated images, (Tov et al., 2021) propose an encoder regularized to control the proximity of the inversions to regions that StyleGAN was originally trained on. (Alaluf et al., 2021) develop an encoder iteratively predicting residuals to add to the latent code.

Another way is to use optimization methods to find the latent code, minimizing the error between the target image and the generated image. The original inversion method for StyleGAN2 uses optimization to find latent codes in W (Karras et al., 2020b). Hybrid approaches encode an image into a latent code used as a starting point for optimization (Zhu et al., 2016).

Different approaches are also possible. (Dinh et al., 2022; Alaluf et al., 2022b) use a hypernetwork that predicts how to modify the generator weights to generate the target image, starting from an initial latent code obtained by an encoder. (Parmar et al., 2022) inverts complex images and allows editing by using different latent layers for different regions of the images. Pivotal Tuning Inversion finds an initial latent code by optimization and then fine-tunes the generator with regularization so that the generated image gets closer to the target image (Roich et al., 2022). It is also possible to invert OOD images by combining the edited part (face) with original details, such as a specific background (Song et al., 2022).

The limitations of GAN inversion techniques are discussed in section 3.5.

Adversarial Autoencoders Related to GAN inversion, which operates on pre-trained models, another family of approaches aims to learn an encoding jointly with the generator. Such approaches should provide better reconstruction results as the generator is constrained by the encoder.

(Makhzani et al., 2015) propose an autoencoder architecture containing an encoder that projects data in a latent space, a decoder that reconstructs data from latent codes, and a discriminator. The discriminator predicts whether a latent code comes from an encoded image or from the target prior distribution. The discriminator and the encoder (which generates "fake" latent codes) are trained to minimize a standard adversarial loss, as in GANs. A reconstruction loss is used to train the encoder and decoder. The resulting models can be used to disentangle the style and content of images, perform unsupervised clustering, reduce dimensionality, and visualize data. The concurrent works of (Donahue

et al., 2017) and (Dumoulin et al., 2017) also propose an architecture comprising an encoder, a decoder, and a discriminator. However, there is no need for a reconstruction loss. The discriminator now receives pairs of either (image, encoded latent code) or (fake image, latent code). The adversarial loss for this discriminator optimizes both the encoder to generate realistic latent codes and the generator to generate realistic images. Such joint encoder architectures can be scaled up to BigGAN (Donahue and Simonyan, 2019) or StyleGAN (Pidhorskyi et al., 2020).

Related to section 3.3 of this thesis, (Lang et al., 2021) use an adversarial autoencoder constrained by a given classifier to explain the classifier's decision process. Figure 2.8 describes the architecture.

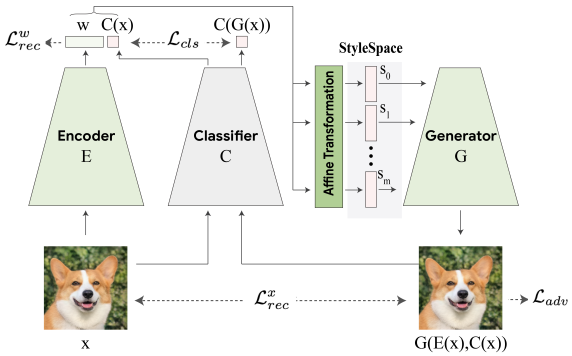


Figure 2.8: StyleEx architecture. The encoder, generator, and discriminator (not shown) are jointly trained. Knowledge of the classifier is embedded into the encoder and generator through specific loss functions. From (Lang et al., 2021)

2.7.2 Diffusion and Text-to-Image Models

Diffusion Models A family of generative models called *diffusion models* has been proposed in recent years and has achieved remarkable results. Nowadays, many tech companies have released text-to-image generation products, such as DALL-E from OpenAI based on this model. The original idea is inspired by non-equilibrium statistical physics (Sohl-Dickstein et al., 2015). A forward diffusion process iteratively destroys the data distribution structure, typically by adding Gaussian noise. The reverse diffusion process, restoring the data structure, is learned. This generative model can progressively transform complete noise into structured data. The idea was improved by (Song and Ermon, 2019; Ho et al., 2020). Denoising Diffusion Probabilistic Models (DDPMs) are a simpler variation of diffusion models. During training, images are sampled and corrupted with random noise and a random timestep (which describes the amplitude of the corruption). The neural network learns to predict the noise that was added to the data by minimizing the mean squared error between the predicted and real noise. When a small timestep is chosen, the data is slightly corrupted, making the task easy for the network. But for high timesteps, the data is barely recognizable. When sampling, a random starting point is sampled and progressively restored by the network. To generate an image, the model must perform the denoising operation many times, e.g., 1000. This causes diffusion models to be slow for sampling a distribution, compared to, e.g., GANs, which only need one forward pass of the network. (Nichol and Dhariwal, 2021) also learn the noise variance and use a cosine noise schedule to improve the original DDPM. They also studied the scalability and showed that more compute leads to better sample quality. (Dhariwal and Nichol, 2021)

improved the architecture of diffusion models, which made them outperform GANs and Variational Autoencoders (Kingma and Welling, 2013) in image generation for ImageNet. In particular, time-dependent classifiers guide the reverse process to control the sample fidelity versus diversity tradeoff. The same benefit can also be achieved with classifier-free guidance (Ho and Salimans, 2022). Work has been done to accelerate the sampling. For instance, (Song et al., 2021a) makes the sampling more than 10 times faster, with a trade-off on generation quality.

Controlling the Generation An important characteristic of generative models is their ability to *conditionally* generate data. A first type of conditioning is class conditioning. Class information can be incorporated into normalization layers during training (Dhariwal and Nichol, 2021) to create a conditional model. At test time, the reverse diffusion process can be guided towards a given class (Dhariwal and Nichol, 2021; Song et al., 2021b; Ho and Salimans, 2022)

Textual conditioning allows for generating complex and diverse images by prompting them with text. Text information can be incorporated into the model through an embedding used in a similar way as a class embedding, with classifier-free guidance controlling the conditioning strength (Nichol et al., 2021). Using CLIP (Radford et al., 2021) as a pre-trained frozen text encoder improves the generation quality (Ramesh et al., 2022; Rombach et al., 2022). However, the numerous multimodal models based on CLIP all share the same systematic failures (Tong et al., 2023). Another option is to use a pre-trained frozen LLM as text encoder (Saharia et al., 2022). A major limitation of text-to-image models is their need for enormous amounts of image-text pairs data, such as the LAION-5B dataset (Schuhmann et al., 2022), which contains billions of examples. More details for text-to-image models are found in section 5.2.

Besides text, conditioning can be segmentation maps, canny edges, or human pose (Zhang et al., 2023) as seen in Figure 2.9. The generation can also be conditioned by a concept illustrated by some example images. The images can be used to fine-tune the generative model (Ruiz et al., 2023). Another way is to find the textual embedding representing these images (Gal et al., 2022).

Chapter 5 leverages the Stable Diffusion model (Rombach et al., 2022), which makes high-quality text-to-image generation accessible as it was published in open-source.

Figure 2.9: Stable Diffusion can be controlled by text and ControlNet adds conditions such as canny edges and human poses through fine-tuning. From (Zhang et al., 2023).

Chapter 3

A Qualitative View: Use Generative Models to Illustrate Failures

3.1 Introduction

The growing use of image classifiers in many, sometimes critical, applications (e.g., medical diagnosis, autonomous driving, autonomous aircraft landing) reinforces the need to understand their behavior. A key issue is to identify the causes for which such systems are likely to fail to ensure the safety of their use. A related issue is to describe the nature of uncertain data for a given classifier, as one can consider uncertainty as a measure of potential failure.

XAI is currently considered as a tool to improve the trustworthiness of AI predictive systems (Arrieta et al., 2020; Linardatos et al., 2021). Explainability studies have mainly focused on providing so-called "post-hoc" explanations that are expected to somehow justify the actual prediction of a trained model. Very few studies have addressed the issue of identifying failure conditions. A related explanatory strategy is the design of counterfactuals (Wachter et al., 2017; Goyal et al., 2019), which aim to identify what minimal and meaningful input modification leads to a desired prediction change. In particular, several works (Zhao et al., 2018; Sauer and Geiger, 2021; Lang et al., 2021; Jeanneret et al., 2023) leverage generative models such as Generative Adversarial Network (GAN) or diffusion models. Generative models have also been used to quantify the uncertainty of a classifier (Oberdiek et al., 2022) or to discover causes of failures (Wiles et al., 2022).

In this chapter, the main goal is to leverage generative models to identify the failure conditions and reasons for uncertainty of a given image classifier. Such generative models can generate infinite amounts of failure cases or uncertain data (as seen by the classifier) and provide a representation – an explanation – of what makes some data hazardous for the classifier.

Section 3.2 introduces the generative models used. It first describes the standard GAN, then the StyleGAN model and its latent space properties, which are exploited in the chapter.

Section 3.3 addresses the question of identifying the failure conditions of a given image classifier. To do so, I exploit the capacity of producing controllable distributions of high-quality image data made available by recent GANs (StyleGAN2): the failure conditions are expressed as directions of strong performance degradation in the generative

model latent space. This analysis strategy is used to discover extreme examples that combine multiple sources of corruption and to compare, in more detail, the behavior of different classifiers. The directions of degradation can also be rendered visually by generating data for better interpretability. Some degradations, such as noise, can affect all classes, whereas others, such as shape, are more class-specific. The approach is demonstrated on the MNIST dataset, which has been completed by two sources of corruption: noise and blur. It shows a promising way to better understand and control the risks of exploiting artificial intelligence components for safety-critical applications.

Section 3.4 tackles a related challenge: identifying the sources of uncertainty in an image classifier. Here, a GAN generates images conditionally to the classifier decision. More specifically, I consider the classifier maximum softmax probability as an uncertainty estimate and use it as an additional input to condition the generative model. The generative model is trained with this additional condition representing the classifier behavior. This allows the generation of images that result in uncertain predictions, giving a global view of which images are harder to classify. Increasing the uncertainty of a given image illustrates the impact of an attribute, providing a more local understanding of the decision process. The experiments performed on the MNIST dataset, augmented with corruptions, are described in subsection 3.3.3.

Section 3.5 discusses the main findings, proposes a way to define a domain in a latent space, and argues why it happens to be difficult to achieve.

3.2 Background

Generative Adversarial Networks (GANs) GANs are a type of generative model known for their generation quality and the controllability offered by their latent (input) space. In particular, they were shown to generate full-size images with high rendering quality. See chapter 2 for more information. They are composed of two neural networks. A generator G generates fake images from an input noise vector z. A discriminator D distinguishes fake images from real ones from the training data, with $D(x)$ being the predicted probability that x is a real image. The training is a competition between the two: the generator tries to fool the discriminator, which in return seeks not to be fooled. The discriminator aims to assign a probability of 0 for a fake image $G(z)$ and a probability of 1 for a real image x. More formally, the generator is trained to minimize the loss $\log(1 - D(G(z)))$, while the discriminator is trained to minimize the loss $\log(1 - D(x)) + \log(D(G(z)))$.

During the training process, the two neural networks improve each other until the discriminator no longer distinguishes between real and fake data: it assigns 0.5 probability to both real and fake images. GAN training is known to be unstable, so an equilibrium has to be found: if the discriminator becomes much better than the generator, it "wins", and it is hard for the generator to improve and fill the gap, and vice-versa. Several losses and regularization terms have been proposed to fix the issue (Arjovsky et al., 2017).

After training, only the generator is used to generate images from noise vectors ("latent codes"). Interestingly, the generator is structured so that interpolations between two latent codes result in a smooth semantic shift of an image into another; for instance, a digit image of "8" is progressively transformed into a mixture of "1" and "8" before ultimately becoming a "1", as shown in Figure 3.6. This is not the case if the interpolation is done in the pixel space.

StyleGAN This chapter uses in particular the model StyleGAN2, widely used for high-quality face generation and edition. It has a unique architecture. Indeed, three different "levels" of latent spaces can be considered. The first latent space, $\mathcal{Z}$, is typically normally distributed like many GANs and is the initial input space of the generator. Samples $\mathbf{z} \in \mathcal{Z}$ are forwarded to an intermediate latent space $\mathcal{W}$ using fully connected layers, resulting in a more disentangled representation than $\mathcal{Z}$ (Karras et al., 2019). Using learned affine transformations, samples $\mathbf{w} \in \mathcal{W}$ are specialized into *styles* $\mathbf{s}$. The space of styles, called StyleSpace, shows a high degree of disentanglement (Wu et al., 2021). Latent spaces $\mathcal{W}$ and $\mathcal{S}$ are highly disentangled, meaning that they encode distinct visual attributes along different dimensions. This allows image editing, one attribute at a time (Wu et al., 2021).

A generated image results from an initial learned constant tensor that is progressively up-sampled and transformed by residual convolution layers. The style vectors modulate the convolution weights of each feature map for each generator layer. Styles applied at low resolution affect high-level aspects (e.g., pose, hairstyle of a face), while at higher resolutions, they affect small details (microstructure). The style vector determines what the generated image looks like. The architecture is illustrated in Figure 3.1.

To give an idea of the complexity of the generative model, in the original StyleGAN2 version that generates images of size 1024^2, $\mathcal{Z}$ and $\mathcal{W}$ have 512 dimensions, $\mathcal{S}$ has 9088 dimensions, and the initial constant tensor has a size of 4^2 with 512 channels. This makes a total of ≈ 25 million parameters.

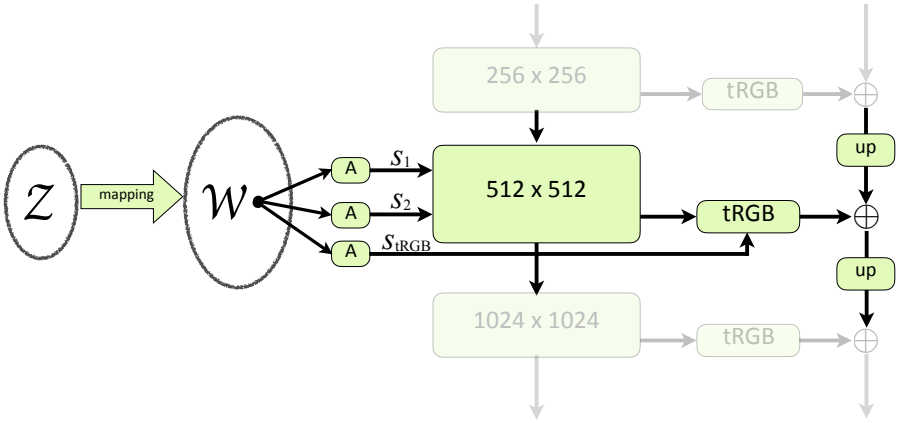


Figure 3.1: Illustration of StyleGAN architecture. The gaussian latent space $\mathcal{Z}$ is mapped to an intermediate latent space $\mathcal{W}$ which is projected to styles $\mathbf{s}$ which modulate convolution weights. tRGB blocks project tensors of arbitrary channels into RGB images that are upsampled and combined to form the final image. From (Wu et al., 2021).

3.3 Generate Synthetic Failure Cases

3.3.1 Generative Models to Explore the Data Space

Designing a probabilistic model in high-dimensional data space, such as image, video, or sound, that is able to faithfully account for their diversity and informative features is a difficult objective. Generative models such as GANs or generative invertible flows (Kingma and Dhariwal, 2018) are ML techniques that provide means to give access to such a distribution by direct sampling. They learn the parameters of a sampling process and are able to generate data that mimic a given random distribution.

GANs exploit a representational *latent space* that can be sampled from a known low-dimension distribution, often Gaussian, that is expected to encode enough information to generate complete images. Generation is then produced by a decoding network that is learned from target data samples. Recent approaches (Wang et al., 2021; Jabbar et al., 2021; Saxena and Cao, 2021) can now generate high-quality, high-dimension data with a photo-realistic rendering when applied to images and with good diversity and fidelity levels. One possible application of generative models for safety objectives is to augment data for testing various operational conditions as in (Zhang et al., 2018).

The latent space can also be used as a way to control the generation process, for instance, to edit images (Härkönen et al., 2020; Wu et al., 2021; Ling et al., 2021). When correctly disentangled, the latent space can be interpreted as a representation space where each dimension encodes some interpretable visual attribute (Wu et al., 2021; Patashnik et al., 2021). In the case of face image generation, these attributes could be hairstyle, head orientation, eye color, glasses, etc. Navigating in the representational latent space can also be used to identify the attributes that best characterize a given class (Lang et al., 2021).

3.3.2 GAN's Latent Space Exploration Guided by a Classifier's Gradient

The proposed approach is based on the exploration of how the latent space of a generative model differentiates between well and poorly-classified data by a given classifier, as illustrated by Figure 3.2. In the following, I briefly describe the chosen generative model and its latent space structure; explain how to find the dimensions of the latent space that differentiate well-classified from misclassified data; describe how to manipulate images to visualize the attributes; and see how one can estimate the accuracy of the classifier conditionally to the location of the data in the latent space.

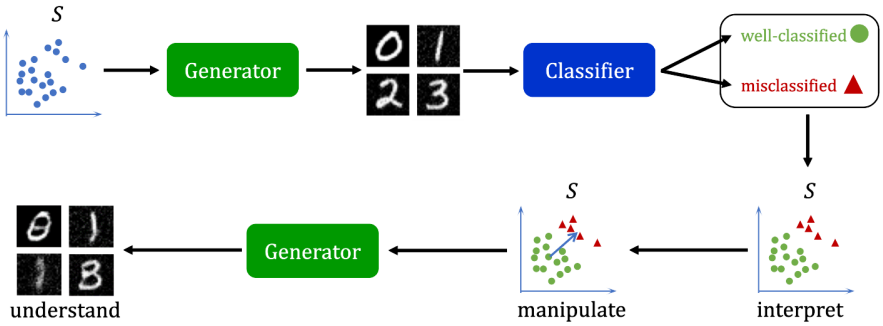


Figure 3.2: An illustration of the proposed approach. Starting from the latent space $\mathcal{S}$ of StyleGAN, a population of images is generated. The images are classified and the information on classification success is added in the space $\mathcal{S}$, where the dimensions discriminating well-classified vs. mis-classified images are found. These dimensions can then be used to visually render the corresponding influential attributes.

Classifier and Data The first input of the approach is a learned image classifier C to be analyzed. The approach assumes having access to model architecture and weights ("white box" hypothesis). The application domain is handwritten digit recognition, and a corresponding dataset is available (which was not necessarily used for learning the classifier).

Generative Model The second ingredient of the approach is a generative model that can be controlled meaningfully. The StyleGAN2 architecture, described in section 3.2, is used. This choice is motivated by several reasons: the quality of generated data, the scalability to complex datasets, and the high degree of disentanglement of the latent space StyleSpace. This disentanglement is particularly useful here, where it is exploited to identify directions of classification performance degradation.

Finding Influential Dimensions in the Latent Space The dimensions of the latent space $\mathcal{S}$ are expected to encode image attributes, such as shape, thickness, orientation, and noise, in a rather disentangled way. This property allows defining a simple search method that is able to identify the most influential dimensions regarding the accuracy of a given classifier.

Gradient-Based Approach The proposed strategy ranks the dimensions according to the gradient of the classifier output with respect to the StyleSpace input. The idea is to score each dimension based on its ability to lower the output score of the true class. More precisely, for each sample $\mathbf{s}$ in the StyleSpace, for which the true class is known, the corresponding image $\mathbf{x} = G(\mathbf{s})$ is generated, and then classified according to $C(\mathbf{x})$. Then we can compute the gradient with respect to the dimension j in the style space of the i-th classification output: $\nabla_{s_j}(C_i(G(\mathbf{s}))$, where i is the index of the true class encoded by $\mathbf{s}$. The gradient can be computed exactly by using an `autograd` algorithmic differentiation provided in standard Deep Learning software environments – both the classifier and the generator being available in such framework. The gradient is averaged over a population of data as the score used to rank the dimensions.

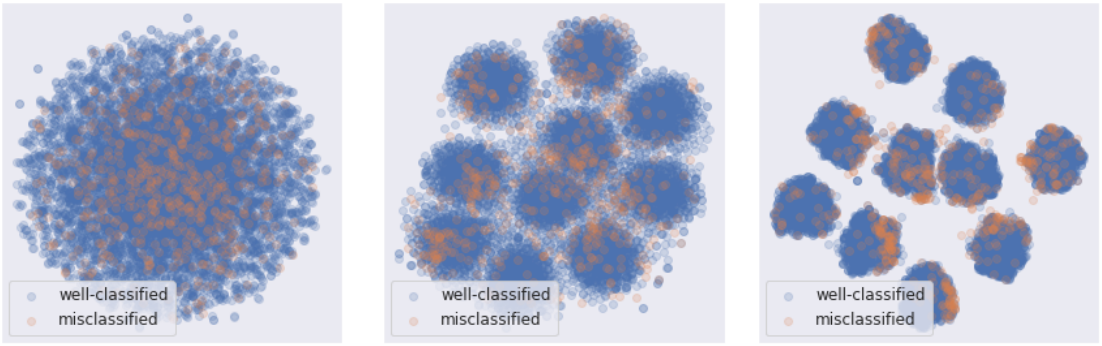
Global and Class-Based Analysis Not all classes behave similarly when corrupted. For instance, a 1 digit, usually written as a single stroke, is more easily identified than a 3, which can be mis-classified as an 8 when there is noise. The impact of corruption potentially depends on the class.

The approach to compute influential directions relies on an average over a population. This population can be global or conditioned by the class, allowing a class conditional or global discovery of influential directions.

Image Manipulation and Extreme Examples Starting from an image where the latent space representation – the style vector – is known, we can modify this representation to generate a modified image. In fact, once the influential dimensions are computed and if the values of the style vector for those dimensions are changed, then the corresponding visual attributes are modified for the generated image. Generating data that follow a high-performance degradation is a simple heuristic: (1) start from a given point $\mathbf{s_0}$ in the StyleSpace, (2) increment the influential dimension by a given amount, and (3) monitor the sign of the increment being given by the sign of the gradient. Note that the starting point $\mathbf{s_0}$ for exploration can be any point in the StyleSpace: it can be a "true" style, computed by mapping to $\mathcal{S}$ a random $\mathbf{z}$ sampled in the input latent space $\mathcal{Z}$, or any other point directly sampled in $\mathcal{S}$, for instance, an average of a given population of $\mathbf{s}$ data. In the experiments of subsection 3.3.3, an "average" digit in the StyleSpace is computed as the mean over a class conditional population.

The data space exploration along influential dimensions also allows the discovery of *extreme examples* defined here as the examples for which a small degradation shifts

Figure 3.3: Samples of real corrupted data (left side) vs. generated data (right side)



(a) t-SNE in $\mathcal{Z}$ (b) t-SNE in $\mathcal{W}$ (c) t-SNE in $\mathcal{S}$

Figure 3.4: t-SNE of generated samples in different latent spaces. $\mathcal{Z}$ does not encode the class as class information is concatenated to latent codes z to form the input of the conditional generator, and does not clearly differentiate well-classified from mis-classified samples. $\mathcal{W}$ and $\mathcal{S}$ are able to separate the classes (the 10 clusters) and well-classified from mis-classified samples, $\mathcal{S}$ doing it better than $\mathcal{W}$.

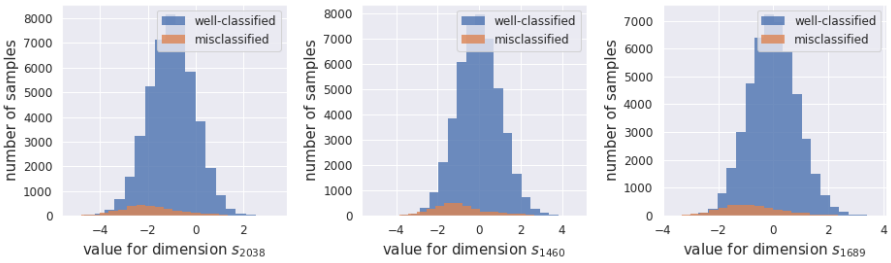
the classifier output from correct to incorrect classification. The experiments of subsection 3.3.3 show several examples of extreme examples discovered by this approach.

Accuracy Conditioned by Latent Space One can also use the latent space to understand better the classifier accuracy. The data along the identified influential dimensions potentially correlates with the classifier accuracy. A classifier can be characterized globally by its accuracy decrease when moving in the StyleSpace along influential dimensions, which vary the amount of corruption. The decreasing slope characterizes globally the resilience to corruption of a classifier and can be used for comparison.

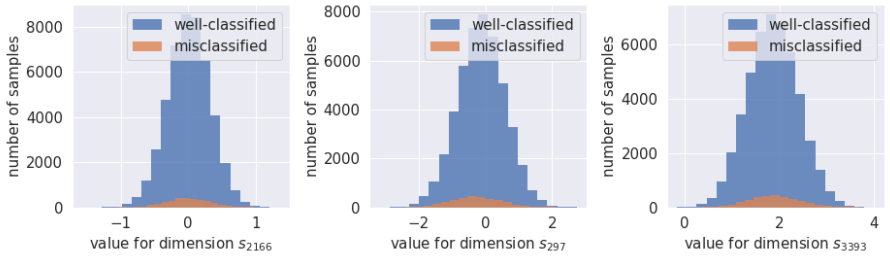
3.3.3 Experiments and Results

Implementation The dataset used to evaluate the approach is MNIST (LeCun et al., 2010). More precisely, the original data was augmented by introducing corruptions to simulate poor-quality data acquisition that may have an influence on class prediction. In particular, corruptions are Gaussian Noise and Gaussian Blur from (Hendrycks and Dietterich, 2019) because they have a significant impact on classification accuracy for a classifier trained on clean data. Data are corrupted in the following way. The first half of the dataset remains clean, and the second half is corrupted by two factors: blur and noise. Both factors' severity levels are randomly chosen between 1, 2, and 3. It ensures that most of the samples remain visually recognizable. Random samples are shown in Figure 3.3.

The StyleGAN2 generative models contain three different latent spaces. It is generally admitted that the so-called StyleSpace $\mathcal{S}$ is a more disentangled representation space. Experiments show that the well-classified and misclassified samples are better separated in this space, even though the generation was not constrained in any way by the classifier.



(a) Top dimensions



(b) Random dimensions

Figure 3.5: (3.5a) Histograms of values for the top 3 dimensions of $\mathcal{S}$ that discriminate the most between well-classified and mis-classified images after generation. For those top dimensions, it is clear that latent codes resulting in well-classified and mis-classified images follow different distributions.
(3.5b) Histogram of values for 3 random dimensions of $\mathcal{S}$. For those dimensions, no difference is visible between the well-classified and mis-classified distributions.

We can visualize this in Figure 3.4 by using t-SNE projection (van der Maaten and Hinton, 2008).

After training on clean data, the classifier (a simple Convolutional Neural Network) reaches an accuracy of 97% on the test data. The metric used to quantify the performance of the generative model is the Fréchet Inception Distance (FID) (Heusel et al., 2017). The generative model trained on corrupted data reaches an FID of 1.63 (computed by comparing $50k$ generated images, unfiltered and without using truncation, to the $60k$ images of the whole training dataset). This low value means a high generation quality. A few samples, shown in Figure 3.3, demonstrate the capacity of the generative model to encode various levels and types of corruption.

Influential Dimensions The method described in 3.3.2 is applied to rank dimensions in the learned StyleSpace. To verify that several dimensions have a bigger impact on performance than others, Figure 3.5 depicts a selection of histograms. The histograms are computer for a specific dimension for the two populations, S_{mis} and S_{well}, where S_{mis} and S_{well} define latent codes resulting in misclassified and well-classified images, respectively. We can see that the values for the top dimensions follow different distributions for well-classified vs. mis-classified images, whereas random dimensions do not discriminate, meaning that the corresponding style attribute does not influence performance.

Figure 3.6 shows the impact of manipulating the latent codes by shifting values along the most influential dimensions. Each column represents one of the top ten influential dimensions, and each line represents a different shift value. We clearly observe various types of image corruption that can be interpreted a posteriori when increasing the shift

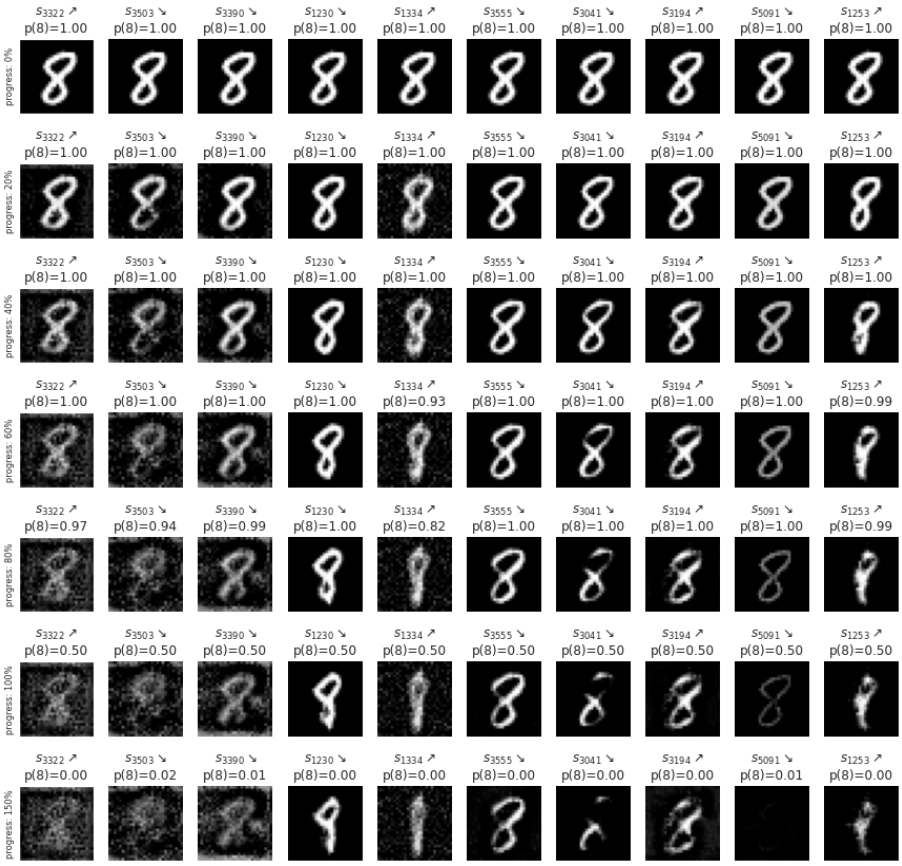
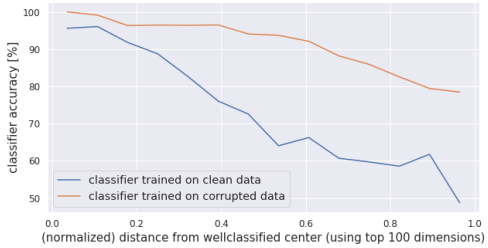


Figure 3.6: Illustration of the degradation evolution starting from the same original image for the ten most influential dimensions. Each column represents one of the top ten influential dimensions; each line represents a different shift value (which also varies per dimension). More specifically, a shift reference value is defined for each dimension as the value that makes the classifier output equal to 0.50 for the corresponding generated image, and each line represents a fraction of the dimension-specific shift reference value, written on the left as *progress*. Above the images are displayed the StyleSpace dimension index, an arrow representing the direction to follow (augment or reduce the value), and the classifier output for the true class.

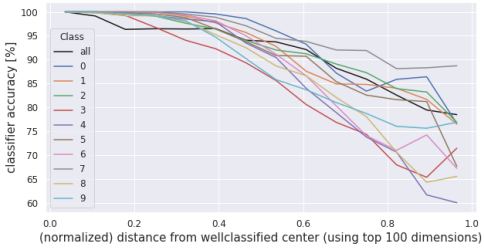
value: the first three dimensions seem to introduce more noise, dimensions 4, 5, and 10 deform the original shape, dimension 9 lowers the intensity, dimensions 6, 7 and 8 introduce partial occlusions. Furthermore, using a generative model allows a large corruption vocabulary and, in particular, allows shape deformation, a capacity that is not available in filter-based frameworks like Imagenet-C (Hendrycks and Dietterich, 2019).

The last three lines of Figure 3.6 show images corresponding to a steep decrease of the classifier output score (from 1.0 to 0). This is where the class prediction shifts and where the generated image can be considered as an extreme example.

Accuracy in the Latent Space As explained in section 3.3.2, we can look at the classifier accuracy evolution when fed with populations of various corruption levels sampled in the StyleSpace. Figure 3.7a shows this evolution on two different classifiers. The accuracy degradation is representative of the robustness to corruptions of a classifier: the classifier trained on clean data sees its accuracy decrease faster than the classifier trained



(a) Comparison of classifiers. The classifier trained on clean data is less robust to corruptions: its accuracy decreases faster than the classifier trained on corrupted data.



(b) Comparison of classes, for the classifier trained on corrupted data. The robustness depends on the class. For instance, predicting correctly 3 or 9 is harder than 0 or 7.

Figure 3.7: Classifier accuracy decreases when samples are far from the well-classified center in the latent space.

Figure 3.8: Illustration of extreme examples found for each class. By starting from the average image in StyleSpace for the class, its latent code in one of the top ten influential dimensions is modified until the classifier shifts its prediction (softmax probability for the given class falls to 0.50). Above the images are displayed the StyleSpace dimension index, an arrow representing the direction to follow (augment or reduce the value), and the classifier output for the true class.

on corrupted data. It also shows that it depends on the class: some classes are more difficult to predict than others. For Figure 3.7b, the lower start for the curve of *all* classes can be explained by the fact that the well-classified center cannot work as well for all classes as for one class (the center and dimensions vary for each curve).

Using most or all dimensions of $\mathcal{S}$ to compute the distance makes the curve not monotonically decreasing. It is better to use fewer dimensions, e.g. 100, as it makes the accuracy curve decrease faster and monotonically. To clarify the curve, samples at too high distance values are filtered out. For such values, the generation quality decreases and the lower number of samples degrades the accuracy computation.

Identification of Extreme Examples The identification of extreme examples, as described in section 3.3.2, is illustrated for the 10 classes in figure 3.8. For each class, the impact of the 10 most influential dimensions is shown. The first column represents the average image of each class, and the ten following columns represent the result of

a corrupted image for one single dimension. Dimensions are selected among the most influential ones, by skipping those with no effect on the classification. It can be seen from the figure that the visual results of extreme examples are specific for each class. Then, if we compare digits 3 and 4, we can see that extreme examples of digit 4 are built by noise adding or structure deformation, whereas the construction of extreme examples of digit 3 is characterized by class switching into digits 8, 1, and 5.

It is important to highlight that the obtained results from figure 3.8 are showing the impact of each single dimension by keeping all other dimensions in their optimal values (average image). The manipulation of a large set of dimensions could combine several types of degradation and allow the identification of new extreme examples.

Analysis The current work addresses a relationship between data quality and model performance by exploring the latent feature space of a generative model. Indeed, the proposed approach allows for the identification of influential directions that deteriorate the classifier performances and the discovery of extreme examples in this space. The proposed approach is based on ranking the latent space dimensions using the classifier output gradient with respect to the StyleSpace input.

Results show the impact and the influence of each identified direction in terms of performance degradation of the classifier. These identified directions, separately or jointly, allow a visual account of the degradation which could help in the interpretability and explainability of deep learning classifiers.

Despite the first promising conclusions of this work, the approach has been demonstrated only for generated and synthetic images. Its application to real data requires a capacity to encode – or invert – any data in the latent space (Xia et al., 2022), to be able to apply the degradation encoded by the influential directions.

Another perspective, is to evaluate the approach on more complex data to identify other types of degradation attributes. Recent works on image manipulation show that visual attributes can be controlled for more complex images (Wu et al., 2021; Patashnik et al., 2021; Härkönen et al., 2020; Ling et al., 2021) and that generative models can be applied to larger datasets such as ImageNet (Sauer et al., 2022). Those two advances indicate the possibility of scaling-up the approach.

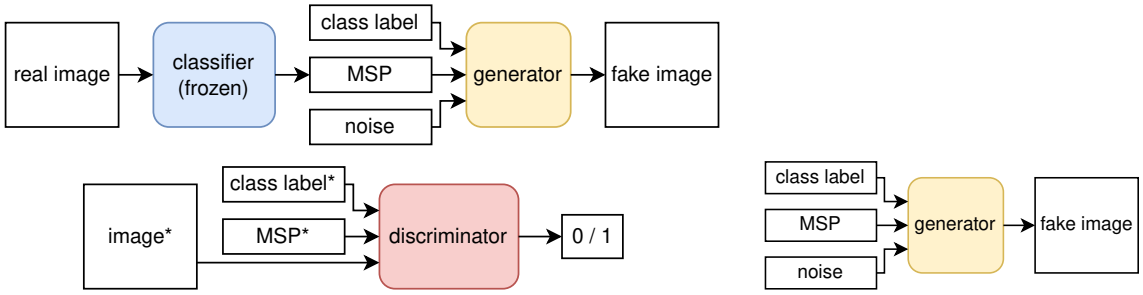
3.4 Generate Synthetic Uncertain Data

3.4.1 GAN Conditioned by a Classifier's Confidence

In section 3.3, we have seen how to navigate a GAN's latent space guided by a classifier's gradient to identify the visual attributes most influencing the classifier's prediction. The GAN was trained independently from the classifier. An alternative way to link the GAN and the classifier is to directly incorporate classifier knowledge into the GAN during training. This can be done through conditioning. A simple way to create a conditional GAN (Mirza and Osindero, 2014) is to concatenate the condition embedding to the noise vector as inputs for the generator and to the real or fake image as inputs for the discriminator. Typically, the condition embedding is a one-hot embedding of the class to create a class conditional GAN. Here, the conditioning is made with a classifier uncertainty score, which embeds knowledge of the classifier into the GAN. The model architecture is depicted in Figure 3.9.

There are several ways to define the classifier prediction uncertainty, e.g., entropy, Maximum Softmax Probability (MSP) (Hendrycks and Gimpel, 2017), or true class probability (Corbière et al., 2021). Here, simple MSP is used as an uncertainty estimation, which performs well to discriminate well-classified from misclassified data as seen in section 4.3. MSP is added as an input condition to the generator. Then, after training, the model can conditionally generate uncertain data to get insights into what impacts the uncertainty. Data can also be manipulated to increase or decrease the uncertainty and exhibit sources of uncertainty.

MSP values are computed from the classifier (with frozen weights). Several populations of uncertainty values are required to learn the generator. For the discriminator used on real images, their associated MSP is computed. For the discriminator used on fake images, the MSP value comes from the condition used by the generator. Note that these values differ from the classifier's outputs because of the imperfect generation. The generator is conditioned by values sample from the real distribution of MSP. This distribution comes from computing the MSP of real images. Otherwise, the discriminator would more easily make the difference between real and fake data, causing the generator training to be harder. It is important to mention that the approach does not distinguish between aleatoric and classifier-dependent epistemic uncertainty: the generative model is used to sample globally uncertain data, whatever the origin of the uncertainty is.



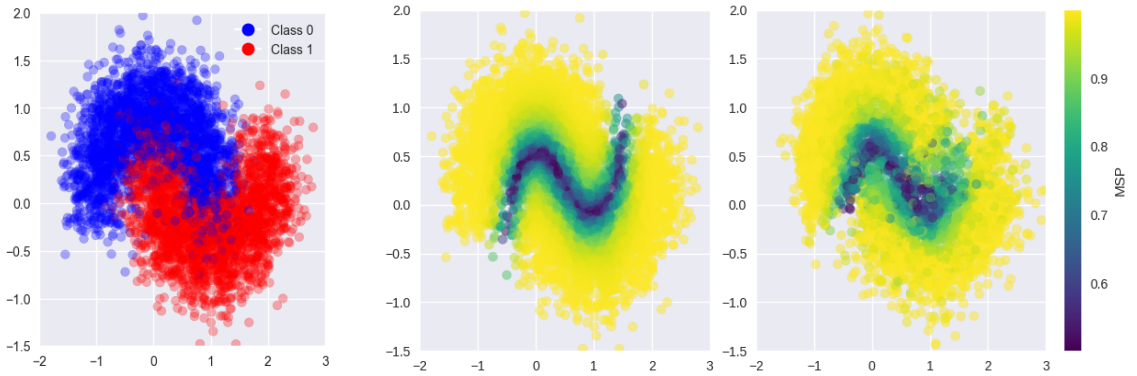
(a) During training time, the additional input MSP conditions the generator. The discriminator evaluates if the combination (class, MSP, image) is realistic.
For the discriminator, inputs are alternatively (class label, MSP from classifier, real image) and (class condition, MSP condition, fake image generated).

(b) After training, we can generate uncertain images (fix low MSP and vary noise) or identify sources of uncertainty for given images (fix the noise and vary MSP).

Figure 3.9: Training process and structure of the generator.

3.4.2 Experiments and Results

Two-Dimensional *Moons* Data The approach is first demonstrated with a simple problem using the moons dataset (Pedregosa et al., 2011). The data is 2-dimensional and looks like two interleaving half-circles corresponding to the upper and lower moon classes. The noise level can be adjusted, and here it is fixed to 0.3 to have an area where the two classes are mixed, as seen in Figure 3.10a. A simple fully connected neural network is trained as a classifier. The generator is based on a fully connected network conditioned by one-hot class embedding and the MSP. The network has a latent space of dimension 8 and 5 layers. The conditioning is a concatenation of the class information as a one-hot embedding vector (of dimension 2) and the MSP as a continuous value.



(a) Moons data with noise level set to 0.3. The two classes overlap in the middle.

(b) (left) Real data with colors representing the MSP computed by the classifier. (right) Generated data with colors representing the MSP used to condition the generation.

Figure 3.10: Results for moons dataset.

Figure 3.10b on the left shows the data, with colors representing the MSP obtained when classifying the data. We can see that the MSP is close to 1 in the area where the classes do not mix but gets lower in the middle area where the classes mix, representing higher uncertainty. We can note that this uncertainty is mostly aleatoric: data can be of either class in the middle region. Figure 3.10b on the right shows synthetic data conditioned by MSP. The values are sampled from MSP computed on real data to follow the same distribution. We can see similarities between the locations of real data with high MSP and synthetic data conditioned by high MSP, and likewise for low MSP. The generator captured which kind of data is uncertain and can generate such data when conditioned with low MSP.

For more quantitative results, the following process is applied: fix some MSP values as conditions ("input confidence"), generate fake data, classify it, and obtain the MSP of the classifier ("output confidence"). Ideally, both values should be the same every time. As seen in Figure 3.12a, it is not necessarily the case, especially for lower values. Yet, the two are correlated.

MNIST Let us now consider more complex data: images. The MNIST dataset contains grey-value images of handwritten digits with ten classes (digits from 0 to 9). A Convolutional Neural Network is trained to classify digits from images. First, clean MNIST data is considered, but to make the problem more difficult, corrupted MNIST images are also studied. Corruptions are Gaussian blur and noise from ImageNet-C (Hendrycks and Dietterich, 2019). These corruptions are applied on a random half of the images, with a random severity level out of 5 possible levels.

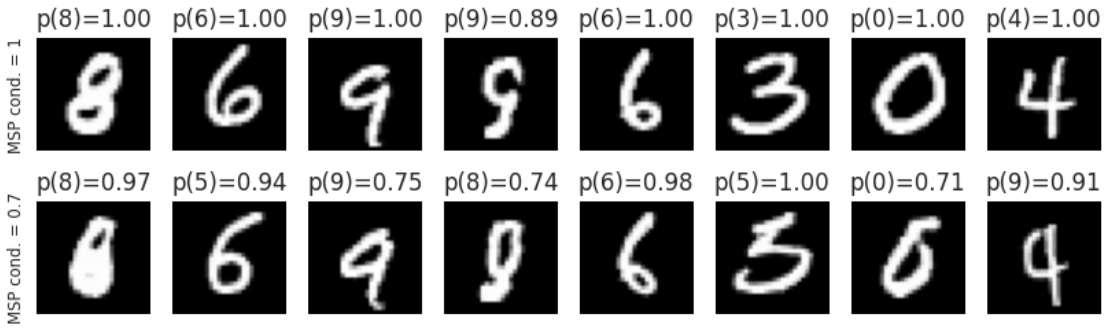
Introducing corruption results in a reasonable accuracy reduction compared to clean MNIST: now 94.0% on the train set and 93.2% on the validation set (instead of 98.8% and 98.5%, respectively). Also, MSP values are more spread out. The GAN is now based on the StyleGAN2 architecture to handle images, with additional conditioning for the MSP. The default latent space dimension of 512 (for the noise) is used, as reducing it made the training more difficult. The conditioning is a concatenation of a class embedding and the MSP value.

We can generate uncertain images by fixing a low MSP value and varying the noise input, as illustrated in Figures 3.11a and 3.11b bottom. Also, comparing Figures 3.11a

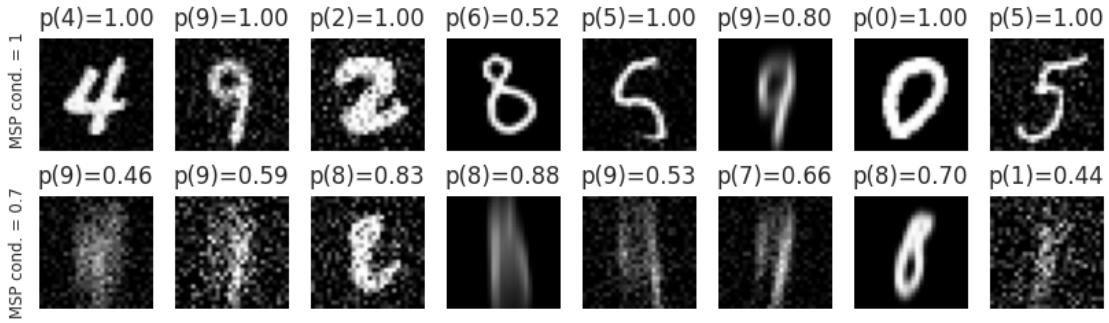
and 3.11b top versus bottom allows gaining insights into the classifier's sources of uncertainty by observing what makes given images more uncertain (by fixing noise inputs and lowering the MSP condition). In this case, it is primarily shape, Gaussian noise, and blur that perturbates the classifier.

The qualitative results in Figure 3.11a and 3.11b show that images generated with the conditioning of MSP $= 1$ mainly result in "output" MSPs close to 1. "Output" MSP values are more spread-out when conditioned with MSP $= 0.7$. Figure 3.12b shows that "input" MSP and "output" MSP can be quite different. While not as good as on the moons dataset, there is still some correlation. I hypothesize that the MSP is much less well-defined on MNIST images than on the moons dataset. More substantial constraints on the conditioning should be considered to improve the results.

Analysis The conditional GAN is an explicit generator of uncertain data that can be used in several ways. It can give a global outlook by generating uncertain images on demand. It can also corrupt images (transform them into their more uncertain form) to visualize sources of local uncertainty. However, the MSP condition is not very well respected for MNIST images. An explanation might be because the MSP does not contain sufficient information to capture the classifier behavior, making the conditioning hard to learn. One possible way to improve the generation would be to introduce another constraint and loss to align the input and classifier dependent MSP computed on the generated image.

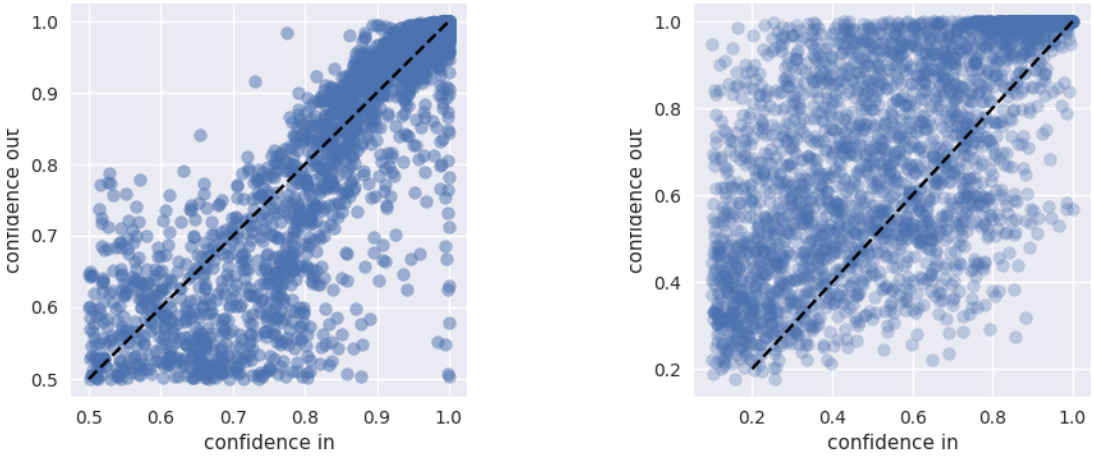


(a) clean MNIST



(b) corrupted MNIST

Figure 3.11: Samples of images generated with MSP condition fixed at 1 (top) and 0.7 (bottom). Above each image is shown the classifier prediction and probability. Images at the bottom look harder, and the classifier is more uncertain.



(a) Moons data.

(b) MNIST data.

Figure 3.12: MSP condition ("in") vs. MSP computed by classifying the generated data ("out"). Uncertainty conditioning works well when points are close to the diagonal: this shows that the MSP condition corresponds roughly to the real MSP.

3.5 Discussion

In this chapter, generative models, more specifically GANs, were used to identify and illustrate the failure conditions of a given image classifier to characterize. Two approaches have been developed.

The first approach is described in section 3.3. A GAN is trained independently from the classifier to characterize. Then, the classifier's gradient is used to guide the navigation in the latent space to identify directions of strong performance degradation. Because the latent space is highly disentangled, each direction encodes different visual attributes. Visualizing these attributes helps understand what factors most impact the classifier's performance. One can also generate images at the limit of the classifier decision boundary, helping identify specific zones in the latent space corresponding to high classification accuracy.

The second approach is described in section 3.4. Now, knowledge of the classifier is embedded into the GAN directly during training. The GAN is conditioned by the classifier's confidence/uncertainty score. This model can be used in two ways. Fix the latent noise input and vary the uncertainty input to identify the main source of uncertainty for a given image. Or, fix the uncertainty input and vary the latent noise input to generate several images of a given uncertainty level to understand the difference between images with high and low uncertainty.

These two approaches allow for a better understanding of a classifier by illustrating what makes it fail or be more uncertain. This is a qualitative approach, making it closer to the field of XAI. However, this thesis aims to define a domain, which requires more than only a qualitative view. The initial idea was to define a domain in a latent space. I have explored this idea, but several reasons made it difficult to achieve. I discuss these reasons below.

The latent space Z would represent images according to the classifier predictions. Work from section 3.4 is a step in that direction: classifier knowledge is embedded in the GAN's latent space thanks to specific conditioning during training. An ideal representation would be a hypersphere where the distance to the center is linked with the classification correctness: well-classified images near the center and misclassified images farther from it. Figure 3.7 is a preliminary result in that direction.

A parameter λ would characterize a zone in the latent space Z, which defines the domain Z_λ. An image would belong to the domain if a selection function selects it. For instance, a selection function could assess if the distance of the image's latent code to the hypersphere center is below a threshold λ. This domain would have a certain accuracy (proportion of well-classified images in the domain) and a certain coverage of the dataset (proportion of the dataset images that are in the domain). A compromise between accuracy and coverage can then be tuned with λ. This is similar to selection functions from the field of selective classification, which is the topic of section 4.3. The process is illustrated in Figure 3.13.

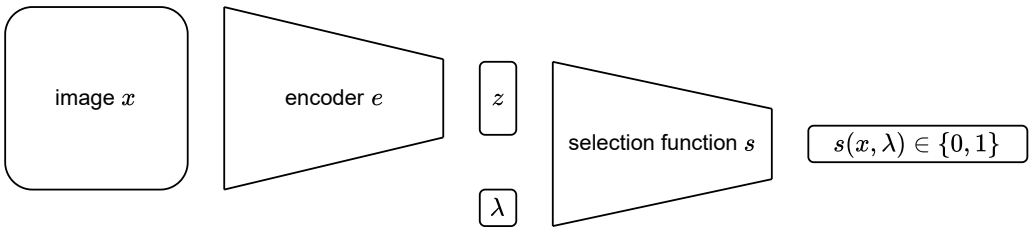


Figure 3.13: Image x is encoded into its latent representation $z \in Z$. A selection function determines whether image x belongs to the domain Z_λ.

A generative model would decode the latent codes from Z to images and help structure the latent space according to visual attributes. This would allow evaluating the classifier performance for different "zones" of the latent space. It would also illustrate the domain boundaries, similar to the qualitative results of sections 3.3 and 3.4. The process is illustrated in Figure 3.14.

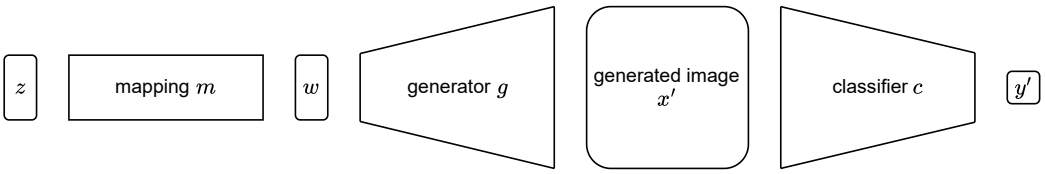


Figure 3.14: Latent code z is mapped to a representation w in the generator's latent space, and transformed into a synthetic image. The synthetic image is visualized for qualitative purposes or classified for performance evaluation.

Defining a domain following this concept requires many different elements that have to be orchestrated together, including several neural networks: the encoder, the generator (and the discriminator used during training), and the mapping. Training these networks requires several loss functions, e.g., a reconstruction loss (encoded images should be reconstructed faithfully by the generator), an adversarial loss (for the generator), and a loss to structure Z according to the classifier behavior.

However, despite the concept's potential, several limits prevent it from achieving the objective of defining a domain.

Encoder Limits The encoder's task is to map images according to the classifier prediction: well-classified images should be far from misclassified ones in the latent space. This means that the encoder needs to know the true class to some extent. If distinguishing well-classified from misclassified images was easy for a network to learn, then the classifier should not have made the mistakes in the first place. The encoder having access to the same training data as the classifier means that there is no reason that it could interpret images better (otherwise, why not use the encoder as a better classifier?). Images that are difficult to classify are also difficult to correctly encode. The approach would thus be interesting only for specific settings, such as a small classifier in an embedded system.

Generator Limits The generator aims to reconstruct images from latent codes. However, for datasets with several classes, GANs are much harder to learn when not class-conditioned. Class-conditional GANs are implicitly image classifiers. During training, discriminators predict whether the combination of an image x and a label y is realistic: they learn the relation between image and class. Generators aim to generate an image of a given class that fools such imperfect discriminators. Class-conditional GANs sometimes generate ambiguous images of a different class from the condition (e.g., an MNIST digit looking more to a 4 than a 9, which was the condition). It means that evaluating a classifier on such images is flawed: the class condition is not necessarily respected. Again, images that are difficult to classify are also difficult to generate correctly (i.e., for which the class in the condition corresponds to the class of the image).

GAN Inversion Limits Another issue is the gap between real and synthetic data, which is tackled by the field of GAN inversion. An image encoded in the latent space and an image generated from this latent code can be quite different. Not *all* possible images are represented in the latent space. When considering difficult data, every detail matters and can explain the classification failure. The gap between real and synthetic data should be as small as possible, but existing approaches are insufficient for what is required here. See Figure 3.15 for examples of reconstructed images compared to the original ones.

In GAN inversion, there is usually a compromise between reconstruction error (how close is the image generated from the latent code to the target image) and editing capabilities (how well the image can be edited from latent code manipulations, some entanglement or artifacts might appear) (Tov et al., 2021). Let us use the StyleGAN architecture as an example. As target image representations might not exist in the original latent space W of StyleGAN, more degrees of freedom are required to lower the reconstruction error. For instance, it is possible to invert in $W+$, a higher dimensional latent space corresponding to feeding different latent codes $w \in W$ for each generator layer. As each layer handles different levels of features (from high-level features like pose to low-level features like hair color), this extended latent space allows the combining of features from different latent codes. Inverting in this latent space results in lower reconstruction errors, but the latent code in $W+$ might reside in regions not naturally explored during training (which happens in W) and thus not well-behaved. This causes problems with editing, which relies on the structure and disentanglement developed during the training. Conversely, inverting in W results in more "natural" latent codes in regions explored during training, but fewer degrees of freedom means less fidelity for the reconstructed image.

Figure 3.15: Examples of unseen images reconstructed by the generator after encoding in the latent space. While images look similar, major attributes can differ, such as eye or hair or skin color, or the presence of glasses. From (Pidhorskyi et al., 2020).

Optimization-based methods are noninjective: different latent codes can be found for the same image, for instance, depending on the initialization.

Similarly, encoder-based methods usually provide better editability than optimization methods but worse reconstruction. This is because encoder-based methods are more likely to project images into "natural" zones of the latent space, as seen during training. Optimization-based methods can explore the latent space more freely, including zones not explored during training. These zones might allow a good reconstruction but poor editability as they do not follow the latent space structure built during training.

One of the initial intuitions was to define a domain as a convex hull delimited by extreme examples. Projecting new data in the space would indicate if it belongs to the domain. However, here is an interesting insight from Balestriero et al. (2021): "the behavior of a model within a training set's convex hull barely impacts that model's generalization performance since new samples lie almost surely outside of that convex hull". This might mean that the original intuition was flawed. It can also help explain the difficulty of GAN inversion.

For all these reasons, the initial idea of defining a domain in a GAN's latent space seems harder than expected. The results of this chapter thus remain limited to a qualitative study. The next chapter looks at the fields of selective classification and calibration, which brings a quantitative view.

Chapter 4

A Quantitative View: Selective Classification and Calibration

4.1 Introduction

In the previous chapter, we leveraged generative models to create hard-to-classify images to give hints about the classifier limits. Such extreme examples help understand what can lead to classifier failure. However, they do not help predict the reliability of a given prediction nor identify which data belongs to the reliability domain. To tackle those issues, we need a different perspective. Selective Classification (SC) and calibration are related research fields. SC allows a model to abstain when in doubt. The accuracy (the proportion of correct predictions among the selected samples) is increased at the expense of coverage (the proportion of selected samples). Calibration aims to obtain predicted probabilities representative of the probability of making a correct prediction. These two fields have different goals but share a similar concern: they both provide tools to anticipate when classifier predictions are accurate or not, which are essential to define a reliability domain.

Section 4.2 describes the notions of SC and confidence calibration with more technical details than in the related work.

Section 4.3 is about comparing different scoring functions for SC through some experiments. A standard baseline uses the maximum softmax probability predicted by the classifier as a confidence score to decide whether to keep or reject the data point. Preliminary experiments on the dataset CIFAR-10 show that this baseline has a strong performance and is not easily beaten by more advanced approaches.

Section 4.4 focuses on confidence calibration. For classification models based on neural networks, the maximum predicted class probability is often used as a confidence score. This score rarely predicts well the probability of making a correct prediction and requires a post-processing calibration step. However, many confidence calibration methods fail for problems with many classes. To address this issue, I transform the problem of calibrating a multiclass classifier into calibrating a single surrogate binary classifier. This approach allows for more efficient use of standard calibration methods. Extensive experiments on numerous neural networks used for image or text classification show that the approach significantly enhances existing calibration methods.

Section 4.5 considers another approach to improve confidence calibration by using generative models. Synthetic data can be used as calibration data to improve the calibration of CIFAR-10 image classifiers, with comparable or better results than using standard validation data. Synthetic images are generated by a GAN trained on CIFAR-10's training data. However, GANs struggle with complex natural images, which restricts the approach to simple and well-structured images.

Finally, section 4.6 discusses the chapter's findings in the broader thesis context.

4.2 Background

Let us consider the classification problem where an input $x \in \mathcal{X}$ is associated with a class label $y \in \mathcal{Y} = \{1, 2, ..., L\}$. The neural network classifier f provides a class prediction from a final softmax layer σ, transforming intermediate logits z into probabilities. The classifier prediction is the most probable class $\hat{y} = \arg\max_{k \in \mathcal{Y}} f_k(x)$ with $f_k(x)$ referring to the probability of class k, and the confidence score is defined as $s(x) = \max_{k \in \mathcal{Y}} f_k(x)$. Note that I use the term *confidence* to denote the Maximum Softmax Probability (MSP).

Selective Classification A *selective classifier* (El-Yaniv and Wiener, 2010; Geifman and El-Yaniv, 2017) is a pair (f, g), where f is a classifier and $g : \mathcal{X} \to \{0, 1\}$ is a binary *selection function*:

$$(f, g)(x) = \begin{cases} f(x), & \text{if } g(x) = 1; \\ \text{don't know}, & \text{if } g(x) = 0. \end{cases}$$

The selection function is typically built from a confidence score s and a threshold α, with $\mathbf{1}$ being the indicator function:

$$g_\alpha(x) = \mathbf{1}[s(x) > \alpha]$$

Ideally, the confidence score reflects true loss monotonicity, meaning that higher confidence is equivalent to a lower loss: $s(x_1) \geq s(x_2) \iff \ell(f(x_1), y_1) \leq \ell(f(x_2), y_2)$. In practice, the ideal confidence score is not available, but surrogate functions are possible. Using the classifier maximum predicted probability as confidence score $s(x) = \max_{k \in \mathcal{Y}} f_k(x)$ is highly effective (Geifman and El-Yaniv, 2017).

A selective classifier is evaluated in terms of risk and coverage. The coverage is the proportion of selected samples over the data distribution P: $\phi(g) = \mathbb{E}_P[g(x)]$. The (selective) risk is the average loss for the selected samples $R(f, g) = \frac{\mathbb{E}_P[\ell(f(x), y) g(x)]}{\phi(g)}$. Instead of using the risk with ℓ being the 0/1 loss, I prefer to use the accuracy as it is a more widely used, which is $1 - R(f, g)$.

To compare different confidence score definitions, we can plot accuracy-coverage curves. Accuracy and coverage pairs are computed when varying the threshold values α. Each value corresponds to a different set of selected data samples, for which the accuracy and coverage are computed. As α decreases, more data samples are selected, which increases the coverage, and the accuracy should decrease. When using random selection, the accuracy is a noisy estimate of the global accuracy for low coverages, and it converges to the global accuracy for higher coverages.

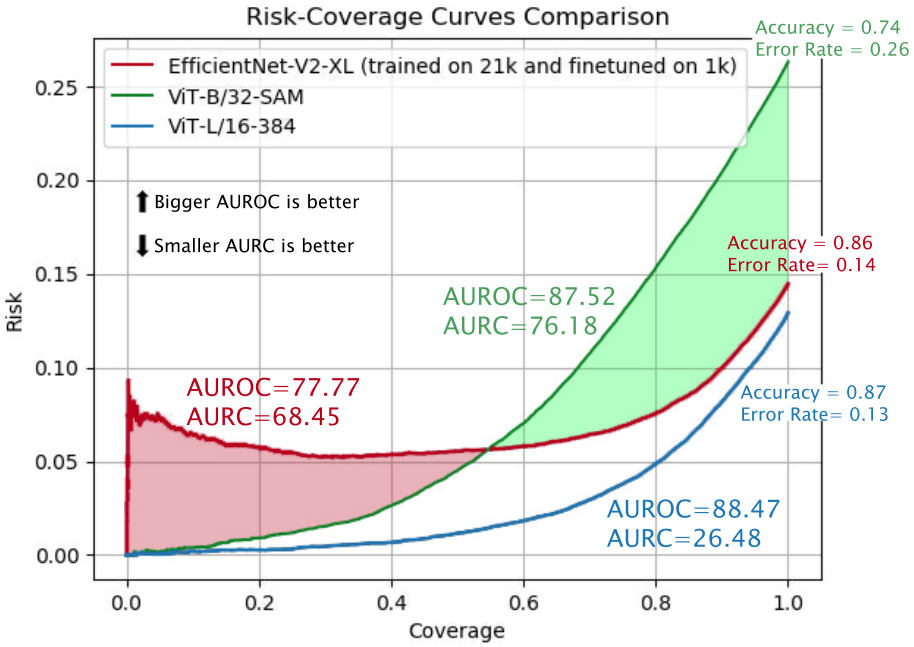


Figure 4.1: Illustration of risk-coverage curves for different models. From (Galil et al., 2023b).

An ideal oracle selection can be defined as first selecting all correctly classified samples and then selecting all the rest. This way of selecting data results in the best accuracy-coverage curve. The accuracy is 100% for coverages up to the global accuracy value and then converges to the global accuracy. The ideal selective accuracy a^* is a function of the coverage ϕ and the global accuracy A:

$$a^*(\phi) = \begin{cases} 1 & \text{if } 0 \leq \phi \leq A; \\ A/\phi & \text{if } A \leq \phi \leq 1. \end{cases}$$

Here is where this equation comes from. The selective accuracy a^* for a given coverage ϕ is the accuracy of the ϕN samples selected out of the total number N of samples. Note that the coverage is a set of discrete values: $\phi \in \{1/N, 1/(N-1), \ldots, 1\}$.

$$a^*(\phi) = \frac{1}{\phi N} \sum_{i=1}^{\phi N} \mathbf{1}[\hat{y}_i = y_i]$$

For a coverage $0 \leq \phi \leq A$, the oracle selects all samples that are well-classified: $\forall i \in \{1, \ldots, \phi N\}, \hat{y}_i = y_i$ which means that $a^*(\phi) = \frac{\phi N}{\phi N} = 1$.

For a coverage $A \leq \phi \leq 1$, all samples are misclassified: $\forall i \in \{\phi N, \ldots, N\}, \hat{y}_i \neq y_i$ which means that $a^*(\phi) = \frac{1}{\phi N} \sum_{i=1}^{\phi N} \mathbf{1}[\hat{y}_i = y_i] = \frac{1}{\phi N}(\sum_{i=1}^{AN} \mathbf{1}[\hat{y}_i = y_i] + \sum_{i=AN+1}^{\phi N} \mathbf{1}[\hat{y}_i = y_i]) = \frac{1}{\phi N}(AN + 0) = A/\phi$.

Figure 4.2 shows the accuracy-coverage curve of an oracle selector.

Instead of comparing curves, we can also use single-number metrics. A risk-coverage curve can be summarized in a single number by computing the Area Under the Risk-Coverage curve (AURC), optionally normalized (E-AURC) (Geifman et al., 2019). A good selective classifier quickly rejects misclassified points as the coverage decreases. Its risk-coverage curve decreases quickly and thus has a low AURC. Because there is some irreducible risk (unless the classifier is perfect), the AURC can never reach 0. By

subtracting the AURC of an oracle classifier that knows which points are misclassified, we get the E-AURC. A low value is desirable, with 0 being optimal.

Unfortunately, these metrics depend on the global accuracy value, making it hard to compare classifiers with different accuracies, contrary to using the coverage at a Selective Accuracy Constraint (SAC) or Area Under the Receiver Operating Characteristic curve (AUROC) (Galil et al., 2023b).

SAC is, for instance, the coverage for an accuracy of 99%. It allows for comparing different classifiers; the best selective classifier is not necessarily the one with the highest global accuracy. The primary metric used in this chapter, and also by (Galil et al., 2023b), is the AUROC. The Receiver Operating Characteristic curve is a graphical representation of a binary classifier's performance at various confidence thresholds. It is created by plotting the True Positive Rate (TPR) against the False Positive Rate (FPR) at various threshold settings. The TPR is the ratio of correctly predicted positive observations to all actual positives. The FPR is the ratio of incorrectly predicted positive observations to all actual negatives. A perfect classifier achieves an AUROC of 1 and a random classifier has an AUROC of 0.5. A high AUROC means the confidence discriminates well between the positive and negative observations. In this case, positive observations are the correct predictions, and negative observations are the incorrect predictions.

Confidence Calibration of a Classifier This section uses the confidence calibration definition from (Guo et al., 2017) that says that the classifier f is calibrated if:

$$P(\hat{y} = y | s = p) = p, \quad \forall p \in [0, 1] \tag{4.1}$$

where the probability is over the data distribution, $\hat{y}$ is the predicted label, s is the confidence, and y is the real label. Equation (4.1) expresses that the probability of being correct when the confidence is around p is indeed p. For instance, if we consider the set of predictions with a confidence of 90%, they should be correct 90% of the time. The conditional probability of (4.1) is not rigorously defined mathematically (the event $\{s = p\}$ has zero probability), and interval-based empirical estimators are often used to define metrics capable of evaluating how well (4.1) is satisfied. This is the case of ECE, which approximates the calibration error by partitioning the confidence distribution into B bins. The absolute difference between the accuracy and confidence is computed for each subset of data in the bins. The final value is a weighted sum of the differences of each bin:

$$\text{ECE} = \sum_{b=1}^{B} \frac{n_b}{N} |\text{acc}(b) - \text{conf}(b)| \tag{4.2}$$

where n_b is the number of samples in bin b, N is the total number of samples, $\text{acc}(b)$ is the accuracy in bin b, and $\text{conf}(b)$ is the average confidence in bin b. ECE can be interpreted visually by looking at diagrams such as those of Figure 4.4: ECE computes the sum of the red bars (difference between bin accuracy and average confidence) weighted by the proportion of samples in the bin.

This chapter considers *post-processing calibration*, the scenario where a classifier has already been trained, and the objective is to enhance its calibration. Post-processing calibration methods aim to remap the classifier probabilities to better-calibrated values without modifying the classifier. They typically use a calibration set different from the training set to optimize parameters or learn a function. The calibration data is noted $D_{cal} = \{(x_i, y_i)\}_{i=1}^{N}$. I focus on post-processing calibration because it enables better

utilization of off-the-shelf models and separates model training (optimized for accuracy) from calibration. These advantages significantly reduce the development cost of obtaining a well-performing and well-calibrated model, contrary to optimizing calibration during training.

I categorize the post-processing calibration methods considered in this section into two groups: scaling methods and binary methods. Two standard methods are described in the following paragraphs.

Temperature Scaling A standard scaling method is Temperature Scaling (TS). It introduces a single learnable parameter T, called the temperature, that scales the classifier logits. Classifier logits are unnormalized class scores output by the classifier before applying the softmax function. Given the logits vector z scaled by T, the softmax function converts it into class probabilities, with the class probability vector written as $p(z, T) = \exp(z/T)/\sum_j \exp(z_j/T)$. A temperature $T = 1$ corresponds to the standard softmax. A temperature $T > 1$ softens the probabilities and makes the distribution more uniform, typically reducing confidence. A temperature $T < 1$ sharpens the probabilities, making them tend to more extreme values (0 or 1).

An extension of TS is Vector Scaling (VS) where there is one coefficient per class, contained in vector v. The vector multiplies the logits vector term by term. Calibrated probabilities are written $p(z, v) = \exp(z \circ v)/\sum_j \exp(z_j.v_j)$.

The temperature or vector coefficients are learned with optimization on a validation set, typically with stochastic gradient descent. TS is a very effective and simple method, usually used as the standard baseline in many calibration papers.

Because two predictions with the same pre-calibration confidences can differ regarding the remaining class probabilities, calibrated confidences become different. Confidence-based ranking of the data is thus impacted by TS, making the method influence SC performance.

Histogram Binning Histogram Binning (HB) is a standard method used in the binary setting (Zadrozny and Elkan, 2001). Examples from a validation set are sorted according to their confidences and divided into B subsets called bins. The bins can be of equal size (same range of confidence values) or equal mass (same number of examples per bin). Bin accuracies are computed from the validation examples assigned in each bin. For instance, one bin contains samples with confidence values between 0.6 and 0.7, and its estimated accuracy is 0.72. At test time, a prediction with pre-calibration confidence falling in the bin, e.g., 0.61, will be assigned the calibrated probability of 0.72.

HB discretizes the confidence values, as they now can take only B values. However, these values are better calibrated.

Applying HB to the multiclass setting requires some multiclass to binary adaptation. Typically, HB is applied to each class separately by considering the multiclass classifier as a set of binary OvA classifiers.

4.3 Selective Classification Experiments

4.3.1 Selection Functions

A key aspect of SC is the selection function. It decides which data belongs to the domain. When data is not selected, the selective classifier abstains. As in standard SC, only in-distribution data is considered. The selective function should not be expected to accurately filter out Out-of-Distribution data as it is a different problem. See (Narasimhan et al., 2024) for a proposition of unifying these two settings. While SC can be improved by specific classifier training approaches (Geifman and El-Yaniv, 2019; Feng et al., 2023), this section studies how to improve SC for a given classifier. It allows for better use of off-the-shelf classifiers without requiring important development efforts.

Selection Function From Confidence Scores A selection function based on thresholding on the classifier Maximum Softmax Probability (MSP) is a powerful baseline (Geifman and El-Yaniv, 2017). Another choice is MC-dropout (Gal and Ghahramani, 2016), which applies dropout (Srivastava et al., 2014) during both training and inference and performs multiple forward passes to generate a distribution of predictions from which a confidence score is computed. A different option is predictive entropy, which is computed from the predicted probabilities and represents model uncertainty. Because MSP outperforms both MC-dropout (Geifman and El-Yaniv, 2017) and predictive entropy (Feng et al., 2023), it is use here as the baseline for the confidence score.

Selection Function as a Trainable Model To improve over the MSP baseline, one possibility is to train an auxiliary model to predict the failure of a fixed classifier (Corbière et al., 2019, 2021), as mentioned in subsection 2.5. In particular, the model learns the True Class Probability (TCP) of a given classifier. TCP is the probability assigned to the true class (in opposition to the predicted class, whose probability is the MSP. TCP is an indicator of failure as when the classifier fails, TCP $<$ MSP (when the classifier is correct, TCP = MSP). Also, because it is a continuous value, it contains more information than a binary failure label. Below, I reproduce this approach and variations using different inputs and outputs for the auxiliary model.

The auxiliary model predicts continuous values that are interpreted as a confidence (or uncertainty) score, used by the selection function of Equation 4.2. Two different outputs are studied:

- Failure prediction (denoted `wellClassified`): a binary classification to discriminate well-classified from misclassified samples.
- TCP: the classifier probability of the true class: as shown by (Corbière et al., 2019), it can outperform using the classifier confidence (MSP).

An auxiliary model trained to predict these outputs can use different inputs for the prediction. In (Corbière et al., 2019), the model first learns the head that predicts TCP from features and then fine-tunes the features encoder initialized with a copy of the original classifier feature encoder. The network thus combines information from features and images. Here, the following inputs are considered:

- Classifier logits: the MSP is computed from logits, so they contain enough information to perform at least as well as the MSP.

- Classifier features: should include more information than the logits.
- Image: directly use the raw image to make predictions (by fine-tuning a copy of the whole classification model).

Selection Function with Distance to Clusters in Feature Space Selection functions can also be inspired by work outside the scope of SC. (Sorscher et al., 2022), an article about data pruning, shows that pruning some data can be done without much impact on performance if the data chosen is easy and redundant and the dataset is big (for small datasets, easy examples are more important than hard ones). The criterion to distinguish "easy" from "hard" data is looking at distances between samples and their closest cluster in feature space (clusters obtained from k-means). This criterion might be applied to SC, which is a similar problem if we suppose that failures come from "hard" data.

After performing k-means clustering of training data in the embedding space of the classifier, the difficulty of a data point is defined with the Cosine Similarity (CoSim) to its nearest cluster centroid or prototype. Easy examples are the most prototypical. The number of clusters is set to the number of classes. The method is not sensitive to the exact value unless it differs from the number of classes by more than an order of magnitude.

4.3.2 Experiments and Results

Classifier The models are pre-trained classifiers from GitHub, using VGG (Simonyan and Zisserman, 2015) and ResNet (He et al., 2016) architectures.

Data The data used is CIFAR-10 with modifications. If the auxiliary model is trained on the standard training data, no learning is made, probably because the dataset contains too few misclassified samples (the classifier train accuracy is above 99%). To solve this problem, validation data (10000 samples, with an accuracy of $\approx 94\%$) is split into two halves: the first for training and the second for testing. Data augmentation is necessary on the training split to avoid overfitting. Note that the issue might not occur on complex datasets for which the training accuracy is not as high, such as ImageNet.

Results An auxiliary model is trained to predict outputs representing failure from given inputs. As mentioned in the previous section, different input and output choices are possible. When inputs are features, the classifier head is copied and fine-tuned; when inputs are logits, the model is a small fully connected network; and when inputs are images, a full classifier copy is fine-tuned. When the output is `wellClassified`, the model is trained on the binary classification task of failure prediction using the binary cross-entropy loss; when the output is TCP, the model is trained on a regression task using a mean squared error loss.

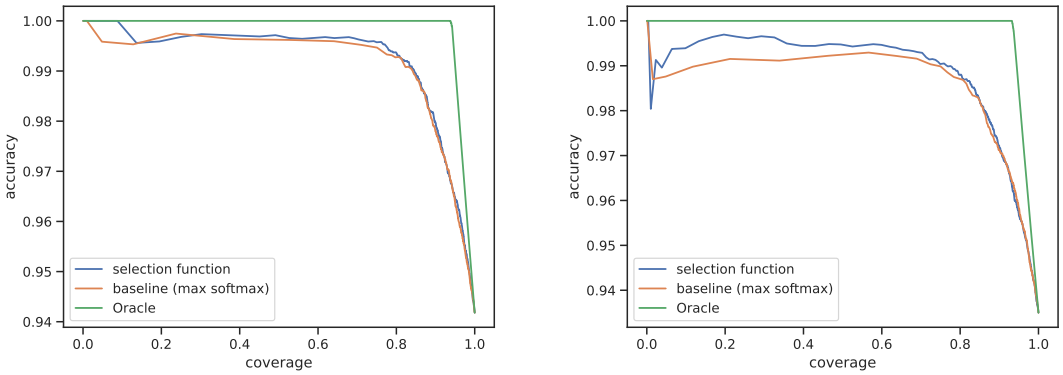
Table 4.1 summarizes the results. For both models, the best results are obtained by using logit inputs, and the output choice is not so significant. Using images as input is not a good option, probably because the training is more complex. Gains from the baseline are more important for ResNet, but it is because ResNet is not a good selective classifier to begin with, compared with VGG, as shown in Figures 4.2 and 4.3. Using CoSim to feature clusters does not outperform the baseline either.

These results can surely be improved with tweaks to the method and hyperparameter optimization, but significant improvements are probably not to be expected. Work by

(Corbière et al., 2019) first learns a network from features to TCP, and then fine-tune the whole feature encoder initialized as a copy of the classifier. Their results are, in my opinion, not good enough to justify the effort compared to just using the MSP baseline, which all classifiers provide without needing further development.

Table 4.1: Selection function models success

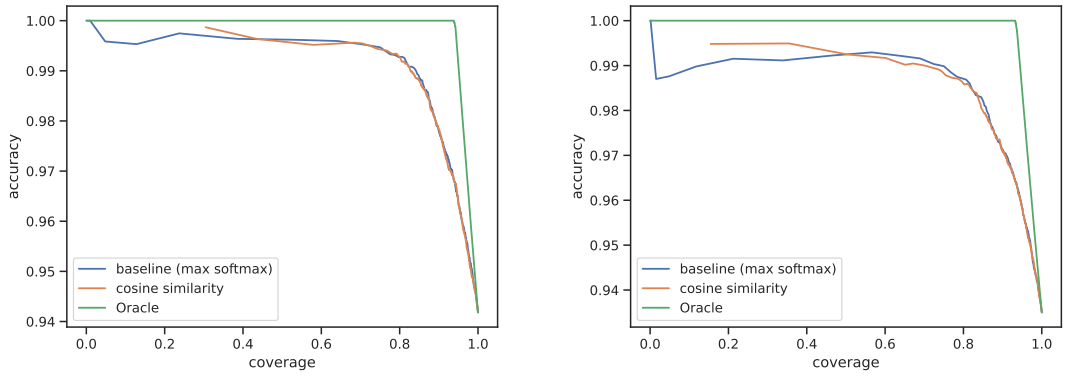
Model	Method	AUROC (↑)	AURC (↓)
VGG-16-BN	baseline (MSP)	0.922	0.008
VGG-16-BN	images in - `wellClassified` out	0.706	0.027
VGG-16-BN	features in - `wellClassified` out	0.923	0.008
VGG-16-BN	logits in - `wellClassified` out	0.925	0.008
VGG-16-BN	images in - TCP out	0.710	0.028
VGG-16-BN	features in - TCP out	0.921	0.008
VGG-16-BN	logits in - TCP out	**0.928**	**0.007**
VGG-16-BN	CoSim between feature clusters	0.920	0.008
ResNet-50	baseline (MSP)	0.893	0.013
ResNet-50	images in - `wellClassified` out	0.630	0.042
ResNet-50	features in - `wellClassified` out	0.654	0.039
ResNet-50	logits in - `wellClassified` out	**0.911**	**0.010**
ResNet-50	images in - TCP out	0.566	0.051
ResNet-50	features in - TCP out	0.557	0.061
ResNet-50	logits in - TCP out	0.907	0.011
ResNet-50	CoSim between feature clusters	0.895	0.014



(a) VGG (b) ResNet

Figure 4.2: Comparison of VGG and ResNet selective classifiers using logits as inputs and TCP as the predicted confidence score.

Analysis - MSP is a Strong Baseline I tested two methods to improve SC performance. The first one, training a model to predict the classification result (through binary classification or regression of the loss or TCP), can beat MSP by a small margin in the best case. The second method, using distances to clusters in a feature space, does not perform better than MSP. The main conclusion of these preliminary experiments is that



(a) VGG (b) ResNet

Figure 4.3: Comparison of VGG and ResNet selective classifiers using distance to feature clusters as the predicted confidence score.

MSP is a strong baseline for SC. The simple methods here are not worth using because of only minor improvements in the best case.

The main takeaway is that it is hard to beat the simple baseline of thresholding the classifier's confidence in SC. I hypothesize that a well-trained classifier is already aware of its uncertainty through its confidence: it already uses most information in the available data. An external process (learning a selection function) using the same data does not work much better. The confidence discriminates quite well incorrect predictions from correct ones, even if not well-calibrated. A good selective function requires a score discriminating incorrect predictions from correct ones; only relative values matter. Calibration measures if the absolute values of the confidence score reflect the correctness probability.

4.3.3 Towards Better Calibration

The SC strategies evaluated above and the post-processing calibration methods share similarities. Both use validation data. In the first case, data is used to train a network for predicting a confidence score, and in the second case, to learn a remapping function that improves the confidence score calibration. Noting this similarity, I applied simple and powerful calibration methods (temperature and vector scaling) to assess their impact on SC. Because in SC, the important thing to predict is whether the prediction is correct or not, I adapted the calibration loss to reflect only the binary task failure prediction. This did not really improve the SC (measured by AUROC) but had a significant impact on confidence calibration (measured by ECE). The following section, 4.4, studies this insight in more detail.

4.4 Making Confidence Calibration Methods More Data-Efficient

4.4.1 Issues Related to Current Approaches

Many post-processing calibration methods have been developed for binary classification models (Platt, 1999; Zadrozny and Elkan, 2001, 2002). Applying these methods to

multiclass classifiers requires some adaptation. One standard approach reformulates the multiclass setting into many OvA binary problems (one per class) (Zadrozny and Elkan, 2002). One limitation of this approach is that it does not scale well. When the number of classes is large, the calibration data is divided into highly unbalanced subsets that do not contain enough positive examples to solve the OvA binary problems.

Other methods based on Platt scaling (Platt, 1999) involve learning a set of parameters whose size grows with the number of classes. For problems with many classes, they tend to overfit, as demonstrated in this work. The following paragraphs describe more precisely the issues of current approaches.

Behavior of Current Scaling Methods Scaling methods for calibration optimize one or more coefficients that scale the logits vector to minimize the cross-entropy loss on calibration data defined as $l_{CE} = -\sum_{k=1}^{L} 1_{k=y} \cdot \log(f_k(x)) = -\log(f_y(x))$. Minimizing l_{CE} therefore increases the probability of the true class.
We can distinguish two cases to understand what happens during the optimization: whether the prediction $\hat{y}$ is correct or not. In the first case, the confidence score is $s = f_y(x)$: minimizing l_{CE} increases the confidence $f_y(x)$. In the second case, the prediction is incorrect, which implies that $f_y(x) < s$. Minimizing L_{CE} increases the probability of the true class $f_y(x)$ but does not directly change the confidence (because $s \neq f_y(x)$). Instead, the confidence (which was attributed to a wrong class) is *indirectly* lowered through the normalization of the softmax layer.

↪ ***Issue 1:*** *Cross-entropy loss is inefficient for lowering confidence in wrong predictions.*

Some scaling methods have another issue. By design, the number of parameters optimized by Vector Scaling and Dirichlet Calibration grows with the number of classes. When the number of classes is high, these methods overfit the calibration set as shown in Figure 4.6.

↪ ***Issue 2:*** *Vector Scaling and Dirichlet Calibration overfit calibration sets with many classes.*

One-versus-All approach for binary methods The OvA calibration approach allows adapting calibration methods for binary classifiers to multiclass classifiers. To do so, it decomposes the calibration of multiclass classifiers into sets of L binary calibration problems: one for each class k. For each problem, the considered probability is $f_k(x)$, and the associated label $1_{y=k} \in \{0, 1\}$. When calibrating a classifier from data, each binary problem is highly imbalanced, with a ratio between positive and negative examples equal to $\frac{1}{L-1}$ if the classes are equally sampled. For instance, for calibration on ImageNet data, the ratio is 1/999. Out of 25000 examples, only 25 of them have a positive label.

↪***Issue 3:*** *OvA approach leads to highly imbalanced binary problems.*

At test time, each of the L class probabilities is calibrated by a separate calibration model. The resulting probability vector can be normalized to ensure a unit norm. Because each class-conditioned probability is calibrated independently, an OvA strategy can change their ranking, modifying the predicted class. In Table 7.7, reported in the Appendix, we see that accuracy is often negatively impacted in practice.

↪***Issue 4:*** *OvA approach can change the predicted class and negatively impact the accuracy.*

4.4.2 Top-versus-All Approach to Confidence Calibration

General Presentation The proposed approach aims to solve all these identified issues. It is based on one observation: in the calibration definition (4.1) evaluated using the standard ECE metric, only the confidence, i.e., the maximal probability, reflects the likelihood of making an accurate prediction. The probabilities of other classes are not taken into account. However, the standard approach to calibration uses the entire set of probabilities, not just confidence, which introduces unnecessary complexity. The approach can be simplified by reformulating the problem of calibrating multiclass classifiers into a *single* binary problem. This problem can be phrased as: "Is the prediction correct?". In this setting, it is not the predicted probabilities vector that is calibrated; it is only a scalar: the confidence. The remaining probabilities are discarded. This is equivalent to calibrating a *surrogate* binary classifier that predicts whether the class prediction is correct. Since this approach looks at the maximal probability, I call it *Top-versus-All (TvA)*.

Replacing the standard approach by TvA is straightforward. Given the standard calibration data $D_{cal} = \{(x_i, y_i)\}_{i=1}^{N}$, a few data preprocessing steps are added. First, compute the predictions $\hat{y}$ and their correctness: $y^b = 1_{\hat{y}=y}$. Second, create the surrogate binary classifier $f^b(x) = \max_{k \in \mathcal{Y}} f_k(x)$. Finally, build the calibration set for the surrogate binary classifier:

$$D_{cal}^{\text{TvA}} = \{(x_i, y_i^b)\}_{i=1}^{N} \tag{4.3}$$

After this preprocessing, a standard calibration function g, e.g., Temperature Scaling, is used to calibrate the binary classifier. The learning of the calibration function follows its original underlying algorithm but uses the modified calibration data D_{cal}^{TvA}.

Algorithm 1 describes the standard approach, and Algorithm 2 describes the TvA approach in more detail and shows in blue the differences with the standard approach. The TvA approach adds a preprocessing step to keep only the confidences instead of the full probabilities vector. It can be seen as creating a surrogate "correctness" classifier and its associated calibration data. The calibrator is learned for the surrogate classifier and applied to the original classifier at inference time. Also, regularization is added for some scaling methods. For binary methods, there is now only one binary calibrator instead of one per class.

After this general presentation, the following paragraphs explain how TvA impacts the two categories of calibration methods, scaling and binary.

Top-versus-All Approach for Scaling Methods Because the TvA setting reformulates the calibration of multiclass classifiers into a binary problem, the natural loss is the binary cross-entropy:

$$l_{BCE} = -\big(y^b \cdot \log s + (1 - y^b) \cdot \log(1 - s)\big) \tag{4.4}$$

Minimizing this loss results in confidence estimates that more accurately describe the probability of being correct, regardless of the $L - 1$ less likely class predictions. Using the binary cross-entropy as a calibration loss makes an important difference compared to the usual multiclass cross-entropy. The cross-entropy loss takes into account the probability of the *correct* class, while with TvA the binary cross-entropy takes into account the probability of the *predicted* class (i.e., the confidence).

As for the standard approach, only two cases are possible. When the prediction is correct, $l_{BCE} = -\log(s) = -\log(f_y) = l_{CE}$. We get the same result as the cross-entropy loss: minimizing it directly increases the confidence. But when the prediction is incorrect, $l_{BCE} = -\log(1 - s) \neq l_{CE}$. Minimizing the loss now *directly* decreases the confidence.

Algorithm 1 Standard approach

Input:
D_{cal}: $\{(x_i, y_i)\}_{i=1}^N$ the calibration data
f: the multiclass classifier
g: a calibration function ▷ *e.g., Temperature Scaling*

Learn calibration function:
if g **is** scaling method **then**
  loss $l :=$ Cross-Entropy
  Learn g to calibrate f by minimizing l on D_{cal}
else if g **is** binary method **then**
  for $k = 1$ to L **do** ▷ *One-versus-All approach*
    $y_i^b \leftarrow 1_{\mathrm{y_i=k}}$
    $D_{cal}^k \leftarrow \{(x_i, y_i^b)\}_{i=1}^N$
    Learn g_k to calibrate f on D_{cal}^k
  end for
  $g \leftarrow (g_1, g_2, \ldots, g_L)$
end if

Inference:
Use g to calibrate confidences from f

Algorithm 2 Top-versus-all approach

Input:
D_{cal}: $\{(x_i, y_i)\}_{i=1}^N$ the calibration data
f: the multiclass classifier
g: a calibration function ▷ *e.g., Temperature Scaling*

Preprocessing:
$\hat{y}_i \leftarrow \arg\max_{k \in \mathcal{Y}} f_k(x_i)$ ▷ *Compute class predictions*
$y_i^b \leftarrow 1_{\hat{\mathrm{y}}_\mathrm{i}=\mathrm{y_i}}$ ▷ *Compute predictions correctness*
$f^b \leftarrow \max_{k \in \mathcal{Y}} f_k$ ▷ *Create surrogate binary classifier*
$D_{cal}^{\mathrm{TvA}} \leftarrow \{(x_i, y_i^b)\}_{i=1}^N$ ▷ *Build binary calib. set*
Learn calibration function:
if g **is** scaling method **then**
  loss $l :=$ Binary Cross-Entropy
  if g **is** vector or Dirichlet scaling
    loss $l \leftarrow l + \lambda l_{reg}$ ▷ *Add regularization*
  end if
  Learn g to calibrate f^b by minimizing l on D_{cal}^{TvA}
else if g **is** binary method **then**
  Learn g to calibrate f^b on D_{cal}^{TvA}
end if

Inference:
Use g to calibrate confidences from f

This is a key difference compared to using the multiclass cross-entropy loss.
The impact of the reformulation can be seen for Temperature Scaling, which optimizes a coefficient T that scales the logits z_k. The reformulation generates stronger gradients when the prediction is incorrect:

$$\left|\frac{\partial l_{BCE}}{\partial T}\right| > \left|\frac{\partial l_{CE}}{\partial T}\right| \text{ for } s > 0.5 \tag{4.5}$$

with $\frac{\partial l_{BCE}}{\partial T} = \frac{1}{T^2} \cdot \frac{1}{s-1} \cdot (\max_k z_k - \sum_k z_k \cdot f_k)$ and $\frac{\partial l_{CE}}{\partial T} = \frac{1}{T^2}(z_y - \sum_k z_k \cdot f_k)$. See Appendix 7.2.1 for the proof. Because for interesting problems, the confidence satisfies $s > 0.5$ most of the time (as shown in Figure 4.4), the TvA approach strengthens the gradients. The optimization of the temperature T is more efficient as confident incorrect predictions are more heavily penalized. Applying Temperature Scaling usually results in overconfident probabilities, but the TvA approach limits this overconfidence. This can be seen experimentally in Table 7.6, which displays the average confidence of both methods.
↪***Solution for Issue 1:*** *Use the binary cross-entropy loss resulting from TvA approach.*

Regularization of Scaling Methods Overfitting of Vector Scaling and Dirichlet Calibration can be reduced with a simple L2 regularization that penalizes the coefficients of the vector v that are far from the reference value 1.

$$l_{reg}(v) = \frac{1}{L}\sum_{i=1}^{L}(v_i - 1)^2 \tag{4.6}$$

This regularization allows these methods to take advantage of their additional expressiveness without being subject to overfitting. The loss for Vector Scaling becomes $l_{BCE} + \lambda l_{reg}(v)$ where λ is a hyperparameter. Dirichlet Calibration uses additional matrix regularization terms.

↪***Solution for Issue 2:*** *Use L2 regularization.*

Top-versus-All Approach for Binary Methods The TvA approach replaces the OvA approach to apply binary methods to the multiclass setting. TvA transforms the multiclass setting into a *single* binary problem that uses the binary calibration dataset (4.3). In this dataset, the proportion of positive labels equals the classifier's accuracy a. The ratio between negative and positive examples is $\frac{(1-a)N}{aN} = \frac{1}{a} - 1$. For a classifier with 80% accuracy on ImageNet and a calibration dataset of 25000 examples, there are 5000 negative and 20000 positive examples (ratio of 1/4). This is still a bit imbalanced but orders of magnitude smaller than the class-wise binary calibration datasets of the OvA approach (ratio of 1/999).

↪***Solution for Issue 3:*** *By not dividing the calibration data into class-wise datasets, the TvA approach yields a much better balanced binary calibration problem.*

The TvA approach operates on confidence alone, not the full class probabilities vector. This means that the class prediction is already done, and the ranking of the class probabilities does not change. The classifier's prediction and accuracy are unaffected. This scheme allows decoupling accuracy improvements (during training time) and calibration (during post-processing calibration), thus avoiding compromises and reducing development time.

↪***Solution for Issue 4:*** *By operating on confidence alone, the TvA approach does not impact the classifier's prediction or accuracy for binary methods applied to the multiclass scenario.*

Comparison with Competing Approaches Works from (Patel et al., 2020) and (Zhang et al., 2020) are competing methods because, like TvA, they are multiclass-to-binary reductions. This is why TvA cannot be applied on top of them: they already transform the multiclass problem into a binary one using a different strategy.

The shared class-wise strategy of (Patel et al., 2020) and the data ensemble strategy of (Zhang et al., 2020) are described very briefly in subsections 3.2 and 3.3.2 of their respective papers and not rigorously justified. My understanding is that these two strategies do exactly the same thing. To build the calibration set, they concatenate all the class probability vectors so that we get a big probability vector of size $N.L$ (N samples and L classes) as predictions and similarly concatenate the one-hot embedding of the target class (a big vector with N ones and $N.(L-1)$ zeros) as targets. Then, they learn a single calibrator. For each example, this calibrator aims to simultaneously increase the probabilities for the target class (target is 1) and decrease all the other class probabilities (target is 0). The single calibrator is applied to each class probability separately, meaning that the ranking of class probabilities can change, modifying the classifier prediction.

The TvA strategy derives from transforming the multiclass calibration into a single binary problem. The intuition is to learn the calibrator on a surrogate binary classifier and apply this calibrator to the original classifier. This binary classifier is built on top of the original classifier (by applying the max function to the class probabilities vector). They thus share their confidence. However, the binary classifier aims to solve a different task: predicting the correctness of the original classifier. To build the calibration set, all the

confidences are concatenated (a vector of size N) as predictions and concatenate all the correctnesses as targets (also a vector of size N). The correctness value of a given example is 1 if the class prediction is correct; otherwise, it is 0. Then, a single calibrator is learned, similar to the strategy above. However, there is a key difference: this calibrator aims to increase the probabilities for correct predictions and decrease them for incorrect predictions. Note that probabilities here are all confidences (the maximum class probabilities), meaning only the confidences that the calibrator directly increases or decreases are considered. In the strategy from (Patel et al., 2020) and (Zhang et al., 2020), the calibrator has to manage all class probabilities (L times more), even the ones that do not matter, including the lowest class probabilities close to 0. This is less efficient (actually, while this can surely be fixed, the original implementations of IRM and I-Max could not run on ImageNet-21K). This point is closely linked to Issue 1: when the prediction is incorrect, increasing the probability of the correct class indirectly decreases the confidence (strategy from (Patel et al., 2020) and (Zhang et al., 2020)) while the TvA strategy directly decreases the confidence.

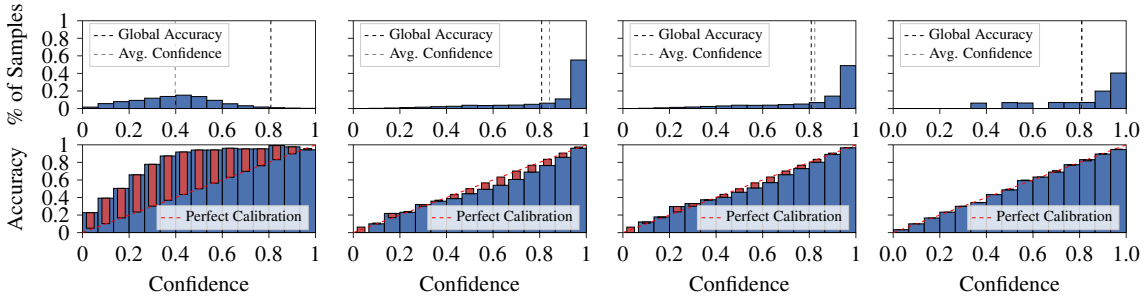
I-Max is more complex because it modifies the Histogram Binning algorithm, while the TvA approach does not. Additionally, (Lin et al., 2022) found that I-Max produces unusable probability vectors. Indeed, they do not sum up to 1, and normalizing them degrades the method's performance.

The TvA approach with practicality and generality in mind. Contrary to (Patel et al., 2020) and (Zhang et al., 2020), the generality of the TvA strategy is demonstrated by applying it on top of existing calibration baselines of different natures (scaling and binary). One of the main goals is that practitioners can easily and quickly try the TvA approach, using just a few lines of code, which can significantly improve the calibration performance of their existing calibration pipeline while having no impact on the predicted class by design (except for VS and Dirichlet calibration (DC)).

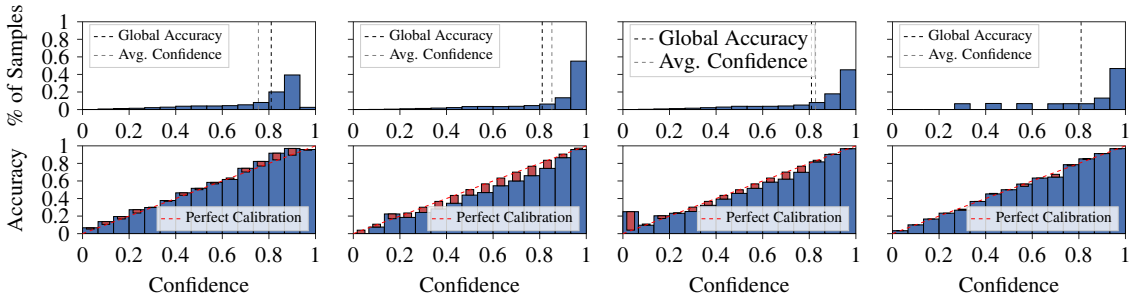
4.4.3 Experiments and Results

Datasets and Models For image classification, the datasets are *CIFAR-10 (C10)* and *CIFAR-100 (C100)* (Krizhevsky et al., 2009) with 10 and 100 classes respectively, *ImageNet (IN)* with 1000 classes, and *ImageNet-21K (IN21K)* (Ridnik et al., 2021) with 10450 classes. For text classification, the datasets are *Amazon Fine Foods (AFF)* (McAuley and Leskovec, 2013) and *DynaSent (DF)* (Potts et al., 2021) for sentiment analysis with 3 classes, *MNLI* (Williams et al., 2018) for natural language inference with 3 classes, and *Yahoo Answers (YA)* (Zhang et al., 2015b) for topic classification on 10 classes. Experiment results are averaged over five random seeds that randomly split the concatenation of the original validation and test sets into calibration and test sets.
The image classification models are *ResNet* (He et al., 2016), *Wide-ResNet-26-10 (WRN)* (Zagoruyko and Komodakis, 2016), *DenseNet-121* (Huang et al., 2017), *MobileNetV3 (MN3)* (Howard et al., 2019), *ViT*, *ConvNeXt* (Liu et al., 2022b), *EfficientNet* (Tan and Le, 2019, 2021), *Swin* (Liu et al., 2021, 2022a), and *CLIP* (Radford et al., 2021). For text classification, the models are *RoBERTa* (Liu et al., 2019) and *T5* (Raffel et al., 2020).
More details about datasets, calibration set sizes, and model weights are in Appendix 7.1.

Baselines The Top-versus-All ($_{\text{TvA}}$) reformulation and regularization ($_{\text{reg}}$) can be applied to different calibration methods. The following scaling methods were tester: *TS* and *VS*, and *DC* (Kull et al., 2019) with the best-performing variant Dir-ODIR, which regular-



(a) ResNet-50 – original (b) ResNet-50 – TS (c) ResNet-50 – TS_{TvA} (d) ResNet-50 – HB_{TvA}



(e) ViT-B/16 – original (f) ViT-B/16 – VS (g) ViT-B/16 – $\text{VS}_{\text{reg\_TvA}}$ (h) ViT-B/16 – HB_{TvA}

Figure 4.4: Reliability diagrams for ResNet-50 and ViT-B/16 when using TS, VS, and HB on ImageNet. The subscript $_{\text{TvA}}$ signifies that the TvA reformulation was used, and $_{\text{reg}}$ means regularization (4.6) was applied. As the methods improve the calibration, the accuracy per bin gets closer to the true accuracy, and the average confidence gets closer to the global accuracy.

izes off-diagonal and bias coefficients. Also, the following binary methods were tested: *HB* (Zadrozny and Elkan, 2001) using for each case the best-performing variant between equal-mass or equal-size bins, *Isotonic Regression (Iso)* (Zadrozny and Elkan, 2002), *Beta Calibration (Beta)* (Kull et al., 2017), and *Bayesian Binning into Quantiles (BBQ)* (Naeini et al., 2015). For comparison, methods with state-of-the-art results on problems with many classes are included: *I-Max* (Patel et al., 2020) and *IRM* (Zhang et al., 2020). More details on code implementations can be seen in Appendix 7.1.

Metrics The primary metric is the ECE (4.2) with 15 equal-width bins, and more metrics are provided in the Appendix. Class-wise metrics are not used because the problem tackled here is confidence calibration. Also, they do not scale well when per-class data is scarce, as discussed in the next paragraph.

Limits of classwise-ECE and top-label-ECE for a High Number of Classes Let us define the ECE for class j: $\text{ECE}_j = \sum_{b=1}^{B} \frac{n_b}{N} |\text{acc}(b,j) - \text{conf}(b,j)|$. The difference compared to (4.2) is that now $\text{acc}(b,j)$ corresponds to the proportion of class j in the bin. Also, $\text{conf}(b,j)$ now is the average probability given to class j for all samples in the bin. Classwise-ECE (Kull et al., 2019) takes the average for all classes: $\text{ECE}_{\text{cw}} = \sum_{j=1}^{L} \text{ECE}_j$. Classwise-ECE considers the full probabilities vectors: all the class probabilities for each prediction. However, this metric does not scale to large numbers of classes. Let us see why with an example.

Let us use a test set of N samples, N/L for each of the L classes (the dataset is

balanced), and a high-accuracy classifier fairly calibrated. The classifier predicts N probability vectors of length L. Predicted probabilities for class j are all the values of the vector at dimension j. Because the classifier has a high accuracy and is fairly calibrated, around N/L values are close to 1 (corresponding to mostly correct predictions), and the remaining ones, around $N - N/L$, are close to 0 (because the predicted class is not class j, and the predicted probability is high for another class).

To compute ECE_j with equal size 15 bins, the predicted probabilities for class j are partitioned into 15 bins. The first bin (with probabilities close to 0), contains $n_1 \approx N - N/L$ samples while the last one (with probabilities close to 1) contains $n_B \approx N/L$ samples. The remaining bins are usually even more empty. That means that the calibration error in the first bin is weighted $n_1/n_B = L - 1$ times more than the last one. For the 1000 classes of ImageNet, $L - 1 = 999$. Figure 4.5 shows the number of samples (n_b) in each bin for an ImageNet classifier.

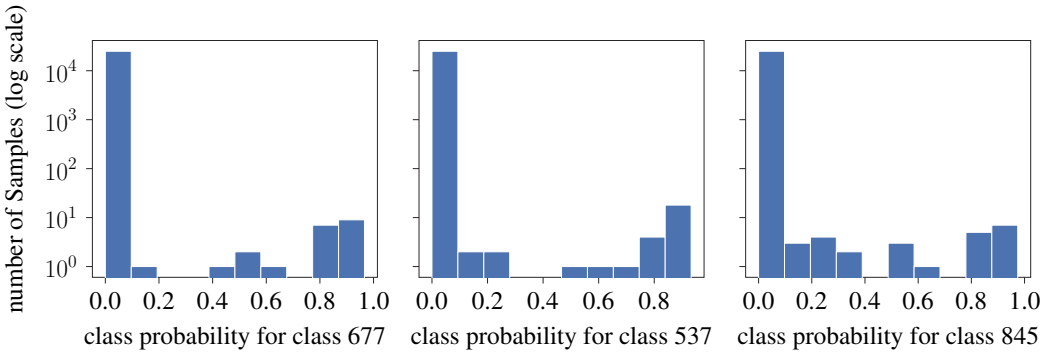


Figure 4.5: Histograms of probabilities for 3 random classes (ViT-16/B on ImageNet).

Because the impact of the calibration error in the bin with the high probabilities is negligible relative to the bin with the low probabilities, the classwise-ECE mostly measures whether probabilities close to 0 are calibrated. I argue this is not what we are interested in: what matters more is the calibration of higher values of the probabilities.

Top-label ECE (Gupta and Ramdas, 2022) is another interesting metric that does not scale to large numbers of classes either. Top-label-ECE divides data into subsets according to the predicted class, computes the ECEs of these subsets, and averages them. For an ImageNet test set of 25000 samples (25 per class), data is divided into 1000 subsets of ≈ 25 samples each (the classifier is high-accuracy, most of the time the predicted class is equal to the true class). The ECE is computed for each subset containing only 25 samples. To compute the ECE, samples are typically partitioned into 15 bins. The number of samples per bin does not allow a correct estimation of the average confidence or accuracy.

Top-versus-All Results For visual qualitative results, Figure 4.4 displays reliability diagrams (Niculescu-Mizil and Caruana, 2005). We can observe that initially, ResNet-50 is highly underconfident, and ViT-B/16 is slightly underconfident. Applying TS and VS solves the underconfidence and makes the models slightly overconfident. TvA further improves these methods, and the average confidence gets closer to the accuracy. HB_{TvA} is even better and approaches perfect calibration.

Table 4.2 shows the results of applying the TvA reformulation to several calibration methods. For clarity, results are averaged over families of models (models based on the same architecture) and the full results are available in Tables 7.2 and 7.3 of the Appendix.

Table 4.2: ECE in % (lower is better). The subscript $_{\text{TvA}}$ denotes that the reformulation was applied to the calibration method. IRM and I-Max are competing methods. The best method for a given model is in bold. Methods in purple impact the model prediction, potentially degrading accuracy; methods in teal do not. Values are averaged over five random seeds. Results are averaged over models of the same family. Detailed results for all models can be seen in Tables 7.2 and 7.3 of the Appendix.

					scaling methods						binary methods					
Dataset	**Models**	**Uncal.**	**IRM**	**I-Max**	**TS**	**TS$_{\text{TvA}}$**	**VS**	**VS$_{\text{reg,TvA}}$**	**DC**	**DC$_{\text{reg,TvA}}$**	**Iso**	**Iso$_{\text{TvA}}$**	**BBQ**	**BBQ$_{\text{TvA}}$**	**HB**	**HB$_{\text{TvA}}$**
C10	ConvNets	1.77	0.67	0.61	1.20	1.25	1.17	1.17	1.16	1.17	1.12	0.68	1.17	0.77	1.06	**0.43**
	CLIP	5.03	1.03	0.94	0.78	**0.73**	2.56	1.85	2.56	1.84	1.05	0.86	1.39	0.86	1.82	**0.73**
C100	ConvNets	6.04	1.30	1.15	4.65	2.96	4.91	2.35	4.89	2.35	5.33	1.38	9.63	1.35	9.56	**1.02**
	CLIP	10.37	2.90	2.57	2.54	2.51	7.78	2.86	7.55	1.84	2.53	1.61	7.48	1.48	7.23	**1.39**
IN	ResNet	15.26	1.31	1.07	2.65	1.89	2.77	1.67	3.59	2.23	3.05	0.79	8.41	0.76	7.49	**0.55**
	EffNet	15.72	0.68	0.48	3.48	2.59	3.67	1.26	3.65	1.23	2.83	0.68	6.55	0.64	4.39	**0.43**
	ConvNeXt	16.46	0.82	0.58	3.67	2.25	4.05	1.37	4.04	1.35	2.97	0.75	7.41	0.68	5.13	**0.52**
	ViT	4.40	0.81	0.61	4.09	2.96	4.31	2.02	4.31	1.99	3.60	0.77	6.64	0.73	6.59	**0.52**
	Swin	5.85	0.75	0.49	3.63	2.91	4.04	1.70	4.03	1.67	3.19	0.74	7.09	0.71	5.39	**0.48**
	CLIP	1.96	1.08	**0.72**	1.89	1.82	1.63	1.05	32.03	67.65	2.35	0.92	8.31	0.93	7.16	0.80
IN21K	MN3	12.34	err.	err.	8.69	4.39	2.52	2.40	58.84	81.16	2.00	0.21	err.	0.20	5.50	**0.17**
	ViT-B/16	6.27	err.	err.	8.92	6.55	2.38	1.54	8.22	3.20	2.14	0.22	err.	0.24	7.89	**0.12**
AFF	T5	5.47	0.27	0.26	1.10	1.15	1.52	1.42	1.18	1.31	0.37	0.27	0.39	0.28	2.87	**0.17**
	RoBERTa	7.37	0.30	0.28	2.40	2.33	1.41	1.85	1.38	1.68	0.52	0.27	0.75	0.35	4.02	**0.20**
DS	T5	8.86	1.39	1.38	2.19	2.17	6.13	2.00	5.91	2.02	1.50	1.55	1.38	1.58	1.90	**1.12**
	RoBERTa	16.12	1.56	1.50	12.07	12.07	14.66	6.80	13.90	5.57	1.71	1.53	1.64	1.14	1.05	**0.91**
MNLI	T5	7.04	0.72	0.70	2.81	2.80	4.46	1.79	4.31	1.82	0.80	0.74	1.38	0.69	2.09	**0.43**
	RoBERTa	9.22	0.89	0.71	5.72	5.72	6.99	1.92	6.59	1.99	1.00	0.92	1.67	0.84	1.02	**0.60**
YA	T5	7.84	0.80	0.81	1.07	1.35	3.70	1.16	3.75	1.15	1.73	0.82	2.81	0.96	3.65	**0.69**
	RoBERTa	19.59	0.97	0.79	12.39	12.38	16.47	2.52	16.07	2.21	1.92	0.99	5.00	0.75	3.41	**0.58**

In most cases, the TvA reformulation significantly lowers the ECE by dozens of percent. Without TvA, binary methods often introduce noise into the prediction and degrade the classifier's accuracy (see Table 7.7), making them inapplicable in a practical setting. TvA solves the issue as it only scales the confidence (after the prediction is made) and makes binary methods outperform scaling methods. Improvements due to TvA are consistent across models. However, exceptions are observed for the smaller models (ResNet-18, and ResNet-34) and CLIP, for which considering cosine similarities as logits could explain the different behavior. Results show that DC is sensitive to hyperparameter tuning, and its performance is usually not much better than VS, which is consistent with (Kull et al., 2019). In some cases, the optimization diverges and it leads to very poor results, e.g., for CLIP on ImageNet. Improvements due to TvA are also consistent across datasets although they tend to increase with the number of classes. Improvements on ImageNet are usually better than on CIFAR-100, whose improvements are usually better than on CIFAR-10. This is notable with e.g., TS or HB. For text datasets with only three classes (AFF, DS, and MNLI), TS does not benefit from TvA, but other methods do, despite the small number of classes. According to (Chen et al., 2023b), TS is among the best calibration methods for the text classification tasks considered here, even compared to ones that retrain the model. Even so, the method HB$_{\text{TvA}}$ significantly outperforms it.

Some methods' current implementations could not handle the large scale of ImageNet-21K, resulting in out-of-memory errors written as "err." in the Table. For I-Max and IRM, this is because they consider the full probability vectors while TvA efficiently uses data by considering only confidence values. Indeed, TvA handles this scale without difficulty.

Additional results are included in Appendix 7.2.2. Tables 7.2 and 7.3 contain the full results for ECE, while the standard deviations are in Table 7.4. According to Table 7.6, we observe that ImageNet networks are mostly underconfident. This is aligned with (Galil et al., 2023b) and goes against previous knowledge on overconfidence, which was initially believed to be linked to network size (Guo et al., 2017). Table 7.7 provides the accuracies after calibration. Table 7.8 exhibits that ECE with equal-mass bins gives similar values as standard ECE. In most cases, TvA also lowers the Brier score, except for Iso, which has the lowest score overall as shown in Table 7.9. The reason why is unclear.

Calibration methods can also be applied to LLMs using In-Context Learning (ICL) (Zhao et al., 2021b; Han et al., 2022; Jiang et al., 2023; Zhou et al., 2023; Abbas et al., 2024) to tackle text classification tasks. The primary goal of these methods is to improve model accuracy. TvA was not designed for this objective, but it can still be applied on top of an existing method that improves the accuracy. TvA then lowers the calibration error while keeping the accuracy gain. Results are in the Appendix in Table 4.3.

To summarize the results for practical use, the experiments show that Histogram Binning (within the TvA or I-Max setting) is the best calibration method overall, providing ECE values consistently below 1%. This is the method I advise using. However, suppose the underlying application requires a confidence with continuous values, e.g., to rank the predictions in the case of selective classification. In that case, I advise using a method that also improves the AUROC, shown in Table 4.4, such as Temperature Scaling or Isotonic Regression.

Solving Overfitting with Regularization and TvA On ImageNet, VS and DC overfit the calibration set, degrading the calibration on the test set. The lower performance of VS relative to TS indicates this overfitting. As visualized in Figure 4.6, combining the binary cross-entropy loss used in the TvA reformulation and an additional regularization term prevents overfitting. The value $\lambda = 0.01$ works well across models. Initializing the vector coefficients to $\frac{1}{T}$ with T obtained by TS_{TvA} helps further improve performance. On the other hand, I found that penalizing values far from $\frac{1}{T}$ instead of 1 degrades the performance.

Influence of the Calibration Set Size The size of the calibration set influences the performance of the different methods, as seen in Figure 4.7. TS and TS_{TvA} do not benefit from more data due to their low expressiveness. VS does not improve the ECE because of the overfitting problem. In contrast, $\text{VS}_{\text{reg\_TvA}}$ benefits from more calibration data. With enough data ($\approx$ 15000), it outperforms TS_{TvA}. Binary methods using the standard OvA approach have poor performance and need a large amount of data to be competitive. Using TvA, they get excellent performance with little data.

Results for LLMs Using In-Context Learning LLMs exhibit an ICLs capability, meaning they can learn from just a few examples in the context. It works by constructing a prompt that includes input-output pairs demonstrating the considered task, followed by a query for a new input. See (Dong et al., 2022) for a survey. Recent works develop calibration methods whose main goal is to improve the performance of ICLs for LLMs, without requiring a complicated model fine-tuning. (Zhao et al., 2021b) uses a customized variant of Platt scaling (more specifically, Vector Scaling). Their method infers good values of the vector scaling parameters in a data-free procedure. The idea is that for a "content-free" input, e.g., "N/A", the calibrated probability has a 50% chance (for a binary classification

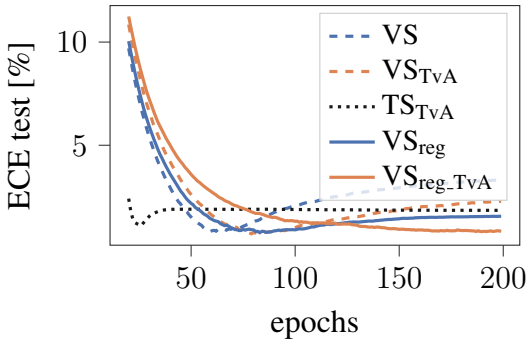


Figure 4.6: Test ECE evolution during training with ResNet-50 on ImageNet. The combination of regularization and TvA prevents overfitting of Vector Scaling. Temperature Scaling with TvA is shown for reference.

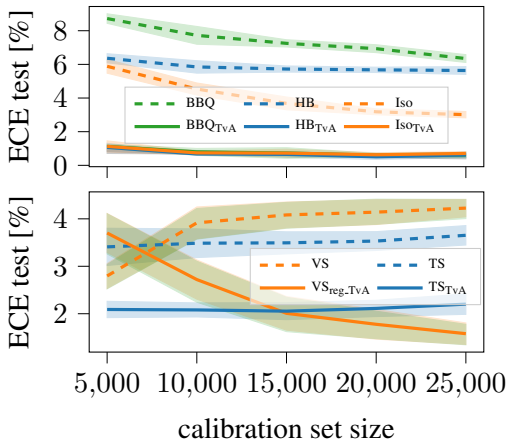


Figure 4.7: Influence of the calibration set size for ResNet-101 on ImageNet. Binary methods at the top and scaling methods at the bottom.

Table 4.3: Calibration methods applied to in-context learning of LLMs. Accuracy and ECE are in %.

		Dataset	TREC		SST-5		DBpedia	
			Acc. (↑)	ECE (↓)	Acc. (↑)	ECE (↓)	Acc. (↑)	ECE (↓)
Model	Shots	Method						
GPT-J 6B	0	Uncalibrated	24.7	29.7	33.7	22.5	19.7	27.4
		ConC	40.0	14.0	40.7	10.3	47.7	24.6
		LinC	**58.9**	26.4	**46.3**	11.0	**62.2**	12.8
		LinC+HB$_{TvA}$	**58.9**	**6.5**	**46.3**	**7.0**	**62.2**	**5.7**
	1	Uncalibrated	43.7	12.1	36.3	30.9	58.7	14.2
		ConC	41.7	13.6	**50.7**	14.2	82.7	6.9
		LinC	**59.9**	9.1	50.1	12.3	**84.4**	6.6
		LinC+HB$_{TvA}$	**59.9**	**3.9**	50.1	**7.3**	**84.4**	**5.1**
	4	Uncalibrated	26.0	41.6	51.3	28.2	89.0	15.7
		ConC	40.3	14.4	**54.3**	8.8	94.0	6.9
		LinC	**57.9**	9.7	53.6	10.6	**94.3**	5.7
		LinC+HB$_{TvA}$	**57.9**	**5.2**	53.6	**7.1**	**94.3**	**4.8**
	8	Uncalibrated	36.0	26.0	48.3	9.7	92.3	9.2
		ConC	46.7	15.5	43.7	11.7	92.0	6.8
		LinC	**60.7**	**6.3**	**51.7**	9.5	**93.9**	5.8
		LinC+HB$_{TvA}$	**60.7**	6.6	**51.7**	**7.6**	**93.9**	**2.6**
Llama-2 13B	0	Uncalibrated	48.7	21.4	34.0	17.6	54.3	19.7
		ConC	71.7	18.7	33.3	17.2	75.3	17.2
		LinC	**73.3**	11.4	**47.6**	11.3	**84.4**	16.2
		LinC+HB$_{TvA}$	**73.3**	**9.3**	**47.6**	**6.7**	**84.4**	**4.1**
	1	Uncalibrated	63.0	8.6	41.3	29.4	90.7	11.4
		ConC	76.0	**5.9**	41.0	12.6	92.3	5.2
		LinC	**79.7**	6.3	**48.7**	12.1	**93.1**	4.4
		LinC+HB$_{TvA}$	**79.7**	6.0	**48.7**	**9.6**	**93.1**	**3.0**
	4	Uncalibrated	60.0	12.0	50.7	37.6	94.0	9.9
		ConC	71.3	6.9	51.3	18.6	**95.3**	3.8
		LinC	**75.6**	8.0	**52.9**	15.1	**95.3**	3.8
		LinC+HB$_{TvA}$	**75.6**	**4.0**	**52.9**	**7.6**	**95.3**	**2.2**
	8	Uncalibrated	70.0	**5.2**	**55.0**	7.0	94.7	5.9
		ConC	**73.7**	12.6	44.0	22.8	94.3	3.9
		LinC	73.5	9.4	50.2	14.4	**95.3**	3.9
		LinC+HB$_{TvA}$	73.5	7.0	50.2	**4.2**	**95.3**	**2.1**

task) of removing a bias toward the positive or negative class. Here, this method is denoted as *ConC*. (Abbas et al., 2024) builds on top of this work but uses a calibration set to learn the scaling parameters by minimizing the cross-entropy loss. This can be considered as Matrix Scaling. This method is denoted as *LinC*. (Zhou et al., 2023) proposes a per-class normalization of the probabilities on a given batch. (Jiang et al., 2023) estimates the in-context model label marginal p(y) from limited data and uses it to calibrate the model probabilities. Paper (Han et al., 2022) uses a Gaussian mixture model.

In this section's experiments, I tested a two-step calibration. First, the state-of-the-art method LinC is applied to maximize the accuracy by learning scaling parameters on a calibration set. Then, HB_{TvA} is applied to scale the confidences to lower the calibration error ECE while preserving the accuracy gains. The same calibration set is used for the two methods. LinC performance depends on hyperparameter values, but to keep the experiments simple, I fixed the following values: 100 epochs, a learning rate of 0.001, and 300 calibration samples. It means that the reported performance of LinC is suboptimal and could be enhanced even more. The experimental setting is the same as (Abbas et al., 2024). The models used are GPT-J with 6B parameters (Wang and Komatsuzaki, 2021) and Llama-2 with 13B parameters (Touvron et al., 2023). The text classification datasets are TREC (Voorhees and Tice, 2000) for question classification with 6 classes, SST-5 (Socher et al., 2013) for sentiment analysis with 5 classes, and DBpedia (Zhang et al., 2015a) for topic classification with 14 classes. The 0-shot, 1-shot, 4-shot, and 8-shot learning settings were tested. Five different sets of 300 test samples were randomly selected, and results are averaged over 5 seeds. The accuracy and ECE were evaluated for each configuration. Please see Table 4.3 for the results. In most cases, LinC+HB_{TvA} achieves the best accuracy and ECE.

Selective Classification Results Even though calibration and SC are linked, improvements in calibration do not directly translate to better SC. Table 4.4 shows the AUROC, a metric for SC. HB_{TvA}, the best calibration method overall, actually degrades the AUROC in most cases. This can be explained by the fact that SC benefits from a continuous score able to finely discriminate between certain and uncertain examples, allowing to control the coverture – risk compromise, but HB discretizes the confidences into, e.g., 10 different values. In general, TvA does not significantly improve the AUROC. The best method overall for SC is the original Isotonic regression.

Limitations The TvA approach tackles confidence calibration and is unlikely to improve performance for stronger notions of calibration, such as class-wise calibration. However, confidence calibration is useful for many practical cases, such as SC (Geifman and El-Yaniv, 2017), OOD detection (Hendrycks and Gimpel, 2017), or active learning (Li and Sethi, 2006). Also, calibration improvements are less significant for problems with few classes (≤ 10) than for problems with many classes, but the TvA approach still provides the best results.

Analysis The experiments demonstrated that reformulating the confidence calibration of multiclass classifiers as a single binary problem significantly improves the performance of baseline calibration techniques. The competitiveness of scaling methods is increased, and binary methods use per-class calibration data more efficiently without altering the model's accuracy. In short, the TvA reformulation enhances many existing calibration

Table 4.4: AUROC in % (higher is better). Methods in purple impact the model prediction, potentially degrading accuracy; methods in teal do not. Improvements from the uncalibrated model are colored in blue and degradations in orange. Results are averaged over models of the same family. Detailed results for all models can be seen in Table 7.5 of the Appendix.

					scaling methods						binary methods					
Dataset	**Models**	**Uncal.**	**IRM**	**I-Max**	**TS**	**TS$_{TvA}$**	**VS**	**VS$_{reg,TvA}$**	**DC**	**DC$_{reg,TvA}$**	**Iso**	**Iso$_{TvA}$**	**BBQ**	**BBQ$_{TvA}$**	**HB**	**HB$_{TvA}$**
C10	ConvNets	91.50	91.34	90.72	91.47	91.45	91.88	**92.20**	91.88	92.19	92.10	91.35	75.53	86.06	74.69	84.43
	CLIP	91.36	91.27	91.59	91.46	91.46	92.07	92.28	92.21	**92.36**	91.25	91.18	88.50	90.57	88.59	90.02
C100	ConvNets	86.15	86.08	85.36	86.10	85.99	86.29	86.70	86.30	86.70	**87.31**	86.06	82.09	85.41	82.63	84.35
	CLIP	83.30	83.30	83.21	84.30	84.29	84.82	86.01	84.88	**86.09**	85.33	83.19	**86.09**	83.19	85.97	83.04
IN	ResNet	84.16	84.16	83.71	85.91	85.79	85.71	85.85	85.52	85.62	**86.63**	84.13	83.30	83.95	82.88	83.64
	EffNet	84.42	84.41	83.73	86.31	86.04	84.98	85.33	84.96	85.32	**86.94**	84.37	81.76	84.37	80.31	83.71
	ConvNeXt	82.31	82.32	81.71	85.17	84.80	84.71	85.01	84.71	85.01	**86.94**	82.27	81.54	82.25	80.07	81.80
	ViT	85.93	85.93	85.21	86.45	86.27	85.51	85.63	85.51	85.65	**87.04**	85.90	81.11	85.71	81.10	85.09
	Swin	85.31	85.33	84.58	86.20	85.97	85.02	85.25	85.02	85.24	**87.01**	85.31	80.67	85.28	80.31	84.48
	CLIP	81.55	81.53	81.23	81.55	81.53	82.56	**83.97**	78.63	80.78	82.30	81.52	82.34	81.47	82.47	81.10
IN21K	MN3	68.79	err.	err.	67.80	65.89	80.00	81.00	61.24	51.94	79.62	68.77	err.	68.77	**90.86**	68.40
	ViT-B/16	72.99	err.	err.	74.36	73.17	79.78	81.42	76.29	78.92	79.66	73.10	err.	73.20	**90.27**	71.95
AFF	T5	83.19	83.16	82.42	82.93	82.92	86.84	**87.96**	87.78	87.91	86.09	83.21	84.44	80.87	79.03	76.83
	RoBERTa	85.70	85.58	81.34	85.67	85.67	86.50	87.50	**87.57**	87.45	85.64	85.69	81.57	75.04	75.71	74.33
DS	T5	77.81	77.74	77.07	77.96	77.96	77.97	78.76	78.59	78.83	**78.88**	77.76	75.80	75.58	71.70	74.68
	RoBERTa	75.67	75.49	72.79	75.82	75.82	75.74	75.99	75.83	76.09	**76.11**	75.56	62.90	66.72	59.50	68.38
MNLI	T5	83.22	83.11	81.50	83.31	83.31	83.32	83.50	83.66	**83.73**	83.50	83.14	73.16	75.09	65.30	75.54
	RoBERTa	82.97	82.91	79.44	83.07	83.07	83.03	83.30	83.22	**83.32**	83.28	82.91	68.34	72.44	60.82	73.72
YA	T5	81.34	81.26	80.77	81.35	81.34	81.40	81.41	81.43	**81.47**	81.42	81.29	79.85	80.35	78.20	79.54
	RoBERTa	78.88	78.83	76.72	79.11	79.11	79.02	79.08	79.14	**79.44**	79.11	78.81	74.72	72.93	73.43	70.43

methods with little to no change in their algorithm. Extensive experiments with state-of-the-art image classification models on complex datasets and with text classification demonstrate the approach's scalability and generality.

Better calibration does not always improve Selective Classification. SC helps define a domain having an estimation of the global accuracy (which holds when no distribution shift happens). While SC is binary (data in or out), calibration offers finer uncertainty quantification for each prediction. Including the notion of calibration in defining a reliability domain is still unclear. One perspective of the thesis is to understand better the link between calibration and SC, as explained in section 6.3.

4.5 Calibrate with Synthetic Data

4.5.1 Use a Class Conditional GAN to Generate Calibration Data

Another idea I briefly explored was to use synthetic data as calibration data. Note that I only conducted preliminary experiments, and work in this section is not a major contribution but rather opens a new perspective. This was inspired by working on calibration in section 4.4 and GANs in chapter 3. Post-hoc calibration methods require data separate from the training set, usually from the validation or test set, called calibration data. The issue is that this data is usually scarce, and using part of the test set might degrade the evaluation quality. In section 4.4, I developed the TvA approach to efficiently apply existing calibration methods, even when calibration data is scarce, which easily happens for problems with many classes. An alternative approach described here is to compensate for scarce real data by using synthetic data.

The proposed method is straightforward: train a class conditional GAN on the same training data as the classifier and generate synthetic data for calibration. See Figure 4.8 for an illustration. The GAN is conditioned by the class: class is encoded as a one-hot vector given to the generator as input. For calibration, the class condition is considered the ground truth. Because some images are ambiguous, the classifier prediction and the class condition can differ. In those cases, the classifier prediction is considered incorrect for the calibration process. Because the class condition is an input, we can choose its value. Here, because a balanced dataset with the same number of samples for each class is used, classes are sampled uniformly.

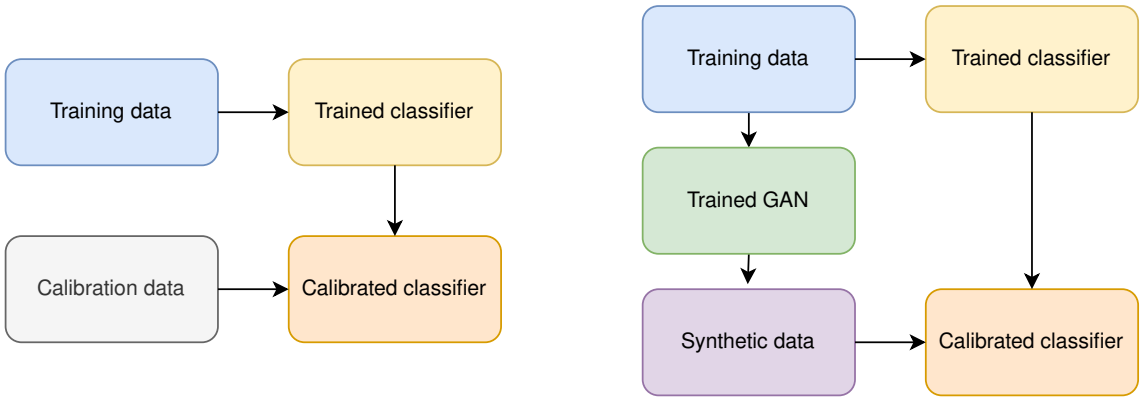


Figure 4.8: The diagram on the left represents the standard pipeline: a trained classifier is calibrated using calibration data usually sampled from the validation or test data. The diagram on the right illustrates how to use synthetic data for calibration.

4.5.2 Experiments and Results

Setting Classifiers are pre-trained on CIFAR-10 (Krizhevsky et al., 2009), a simple image classification dataset with 10 classes. The model architectures considered are: DenseNet (Huang et al., 2017), GoogLeNet (Szegedy et al., 2015), Inception-v3 (Szegedy et al., 2016), MobileNet-v2 (Sandler et al., 2018), ResNet (He et al., 2016), and VGG (Simonyan and Zisserman, 2015). The calibration methods tested are Temperature Scaling and Vector Scaling. Several calibration datasets are compared. The standard calibration approach uses the validation set, which is used for reference. Data augmentation techniques are used for the validation dataset. The augmentations used are random crops, horizontal flips, and synthetic data from the GAN trained on the same training data as the classifier. For this data, the GAN's class condition is the ground truth, and the classifier prediction might differ because some images are ambiguous. The ECE is computed with 15 equal size bins.

Generative Model The GAN used is based on the StyleGAN2-ADA architecture, with a conditional FID (Heusel et al., 2017) of 2.42, a rather good value. Figure 4.9 shows some samples of real and synthetic images for all ten classes. The model implementation and weights are from the official GitHub repository[1].

[1] https://github.com/NVlabs/stylegan2-ada-pytorch

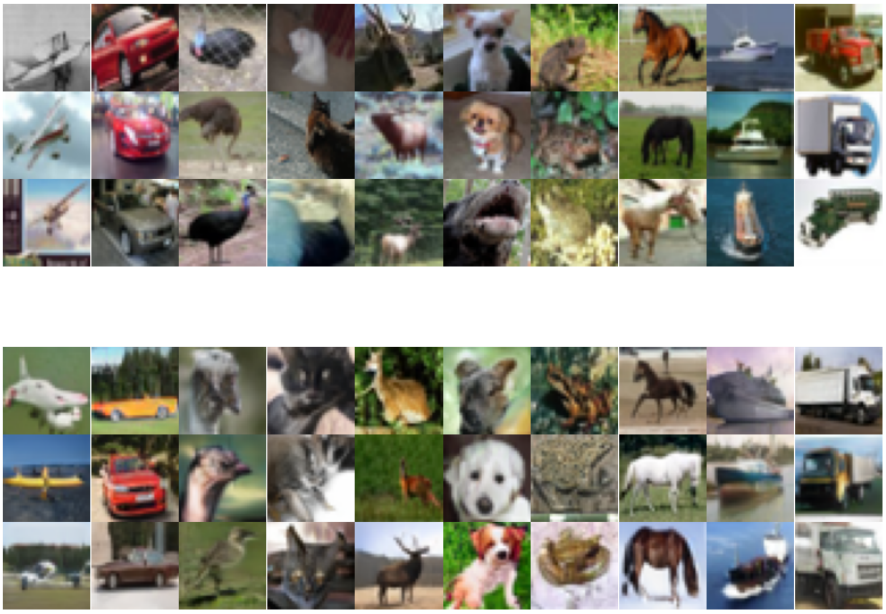


Figure 4.9: CIFAR10 images at the top, synthetic images from StyleGAN2-ADA at the bottom, for all ten classes.

Synthetic Data for Calibration Table 4.5a shows the ECE results for different calibration datasets. All datasets contain 5000 examples. For Temperature Scaling, using the standard validation set is the best choice most of the time. Using data augmentations does not improve the ECE, and using synthetic data is worse. For Vector Scaling, augmenting the validation data similarly does not improve the ECE. However, using the synthetic data is at least competitive with the validation data and often better. Improvements are not huge, but replacing real annotated data with synthetic data can be useful, especially when calibration data is sampled from the test set, such as for ImageNet, leaving less data for thorough evaluation.

Filtering High/Low Confidences I studied whether calibration data with low- or high-confidence samples improves calibration. To do so, calibration data is either a random half of the original validation data or the halves with the lowest or highest confidence samples. All datasets thus contain 2500 examples. Results are shown in Table 4.5b. Using low-confidence samples is rarely better than random sampling. Using high-confidence samples is even worse, probably because most samples correspond to correct predictions, leading the calibration process to increase all probabilities. If controlling the confidence of the calibration samples was useful, the confidence-conditioned GAN developed in section 3.4 could have been used to improve the calibration, but it seems that using random samples is better.

Analysis Using synthetic data for calibration leads to similar or improved results compared to using validation data separate from the training data when using Vector Scaling. Because calibration data needs to be different from the training and test data, generating synthetic calibration data is useful because the model training and evaluation remain unchanged, and the calibration improves. However, there are some limitations. For

Table 4.5: ECE in % computed on the test set for different CIFAR-10 models, lower is better. Temperature Scaling and Vector Scaling methods are compared, and reference values for the uncalibrated model are shown as a reference. Values are averages over five seeds.

(a) Three calibration datasets compared: the validation set, the validation set using data augmentation, and a synthetic dataset

Method	Uncal.	Temperature Scaling			Vector Scaling		
Dataset origin		Val.	Val. aug.	Synth.	Val.	Val. aug.	Synth.
DenseNet-121	2.22	**1.77**	1.84	3.18	1.89	1.88	1.78
DenseNet-161	2.12	1.99	2.05	3.49	1.94	2.01	**1.90**
DenseNet-169	2.54	2.07	2.18	3.36	1.95	**1.93**	1.99
GoogLenet	1.47	1.18	1.39	1.31	1.29	1.41	**0.90**
Inception-v3	1.98	1.55	1.55	2.60	1.46	**1.42**	1.67
MobileNet-v2	2.59	**1.47**	**1.47**	2.50	1.57	1.63	1.59
ResNet-18	2.03	1.72	1.76	2.59	1.44	1.58	**1.25**
ResNet-34	2.71	2.14	2.37	3.23	2.06	2.36	**2.03**
ResNet-50	2.27	1.57	1.48	2.69	1.79	1.73	**1.46**
VGG-11-BN	1.59	1.69	1.78	2.04	1.46	1.44	**1.34**
VGG-13-BN	1.19	1.17	1.22	1.73	**1.12**	1.31	1.29
VGG-16-BN	1.72	1.74	2.00	2.90	1.83	2.00	**1.71**
VGG-19-BN	2.07	2.17	2.47	3.32	2.09	2.38	**2.01**

(b) Three calibration datasets compared: a random half of the validation set, the half with the lowest confidences, the half with the highest confidences

Method	Uncal.	Temperature Scaling			Vector Scaling		
Dataset origin		Val.	Val. low conf.	Val. high conf.	Val.	Val. low conf.	Val. high conf.
DenseNet-121	2.22	**1.73**	3.00	2.80	1.82	2.81	3.75
DenseNet-161	2.12	1.99	2.80	2.36	**1.88**	2.45	3.48
DenseNet-169	2.54	2.29	2.93	2.69	**2.13**	2.51	3.80
GoogleNet	1.47	1.22	**1.02**	2.01	1.25	1.13	5.31
Inception-v3	1.98	1.58	2.29	2.61	**1.53**	2.44	3.01
MobileNet-v2	2.59	**1.49**	2.52	3.44	**1.49**	2.83	8.99
ResNet-18	2.03	1.74	2.30	2.64	**1.56**	1.87	4.25
ResNet-34	2.71	2.19	3.31	3.49	**2.17**	2.81	5.42
ResNet-50	2.27	**1.62**	2.79	3.10	1.81	2.77	5.39
VGG-11-BN	1.59	1.66	1.88	2.79	**1.43**	1.56	7.28
VGG-13-BN	1.19	**1.17**	1.44	2.57	1.19	1.32	4.79
VGG-16-BN	1.72	**1.68**	2.60	3.22	1.72	2.70	4.99
VGG-19-BN	**2.07**	2.18	3.30	2.77	2.19	2.94	4.40

CIFAR-10, there are no significant calibration improvements when using more than a few thousand samples, so generating more data is not useful. Selecting calibration samples according to the classifier confidence does not improve calibration. The main limitation is that generating images is difficult, and GANs do not work as well for more complex data. The StyleGAN architecture was developed for structured images such as faces and does not scale well to, e.g., ImageNet. A perspective is then to look at diffusion models that can generate complex images of high quality.

4.6 Discussion

This chapter looked at SC and calibration. SC directly allows defining a reliability domain using a threshold on a confidence score. One has to choose a good confidence score, and we saw in this chapter that using the classifier maximum predicted probability (MSP) is an excellent baseline. The choice of the threshold is made from an accuracy or coverage constraint using some validation data. There are a few limits to this approach. First, the "guaranteed" selective accuracy for a given threshold is estimated on validation data and becomes invalid if new data does not come from the same distribution: it is not robust to distribution shifts. For instance, let us suppose that a classifier is particularly bad for one specific class, which is rare in the validation data. The estimated selective accuracy is not impacted much by it. However, if this class suddenly becomes more frequent after deployment, the accuracy will be degraded. The selection function is designed with and evaluated on in-distribution data only. It does not work for OOD data (e.g., images of a new class) or against adversarial attacks because the classifier confidence for such data might be high for completely incorrect predictions. This might not be a problem for controlled deployment environments, which prevents OOD data and adversarial attacks. Second, while the function defining in or out of domain is clear (thresholding on confidence), it provides no insight into the semantics of "good" data versus "bad" data. It does not help understand or improve the model.

Calibration helps estimate the uncertainty of single data points better, whereas SC only gives a selective accuracy for all selected data. SC looks at *relative* confidence values to rank data from "easy" to "hard", while calibration requires *absolute* confidence values that truly reflect the prediction uncertainty. Calibration helps quantify the criticality of hard data points: how often do they lead to failure? Usually, better calibration leads to better SC, but it depends on the models and calibration methods (see Table 4.2). It shares the same limitations as SC: it only applies to in-distribution data and does not provide semantic insights.

Chapter 5

Incorporating Textual Descriptions

5.1 Introduction

In the first chapter, we leveraged generative models to create hard-to-classify images that illustrate the classifier limits, following an XAI point of view. In the second chapter, we have explored selective classification and calibration, considering a more quantitative point of view. These fields relate to uncertainty quantification and knowing when predictions are reliable to possibly abstain from making an uncertain decision. This chapter studies another aspect: describing a classifier's reliability with semantic attributes. The goal is to build a link between data descriptions and classifier performance and use it to define a reliability domain.

Section 5.2 describes diffusion models and the Stable Diffusion text-to-image generative model. These models are an essential tool to better apprehend the work done in this chapter.

Section 5.3 looks at the identification of classifier failures using generative models, in line with recent works (Wiles et al., 2022; Vendrow et al., 2023; Metzen et al., 2023). I first describe the approach of Metzen et al. (2023) that identifies text-described subdomains likely to lead to classifier failure. This work uses synthetic data generated from textual descriptions to identify failures that might happen in the real world. Indeed, the real-world performance of image classifiers might differ from what was estimated on a validation set. In particular, classifiers may perform well for conditions frequently encountered during training but poorly for other infrequent conditions.

Recent advances in text-to-image generative models make them valuable for benchmarking computer vision models such as image classifiers: they can generate images conditioned by textual prompts that cause classifier failures, allowing failure conditions to be described with textual attributes. However, the generation cost becomes an issue when many synthetic images need to be generated, which is the case when many different attribute combinations need to be tested.

I propose an image classifier benchmarking method formulated as an iterative process that alternates image generation, classifier evaluation, and attribute selection. Based on Bayesian Optimization, this process efficiently explores the attributes leading to poor behavior. Problematic subdomains – subsets of data sharing a textual description – are identified much more quickly and extensively than in previous work.

Section 5.4 explores how such semantically-described subdomains can be used for se-

lective classification. In the previous section, the goal was to find problematic subdomains likely leading to classification failure. Here, the goal is to identify in which conditions the classifier is reliable to reconnect with the notion of reliability domain. To the best of my knowledge, most related works look at classifier failures, not classifier successes. Defining a domain requires knowing when the classifier is successful, and the ability to identify *some* failures is not enough to do so.

We have seen in chapter 4 that selective classification is a way to define a reliability domain by filtering out data for which the classification is likely to be wrong. The standard way to filter data is to use a threshold on a confidence score computed for each prediction. We have seen in section 4.3 that using the classifier's MSP is a competitive and simple approach. One issue with this approach is that the classifier confidence does not say much about which *kind* of data is likely to work well. One key aspect of this thesis is to link data description with classification performance: we want a semantic description of a reliability domain.

In this section, I present *Semantic Selective Classification*: a way to filter out data likely to be wrong by leveraging textual attributes of the data that are translated into a confidence score using *Semantic Binning*. First, the attributes and their possible values are defined. Then, images are grouped according to their attribute values. Each combination of attribute values, or subdomain, is assigned a confidence score. The score value is the subdomain accuracy estimated with validation data. This confidence score is then used by standard selective classification to plot accuracy-coverage curves. Additionally, for each accuracy-coverage point, we also get a list of attribute values describing the domain. I experimented with synthetic data with annotated attribute values to show the method's potential and then with real data for which attribute values have to be estimated. Experiments on real data bring out the inherent main limitation of the method: the need for massive annotated validation data. Generating synthetic validation data with text-to-image models and Textual Inversion does not produce good enough images to solve the issue.

Finally, section 5.5 discusses the chapter's findings.

5.2 Background

The Related Work chapter contains high-level information and references about diffusion models. This section includes more technical details.

Diffusion Models Diffusion models can generate images from textual descriptions. They are characterized by their ability to produce high-quality images through an iterative denoising process. The core mechanism involves a forward diffusion process that incrementally adds noise to an image until it becomes indistinguishable from Gaussian noise. The reverse process, iteratively reconstructing the image from noise, is learned during training. See Figure 5.1.

Mathematically, the forward process models the addition of noise over T timesteps as $x_t = \sqrt{\alpha_t} x_{t-1} + \sqrt{1-\alpha_t}\epsilon$, where ϵ is sampled from a standard normal distribution. α_t is the noise schedule that defines how much noise is added to the data during the forward process at each time step. The choice of noise schedule, e.g., linear or cosine, affects both the quality of the generated data and the stability of the model's training. A model ϵ_θ is trained to predict the noise added at each timestep. The model architecture is typically a

U-Net (Ronneberger et al., 2015), which is conditioned by time t, e.g., with a sinusoidal embedding. The reverse process aims to recover the original image by removing the predicted noise: $\hat{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1-\alpha_t}{\sqrt{1-\alpha_{t-1}}}\epsilon_\theta(x_t, t))$.

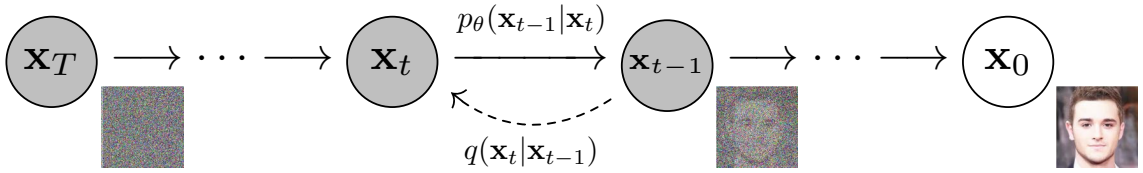


Figure 5.1: Illustration of the diffusion and reverse processes of a DDPM. From (Ho et al., 2020).

Text-to-Image Model Stable Diffusion In this chapter, the Stable Diffusion (SD) model is used. Because it is open-source, it is widely used in research. It is based on the Latent Diffusion Models architecture (Rombach et al., 2022), illustrated in Figure 5.2. The architecture's main novelty is to operate the diffusion and reverse processes in a latent space instead of the pixel space, such as in (Ho et al., 2020). As for DDPMs, the noise predictor uses the U-Net backbone. It makes the training and inference of diffusion models more efficient without degrading quality. Text, images, or semantics maps can condition the generation. Text conditioning uses embeddings that are obtained with a pre-trained text encoder (OpenCLIP-ViT/H for SD 2.1) and integrated into the model with cross-attention (Vaswani et al., 2017). Training a text-to-image model requires a dataset of image-text pairs. For SD, it is a subset of LAION-5B (Schuhmann et al., 2022), which contains more than 5 billion image-text pairs crawled from the Internet.

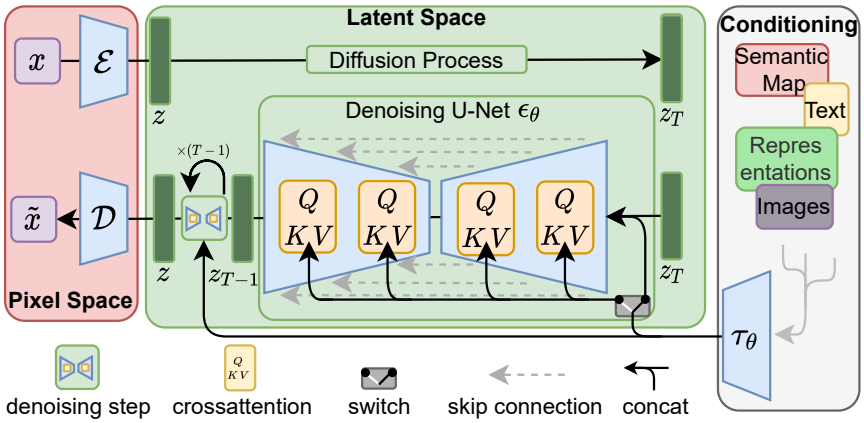


Figure 5.2: Illustration of Latent Diffusion Models, the architecture behind SD. From (Rombach et al., 2022).

5.3 Textual Descriptions of Classifier Failures Using Text-to-Image Models

5.3.1 Leveraging Text-to-Image Generative Models

Recently, there have been massive improvements in multimodal models, especially those combining textual and visual data like text-to-image generative models. These mod-

els have demonstrated an exceptional ability to understand and generate content that captures the nuanced interplay between text and images (Ramesh et al., 2022; Saharia et al., 2022). This allows new ways of benchmarking image classifiers with generative models. Classifier performance can be studied in relation to the textual attributes of the data (Wiles et al., 2022; Metzen et al., 2023; Vendrow et al., 2023). Despite their potential, however, the practical utility of these generative models is still limited by the computationally intensive inference process of the underlying diffusion models. For example, in (Wiles et al., 2022), testing whether the presence of a flower in an image causes the classifier to sometimes mistake flies for bees, requires hardware with 20×4 TPUs.

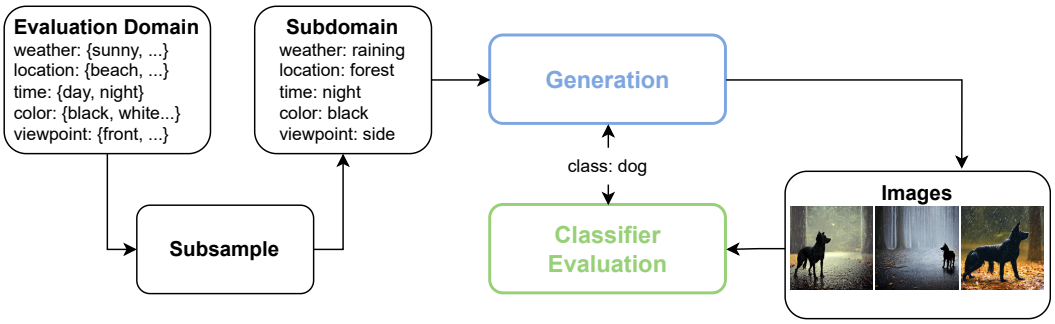
(Metzen et al., 2023) developed a classifier evaluation process that starts with a textual description of the conditions the model is likely to encounter during use, an ODD. It consists of many different combinations of attributes. To test a combination, they use synthetic data from a text-to-image model. They then identify the combinations leading to classifier errors. An example of systematic error identified is that minivans are misclassified as snowplows 30% of the time when they are small, orange, and in a forest with snowy weather. Figure 5.3a describes the approach. One of its major limitations is the combinatorial explosion. For the vehicle experiment in their paper, 18720 combinations are possible, each requiring 16 image generations. Generating one image with Stable Diffusion typically takes several seconds. Thus, there is a need to limit the number of evaluated combinations. The authors use Combinatorial Testing (CT) (Nie and Leung, 2011), and in particular n-wise testing. They test all possible interactions between a specific number of parameters (n) to significantly reduce the number of test cases while maintaining high coverage. For example, pairwise testing (2-wise) ensures that every possible pair of parameters is tested at least once. This method does not permit choosing the number of combinations to test nor ranking them by importance. Also its performance is not great: in the experiments of subsection 5.3.3, it does not select much better than random choice.

5.3.2 Bayesian Optimization to Explore Faster

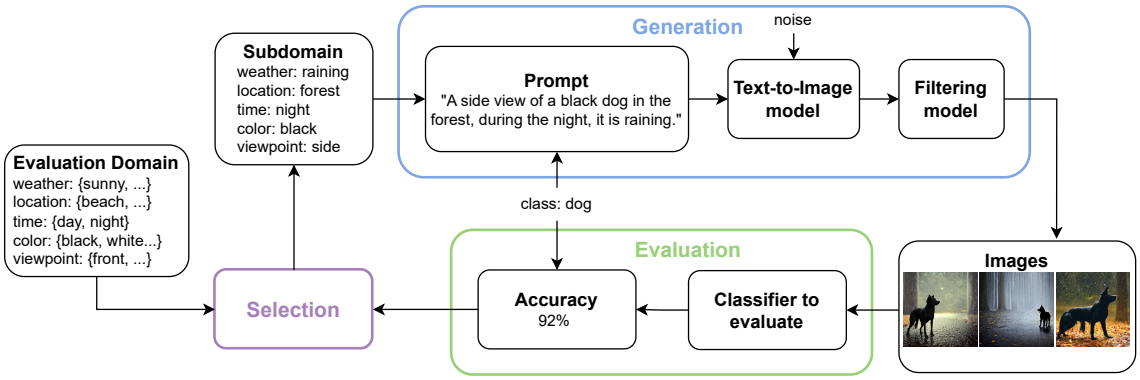
As an alternative to CT, I propose an efficient iterative process to explore the textual attributes leading to classifier failure. Below I describe the classifier studied, define the evaluation domain and subdomains, describe the general pipeline to generate images for the subdomains, and the proposed guided exploration of attributes. The process is presented schematically in Figure 5.3b and more formally in Algorithm 3.

Image Classifier To demonstrate the approach without using considerable computing power, the task considered is simple: binary classification of images containing dogs. I construct a dog classifier from a classifier pre-trained on ImageNet. Out of the 1000 classes, 119 are different dog breeds. The classifier probabilities of these classes are summed to get the *dog* probability and the rest gives the *not-dog* probability.

Define the Evaluation Domain and Subdomains An *evaluation domain* is defined as the ensemble of environmental conditions under evaluation. These conditions are described by textual attributes, each containing a finite number of values. They can be categorical or continuous, but only categorical attributes are considered in this work. The domain comprises all the possible attribute value combinations, which are called *subdo-*



(a) Original method of (Metzen et al., 2023). Some subdomains are subsampled from the evaluation domain using CT: a fixed number of subdomains that cover well the search space are selected. This selection is actually not much better than random selection (see subsection 5.3.3).



(b) Illustration of the proposed method that alternates generation, evaluation, and selection. The selection function selects the next subdomain to evaluate based on the feedback of the previous evaluated subdomains. The right choice of the selection function achieves an efficient exploration of the evaluation domain.

Figure 5.3: Comparison of the original method and the proposed one.

mains. The number of subdomains grows exponentially with the number of attributes considered.

Here is a simple example. Two attributes are considered: the `weather`, which takes two values: (sunny, cloudy), and the `location`, which takes two values: (city, forest). The domain contains four subdomains, which are all the attribute value combinations: (sunny & city), (sunny & forest), (cloudy & city), and (cloudy & forest).

As a starting point, the textual attributes to explore have to be defined. Expert knowledge about the environmental conditions likely to perturbate the classification is required. As the task is image classification of natural images of dogs, I define the following attributes and associated values in parenthetis: `weather` (sunny, cloudy, raining, snowing), `location` (at the beach, in the forest, in the city, inside a house, in a garden, in the desert, in the mountains), `time` (day, night), `color` (white, black, brown, beige, gray, red, green, blue), and `viewpoint` (front, side, rear). Invalid combinations, such as sunny `weather` during night `time`, must be removed.

Generate Data Conditioned by Attributes The process contains the following steps:

- **Prompt** The first step is to create a textual prompt describing a single subdomain. Following Metzen et al. (2023), the prompt template is: "A {`viewpoint`} view of a

Algorithm 3 Efficient Exploration of Image Classifier Failures

Input:
D_{to_eval} the evaluation domain
S_{to_eval} the list of subdomains to evaluate
f: the classifier
g: the generative model
h: the selection function
n: the number of allowed evaluations
$S_{eval} = \emptyset$ the dataset of subdomains evaluations
$s_0 \in S_{to\_eval}$: the initial selected subdomain
Explore subdomains:
for $i = 0$ **to** n **do**
  Generation
  build prompt p_i from selected subdomain s_i
  $\hat{x}_i \leftarrow g(p_i)$ ▷ generate and filter images from prompt
  Evaluation
  $\hat{y}_i \leftarrow \arg\max f(\hat{x}_i)$ ▷ compute predicted classes
  $a_i \leftarrow acc(y, \hat{y})$ ▷ compute classifier accuracy
  Selection
  $S_{eval} \leftarrow S_{eval} \cup \{(s_i, a_i)\}$ ▷ add result to dataset
  $S_{to\_eval} \leftarrow S_{to\_eval} \setminus s_i$ ▷ remove from list
  $s_{i+1} = h(S_{eval}, S_{to\_eval})$ ▷ update h and select next
end for

{`color`} dog {`location`}, during the {`time`}, it is {`weather`}.". Further prompt engineering might be required to improve generation quality, as seen in Figure 5.9.

- **Generate** A text-to-image model generates images conditioned by the textual prompt. The generation is not deterministic: the starting noisy image is random, and noise is applied to each step of the reverse diffusion process. This means that one textual conditioning leads to a variety of aligned images.

- **Filter** The generation is imperfect; sometimes, the synthetic image does not align well with the textual prompt input. A filtering process is applied to limit this issue using CLIP as a zero-shot subdomain classifier where a textual prompt defines each of the subdomains. The Cosine Similarity between a generated image and all subdomains prompts provides logits. Applying the softmax function to the logits provides the predicted probabilities that the image corresponds to one subdomain rather than another. If the prompt with the maximum probability is indeed the prompt used to generate the image, the image is considered correct; otherwise, it is filtered out.

Guided Exploration of Attributes That Matter Because generating data conditioned by the attributes described above is time-consuming, I propose an efficient exploration of the critical attributes. An iterative process alternates the generation of images for a subdomain, evaluates the classifier on the subdomain, and selects the next subdomain to evaluate based on this feedback. I propose several selection functions below.

Genetic Algorithm Genetic Algorithm (GA) is an efficient optimization method based on natural selection (Holland, 1992). Its principle is to start with a population of solutions. The top performers are preserved, and a crossover operation generates *children* solutions from pairs of *parents*. This new generation of solutions undergoes mutations with a small probability, adding diversity.

Bayesian Optimization Bayesian Optimization (BO) (Jones, 2001; Garnett, 2023) is often discussed in the context of Surrogate-Model Based Optimization (Zaefferer, 2018). The aim is to evaluate the costly objective function as few times as possible. To this end, an efficient model is used as its surrogate. BO typically relies on regression using Gaussian Processs (GPs), a computational process generally known as Kriging. Despite their ubiquity, thanks to many positive attributes, GPs have several drawbacks. The most important one is its cubic complexity of estimating the probabilities, making them inefficient as the observed data points increase. Their use is also contingent on selecting a kernel and possibly a distance function. It is however possible to effectively apply the general BO loop with alternative models, such as DNNs (Snoek et al., 2015), as well as random forests (Hutter et al., 2011) and Bayesian neural networks (Garnett, 2023). The method to efficiently explore the space of subdomains involves the same core loop at the center of BO, relying on a predictive model to guide the search towards the critical subdomains.

1. Selection: choose the next subdomain to evaluate using the model
2. Observation: evaluate the subdomain
3. Model update: add the new observation to the dataset

The selection policy generally means selecting the point which maximizes an acquisition function. Many acquisition functions exist in the literature, and each presents a different trade-off between exploration and exploitation. The selection policy is inspired by Expected Improvement (Mockus et al., 1978), a widely used and generally effective acquisition function. Using the model's estimation of each subdomain's quality, the subdomain selected is the one with the highest potential improvement over the current best subdomain.

5.3.3 Experiments and Results

Below are experiments evaluating the different aspects of the approach. I provide information on the experimental setting, compare different selection functions, and display qualitative results of classifier evaluation.

Classifier The classifier under study uses the ViT-B/16 architecture. Weights are from torchvision (maintainers and contributors, 2016), following a pre-training on the ImageNet dataset. The binary dog classifier's accuracy on ImageNet validation data is more than 99%. The goal is to assess if it can generalize well to data that is more diverse than in the original dataset.

(a) "A side view of a brown dog in the city, during the day, it is sunny."

(b) "A front view of a white dog at the beach, during the day, it is sunny."

(c) "A rear view of a black dog in the desert, during the day, it is foggy."

(d) "A front view of a red dog in the city, during the day, it is snowing."

(e) "A front view of a white dog at the beach, during the day, it is sunny."

(f) "A side view of a beige dog in a garden, during the day, it is raining."

(g) "A rear view of a blue dog in the mountains, during the night, it is foggy."

(h) A front view of a beige dog inside a house, during the day, it is sunny.

Figure 5.4: Samples of generated images with their associated prompt. Images on the top row are classified as dogs, while those at the bottom are not. Note that some biases of the generative model appear: sunglasses at the beach and an umbrella when raining.

Subdomains The number of possible attribute combinations is 1 `class` (dog) $\times$ 4 `weathers` $\times$ 7 `locations` $\times$ 2 `times` $\times$ 8 `colors` $\times$ 3 `viewpoints` = 1344. However, some of the combinations are impossible (e.g., "during the night, it is sunny" or "in a house, it is snowing"). After filtering those, 1032 combinations remain, forming all the possible subdomains to evaluate.

Generative Model Stability AI's implementation[1] of SD 2.1 is used as a text-to-image generative model. Its architecture is based on Latent Diffusion Models, and text conditioning uses a fixed pre-trained text encoder based on CLIP ViT/H. Generated images have a 512×512 resolution, but they are resized to 256×256 to save disk space and because resizing images at a lower resolution is already part of the classifier data preprocessing. This model is treated as a black box random image generator conditioned by textual input prompts.

Filtering Model Because the generation is noisy, it is necessary to filter out generated images that do not align well with the textual input prompt. It requires using a subdomain classifier that classifies generated images into one of the subdomains. This classifier is a pre-trained CLIP ViT-L/14 adapted as a zero-shot classifier.

Baselines For comparison, other methods are used as baselines for selecting the subdomains to evaluate.

[1] https://huggingface.co/stabilityai/stable-diffusion-2-1

- The *random selection* simply randomly picks a subdomain to test in the list of the remaining ones.
- The *oracle* knows all the subdomain's accuracies in advance, and it chooses the subdomains by order of increasing accuracy. This is the best way to select the subdomains, but also the most costly as it requires knowing all the subdomain performances.
- *CT* aims to test a limited number of combinations that cover well the search space. In particular, I use n-wise testing from the library `allpairspy` (allpairspy). The variable n varies from 2 (pairwise testing) to 5 (because there are 5 attributes). This approach was used by (Metzen et al., 2023).

Method Details Two different approaches are tested:

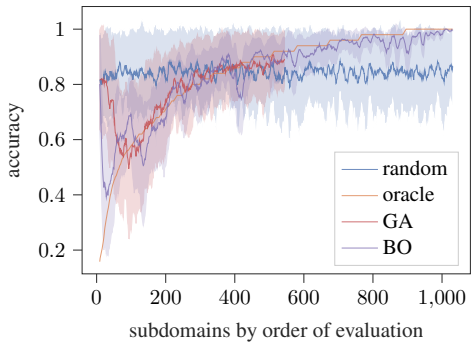
- *GA* The population size is 20 and the library `pymoo` is used (Blank and Deb, 2020).
- *BO* The predictor takes a one-hot embedding of the subdomain attributes as input to predict the accuracy. Random Forest Regressors (Breiman, 2001), Lasso (Tibshirani, 1996), Linear Regression, and Support Vector Regression (SVR) (Drucker et al., 1996) are tested using `scikit-learn` (Buitinck et al., 2013). The methods pre-train on 10 random subdomains to lower the variability of each run.

Metrics The metrics are: selected subdomain accuracies, average accuracy of selected subdomains, and coverage of the 10% lowest accuracies subdomains. The Spearman rank correlation evaluates the quality of the predictors.

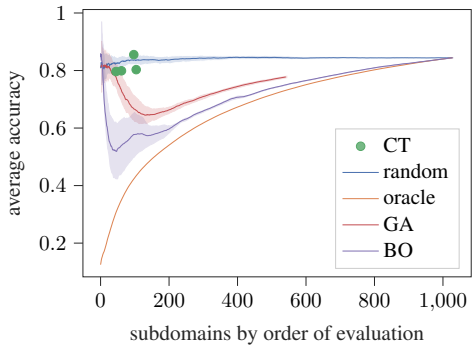
Evaluating All Subdomains for Reference To validate the approach, the performance results of all subdomains are saved as shown in Table 5.1. Because all evaluation results are pre-computed, benchmarking the different selection functions is done by replacing the generation and evaluation parts with a simple table look-up. This allows for comparing different selection functions quickly and removes the variability originating from generating different images each time. 50 valid images were generated for each of the 1032 subdomains. It took approximately 200 hours to generate all images on one NVIDIA V100 GPU. Sometimes, hundreds of images had to be generated to obtain 50 valid ones after filtering. The expected evaluation time of one subdomain is 12 minutes, or 1 hour for 5 subdomains.

Table 5.1: Reference evaluation data. The generation and evaluation steps are precomputed, and the results are saved in a table. A table look-up replaces these costly steps to compare different selection functions quickly.

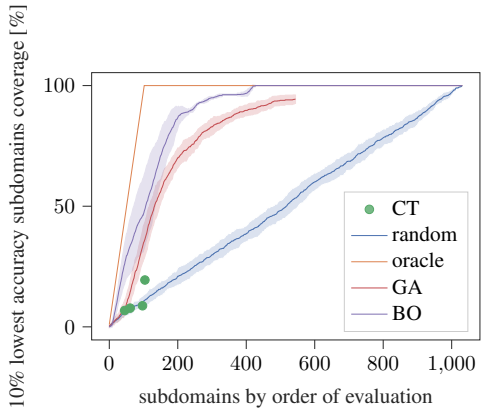
Subdomain #	Viewpoint	Color	Time	Location	Weather	Classif. acc.
0	side	white	day	at the beach	sunny	0.98
1	side	white	day	at the beach	snowing	0.94
2	side	white	day	at the beach	raining	0.86
...	...	...	...	...	...	...
1031	rear	blue	night	in the mountains	foggy	0.66



(a) Evolution of the accuracy of selected subdomains during the exploration (lower is better). A moving average with a window size of 10 was used to improve clarity. GA and the BO quickly select low-accuracy subdomains until only higher-accuracy subdomains remain.



(b) Evolution of the average accuracy on subdomains already evaluated during the exploration (lower is better). All methods converge to the global accuracy. Combinatorial testing is not much better than random selection, compared to the GA and BO.



(c) Evolution of the 10% lowest accuracy subdomains coverage (higher is better). The 10% (103) subdomains with the lowest accuracies are identified, and the proportion covered by the subdomains selected during the exploration is computed. The BO finds all of them after evaluating ≈ 300.

(d) Spearman's rank correlation coefficient for different predictors and training set sizes. For small training sizes, Lasso is the best by a small margin while SVR is the worst. Lasso is chosen as the default predictor because small training sizes are the most important to explore the subdomains with limited time.

Figure 5.5: Different metrics to compare the quality of the subdomain selection when iterating on the loop generation, evaluation, and selection. In general, combinatorial testing is not much better than random selection, and it only gives a few options for the number of subdomains selected. GA and BO are much more efficient and can explore any given number of subdomains according to the computation time available. Note that the x-axis of 5.5a, 5.5b, and 5.5c could be replaced by GPU.hours going from 0 to ≈ 200. All plots are averaged over 10 seeds and the standard deviations are shown.

Figure 5.4 shows samples of generative images with their input prompt. While not perfect depictions of dogs, they are close enough to benchmark the classifier. Some images clearly depict dogs, yet the classifier fails to identify them. This highlights some of its limits: for instance, the presence of sunglasses or a small umbrella piece makes the classifier fail.

Benchmarking the Selection Functions The main goal of selection functions is to quickly identify subdomains with low accuracy. To measure this, Figure 5.5 shows the evolution of different metrics during the exploration . The main conclusion is that combinatorial testing (n-wise testing with $n \in \{2, 3, 4, 5\}$) is not much better than random selection. Also, it has the disadvantage of restricting the number of selected subdomains: this number cannot be tuned. GA is much better, and BO is even better. BO can successfully identify all the 10% most critical subdomains (with lowest accuracies) after evaluating $\approx 40\%$ of all subdomains. This also proves that subdomain performance can be precisely inferred from the domain attributes. This means that classifier failures can be explained from the attributes, providing interesting insights into the classifier decision process.

Figure 5.5d compares the Spearman's rank correlation coefficient for different predictors. It measures the strength and direction of the monotonic relationship between two ranked variables, here the predicted and test accuracies. A value close to 1 means the relationship between the two variables is monotonic. Lasso is chosen as the default predictor for BO because it is the best method for small training set sizes. Indeed, the beginning of the exploration, when the data is limited, is particularly important. Furthermore, it showed less variability than, for example, random forests.

Figure 5.6 details subdomain accuracies for a specific step in the evaluation process: when the number of evaluated subdomains equals 61 (the number of subdomains selected by 3-wise testing). This also shows a clear advantage for GA and BO in quickly identifying low-accuracy subdomains.

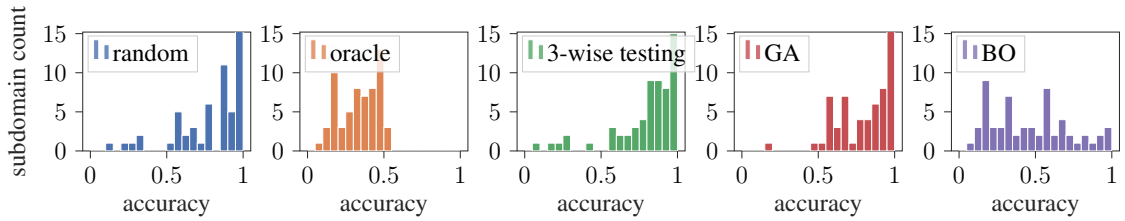


Figure 5.6: 3-wise testing selects 61 subdomains to evaluate. Most of them are high-accuracy. It can be compared to the other methods when allowed to explore 61 subdomains. GA and BO identify much more low-accuracy subdomains.

Qualitative Analysis of Classifier Failures The main focus of this work is to efficiently detect the attributes with the most impact on classification performance. This can be used to perform qualitative assessments of the classifier's behavior when considering different attributes. The BO approach is used to explore 300 subdomains. Figure 5.7 shows the average accuracies for each attribute value. This shows the impact of each individual attribute. Figure 5.8 displays the impact of all the possible combinations of the attributes characterizing `weather` and `location`.

Summary of the Results Performing BO with Lasso as a selection function allows for a much more efficient exploration of subdomains leading to classifier failures. For instance the 100 subdomains with the lowest accuracy are all identified after exploring 400 out of the 1032 subdomains. After only a few subdomains are evaluated, the selection function is able to efficiently guide the exploration process.

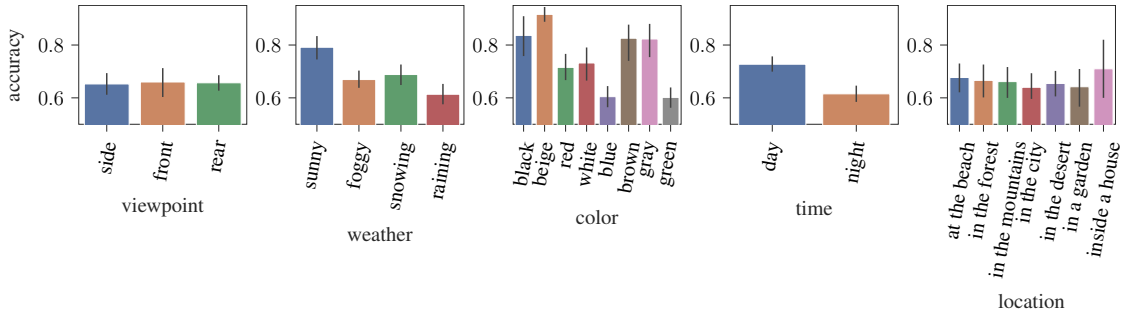


Figure 5.7: Average accuracies for each value of each attribute. The 95% confidence interval is also shown.

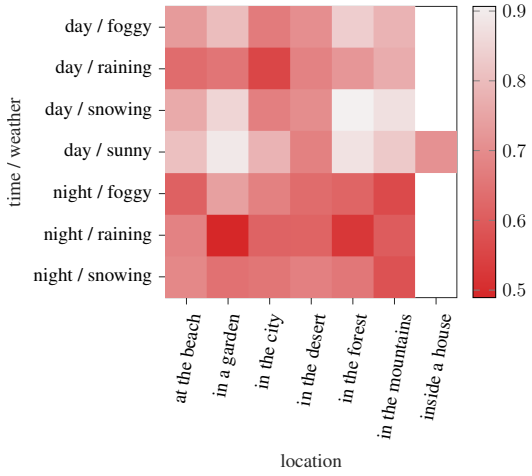


Figure 5.8: Heatplot displaying the average accuracies for different attribute values.

Limitations Benchmarking classifiers with generative models has limitations, as observed by other work (Wiles et al., 2022; Metzen et al., 2023). There can be occasional misalignments between the prompt and the image due to bias or language limitations. For instance, in this work, the `viewpoint` attribute is sometimes not the one requested in the text prompt. Generator failures also happen for a few specific subdomains, e.g., nearly all images for "A front view of a green dog in the mountains, during the night, it is raining." are in a cartoon style, which is not the case for snowing, see Figure 5.9. Prompt engineering is required to allow a rigorous benchmark of the classifier. Also, generated images do not cover everything possible in the real world. The approach tackles the computing time problem. Its main limitation is that there is no guarantee that a good selection function will identify *all* problematic subdomains for an incomplete exploration. For instance, a subdomain might be difficult for completely different reasons than the others. Thus, a selection based on learning a relation between subdomain attributes and performance might miss it.

Figure 5.9: In the top row, images are generated with the prompt "A front view of a green dog in the mountains, during the night, it is raining.". They are mostly in a cartoon style. In the bottom row, the same prompt, but "raining" has been replaced by "snowing". The phenomenon disappears. Is this a generator failure? Careful prompt engineering, e.g., adding "a realistic image", is required to ensure alignment between the textual prompt, the generated images, and the expected images.

5.4 Using Textual Attributes to Define a Reliability Domain

5.4.1 Semantic Binning for Semantic Selective Classification

In the previous section, we saw a method to efficiently find attributes likely leading to failure and establish a link between data description and classifier accuracy. In this section, I associate these concepts with selective classification to instead study when the classifier is reliable. The standard way to do selective classification is to keep only data for which a confidence score is above a given threshold. Here, the idea is to use data description to compute a confidence score used for selective classification.

Semantically Described Domain and Subdomains As in section 5.3, textual attributes describe the images (e.g., `background` and `size`). These textual attributes have a finite number of possible values (e.g., cloudy or sunny for the `weather`; small or big for the `size`). Each image is described by a combination of attribute values (cloudy `weather` and small `size`). One combination of attribute values is called a *subdomain*, and each subdomain contains several images. Semantic Selective Classification (SSC) selects many subdomains for which the classifications are likely to be correct to form a *domain*. The method requires attribute values for each image. They can come from annotations if the method is used to understand the classifier's strengths and weaknesses. If the method is used at test time, attribute values have to be estimated, e.g., by another model like CLIP. Besides using high-level image features as attributes, I also use the classifier's predicted class as an attribute. Indeed, this characteristic can easily be estimated (with the classifier) and is related to a misclassification likelihood.

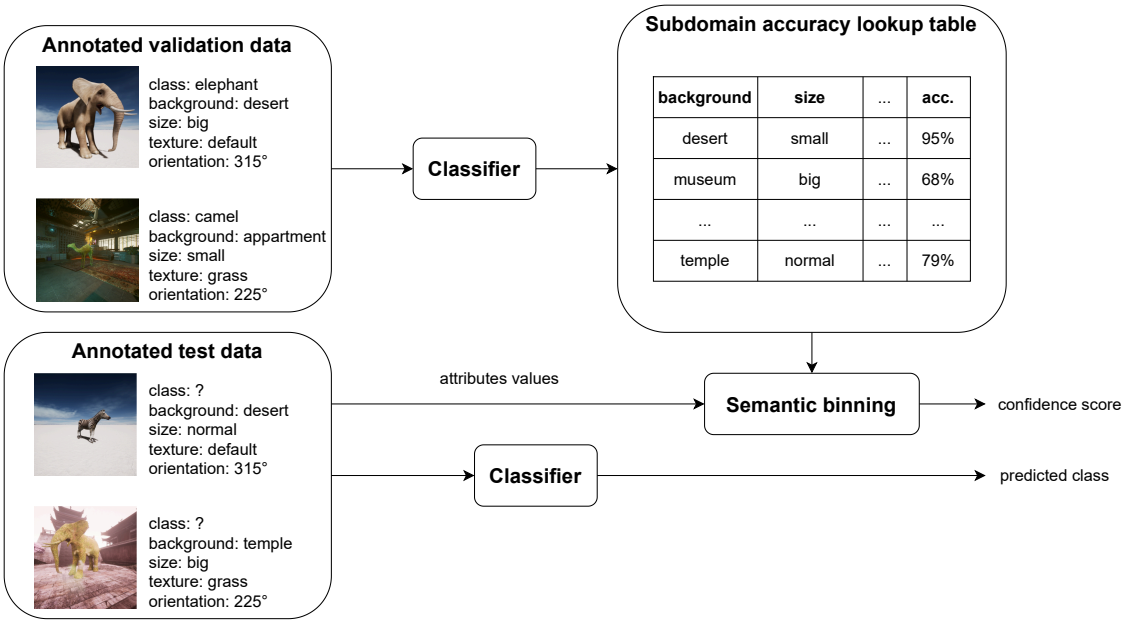


Figure 5.10: Semantic Selective Classification (SSC). Validation data annotated with attribute values is used to create a lookup table of subdomain accuracies. At test time, images are assigned to their subdomain according to the attribute values, and the confidence score is set as the subdomain accuracy, a process called semantic binning, which resembles histogram binning, a calibration method.

Semantic Binning Here is how we can obtain a confidence score from data descriptions. Validation data is first split into different subdomains according to the attribute values. Then, the classification accuracy is computed for each subdomain. The confidence score for a subdomain is set as the subdomain accuracy. At test time, data is assigned to their respective subdomain according to the attribute values, and their confidence score is set accordingly, using previously computed values.

I call this method *Semantic Binning (SB)* due to its similarity with the histogram binning calibration method (mentioned in section 4.4). Histogram binning divides calibration data into bins according to their confidence values. Then, for each bin, the classifier accuracy is computed. The calibrated probability is set as the bin accuracy. At test time, data is assigned to bins according to the confidence values, and their calibrated probability is set as the bin accuracy computed with calibration data.

It can happen that at test time, we get an image belonging to a subdomain whose accuracy was not estimated with validation data, e.g., because we have a fine description that generates a large number of subdomains, some of which are not covered by the validation data. In that case, a *fallback* process is needed to estimate the accuracy. Coarser subdomains are thus considered by dropping some of the attributes. For instance, if a specific combination of a `background` and a `size` was not seen on the validation data, the `size` attribute is dropped. The estimated accuracy is the average accuracy for this specific `background`, regardless of the `size`.

SB computes confidence scores used by a selection function as for standard Selective Classification. The selection function uses a threshold on those confidence values to determine whether each image belongs to the domain. I call Semantic Selective Classification the method of applying SC to confidence scores computed with SB.

Estimating the Attribute Values Below, I experiment with a dataset containing attribute values annotations to validate the approach. However, in practice, such values do not exist. A way to estimate them is by using a general foundation model like CLIP (Radford et al., 2021). CLIP is used as a zero-shot attribute value classifier by using text templates, e.g., "the background is a {forest, desert...}". The attribute value leading to the lowest Cosine Similarity between the text embedding and the image embedding is CLIP's prediction. See Figure 5.11 for an illustration.

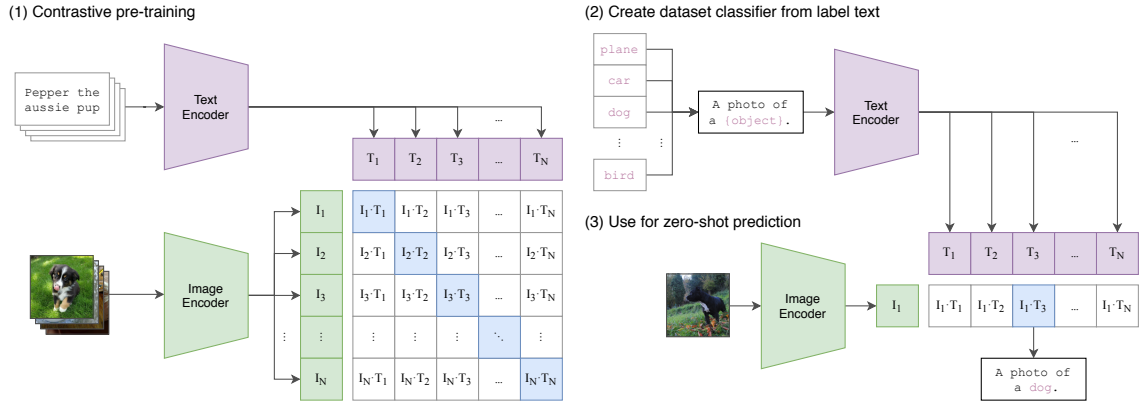


Figure 5.11: CLIP training and use as a zero-shot classifier. From (Radford et al., 2021).

Getting Enough Validation Data per Subdomain Another requirement of the method is to have sufficient validation data to estimate the accuracy for each subdomain reliably. For instance, we would like to have images of birds in uncommon contexts. The idea is to use synthetic data to augment the validation set. In particular, Textual Inversion (TI) (Gal et al., 2023) allows to personalize image generation of text-to-image models. From a few examples of a "concept," e.g., a specific bird, the method finds the "textual embedding" corresponding to the concept, i.e., the embedding leading to generating images close to the few examples. This learned embedding can then be combined into sentences to generate the concept in different contexts. See Figure 5.12 for an illustration.

Figure 5.12: Illustation of Textual Inversion. An optimization process finds the embedding of a pseudo-word S_* that matches the concept present in the input samples. The pseudo-word can then be used in sentences to generate the concept in different contexts or styles. From (Gal et al., 2023).

5.4.2 Experiments with Synthetic Data

Weak Classifier on PUG Animals Dataset The PUG Animals dataset (Bordes et al., 2024) is a photorealistic synthetic image dataset with annotated factors of variations, which are called attributes here. Images are produced with Unreal Engine, a game engine that produces photorealistic environments. The dataset contains 215040 images using 70 animal assets, 64 `backgrounds`, 3 object `sizes`, and 4 `textures` under 4 different camera `orientations`. All combinations of attributes are covered by the images with one image for each combination. I use 50% of the data for training, 25% for validation, and 25% for test.

The classifier under study is based on the ViT-16-B architecture and pre-trained on ImageNet. The classifier head (from features to logits) is fine-tuned on the training set and reaches an accuracy of 73% on the test set. This is, for the purpose of validating the approach, a weak classifier because fine-tuning the whole classifier results in a classifier that is rarely incorrect (accuracy above 99%). For almost perfect classifiers, selective classification is not useful.

As shown in Figure 5.13, some attribute values impact the classifier's accuracy. Values for `orientation` and `texture` attributes do not have a significant impact, but values of `background`, `class`, and `size` matter more.

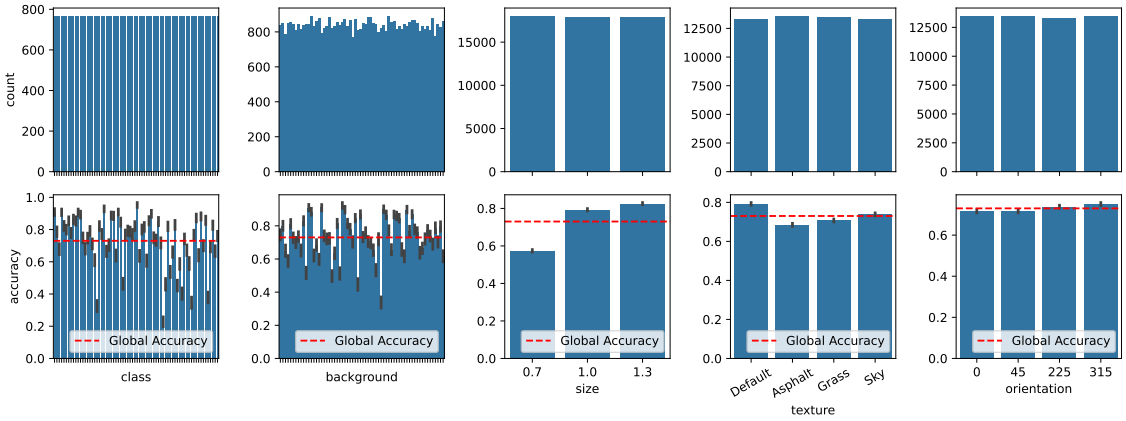


Figure 5.13: Counts and average accuracies when test data is grouped by class and attribute values. For a weak classifier and PUG data.

As a first experiment, very basic SSC can be done by only considering a single attribute. In that case, the number of subdomains is equal to the number of possible attribute values, which is very small. Let us use the object `size` attribute as an example. Three subdomains are possible: images with small, normal, and big `size`. SB assigns a confidence score for each one: high confidence for big `size` (which has high accuracy), and lower for small `size` (which has lower accuracy). SSC defines a domain as all data for which the confidence score is lower than the threshold. For decreasing threshold values, the domains are: only big `sizes`; big and normal `sizes`; big, normal, and small `sizes`. They correspond to the three points in the accuracy-coverage curve of Figure 5.14. The figure also compares different attribute choices. The best attribute is the `predicted class`: it has the most impact on the classifier's correctness. The `background` is also important. This is aligned with the histograms of Figure 5.13, whose variability is linked to the attribute's impact on the classifier's correctness. For a single number measure, the AUROC, a standard metric for selective classification, is shown in Table 5.2, which confirms the ranking

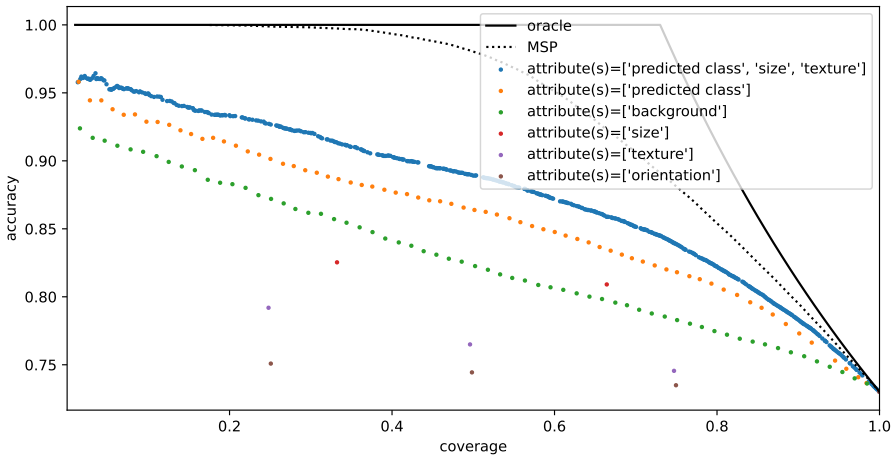


Figure 5.14: Accuracy-coverage curves. Using single attributes and the best attributes combination vs. the baseline (MSP).

of the attributes: `predicted class`, `background`, `size`, `texture`, and `orientation`.

Looking at single attributes is very limited, so now let us combine several attributes. Table 5.2 shows results for different combinations of attributes. The total number of subdomains is the product of all possible attribute values and grows quickly. When the number of subdomains is high, estimating subdomain accuracy with validation data becomes an issue. Because data is split into more subdomains, each subdomain contains less data, and its accuracy is thus not well estimated. I fixed the minimal number of validation examples in a subdomain to 3 in order to avoid extremely bad accuracy estimations. This value is very small but representative of the usually small number of examples per subdomain. There are now many subdomains with bad accuracy estimates on validation data, and many subdomains are not even covered. For these subdomains, test examples are assigned a confidence score using the fallback process described in subsection 5.4.1. According to Table 5.2, the best attribute combination is [`predicted class`, `size`, `texture`]. Figure 5.14 shows its accuracy-coverage curve. Compared to just using the `predicted class`, the additional attributes add information to predict classifier failures without creating too many subdomains (840). Adding the `orientation` degrades the performance because this attribute is not by itself a good predictor of failure and also because it multiplies the number of subdomains. For this combination of attributes, the selected subdomains for a selective accuracy of 95% are listed in Table 5.3.

To conclude, SSC does not select data as well as the MSP but provides semantic information on the data "in-domain" vs. "out-domain." Because of limited data, there is a compromise between the description preciseness and the selective classification performance: too fine descriptions lead to many subdomains for which the accuracy is poorly estimated and does not generalize to test data. Also, the problem studied here is artificial: the classifier is weak, so SSC is useful for detecting its limits that are linked to semantic attributes. Fine-tuning all the classifier's weights results in an almost perfect classifier. The purpose of selective classification is to improve accuracy by trading off coverage, but if the global accuracy is already $> 99\%$, then selective classification has no benefit.

Table 5.2: Comparing different choices of attributes to define subdomains for PUG data. The total number of subdomains is the total number of attribute value combinations. The number of validation subdomains is the number of subdomains containing at least 3 validation examples (required to estimate the subdomain accuracy). The number of fallbacks is the number of times a test example does not belong to any of the validation subdomains and whose confidence score computation has to follow the fallback process. AUROC measures the quality of the selective classification using a given confidence score.
Note that the baseline AUROC when using the classifier MSP as a confidence score is 0.913, and for random selection, the AUROC is 0.5.

Attributes considered	**# subdom. (valid/total)**	**% fallbacks**	**AUROC (↑)**
predicted class	70 / 70	0%	0.742
background	64 / 64	0%	0.668
size	3 / 3	0%	0.642
texture	4 / 4	0%	0.556
orientation	4 / 4	0%	0.520
predicted class, background	4467 / 4480	1.54%	0.781
predicted class, background, size	9978 / 13440	24.3%	0.774
predicted class, background, size, texture	3835 / 53760	91.9%	0.751
predicted class, background, size, texture, orientation	557 / 215040	98.7%	0.745
background, size	192 / 192	0%	0.730
background, size, texture	768 / 768	0%	0.736
background, size, texture, orientation	3072 / 3072	0%	0.726
predicted class, size	210 / 210	0%	0.779
predicted class, size, texture	840 / 840	0%	**0.787**
predicted class, size, texture, orientation	3357 / 3360	0.04%	0.771

Table 5.3: List of 81 subdomains out of the 840 defined by using the combination of the attributes: `predicted class`, `size`, and `texture`. These subdomains are those for which the confidence threshold corresponds to 95% selective accuracy and 9% coverage (4972/53760 test examples). Some subdomains are grouped in one line to save space.

Predicted class	**Size(s)**	**Texture(s)**
Ammonite	Normal	Default
Ammonite	Big	Default; Grass
Ant	Big	Default
Armadillo	Normal	Default; Asphalt; Grass; Sky
Armadillo	Big	Default; Sky
Bear	Normal; Big	Default
BlackRockFish	Small; Normal	Default
Camel	Small	Default; Sky
Camel	Normal; Big	Default
Capybara	Small	Asphalt
Capybara	Normal	Asphalt; Grass
Capybara	Big	Default
Caribou	Normal	Default
Cat	Normal; Big	Default
Cattle	Small	Sky
Dolphin	Big	Default
EarlessSeal	Normal	Default
Elephant	Small	Default; Asphalt; Grass; Sky
GiantAnteater	Normal	Default; Grass; Sky
GiantAnteater	Big	Default
GiantTortoise	Normal	Sky
GiantTortoise	Big	Default; Grass

Predicted class	**Size(s)**	**Texture(s)**
Goat	Big	Default
GoldBeetle	Normal	Default; Asphalt; Sky
GoldBeetle	Big	Default; Asphalt; Sky
Hippopotamus	Normal; Big	Default
Horse	Normal; Big	Default
Impala	Normal	Default
Koi	Small	Asphalt
Koi	Normal	Asphalt; Grass; Sky
Koi	Big	Default; Asphalt; Grass
Lion	Small; Normal	Default
Parasaurolophus	Normal	Default
Parasaurolophus	Big	Grass
Penguin	Normal	Grass; Sky
Penguin	Big	Default; Grass
PoisonDartFrog	Big	Default
Scorpion	Big	Default; Sky
Tapir	Normal; Big	Default
Triceratops	Small	Default; Grass; Sky
Triceratops	Normal	Default; Sky
Triceraptos	Big	Default
Turtle	Normal	Sky
Turtle	Big	Default; Grass; Sky

5.4.3 Experiments with Real Data

Estimating Attribute Values for Real Image Datasets The experiments above use synthetic data for which attribute values are annotated. However, in practice, such annotations are unlikely to exist. This paragraph considers a more realistic setting. The task is fine-grained image classification of bird species with the dataset CUB-200-2011 (Wah et al., 2011). It contains 11788 images of 200 bird species, with approximately 60 images per class. The dataset is split into a training set of size 5994 and a testing set of size 5794. I use the training set as a validation set. Supposing that CUB data was not seen during training, CLIP is used as a zero-shot classifier, with a test accuracy of 63%. It is also used as an attribute value predictor for the `background` (sky, water, ground, plants, and branches) and bird `position` (flying, swimming, and still), as seen in Figure 5.15. To limit the impact of wrong prediction, I use a threshold on the predicted probability: when the value is under 0.4 for the `background` or 0.7 for the `position`, the prediction is set as "unknown". These thresholds maximize the AUROC on validation data when considering each attribute for subdomain definition. Figure 5.16 shows that the attribute `background` and `position` have a moderate impact on the accuracy. Contrary to PUG data, the attributes' distribution is not balanced, e.g., there are few images of flying birds.

Table 5.4 and Figure 5.17 both show similar results. The attributes `background` and `position` have a low selective classification performance: they do not correlate well with classification correctness. Grouping these attributes is not much better. However, the attribute `predicted class` performs well and even outperforms MSP as measured by AUROC and selective accuracy for coverage $> 30\%$. Combining the `predicted class` with other attributes degrades the performance.

To conclude for this dataset, SSC based on the `background` and `position` attributes values is not performing well at selecting data likely to be well-classified, as shown by the accuracy-coverage curves of Figure 5.17 and AUROC in Table 5.4. Two reasons can explain this: the attribute prediction is not reliable, and the attributes are not correlated

Figure 5.15: Samples of CUB data, with zero-shot predictions from CLIP for the `background` and `position` attributes, with the predicted probability in parenthesis.

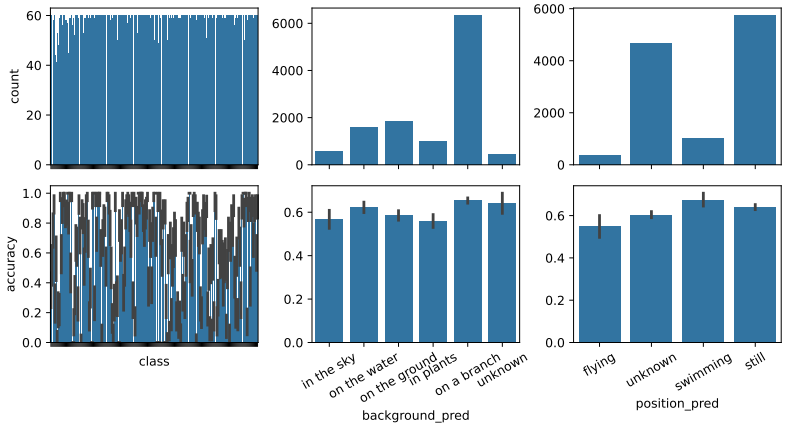


Figure 5.16: Counts and accuracies when test data is grouped by class and attribute values. CLIP is applied as a zero-shot classifier and attribute predictor on CUB data.

Table 5.4: Comparison of different choices of attributes to define subdomains for CUB data. The total number of subdomains is the total number of attribute value combinations. The number of validation subdomains is the number of subdomains containing at least 3 validation examples (required to estimate the subdomain accuracy). The number of fallbacks is the number of times a test example does not belong to any of the validation subdomains and whose confidence score computation has to follow the fallback process. AUROC measures the quality of the selective classification using a given confidence score. Note that the baseline AUROC when using the classifier MSP as a confidence score is 0.810, and for random selection, the AUROC is 0.5.

Attributes considered	**# subdom. (valid/total)**	**% fallbacks**	**AUROC (↑)**
predicted class	192 / 200	0.22%	**0.833**
predicted background	6 / 6	0%	0.535
predicted position	4 / 4	0%	0.527
predicted background, predicted position	23 / 24	0%	0.551
predicted class, predicted position	371 / 800	4.19%	0.829
predicted class, predicted background	423 / 1200	7.58%	0.831
predicted class, predicted position, predicted background	545 / 4800	16.1%	0.829

with classifier correctness. By looking at samples, e.g., in Figure 5.15, we see that the attribute predictions are not perfect but mostly correct. The second reason is thus more likely to explain the low performance: these attributes just do not really correlate with the classifier performance. However, filtering on the `predicted class` outperforms MSP. For this dataset, predictions are mostly wrong for some predicted classes and the classifier should abstain for these classes. Thus, selective classification can be improved when only considering the `predicted class` attribute. As in the previous case, the weak classifier could be improved with few-shot learning approaches.

Augmenting Real Images with Synthetic Data In the previous experiments, we have seen that SSC requires validation data covering each combination of attributes in order to estimate subdomain accuracies reliably. In practice, such data is either unavailable or could be used for other important purposes, such as fine-tuning. To solve this issue, an appealing idea is to use synthetic data. Recent techniques allow personalizing image generation using text and images. In particular, with Textual Inversion applied to text-to-

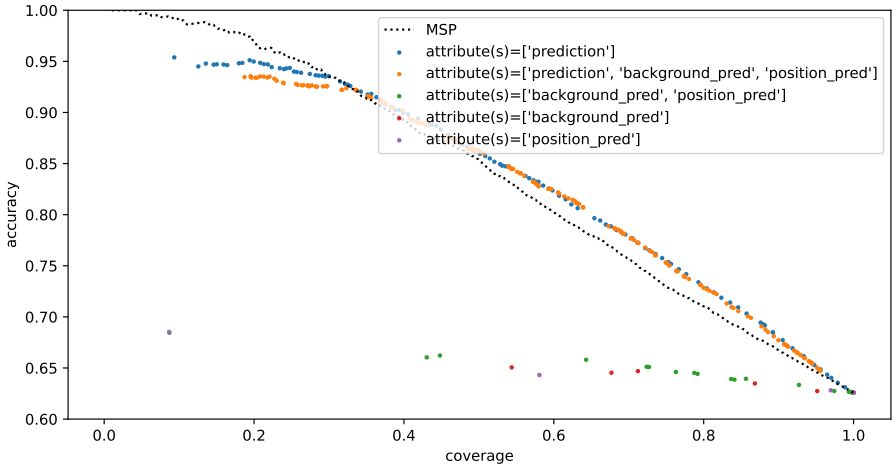


Figure 5.17: Accuracy-coverage curves. Using single attributes and attributes combinations vs. the baseline (MSP). Attributes `background` and `position` are only slightly discriminative for classification correctness, even when combined. Also, they do not improve the performance over only using `predicted class` as attribute.

image models, we can generate variations of existing images using textual descriptions. The approach here requires the generation of CUB-style birds in different contexts defined by the attributes `background` and `position`.

The generative model used is Stable Diffusion 2.1. Code for the model and TI come from the `diffusers` library (von Platen et al., 2022).10 sample images per class are used for TI, and the initializer token is "bird". The scheduler for generating images is DDIM (Song et al., 2021a) with 50 inference steps.

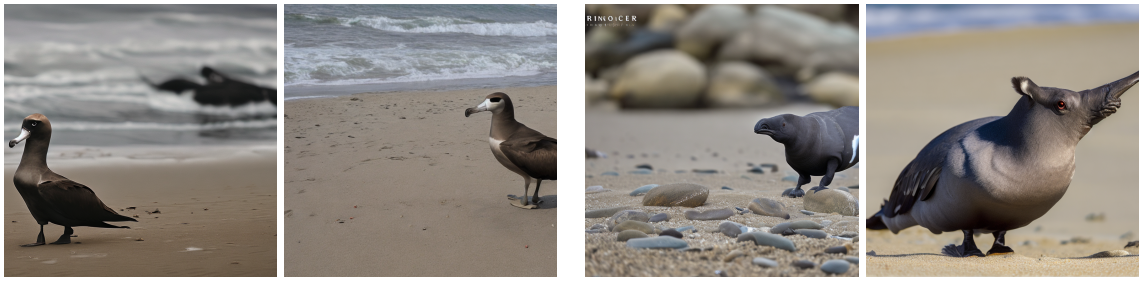
Figure 5.18 shows qualitative samples highlighting the limit of the approach. Using SD only conditioned by text is not good enough, in part because the image style is not the same as CUB images. For instance, many synthetic images show a high level of blur in the background, unlike real images. Also, some specific class names are not well known by the generator. For instance, prompting `rhinoceros auklet` images produces a weird combination of a rhinoceros and a bird. This is what TI aims to solve: learning new concepts from a few examples. Unfortunately, it does not work well enough: birds are not close enough to the original species, and the generator struggles to consider additional conditions (e.g., "on the beach").

To conclude, using synthetic images does not solve the issue of too few data per subdomain simply because the images are not good enough. Neither the correct class nor the context are reliably generated. Future generative models and better inversion techniques would likely overcome these issues, but SD 2.1 and Textual Inversion, both released in 2022, are not good or reliable enough for this use case.

Analysis Experiments on synthetic data with a weak classifier demonstrate promising results for SSC. Combining the attributes of `predicted class`, `size`, and `texture` to select data provides insights on how combinations of attributes impact classifier accuracy.

However, on real data there are the difficulties of annotating the images with attributes and having validation data in large quantity. In the experiments, the attributes of `background` and `position` do not have a big impact on classifier accuracy. Another difficulty is thus the selection of important attributes.

(a) Some image samples used for Textual Inversion.



(b) Synthetic images generated from the textual prompt: "a high quality photo of a [class_name] on the beach".

(c) Synthetic images generated after textual inversion from the textual prompt: "a high quality photo of a [placeholder_token] on the beach". The placeholder token is a string for which the embedding was learned by TI.

Figure 5.18: Comparison between real images and synthetic images, without and with Textual Inversion. Two classes are shown: `black footed albatross` on the left side and `rhinoceros auklet` on the right. Without TI, the image context (beach) is respected, but the birds are not well generated, especially the `rhinoceros auklet`, whose name seems confusing to the generator. TI aims to align the generated birds with the provided samples but fails to do as well; it also struggles to respect the text prompt.

5.5 Discussion

Text-to-image models have great potential as useful tools for benchmarking image classifiers by generating images of failure cases. However, since the highest-quality generators are based on diffusion models, their high inference time prevents large-scale image synthesis for advanced evaluation. Section 5.3 starts from an evaluation domain described by textual attributes. To efficiently explore the critical attribute combinations that cause classifier failures, I propose to create an iterative process that alternates image generation, classifier evaluation, and attribute selection. I compared different selection functions and showed that they all outperformed the method used in previous work. I believe that the approach can be further improved by using Neural Architecture Search methods, taking advantage of low-fidelity evaluations. For example, the accuracy could be estimated with

20 images. The method would then use these low-fidelity evaluations to decide which combination is worth testing with high-fidelity, say 200 images. In addition, for more complex problems, one can use word embeddings from language models instead of one-hot embeddings of finite attributes. The work of section 5.3 can potentially improve the benchmarking of image classifiers with text-to-image models, as it addresses a major limitation: computational time. It allows the exploration of larger domains and more precise estimates of accuracies and class probabilities.

Instead of looking at a classifier's failures, we can look at when it is reliable. This is the topic of section 5.4. I have experimented with the idea of selecting data according to attribute values to add a semantic dimension to selective classification. Results for the synthetic PUG dataset are promising. However, for real images, such as the CUB dataset, one has to find the attributes that matter and reliably estimate their values. Using synthetic images to augment real images did not work because the generated images were insufficient in quality.

Selective classification originally tackles filtering out samples likely to be misclassified among in-distribution data only. In this setting, the classifier is usually mostly robust to the different attribute values, as those values have been seen during training. Because different attribute values do not directly cause errors, SSC does not perform a good selection of data that is likely to be misclassified. SSC makes more sense when there is a form of distribution shift and when the considered attributes influence the classifier's correctness. This is the first limitation of SSC. Estimating the attribute values can also be a problem. Large multimodal models can be used to predict values for general attributes, but might introduce errors. Another limitation is the need for a certain amount of validation data for each subdomain to estimate accuracy reliably. This limits the number of attributes that can be considered. For each new attribute, the number of subdomains is multiplied by the number of attribute values, quickly leading to a combinatorial explosion. Subdomains can also be so precise that they only contain a few to no data. I explored using synthetic data to augment the validation data, but the generated data is unrealistic and does not align with the target attributes. Another issue is the determination of which attributes to consider: some do not impact the accuracy. Works on discovering failures might be a solution.

These limitations prevent Semantic Selective Classification from being generally effective for image classification. However, it should be interesting in particular settings like a classifier under covariate shift, where large multimodal models can accurately predict general attributes but not the class. The idea should benefit from advances in generative AI. An interesting perspective is to study SSC applied to tabular data, where attribute values could simply be the values of the input samples.

Chapter 6

Conclusion / Discussion

6.1 Is the Initial Goal Solved?

The contributions of this thesis are diverse and were divided into three chapters (3, 4, and 5). Each chapter has its set of questions, tools, and uses. Let us now go back to the original goal defined in section 1.4. Did the thesis work successfully solve the questions (Q) asked and demonstrate the targeted uses (U)?

Q1 How can a domain be expressed? Are there boundaries? A domain can be expressed in several ways, but it depends on the definition. In many works on OOD detection, the domain is defined by the examples whose class was seen during training. Any data of a different class should be detected as it is considered out of the domain. Many OOD detection methods are a form of a binary classifier, which expresses the domain. In DA and DG, the domain is defined by a specific dataset with specific characteristics. In this thesis, the domain is defined as the set of data for which predictions are reliable. In sections 3.3 and 3.4, the classifier is coupled with a GAN to generate examples at the limit of the domain. These synthetic examples help identify the boundaries but do not provide a clear way to express the domain, even when using the latent space as discussed in 3.5.

The notion of reliability domain is more aligned with selective classification. Standard selective classification considers in-distribution data only. This is a reasonable hypothesis for controlled production settings. The domain is expressed by the selection function, which aims to predict whether the prediction is likely to be correct. In-domain selective accuracy can be estimated and is higher than the global accuracy, as difficult data is filtered out. The selection function usually uses a threshold on a confidence score, so the domain size and performance can be controlled. Boundaries correspond to all data for which the confidence score value equals the threshold. Section 4.3 showed that simply using the classifier MSP as a confidence score provides great results, not easily improved by more complex methods. It means that a well-trained classifier is well aware of its uncertainty for in-distribution data. Selective classification is a straightforward way to express a domain.

However, it has two limitations. The first one is that selective accuracy is a global metric in the domain and does not give additional information on individual examples. This is what calibration aims to do: provide calibrated confidence scores for each example. A calibrated confidence score not only ranks data for selective classification but also quantifies the uncertainty. Instead of the binary view of selective classification, calibrated confidence scores provide better local insights by quantifying the uncertainty for each example. In this thesis, two new approaches to improve confidence calibration were

developed: use a surrogate binary classifier in section 4.4 and use synthetic data in section 4.5. However, how calibration can improve the domain expression remains unclear. The second limitation is that neither selective classification nor calibration provides semantic insights into the domain. The domain is expressed by a set of numbers describing the data. Section 5.4 describes SSC, a new way to incorporate semantics into selective classification. The domain is not expressed by a set of numbers but rather by textual descriptions.

To summarize, selective classification is a good starting point for expressing a domain, but it requires extensions such as calibration and incorporation of semantics.

Q2 What are good examples to describe these limits? Can they be identified from reference data or generated artificially? Can they be described by interpretable characteristics or attributes? Extreme examples are rare in reference data, by definition. Generative models can be used to create synthetic examples, but they have to be coupled with the classifier in some way. In section 3.3, the classifier's gradient guides the GAN's generation. In section 3.4, the classifier conditions the GAN during training. Both methods allow the generation of artificial extreme examples. Interpreting these examples requires human annotation based on the examples, e.g., noise and contrast are important attributes.

Using text-to-image models helps automatically find textual descriptions. In section 5.3, groups of difficult-to-classify data are generated and described with text. The generator can be independent of the classifier, but a more efficient approach uses Bayesian optimization to choose relevant textual attributes to evaluate using the classifier's feedback. In section 5.4, the goal is complementary to describing limits: describe the in-domain data.

Q3 Can extreme examples be characterized based on their contribution to the different sources of error in learning? (Estimation, approximation, optimization, bias / variance trade-off) Unfortunately, the contributions of this thesis do not have an answer to this question.

U1 As a performance indicator of a given algorithm by providing more detailed information than classical global metrics, and additional information on the local behavior of the algorithms. Selective classification, studied in section 4.3, provides more detailed information than classical global metrics because two models might share the same global accuracy but have different selective accuracies because their confidence scores are not of the same quality. Some models know their uncertainty better than others. Confidence calibration, studied in sections 4.4 and 4.5 improves the uncertainty quantification, improving information on local behavior at the sample level.

Chapter 5 groups data into textually described subgroups with an associated accuracy value. It provides detailed local information at the subgroup level.

U2 As an explainability tool to intuitively analyze the overall behavior of the algorithm. Chapter 3 provides two explainability tools: showing synthetic extreme examples and what visual attributes are the most important. However, human knowledge is required to interpret them.

Chapter 5 also provides an explainability tool. Leveraging text-to-image models allows the generation of subgroups of data with associated textual descriptions. Section 5.3

is about describing failure modes, while section 5.4 is about expressing the domain, which includes data likely to be correctly classified.

U3 As a development tool to control learning and the trade-off between domain extension/performance. The trade-off between domain extension and performance can be controlled with selective classification (section 4.3) or SSC (section 5.4). However, only pre-trained classifiers were studied. The thesis does not explore how expressing a domain can be used during the classifier's learning phase. Addressing this use case could leverage insights gained from training good selective classifiers (Geifman and El-Yaniv, 2019).

U4 As a means of specifying the data to be collected. Explainability results from chapter 3 can be used to identify with human intervention which kind of data leads to failure, and collect training data to fix these failures. Chapter 5 provides a better tool that textually describes data likely to be misclassified or well-classified. Because text-to-image models are used, synthetic data can be generated to evaluate the classifier and estimate the impact of collecting specific real data.

6.2 A Look Back at 3 Years of AI Progress

Large Multimodal Models Large multimodal models integrate multiple modalities of data (e.g., text, images, video, audio) to understand and generate complex representations. In the last few years, many of these models were developed, for example CLIP (Radford et al., 2021), BLIP (Li et al., 2022a), Flamingo (Alayrac et al., 2022), Stable Diffusion (Rombach et al., 2022), Gemini (Gemini Team et al., 2023), and GPT-4 (OpenAI, 2023). While LLMs were a hot topic during the thesis period, it seems multimodal models are the way forward.

For this thesis, models combining images and text are particularly interesting. Chapter 5 leverages such models to describe subgroups of images with textual attributes. Estimating the performance of these subgroups creates a link between data descriptions and classification performance, one of the main aspects of the initial thesis goal. Other recent works mentioned in section 2.4 also use large multimodal models to describe classifier failures. These models will likely continue to be used to better evaluate classifiers' performance.

~~GANs~~ Diffusion Models At the beginning of the thesis, the state-of-the-art image generative models were GANs, e.g., BigGAN and StyleGAN. Many publications developed ways to edit real images through latent space navigation and GAN inversion. However, in the last few years, diffusion models have taken the lead. The research community seems to have shifted toward this model family. Besides higher image quality and more stable training than GANs, the ability to control the generation with text often results in impressive results.

Closed Research The release of OpenAI's ChatGPT at the end of 2022 provoked a massive shift in the field of AI. A big part of research pivoted towards LLMs. Tremendous computing power is now devoted to training such large models with hundreds of millions of parameters. The Bitter Lesson still holds (Sutton, 2019).

One additional unexpected consequence is that research is becoming increasingly closed. OpenAI does not really publish scientific papers anymore: details on their latest models remain secret, with "technical reports" not revealing much information (OpenAI, 2023). The trend is followed by Google, which restructured its research teams a few months after ChatGPT's release. Indeed, its business model relying on search is being threatened by new types of access to information enabled by LLMs. Now the company produces "reports" on their latest models, where little information about model architecture, training, and data are found (Gemini Team et al., 2023). This closing of research is understandable as companies now sell access to these models as a product. On the other hand, Meta does not sell access to models but does benefit from advances in the field, so the company keeps pushing for open source. In short, we can expect future breakthroughs to remain secret inside companies. This trend might tend to slow down research, but it could be compensated by the exponential development of the field.

But When Do DNNs Work? I find it surprising that expressing when an DNN is correct is not well covered by current research. To me, related work mostly tackles the reverse problem: identifying or detecting some failures (see section 2.4). However, the possibilities of failures are infinite, and being able to detect some of them does not guarantee reliable behavior the rest of the time. Furthermore, failure modes are split into separate research areas that only tackle one part of the problem. As mentioned in section 2.3, OOD detection only tackles semantic shifts, not failure detection, and SC aims to detect failures but supposes no shift happens, for instance.

This is why I believe the question "When is the prediction actually correct?" to be of primary importance compared to the reverse problems of identifying failures in some particular setting. Of course, addressing this question might be much harder in practice.

The question of AI reliability arises from deploying models. It thus might be more important for companies selling and using DNNs than for academics. For autonomous driving and medical systems, the consequences of DNNs' prediction errors are deadly. For AI based tools such as GPT-4 or Gemini, the consequences of errors (also called hallucinations) can be financially disastrous as users lose their trust in the product. Accelerating the deployment of AI systems will accelerate research on the question. However, as explained above, such research might not be publicly shared because of the financial stakes.

I think there may be a disconnection between industry and academia on the question of AI's reliability: industry knows the real issues but does not share its solutions, and academia proposes solutions to problems that might not be the ones encountered in practice. Anyway, hopefully, some open or closed source tools will soon make AI more trustworthy and thus more useful.

6.3 Perspectives

The goal of this thesis was not only to solve a problem but to define it clearly. I believe that I have made a small bit of progress on both. Properly expressing a domain is still unclear and perhaps unachievable (humans are imperfect in knowing when they are right). One contribution of the thesis is the perspectives it opens.

Theoretical Link Between Selective Classification and Calibration In chapter 4, the link between SC and calibration resulted in the new TvA approach to calibration. I believe more work can be done to unify these two notions. For instance, perfect calibration might *imply* optimal selective classification.

Recall the definition of perfect calibration from (Guo et al., 2017):

$$\Pr_P(\hat{y} = y | s(x) = p) = p, \quad \forall p \in [0, 1] \tag{6.1}$$

where the probability is over the data distribution $P(X, Y)$ of inputs (e.g., images) and labels, $\hat{y}$ is the label predicted by classifier f, $s(x)$ is the MSP of f for the input x, and y is the real label. What matters is the *absolute* value of s, which should reflect true probability.

From (Geifman et al., 2019), an *optimal* confidence score κ (for f) should reflect true loss monotonicity in the sense that for every two labeled instances $(x_1, y_1) \sim P(X, Y)$ and $(x_2, y_2) \sim P(X, Y)$,

$$\kappa(x_1, \hat{y}_1) \leq \kappa(x_2, \hat{y}_2) \iff \Pr_P[\hat{y}_1 \neq y_1] \geq \Pr_P[\hat{y}_2 \neq y_2]. \tag{6.2}$$

What matters is the *relative* values of κ, which should properly rank the examples.

To me, it seems possible that using the MSP a perfectly calibrated classifier, i.e. s verifying Equation 6.1 implies s is an optimal confidence score for SC as defined by 6.2. At least, there is probably a theoretical link between the two definitions.

However, even if that is the case, better calibration does not necessarily improve selective classification. This is empirically proven in the section 4.4 results and in (Galil et al., 2023b). It would be interesting to understand why. I think that the fields of calibration and SC could benefit from one another regarding methods or metrics, for instance. Calibration (and uncertainty quantification in general) is not an end in itself: it requires a purpose, such as filtering data likely to be misclassified, which is exactly what SC does.

Characterize Foundation Multimodal Models In this thesis, the object of interest is an image classifier to characterize. However, in many cases, it is possible to adapt a foundation model to the classification task, particularly a large multimodal model.

All the image classifier failure identification approaches cited in section 2.4 rely on "oracle" large multimodal models such as CLIP or Stable Diffusion to evaluate a "bad" classifier. Indeed, most of them use a low-performing classifier to evaluate, for instance, based on the ResNet architecture, which is almost a decade old and outperformed by more recent architectures. Then, they leverage large pre-trained models, considered oracles able to perfectly describe images with text, to identify when the classifier fails. My question is: why not just use the large pre-trained models for the classification task if they understand the data so well? This limitation is not discussed in the papers, which usually justify their motivation lightly: classifiers are biased and fail, and we should understand why. In practice, if I have at disposition a bad classifier based on a decade-old architecture and an "oracle" multimodal model, then why choose to use the bad classifier and do extra work evaluating its failures with the multimodal model instead of directly using the multimodal model? There is at least one scenario in which using a bad classifier on purpose would make sense, which is to embed models in low-power devices. However, even then, progress in quantization techniques and distillation, among others, allows the use of powerful multimodal models such as Gemini Nano to run on smartphones (Gemini Team et al., 2023).

In the experiments of section 5.4, the method SSC can filter out data based on attributes in the setting of a weak classifier on synthetic data (subsection 5.4.2). However, a properly trained classifier almost does not fail on this data, so the method becomes useless. Similarly, I expect all methods mentioned in section 2.4 to mostly stop working at identifying failures when the classifier is realistically developed, e.g., by adapting the large multimodal model used to identify failures.

Instead of pursuing this research niche of identifying failures of bad classifiers, I believe pursuing another direction is more useful: characterizing large foundation multimodal models. The advent of large multimodal and foundation models raises new questions. Such models are supposedly easily adaptable to many problems. For instance, CLIP can be used as a zero-shot or few-shot classifier with good performance for many datasets (Radford et al., 2021). It sometimes performs better than an image classifier trained from scratch, especially if training data is scarce. However, despite being trained on an enormous amount of data, it still has many failure modes. Some of the failures are caused by bad text encoding (Tong et al., 2023) or bad image encoding (Tong et al., 2024). The performance of vision-language models on downstream vision applications can even be predicted only from how they encode the class labels (Zohar et al., 2024). Evaluating these models is becoming challenging: some benchmarks include questions that can be solved without any visual information, and models might be trained on evaluation data (Chen et al., 2024). Good benchmarks should consider multi-dimensional aspects of the model evaluation (see (Wang et al., 2023) for an example for LLMs).

Evaluating a large multimodal model's general reliability seems a very hard problem. However, a starting point might be to evaluate the model's reliability when adapted to simpler subtasks such as image classification. In that case, insights from this thesis could be leveraged, as no assumption on the image classifier's structure was made.

Clarify Failure Modes In my opinion, there is a clear lack of clarity of the terms used in the research fields related to model failures. This includes "domain", "out-of-distribution", "corner case", "anomaly", "operational design domain", and many others. In many cases, papers have their definition of the terms, which makes it hard to compare and make sense of all these works.

As illustrated in Figure 2.3, the research fields addressing failures are mostly disjoint and focus on specific subproblems. AI research would benefit from clarifying all these subproblems, e.g., with an exhaustive survey paper. Works pointing in that direction seem rare, but here are some references. Yang et al. (2021) aim to unify OOD detection, open set recognition, novelty detection, and anomaly detection into the notion of generalized OOD. Guérin et al. (2023) propose the notion of model scope, which denotes the set of correct predictions, similar to the reliability domain discussed in the thesis. (Kumar et al., 2019b) categorizes failure modes into intentional (e.g., perturbation attack, poisoning attack) and unintentional (e.g., distribution shifts, corruptions).

A Reliability Benchmark Besides clarifying the different failure modes in a survey, another required work is the development of a comprehensive reliability benchmark. Because reliability is multi-faceted, a comprehensive benchmark should measure different dimensions that might compete with each other: a trade-off between different aspects might exist. Such a benchmark could produce radar charts to compare the different aspects, as illustrated in Figure 6.1.

Existing metrics can directly be reused, typically the AUROC for Selective Classification. However, metrics should be reformulated in some cases: the main goal is to identify failures, not detect if a shift happens, for instance. For covariate shifts, the metrics should measure whether the failure detection works for data under covariate shifts. For adversarial attacks, the goal is not to detect if there is an attack but to detect when the attack causes a failure. When a semantic shift happens, the classifier automatically fails (because it cannot predict a new class): in that case, predicting failure and if there is a shift are equivalent and could again be measured by standard metrics for Out-of-Distribution detection.

Methods to detect failures can come from either field, for instance using MSP as a confidence score (Hendrycks and Gimpel, 2017; Geifman and El-Yaniv, 2017) or Mahalanobis distance (Lee et al., 2018b).

Here are a few references as a starting point. WILDS (Koh et al., 2021) is a curated benchmark for covariate shift. It includes 10 datasets capturing a variety of distribution shifts that commonly occur in real-world scenarios. They include variations across hospitals in tumor identification, across camera traps in wildlife monitoring, and across different times and locations in satellite imaging and poverty mapping. It is mostly for Domain Generalization but can be used for unsupervised Domain Adaptation as well. OpenOOD (Yang et al., 2022) is a benchmark mostly about semantic shift and generalized OOD.

SafeBench is a competition designed to stimulate research on new benchmarks that assess and reduce risks associated with artificial intelligence. (Wang et al., 2023) is an example of a comprehensive benchmark to evaluate the trustworthiness of LLMs. RobustBench (Croce et al., 2020) is a standardized benchmark of adversarial robustness. Liao et al. (2021) list common problems in many evaluation benchmarks.

Quantitative evaluations are useful for making progress, so a reliability benchmark would guide future research on the notion of reliability domain.

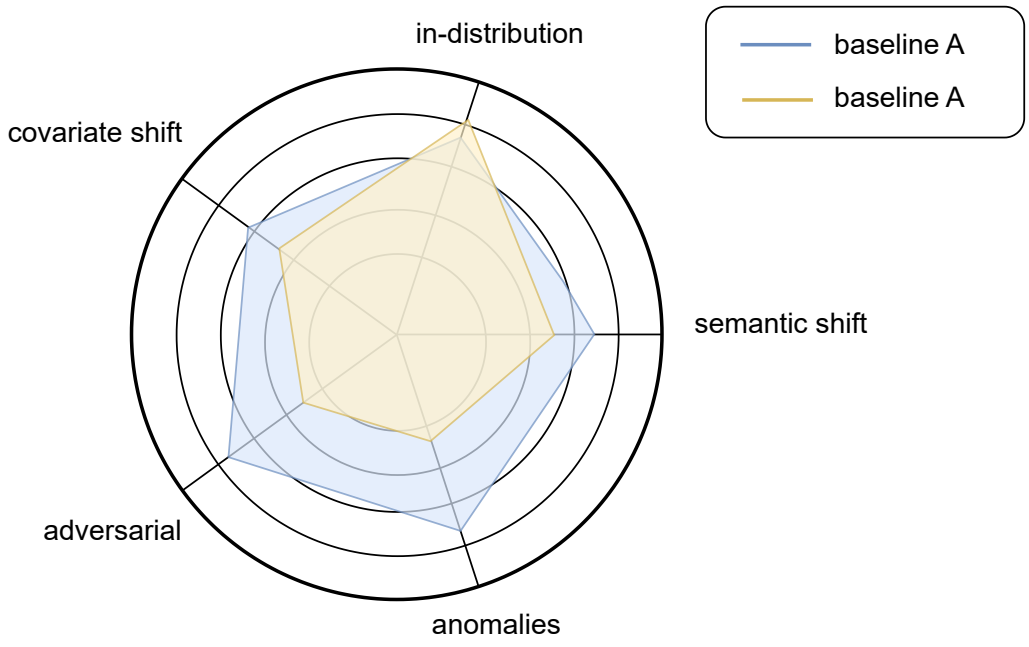


Figure 6.1: Concept of a radar chart evaluating the reliability of a classifier for baseline methods detecting several types of failures.

Chapter 7

Appendix

7.1 Implementation Details

Models weights

- Model weights for StyleGAN come from NVIDIA https://github.com/NVlabs/stylegan2-ada-pytorch.
- Model weights for CIFAR are from (Mukhoti et al., 2020).
- Model weights for ImageNet come from torchvision (maintainers and contributors, 2016).
- Model weights for ImageNet-21K are from (Ridnik et al., 2021).
- CLIP weights are from OpenAI's Hugging Face.
- Stable Diffusion 2.1 weights are from StabilityAI's Hugging Face.
- Original weights for T5 and RoBERTa come from the Transformers library (Wolf et al., 2020). The models are fine-tuned for each task using prompt-based learning (Liu et al., 2023). For more details, see (Chen et al., 2023b).
- GPT-J is from https://huggingface.co/EleutherAI/gpt-j-6b (Apache-2.0 license) and Llama-2 from https://huggingface.co/meta-llama/Llama-2-13b (license).

Datasets

- *MNIST* is an image classification dataset that contains grey-value images of handwritten digits with ten classes (digits from 0 to 9). The training set contains 60000 images, and the test set contains 10000 images.
- *CIFAR-10 (C10)* and *CIFAR-100 (C100)* (Krizhevsky et al., 2009) contain 60000 32x32 images corresponding to 10 and 100 classes, respectively. Data is split into subsets of 45000/5000/10000 images for train/validation/test. When used for calibration, the original validation and test sets are concatenated and randomly split into a calibration set of size 5000, and a test set of size 10000.

- *ImageNet (IN)* (Deng et al., 2009) contains 1.3 million images from 1000 classes. When used for calibration, following (Guo et al., 2017), the original validation test of size 50000 is randomly split into a calibration set and a test set, both of size 25000.

- *ImageNet-21K (IN21K)* (Ridnik et al., 2021), in its winter21 version, contains 11 million images in the train set, and 522500 in the test set (50 for each of the 10450 classes). When used for calibration, the test set is randomly split into equal-sized calibration and test set (261250 samples each, 25 per class).

- *PUG Animals* (Bordes et al., 2024) is a photorealistic synthetic image dataset with annotated factors of variations. Images are produced with Unreal Engine, a game engine that produces photorealistic environments. The dataset contains 215040 images using 70 animal assets, 64 `backgrounds`, 3 object `sizes`, and 4 `textures` under 4 different camera `orientations`. All combinations of attributes are covered by the images with one image for each combination.

- *CUB-200-2011* (Wah et al., 2011) is a dataset for fine-grained image classification of bird species. It contains 11788 images of 200 bird species, with approximately 60 images per class. The dataset is split into a training set of size 5994 and a testing set of size 5794.

- *Amazon Fine Foods* (McAuley and Leskovec, 2013) is a collection of customer reviews for fine foods sold on Amazon. Reviews are categorized into bad, neutral, and good. The original validation set size is 78741 and test size 91606. 78741 samples are randomly chosen for calibration and 91606 for test.

- *DynaSent* (Potts et al., 2021) is a dynamic benchmark for sentiment analysis consisting of sentences annotated as positive, neutral, and negative. The original validation set size is 11160 and test size 4320. 11160 samples are randomly chosen for calibration and 4320 for test.

- *MNLI* (Williams et al., 2018) contains pairs of sentences labeled as contradiction, neutral, and entailment. The original validation set size is 19635 and test size 9815. 19635 samples are randomly chosen for calibration and 9815 for test.

- *Yahoo Answers (YA)* (Zhang et al., 2015b) contains question-answers pairs corresponding to 10 different topics. The original validation set size is 14000 and test size 60000. 14000 samples are randomly chosen for calibration and 60000 for test.

- TREC (Voorhees and Tice, 2000) contains questions categorized into 6 classes. The training set contains 5500 labeled questions, and the test set contains another 500.

- SST-5 (Socher et al., 2013) contains 11855 sentences corresponding to 5 sentiments (from very negative to very positive).

- DBpedia (Zhang et al., 2015a) contains text for topic classification with 14 classes. The training set contains 560000 samples, and the test set 5000.

Codes

- Original implementation of StyleGAN: https://github.com/NVlabs/stylegan2-ada-pytorch
- The library `netcal` (Küppers et al., 2020) (Apache-2.0 license) was used for binary methods for calibration and reliability diagrams.
- Official implementation of temperature scaling: https://github.com/gpleiss/temperature_scaling (MIT license).
- Official implementation of Dirichlet calibration: https://github.com/dirichletcal/experiments_dnn (MIT license).
- Official implementation of I-Max: https://github.com/boschresearch/imax-calibration (AGPL-3.0 license).
- Official implementation of IRM: https://github.com/zhang64-llnl/Mix-n-Match-Calibration (MIT license).
- Calibration evaluation codes are from https://github.com/JeremyNixon/uncertainty-metrics-1 (Apache-2.0 license) and https://github.com/IdoGalil/benchmarking-uncertainty-estimation-performance (MIT license).
- I used `PyTorch` 2.0.0 (Ansel et al., 2024) (BSD-style license).
- CIFAR models are from (Mukhoti et al., 2020) https://github.com/torrvision/focal_calibration (MIT license).
- ImageNet-21K models are from (Ridnik et al., 2021) https://github.com/Alibaba-MIIL/ImageNet21K (MIT license).
- CLIP models are from HuggingFace's Transformers library (Wolf et al., 2020) (Apache-2.0 license).
- Pretrained language models are from Transformers (Wolf et al., 2020) and calibration codes from (Chen et al., 2023b) https://github.com/lifan-yuan/PLMCalibration (MIT license).
- Code for the calibration of LLMs using ICL is from https://github.com/mominabbass/LinC, itself built upon https://github.com/tonyzhaozh/few-shot-learning (Apache-2.0 license).
- Genetic Algorithm code is from `pymoo` (Blank and Deb, 2020).
- Combinatorial Testing code is from `allpairspy` (allpairspy).
- ML models such as SVR, Lasso, Random Forest Regressor, and Linear Regression are from `scikit-learn` (Buitinck et al., 2013).
- Stable Diffusion code is from the `diffusers` library (von Platen et al., 2022).
- Textual Inversion code is from the `diffusers` library (von Platen et al., 2022) https://github.com/huggingface/diffusers/blob/main/examples/textual_inversion/textual_inversion.py.

7.2 Additional Results for Top-versus-All (TvA) Calibration

7.2.1 Theoretical Justification of TvA for Temperature Scaling

L is the number of classes, $f_k(x)$ the classifier estimated probability for class k and data sample x, y the correct class, and the confidence $s(x) := \max_k f_k(x)$. The cross-entropy loss is $l_{CE}(x, y) = -\sum_{k=1}^{L} 1\{k = y\} \cdot \log(f_k(x)) = -\log(f_y(x))$. Because the last layer of the classifier is a softmax function, $f_y(x) = \frac{e^{z_y}}{\sum_k e^{z_k}}$ with z the logits vector. Note that I omit the writing variable x in the following for clarity.

Temperature scaling optimizes a coefficient $T > 0$ that scales the logits vector. Predicted probabilities become $f_y(x) = \frac{e^{z_y/T}}{\sum_k e^{z_k/T}}$.

Let us first develop the standard cross-entropy loss when temperature scaling is applied:

$$l_{CE} = -\log(f_y) = -\log(\frac{e^{z_y/T}}{\sum_k e^{z_k/T}}) = -\left(\log(e^{z_y/T}) - \log(\sum_k e^{z_k/T})\right) = -\frac{z_y}{T} + \log(\sum_k e^{z_k/T})$$

Let us compute its gradient:

$$\begin{aligned}
\frac{\partial l_{CE}}{\partial T} &= \frac{z_y}{T^2} + \frac{\partial \log(\sum_k e^{z_k/T})}{\partial \sum_k e^{z_k/T}} \cdot \frac{\partial \sum_k e^{z_k/T}}{\partial T} \text{by application of the chain rule on the second term.} \\
&= \frac{z_y}{T^2} + \frac{1}{\sum_k e^{z_k/T}} \cdot \sum_k \frac{\partial e^{z_k/T}}{\partial T} \\
&= \frac{z_y}{T^2} + \frac{1}{\sum_k e^{z_k/T}} \cdot \sum_k \frac{\partial (e^{z_k})^{1/T}}{\partial T} \\
&= \frac{z_y}{T^2} + \frac{1}{\sum_k e^{z_k/T}} \cdot \sum_k \frac{\log(e^{z_k})}{-T^2} (e^{z_k})^{1/T} \\
&= \frac{z_y}{T^2} + \frac{1}{\sum_k e^{z_k/T}} \cdot \sum_k \frac{z_k \cdot e^{z_k/T}}{-T^2} \\
&= \frac{1}{T^2}\left(z_y - \frac{\sum_k z_k \cdot e^{z_k/T}}{\sum_k e^{z_k/T}}\right) \\
&= \frac{1}{T^2}\left(z_y - \sum_k \frac{z_k \cdot e^{z_k/T}}{\sum_j e^{z_j/T}}\right) \\
&= \frac{1}{T^2}\left(z_y - \sum_k z_k \cdot f_k\right) \qquad (7.1)
\end{aligned}$$

For the TvA approach, the problem becomes binary. The classification output becomes the confidence $s(x) = \max_{j \in \mathcal{Y}} f_j(x)$ and the ground truth label becomes a binary representation of the prediction correctness: $y^b = 1_{\hat{y}=y}$ with $\hat{y}(x) = \arg\max_{k \in \mathcal{Y}} f_k(x)$ and 1 the indicator function. The loss used is the binary cross entropy $l_{BCE}(x, y) = -\big(y^b \cdot \log s(x) + (1 - y^b) \cdot \log(1 - s(x))\big)$

Let us compute the gradient:

$$
\begin{aligned}
\frac{\partial l_{BCE}}{\partial T} &= \frac{\partial l_{BCE}}{\partial s} \cdot \frac{\partial s}{\partial T} \\
&= -\left(y^b \frac{1}{s} + (1-y^b)\frac{-1}{1-s} \right) \cdot \frac{\partial s}{\partial T} \\
&= -\left(\frac{y^b(1-s)}{s(1-s)} - \frac{s(1-y^b)}{s(1-s)} \right) \cdot \frac{\partial s}{\partial T} \\
&= \frac{s-y^b}{s(1-s)} \cdot \frac{\partial s}{\partial T} \\
&= \frac{s-y^b}{s(1-s)} \cdot \frac{\partial \frac{e^{z_m/T}}{\sum_k e^{z_k/T}}}{\partial T} \text{ because } s = \max_j \frac{e^{z_j/T}}{\sum_k e^{z_k/T}} = \frac{e^{z_m/T}}{\sum_k e^{z_k/T}} \text{ with } z_m = \max_k z_k \\
&= \frac{s-y^b}{s(1-s)} \cdot \frac{\frac{\partial e^{z_m/T}}{\partial T}\sum_k e^{z_k/T} - e^{z_m/T}\frac{\partial \sum_k e^{z_k/T}}{\partial T}}{(\sum_k e^{z_k/T})^2} \\
&= \frac{s-y^b}{s(1-s)} \cdot \frac{\frac{\log(e^{z_m})}{-T^2}(e^{z_m})^{1/T}\sum_k e^{z_k/T} - e^{z_m/T}\sum_k \frac{\log(e^{z_k})}{-T^2}(e^{z_k})^{1/T}}{(\sum_k e^{z_k/T})^2} \\
&= \frac{s-y^b}{s(1-s)} \cdot \frac{e^{z_m/T}}{T^2} \cdot \frac{-\log(e^{z_m})\sum_k e^{z_k/T} + \sum_k \log(e^{z_k})(e^{z_k})^{1/T}}{(\sum_k e^{z_k/T})^2} \\
&= \frac{s-y^b}{s(1-s)} \cdot \frac{e^{z_m/T}}{T^2} \cdot \frac{-z_m\sum_k e^{z_k/T} + \sum_k z_k \cdot e^{z_k/T}}{(\sum_k e^{z_k/T})^2} \\
&= \frac{s-y^b}{s(1-s)} \cdot \frac{1}{T^2} \cdot s \cdot \frac{-z_m\sum_k e^{z_k/T} + \sum_k z_k \cdot e^{z_k/T}}{\sum_k e^{z_k/T}} \\
&= \frac{1}{T^2} \cdot \frac{s-y^b}{1-s} \cdot \frac{-z_m\sum_k e^{z_k/T} + \sum_k z_k \cdot e^{z_k/T}}{\sum_k e^{z_k/T}} \\
&= \frac{1}{T^2} \cdot \frac{s-y^b}{1-s} \cdot \left(\frac{\sum_k z_k \cdot e^{z_k/T}}{\sum_k e^{z_k/T}} - z_m \right) \\
&= \frac{1}{T^2} \cdot \frac{y^b-s}{1-s} \cdot \left(z_m - \frac{\sum_k z_k \cdot e^{z_k/T}}{\sum_k e^{z_k/T}} \right) \\
&= \frac{1}{T^2} \cdot \frac{y^b-s}{1-s} \cdot \left(z_m - \sum_k \frac{z_k \cdot e^{z_k/T}}{\sum_j e^{z_j/T}} \right) \\
&= \frac{1}{T^2} \cdot \frac{y^b-s}{1-s} \cdot \left(z_m - \sum_k z_k \cdot f_k \right) \qquad (7.2)
\end{aligned}
$$

There are two cases:

1. The prediction is correct: $y^b = 1$ and $z_m = z_y$. Let us inject these in (7.2): $\frac{\partial l_{BCE}}{\partial T} = \frac{1}{T^2} \cdot (z_y - \sum_k z_k \cdot f_k) = \frac{\partial l_{CE}}{\partial T}$
   We thus get the same gradient as the standard cross-entropy loss.

2. The prediction is incorrect: $y^b = 0$ and $z_m > z_y$. (7.2) becomes: $\frac{\partial l_{BCE}}{\partial T} = \frac{1}{T^2} \cdot \frac{s}{s-1} \cdot (z_m - \sum_k z_k \cdot f_k)$
   By comparing to (7.1), we have the term $\frac{1}{T^2} \cdot (z_m - \sum_k z_k \cdot f_k) > \frac{1}{T^2}(z_y - \sum_k z_k \cdot f_k) = \frac{\partial l_{CE}}{\partial T}$ and the remaining part of (7.2) $|\frac{s}{s-1}| > 1$ when $s > 0.5$.

So to recapitulate, $|\frac{\partial l_{BCE}}{\partial T}| > |\frac{\partial l_{CE}}{\partial T}|$ when $s > 0.5$, which corresponds to the vast majority of data points as the classifier gets better calibrated. This is shown in Figure 4.4.
We also have $\lim_{s \to 1} |\frac{s}{s-1}| = \infty$. In practice, s is not close enough to 1 to generate exploding gradients, so it just means that as confidences for wrong predictions gets higher, so does the gradient to reduce the confidences.

The conclusion is that for correct predictions, the TvA approach does not change the optimization, but for incorrect predictions, the gradient is stronger and penalizes more heavily confident predictions that are wrong. This is also proven experimentally by looking at Table 7.6 where we see that average confidences of temperature scaling with the TvA approach (TS_{TvA}) are lower than the ones using the standard approach (TS), for almost all networks. This makes the average confidences closer to the accuracy, showing reduced overconfidence.

7.2.2 Additional Tables of Results

Table 7.1: Computing time (in seconds) of the calibration on ImageNet, using one NVIDIA V100 GPU. The first column denotes the data preprocessing time, which includes computing the model logits for all calibration examples. Post-hoc calibration methods do not usually require much computing power compared to classifier training.

Model	Preproc.	IRM	I-Max	TS	TS_{TvA}	VS	$\text{VS}_{\text{reg,TvA}}$	DC	$\text{DC}_{\text{reg,TvA}}$	HB	HB_{TvA}	Iso	Iso_{TvA}	Beta	Beta_{TvA}	BBQ	BBQ_{TvA}
ResNet-50	141	2021	543	215	218	214	217	226	226	129	1	66	1	873	22	1156	2
ViT-B/16	151	7119	524	225	226	217	222	232	235	127	1	61	1	917	23	1169	2

Table 7.2: ECE in % (lower is better, best in bold) – full results for image classification datasets. Averages on 5 seeds. Mean relative improvements from TvA are shown (negative values for reductions of ECE). Methods in purple impact the model prediction, potentially degrading accuracy; methods in teal do not. Values are averaged over five random seeds.

(a) CIFAR-10

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
ResNet-50	1.80	0.77	0.68	1.07	1.09	0.91	0.90	0.91	0.89	2.16	1.50	1.13	0.74	1.27	0.94	1.02	**0.53**
ResNet-110	2.57	0.53	0.54	1.32	1.36	1.35	1.33	1.35	1.34	2.97	1.40	1.20	0.56	1.45	0.67	1.42	**0.37**
WRN	1.21	0.78	0.64	1.10	0.92	1.19	0.97	1.19	0.97	1.75	1.46	1.16	0.82	0.88	0.78	0.79	**0.52**
DenseNet	1.52	0.60	0.59	1.32	1.61	1.21	1.47	1.19	1.47	2.04	2.05	0.98	0.58	1.07	0.71	1.00	**0.30**
Mean improvement ConvNets				3%		0%		1%		-25%		-39%		-31%		-57%	
CLIP (ViT-B/32)	4.77	1.39	1.35	1.02	1.02	2.79	1.88	2.82	1.90	1.64	1.46	1.21	1.16	1.79	1.05	2.32	**0.98**
CLIP (ViT-B/16)	5.39	1.04	0.95	0.64	**0.57**	2.91	1.91	2.92	1.89	1.15	1.83	1.29	0.79	1.58	0.91	2.14	0.75
CLIP (ViT-L/14)	4.93	0.65	0.52	0.68	0.60	1.98	1.76	1.94	1.74	0.96	0.77	0.65	0.62	0.81	0.62	1.01	**0.46**
Mean improvement CLIP				-8%		-26%		-26%		9%		-16%		-36%		-59%	

(b) CIFAR-100

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
ResNet-50	6.56	1.37	1.35	4.93	2.97	5.23	2.24	5.22	2.24	5.59	3.39	5.70	1.40	10.07	1.47	9.62	**1.17**
ResNet-110	7.95	1.40	1.31	5.05	4.04	5.32	2.65	5.30	2.70	6.10	4.74	6.59	1.34	8.53	1.44	10.04	**1.23**
WRN	4.41	1.24	0.95	4.42	2.70	4.57	2.28	4.55	2.27	4.50	2.75	4.42	1.41	10.02	1.26	8.45	**0.94**
DenseNet	5.23	1.20	0.97	4.19	2.12	4.53	2.23	4.51	2.20	4.99	2.84	4.61	1.35	9.91	1.24	10.12	**0.76**
Mean improvement ConvNets				-37%		-52%		-52%		-36%		-73%		-86%		-89%	
CLIP (ViT-B/32)	9.51	2.22	1.80	2.22	2.12	8.74	3.49	7.97	1.98	6.52	2.71	2.35	1.47	8.11	**1.21**	8.13	1.23
CLIP (ViT-B/16)	10.63	3.33	3.04	2.71	2.74	8.64	3.04	8.14	1.80	7.09	2.53	2.78	1.74	7.41	1.76	7.09	**1.48**
CLIP (ViT-L/14)	10.96	3.15	2.86	2.68	2.66	5.96	2.06	6.54	1.74	6.44	1.99	2.46	1.62	6.93	1.47	6.46	**1.46**
Mean improvement CLIP				-1%		-63%		-75%		-64%		-36%		-80%		-80%	

(c) ImageNet

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
ResNet-18	2.71	1.01	**0.57**	1.87	1.86	1.73	2.12	3.43	3.44	4.39	1.44	3.87	0.94	9.68	0.91	9.54	**0.57**
ResNet-34	3.63	0.84	**0.56**	1.77	1.80	1.87	2.02	3.46	2.98	4.97	1.11	4.08	0.83	9.17	0.85	8.42	0.60
ResNet-50	41.15	2.56	2.59	3.25	1.66	3.26	0.90	3.27	0.94	11.30	2.20	1.23	0.68	8.44	0.66	5.80	**0.50**
ResNet-101	13.56	0.82	0.58	3.72	2.22	4.22	1.62	4.20	1.58	9.21	1.87	3.01	0.71	6.35	0.61	6.18	**0.52**
Mean improvement ResNet				-22%		-26%		-37%		-76%		-69%		-91%		-92%	
EffNet-B7	12.61	0.61	**0.40**	3.71	2.96	3.84	1.41	3.82	1.35	9.32	2.27	2.93	0.65	6.93	0.58	4.89	**0.40**
EffNetV2-S	16.92	0.68	**0.44**	3.60	3.34	3.91	1.43	3.90	1.45	8.03	2.57	2.97	0.67	7.66	0.68	5.33	0.47
EffNetV2-M	24.88	0.80	0.70	3.77	2.71	3.84	1.16	3.82	1.14	8.32	1.79	2.89	0.75	6.55	0.75	4.36	**0.49**
EffNetV2-L	8.48	0.63	0.39	2.86	1.34	3.08	1.05	3.06	0.98	9.45	0.99	2.51	0.64	5.06	0.54	2.99	**0.37**
Mean improvement EffNet				-27%		-66%		-66%		-78%		-76%		-90%		-90%	
ConvNeXt-T	16.95	1.11	0.84	3.08	1.52	3.49	1.18	3.48	1.15	8.95	1.66	2.55	0.87	7.34	0.70	5.63	**0.61**
ConvNeXt-S	17.60	0.75	0.59	3.76	2.29	4.19	1.32	4.18	1.31	8.77	1.73	3.06	0.70	7.46	0.68	5.32	**0.48**
ConvNeXt-B	18.78	0.74	**0.41**	3.83	2.51	4.10	1.33	4.09	1.31	9.44	1.84	3.03	0.77	7.72	0.70	5.02	0.52
ConvNeXt-L	12.52	0.66	0.47	4.02	2.69	4.42	1.64	4.42	1.63	7.97	1.37	3.26	0.67	7.12	0.62	4.55	**0.46**
Mean improvement ConvNeXt				-39%		-66%		-67%		-81%		-74%		-91%		-90%	
ViT-B/32	6.37	0.77	0.60	4.02	2.17	4.67	1.82	4.66	1.77	6.58	1.68	3.58	0.84	9.51	0.73	7.76	**0.53**
ViT-B/16	5.61	0.86	0.54	3.80	3.25	4.29	1.93	4.27	1.92	7.36	2.29	3.39	0.79	5.88	0.71	6.79	**0.51**
ViT-L/32	4.27	0.83	0.75	5.00	3.89	5.37	2.53	5.37	2.49	6.33	2.57	4.43	0.76	9.31	0.79	7.39	**0.61**
ViT-L/16	5.17	0.99	0.77	5.77	4.63	5.29	2.62	5.27	2.58	7.44	3.05	4.10	0.85	6.83	0.78	7.38	**0.54**
ViT-H/14	0.60	0.60	**0.40**	1.84	0.88	1.95	1.22	2.00	1.17	7.84	0.75	2.48	0.62	1.67	0.63	3.62	0.42
Mean improvement ViT				-31%		-51%		-53%		-70%		-78%		-85%		-92%	
Swin-T	6.82	0.76	0.45	3.08	1.85	3.45	1.38	3.44	1.37	7.72	1.61	2.94	0.72	6.72	0.67	6.31	**0.43**
Swin-S	3.65	0.78	0.54	3.63	2.95	4.17	1.77	4.17	1.76	7.91	2.31	3.29	0.77	7.20	0.80	5.72	**0.44**
Swin-B	4.77	0.72	**0.45**	3.88	3.43	4.22	1.98	4.21	1.95	7.98	2.27	3.33	0.75	6.83	0.68	4.70	0.52
SwinV2-T	8.31	0.80	**0.46**	3.61	2.25	3.92	1.51	3.91	1.49	8.68	1.76	3.08	0.81	7.81	0.79	6.20	0.52
SwinV2-S	6.07	0.75	0.46	3.79	3.32	4.24	1.74	4.23	1.71	8.51	2.26	3.16	0.74	7.18	0.67	5.02	**0.41**
SwinV2-B	5.50	0.69	0.59	3.82	3.68	4.25	1.80	4.22	1.73	7.53	2.68	3.34	0.67	6.78	0.63	4.42	**0.55**
Mean improvement Swin				-21%		-58%		-59%		-73%		-77%		-90%		-91%	
CLIP (ViT-B/32)	1.50	0.96	**0.75**	1.70	1.58	1.38	0.92	36.01	70.52	3.57	0.82	2.22	0.84	8.12	0.88	6.63	0.82
CLIP (ViT-B/16)	1.80	1.31	0.75	1.92	1.89	1.61	0.87	34.02	66.04	4.53	1.08	2.35	1.01	8.48	0.91	6.98	**0.74**
CLIP (ViT-L/14)	2.57	0.97	**0.67**	2.04	1.99	1.89	1.36	26.06	66.40	5.95	1.35	2.49	0.92	8.33	1.01	7.86	0.84
Mean improvement CLIP				-4%		-36%		115%		-77%		-61%		-89%		-89%	

(d) ImageNet-21K

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
MN3	12.34	err.	err.	8.69	4.39	2.52	2.40	58.84	81.16	err.	1.02	2.00	0.21	err.	0.20	5.50	**0.17**
ViT-B/16	6.27	err.	err.	8.92	6.55	2.38	1.54	8.22	3.20	err.	3.72	2.14	0.22	err.	0.24	7.89	**0.12**
Mean improvement				-38%		-20%		-12%		err.		-90%		err.		-98%	

Table 7.3: ECE in % (lower is better, best in bold) – full results for text classification datasets. Averages on 5 seeds. Mean relative improvements from TvA are shown (negative values for reductions of ECE). Methods in purple impact the model prediction, potentially degrading accuracy; methods in teal do not. Values are averaged over five random seeds.

(a) Amazon Fine Foods

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
T5	5.18	0.28	0.25	1.24	1.26	1.34	1.34	0.99	1.28	5.44	0.80	0.41	0.28	0.38	0.30	2.45	**0.21**
T5-large	5.76	0.26	0.26	0.97	1.04	1.70	1.49	1.36	1.34	5.71	1.63	0.33	0.26	0.40	0.26	3.29	**0.14**
Mean improvement T5				4%		-6%		14%		-78%		-26%		-28%		-94%	
RoBERTa	7.90	0.28	0.30	2.27	2.21	1.37	1.93	1.51	1.78	7.48	4.30	0.31	0.28	1.11	0.37	4.07	**0.24**
RoBERTa-large	6.83	0.32	0.25	2.52	2.44	1.45	1.78	1.24	1.57	6.36	4.45	0.72	0.26	0.38	0.34	3.96	**0.16**
Mean improvement RoBERTa				-3%		32%		22%		-36%		-37%		-39%		-95%	

(b) DynaSent

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
T5	7.99	1.48	1.40	1.18	**1.16**	4.88	1.97	4.66	2.04	10.95	2.81	1.44	1.66	1.32	1.53	1.99	1.32
T5-large	9.73	1.30	1.36	3.20	3.19	7.38	2.03	7.15	2.00	11.81	4.72	1.56	1.45	1.44	1.62	1.80	**0.92**
Mean improvement T5				-1%		-66%		-64%		-67%		4%		14%		-41%	
RoBERTa	17.37	1.67	1.59	13.20	13.20	15.84	7.83	15.02	6.40	18.36	10.48	1.68	1.64	1.76	1.19	1.26	**1.06**
RoBERTa-large	14.88	1.46	1.42	10.94	10.94	13.49	5.77	12.78	4.73	15.69	9.30	1.74	1.43	1.52	1.08	0.85	**0.75**
Mean improvement RoBERTa				0%		-54%		-60%		-42%		-10%		-31%		-14%	

(c) MNLI

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
T5	6.48	0.71	0.67	1.17	1.15	3.25	1.88	3.21	2.01	7.74	2.12	0.84	0.73	0.98	0.80	1.91	**0.47**
T5-large	7.59	0.74	0.72	4.46	4.45	5.66	1.71	5.41	1.64	8.20	4.43	0.77	0.76	1.78	0.57	2.28	**0.40**
Mean improvement T5				-1%		-56%		-54%		-59%		-7%		-43%		-79%	
RoBERTa	10.26	0.90	0.81	6.52	6.52	7.83	2.03	7.38	2.12	11.06	6.16	0.87	0.94	1.25	0.93	1.20	**0.60**
RoBERTa-large	8.18	0.87	0.61	4.93	4.92	6.15	1.80	5.81	1.85	8.80	5.39	1.12	0.90	2.09	0.75	0.84	**0.60**
Mean improvement RoBERTa				-0%		-72%		-70%		-42%		-6%		-45%		-39%	

(d) Yahoo Anwsers

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
T5	6.64	0.74	0.90	**0.67**	0.97	2.70	1.01	2.66	0.94	7.70	1.64	1.57	0.79	2.61	0.95	4.15	0.71
T5-large	9.04	0.87	0.72	1.47	1.73	4.70	1.31	4.84	1.36	10.34	2.39	1.90	0.85	3.01	0.98	3.15	**0.67**
Mean improvement T5				31%		-67%		-68%		-78%		-52%		-66%		-81%	
RoBERTa	19.53	1.03	0.72	12.02	12.00	16.26	2.29	15.85	1.73	20.13	9.41	1.96	1.05	5.05	0.72	3.56	**0.60**
RoBERTa-large	19.65	0.90	0.86	12.77	12.75	16.67	2.75	16.30	2.70	20.18	10.19	1.87	0.94	4.96	0.78	3.26	**0.57**
Mean improvement RoBERTa				-0%		-85%		-86%		-51%		-48%		-85%		-83%	

Table 7.4: Standard deviations of ECE in % for 5 seeds.

(a) CIFAR-10

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
ResNet-50	0.16	0.15	0.15	0.14	0.26	0.25	0.18	0.24	0.16	0.29	0.29	0.19	0.14	0.39	0.12	0.37	0.09
ResNet-110	0.10	0.22	0.08	0.12	0.10	0.14	0.16	0.12	0.15	0.14	0.21	0.22	0.18	0.24	0.20	0.17	0.19
WRN	0.09	0.17	0.21	0.29	0.21	0.25	0.07	0.25	0.08	0.09	0.57	0.23	0.20	0.19	0.22	0.26	0.24
DenseNet	0.11	0.08	0.16	0.12	0.18	0.09	0.09	0.09	0.08	0.13	0.78	0.14	0.05	0.15	0.08	0.19	0.10
CLIP (ViT-B/32)	0.12	0.49	0.35	0.17	0.13	0.31	0.37	0.30	0.32	0.29	0.32	0.13	0.44	0.48	0.41	0.39	0.32
CLIP (ViT-B/16)	0.17	0.36	0.33	0.19	0.10	0.22	0.15	0.22	0.13	0.13	0.91	0.18	0.27	0.09	0.26	0.11	0.29
CLIP (ViT-L/14)	0.05	0.22	0.21	0.11	0.09	0.11	0.15	0.10	0.16	0.22	0.15	0.19	0.18	0.21	0.11	0.18	0.09

(b) CIFAR-100

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
ResNet-50	0.22	0.50	0.60	0.53	0.47	0.52	0.46	0.50	0.44	0.47	0.48	0.32	0.50	0.37	0.33	0.49	0.25
ResNet-110	0.28	0.23	0.25	0.38	0.39	0.35	0.31	0.33	0.31	0.33	0.59	0.28	0.34	0.60	0.21	0.56	0.11
WRN	0.19	0.24	0.38	0.27	0.14	0.25	0.30	0.24	0.33	0.16	0.28	0.23	0.46	0.71	0.57	0.33	0.32
DenseNet	0.10	0.24	0.17	0.26	0.13	0.30	0.20	0.35	0.22	0.60	0.79	0.24	0.45	0.68	0.28	0.31	0.22
CLIP (ViT-B/32)	0.14	0.25	0.20	0.05	0.14	0.13	0.27	0.48	0.45	0.32	1.17	0.26	0.25	0.49	0.23	0.40	0.46
CLIP (ViT-B/16)	0.21	0.35	0.42	0.30	0.37	0.40	0.37	0.42	0.48	0.74	1.00	0.50	0.42	0.54	0.42	0.37	0.51
CLIP (ViT-L/14)	0.17	0.33	0.53	0.16	0.19	0.23	0.24	0.19	0.32	0.34	1.14	0.25	0.24	0.39	0.30	0.27	0.09

(c) ImageNet

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
ResNet-18	0.14	0.12	0.14	0.11	0.11	0.13	0.13	0.15	0.14	0.38	0.19	0.03	0.17	0.23	0.24	0.22	0.17
ResNet-34	0.12	0.16	0.24	0.12	0.10	0.14	0.20	0.13	0.17	0.54	0.10	0.24	0.17	0.19	0.21	0.06	0.13
ResNet-50	0.21	0.20	0.18	0.26	0.34	0.32	0.15	0.31	0.15	1.71	0.18	0.22	0.16	0.15	0.26	0.21	0.09
ResNet-101	0.15	0.11	0.22	0.11	0.18	0.18	0.25	0.17	0.25	1.02	0.08	0.19	0.11	0.23	0.16	0.25	0.14
EffNet-B7	0.07	0.13	0.11	0.10	0.12	0.15	0.18	0.14	0.22	1.91	0.17	0.13	0.10	0.16	0.16	0.11	0.06
EffNetV2-S	0.15	0.19	0.08	0.17	0.19	0.22	0.21	0.22	0.17	1.10	0.24	0.26	0.15	0.32	0.15	0.20	0.24
EffNetV2-M	0.18	0.12	0.13	0.17	0.15	0.16	0.26	0.18	0.22	1.10	0.47	0.13	0.10	0.24	0.15	0.26	0.06
EffNetV2-L	0.11	0.07	0.12	0.15	0.12	0.21	0.17	0.18	0.19	1.64	0.23	0.25	0.07	0.33	0.17	0.41	0.13
ConvNeXt-T	0.16	0.08	0.12	0.25	0.28	0.29	0.30	0.28	0.33	1.57	0.42	0.31	0.12	0.22	0.21	0.15	0.10
ConvNeXt-S	0.14	0.26	0.18	0.23	0.17	0.24	0.27	0.23	0.28	1.17	0.21	0.15	0.12	0.14	0.14	0.29	0.08
ConvNeXt-B	0.20	0.09	0.16	0.30	0.26	0.36	0.33	0.36	0.32	2.12	0.36	0.33	0.10	0.11	0.06	0.40	0.12
ConvNeXt-L	0.16	0.10	0.09	0.26	0.17	0.21	0.27	0.20	0.28	1.33	0.12	0.28	0.14	0.26	0.19	0.33	0.13
ViT-B/32	0.20	0.17	0.19	0.29	0.17	0.34	0.34	0.37	0.30	0.99	0.12	0.31	0.15	0.21	0.16	0.28	0.14
ViT-B/16	0.15	0.07	0.13	0.26	0.17	0.35	0.36	0.35	0.34	0.54	0.24	0.28	0.08	0.23	0.11	0.23	0.10
ViT-L/32	0.10	0.15	0.11	0.19	0.10	0.28	0.22	0.28	0.22	0.31	0.17	0.29	0.11	0.20	0.16	0.23	0.14
ViT-L/16	0.22	0.24	0.39	0.19	0.15	0.35	0.36	0.35	0.37	0.81	0.12	0.30	0.28	0.31	0.23	0.21	0.32
ViT-H/14	0.15	0.18	0.09	0.21	0.28	0.19	0.20	0.20	0.22	0.75	0.15	0.20	0.19	0.34	0.16	0.29	0.18
Swin-T	0.17	0.13	0.19	0.19	0.12	0.20	0.16	0.20	0.16	0.70	0.10	0.23	0.15	0.34	0.20	0.12	0.17
Swin-S	0.10	0.18	0.18	0.19	0.19	0.21	0.25	0.22	0.20	0.48	0.27	0.24	0.16	0.22	0.27	0.30	0.14
Swin-B	0.07	0.14	0.16	0.21	0.25	0.24	0.30	0.21	0.32	0.80	0.11	0.28	0.21	0.24	0.15	0.34	0.13
SwinV2-T	0.10	0.09	0.14	0.15	0.10	0.22	0.20	0.20	0.20	1.63	0.07	0.29	0.13	0.32	0.22	0.11	0.16
SwinV2-S	0.14	0.06	0.17	0.18	0.16	0.26	0.21	0.25	0.27	1.29	0.08	0.31	0.09	0.29	0.14	0.19	0.21
SwinV2-B	0.10	0.10	0.18	0.14	0.13	0.24	0.17	0.24	0.20	0.57	0.09	0.16	0.13	0.11	0.11	0.21	0.04
CLIP (ViT-B/32)	0.20	0.33	0.34	0.25	0.20	0.18	0.17	0.73	1.83	0.51	0.18	0.36	0.35	0.23	0.24	0.17	0.37
CLIP (ViT-B/16)	0.11	0.27	0.12	0.12	0.10	0.17	0.10	1.08	3.69	0.71	0.62	0.18	0.17	0.22	0.13	0.10	0.15
CLIP (ViT-L/14)	0.16	0.10	0.11	0.07	0.16	0.10	0.05	0.96	18.76	1.01	0.45	0.15	0.15	0.23	0.15	0.29	0.21

(d) ImageNet-21K

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
MN3	0.04	err.	err.	0.10	0.07	0.07	0.08	7.34	17.87	err.	0.23	0.08	0.06	err.	0.04	0.04	0.05
ViT-B/16	0.08	err.	err.	0.09	0.16	0.08	0.04	0.23	0.18	err.	0.29	0.10	0.06	err.	0.06	0.05	0.05

(e) Amazon Fine Foods

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
T5	0.09	0.06	0.10	0.05	0.05	0.07	0.15	0.08	0.14	0.08	0.09	0.11	0.06	0.06	0.04	0.14	0.06
T5-large	0.09	0.06	0.07	0.06	0.05	0.02	0.18	0.07	0.23	0.09	0.28	0.07	0.05	0.05	0.07	0.13	0.05
RoBERTa	0.12	0.08	0.06	0.14	0.14	0.07	0.05	0.06	0.12	0.15	0.04	0.07	0.07	0.06	0.09	0.12	0.08
RoBERTa-large	0.10	0.11	0.06	0.09	0.10	0.05	0.16	0.09	0.26	0.13	0.07	0.08	0.04	0.07	0.13	0.12	0.05

(f) DynaSent

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
T5	0.51	0.21	0.39	0.35	0.28	0.60	0.54	0.52	0.47	0.69	0.55	0.22	0.20	0.42	0.33	0.49	0.54
T5-large	0.28	0.32	0.31	0.20	0.20	0.28	0.20	0.38	0.33	0.60	0.48	0.21	0.35	0.31	0.58	0.32	0.47
RoBERTa	0.67	0.41	0.58	0.65	0.65	0.63	0.49	0.60	0.63	0.54	0.28	0.57	0.45	0.33	0.30	0.16	0.17
RoBERTa-large	0.48	0.25	0.44	0.43	0.43	0.45	0.42	0.45	0.35	0.40	0.69	0.48	0.31	0.94	0.25	0.34	0.34

(g) MNLI

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
T5	0.17	0.19	0.20	0.23	0.22	0.20	0.21	0.23	0.18	0.18	0.35	0.23	0.18	0.37	0.24	0.17	0.15
T5-large	0.27	0.16	0.13	0.35	0.34	0.30	0.40	0.34	0.38	0.21	0.21	0.21	0.14	0.12	0.23	0.23	0.26
RoBERTa	0.41	0.33	0.23	0.36	0.36	0.37	0.38	0.36	0.40	0.34	0.20	0.17	0.34	0.18	0.27	0.45	0.21
RoBERTa-large	0.13	0.13	0.18	0.12	0.12	0.13	0.16	0.13	0.17	0.19	0.21	0.28	0.16	0.19	0.26	0.27	0.16

(h) Yahoo Answers

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
T5	0.04	0.20	0.36	0.07	0.24	0.18	0.25	0.17	0.29	0.22	0.74	0.18	0.23	0.22	0.31	0.76	0.23
T5-large	0.07	0.12	0.13	0.10	0.20	0.15	0.08	0.18	0.09	0.33	0.23	0.15	0.13	0.25	0.28	0.22	0.09
RoBERTa	0.09	0.19	0.26	0.09	0.09	0.09	0.25	0.11	0.29	0.11	0.44	0.21	0.11	0.30	0.26	0.36	0.30
RoBERTa-large	0.06	0.15	0.18	0.06	0.06	0.07	0.29	0.10	0.32	0.10	0.20	0.17	0.13	1.25	0.24	0.26	0.18

Table 7.5: AUROC in % (higher is better). Methods in purple impact the model prediction, potentially degrading accuracy; methods in teal do not. Improvements from the uncalibrated model are colored in blue and degradations in orange.

(a) CIFAR-10

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
ResNet-50	92.09	91.79	91.30	92.01	91.98	92.71	92.76	92.71	92.76	90.94	92.09	92.29	91.87	75.63	85.53	75.78	84.69
ResNet-110	92.26	92.25	91.53	92.19	92.18	92.18	92.31	92.16	92.30	91.33	92.26	92.40	92.20	73.44	85.01	74.46	84.56
WRN	91.17	91.17	90.36	91.21	91.19	91.80	92.32	91.82	92.31	90.45	91.17	92.28	91.18	75.99	86.62	73.91	85.38
DenseNet	90.46	90.15	89.68	90.48	90.45	90.84	91.40	90.84	91.39	89.98	90.46	91.46	90.12	77.07	87.07	74.61	83.07
CLIP (ViT-B/32)	89.85	89.72	89.73	89.97	89.97	90.73	91.07	90.95	91.14	90.58	89.85	90.28	89.66	88.86	89.41	88.01	88.76
CLIP (ViT-B/16)	91.00	90.96	90.72	91.10	91.10	91.79	91.99	91.88	92.06	91.75	91.00	90.69	90.83	89.36	90.57	89.12	89.97
CLIP (ViT-L/14)	93.22	93.15	94.32	93.32	93.31	93.69	93.78	93.79	93.87	93.17	93.22	92.78	93.05	87.29	91.73	88.65	91.33

(b) CIFAR-100

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
ResNet-50	85.80	85.78	85.03	85.73	85.62	85.92	86.50	85.91	86.51	85.49	85.80	87.19	85.69	81.21	85.31	81.90	83.90
ResNet-110	85.03	84.99	83.76	84.94	84.85	84.82	85.34	84.84	85.33	84.57	85.03	86.32	84.97	80.54	84.47	81.40	82.93
WRN	87.59	87.52	87.02	87.59	87.48	87.83	88.17	87.85	88.14	87.45	87.59	88.65	87.46	83.59	86.46	83.77	85.69
DenseNet	86.17	86.02	85.64	86.12	86.00	86.60	86.79	86.59	86.79	86.01	86.17	87.09	86.11	83.02	85.42	83.47	84.86
CLIP (ViT-B/32)	83.06	83.14	82.99	83.74	83.70	83.95	85.16	83.99	85.02	84.17	83.06	84.61	82.95	85.59	83.00	85.61	82.82
CLIP (ViT-B/16)	82.57	82.55	82.51	83.65	83.66	83.95	85.29	84.04	85.76	84.18	82.57	84.45	82.51	85.76	82.46	85.62	82.24
CLIP (ViT-L/14)	84.26	84.22	84.14	85.51	85.50	86.56	87.57	86.62	87.50	85.64	84.26	86.94	84.11	86.92	84.12	86.67	84.07

(c) ImageNet

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
ResNet-18	85.73	85.70	85.37	85.64	85.65	85.63	85.88	85.30	85.39	84.39	85.73	86.12	85.69	83.19	85.40	83.73	85.03
ResNet-34	86.18	86.19	85.78	86.11	86.10	86.25	86.38	85.86	85.99	84.64	86.18	86.41	86.14	82.26	85.77	83.03	85.77
ResNet-50	80.53	80.54	80.12	85.93	85.69	85.60	85.57	85.58	85.57	82.34	80.53	86.91	80.49	85.27	80.52	83.63	79.96
ResNet-101	84.18	84.20	83.57	85.96	85.71	85.38	85.56	85.34	85.53	82.38	84.18	87.09	84.18	82.48	84.11	81.16	83.80
EffNet-B7	84.92	84.84	84.10	86.61	86.34	85.18	85.51	85.18	85.52	81.91	84.92	87.14	84.87	81.57	84.94	80.38	84.26
EffNetV2-S	85.77	85.87	85.29	87.02	86.86	85.30	85.67	85.28	85.68	82.55	85.77	87.42	85.72	82.44	85.75	80.86	84.80
EffNetV2-M	82.36	82.36	81.58	85.26	84.92	83.66	84.19	83.64	84.23	80.51	82.36	86.51	82.32	81.24	82.23	79.45	81.71
EffNetV2-L	84.63	84.58	83.96	86.33	86.05	85.77	85.94	85.75	85.87	82.27	84.63	86.70	84.55	81.78	84.56	80.56	84.08
ConvNeXt-T	82.35	82.31	81.84	85.47	85.18	85.60	85.57	85.59	85.59	81.93	82.35	86.97	82.29	82.58	82.30	81.44	81.88
ConvNeXt-S	82.29	82.36	81.87	85.27	84.88	84.81	85.01	84.81	85.01	81.22	82.29	86.98	82.26	81.29	82.20	79.78	81.86
ConvNeXt-B	82.27	82.27	81.70	85.13	84.75	84.40	84.88	84.40	84.90	80.87	82.27	87.01	82.22	81.79	82.25	79.89	81.69
ConvNeXt-L	82.35	82.35	81.42	84.81	84.38	84.04	84.58	84.04	84.54	80.24	82.35	86.79	82.32	80.49	82.23	79.15	81.75
ViT-B/32	85.57	85.60	85.10	86.31	86.13	85.95	85.98	85.93	85.98	83.56	85.57	87.16	85.54	83.39	85.55	82.69	85.11
ViT-B/16	85.52	85.55	84.92	86.32	86.12	85.36	85.53	85.34	85.56	81.82	85.52	87.19	85.50	81.56	85.39	81.21	85.09
ViT-L/32	85.42	85.45	84.78	85.93	85.73	85.19	85.29	85.20	85.30	82.07	85.42	87.25	85.41	81.51	85.40	81.42	85.09
ViT-L/16	85.85	85.83	84.63	86.16	86.00	84.33	84.65	84.32	84.64	80.96	85.85	86.97	85.83	79.76	85.66	80.08	84.85
ViT-H/14	87.28	87.24	86.60	87.53	87.34	86.74	86.71	86.77	86.78	82.15	87.28	86.65	87.22	79.33	86.54	80.08	85.32
Swin-T	85.68	85.71	85.04	86.50	86.34	85.79	85.86	85.80	85.86	83.14	85.68	87.10	85.66	82.15	85.64	81.39	85.16
Swin-S	85.37	85.36	84.75	85.99	85.78	84.99	85.20	85.00	85.20	80.92	85.37	86.92	85.35	80.25	85.32	80.05	84.80
Swin-B	84.11	84.26	83.38	85.26	84.91	83.93	84.18	83.92	84.20	79.78	84.11	86.55	84.19	78.94	84.24	79.09	83.09
SwinV2-T	85.80	85.79	85.10	86.74	86.56	85.82	86.04	85.80	86.02	83.06	85.80	87.29	85.78	82.58	85.76	81.47	85.26
SwinV2-S	85.75	85.75	85.03	86.61	86.39	85.19	85.51	85.19	85.50	81.77	85.75	87.18	85.74	80.54	85.75	80.41	84.73
SwinV2-B	85.15	85.13	84.19	86.07	85.82	84.43	84.70	84.40	84.65	80.31	85.15	86.99	85.16	79.57	84.98	79.42	83.86
CLIP (ViT-B/32)	80.56	80.56	80.24	80.59	80.57	81.73	83.21	77.72	77.92	81.14	80.56	81.43	80.52	81.85	80.53	82.14	80.18
CLIP (ViT-B/16)	81.12	81.08	80.96	81.16	81.15	82.28	83.64	78.10	84.57	81.52	81.12	82.17	81.10	82.33	81.07	82.42	80.76
CLIP (ViT-L/14)	82.98	82.95	82.50	82.90	82.86	83.66	85.07	80.09	79.86	82.75	82.98	83.29	82.93	82.85	82.80	82.87	82.36

(d) ImageNet-21K

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
MN3	68.79	err.	err.	67.80	65.89	80.00	81.00	61.24	51.94	err.	68.79	79.62	68.77	err.	68.77	90.86	68.40
ViT-B/16	72.99	err.	err.	74.36	73.17	79.78	81.42	76.29	78.92	err.	72.99	79.66	73.10	err.	73.20	90.27	71.95

(e) Amazon Fine Foods

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
T5	85.04	84.99	84.32	85.11	85.11	87.84	87.99	88.09	88.02	87.46	85.04	87.03	85.04	85.53	83.47	80.53	79.41
T5-large	81.33	81.34	80.52	80.75	80.74	85.84	87.94	87.47	87.80	87.27	81.33	85.15	81.38	83.35	78.27	77.53	74.25
RoBERTa	83.52	83.50	80.50	83.34	83.34	84.96	86.75	86.44	86.56	86.27	83.51	83.93	83.51	81.62	75.17	75.80	73.35
RoBERTa-large	87.88	87.67	82.18	87.99	87.99	88.04	88.25	88.70	88.33	88.40	87.88	87.34	87.88	81.52	74.92	75.63	75.30

(f) DynaSent

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
T5	78.01	77.80	77.51	78.12	78.12	78.11	78.34	78.31	78.31	78.20	78.01	78.57	77.91	76.88	76.81	74.14	76.04
T5-large	77.61	77.69	76.63	77.81	77.81	77.82	79.18	78.87	79.35	78.83	77.61	79.20	77.61	74.72	74.34	69.26	73.32
RoBERTa	75.16	75.15	72.77	75.30	75.30	75.23	75.47	75.60	76.05	75.31	75.16	76.10	75.06	64.47	66.97	59.47	68.49
RoBERTa-large	76.18	75.83	72.80	76.34	76.34	76.25	76.52	76.05	76.12	75.67	76.18	76.13	76.06	61.33	66.46	59.53	68.27

(g) MNLI

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
T5	84.44	84.39	83.31	84.51	84.51	84.55	85.01	85.07	85.15	84.59	84.44	84.89	84.37	79.85	80.50	70.97	78.39
T5-large	82.01	81.83	79.68	82.10	82.10	82.08	81.99	82.26	82.30	82.14	82.01	82.11	81.92	66.48	69.68	59.63	72.69
RoBERTa	82.36	82.33	79.46	82.50	82.50	82.48	82.83	82.82	82.89	82.78	82.36	82.99	82.32	67.99	72.67	60.25	73.19
RoBERTa-large	83.58	83.49	79.43	83.63	83.63	83.58	83.78	83.63	83.75	83.49	83.58	83.58	83.50	68.68	72.21	61.39	74.25

(h) Yahoo Anwsers

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
T5	81.60	81.53	81.10	81.61	81.61	81.64	81.65	81.63	81.65	81.59	81.60	81.70	81.53	80.46	80.81	78.99	80.03
T5-large	81.09	81.00	80.45	81.08	81.07	81.16	81.17	81.22	81.29	81.07	81.09	81.15	81.05	79.24	79.89	77.41	79.06
RoBERTa	78.63	78.59	76.52	78.80	78.80	78.73	78.70	78.90	79.07	78.68	78.63	78.86	78.57	74.45	73.17	73.20	67.91
RoBERTa-large	79.13	79.08	76.92	79.41	79.41	79.30	79.46	79.37	79.81	79.34	79.13	79.35	79.04	75.00	72.69	73.67	72.96

Table 7.6: Average confidence in %. Methods in purple impact the model prediction; methods in teal do not. Overconfidence (average confidence > accuracy) is shown in violet and underconfidence (average confidence < accuracy) in brown.

(a) CIFAR-10

Model	Acc.	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
ResNet-50	94.9	96.6	94.9	94.8	95.6	95.2	95.6	95.5	95.6	95.5	96.8	95.1	95.3	94.9	94.9	94.9	94.9	95.0
ResNet-110	94.6	97.1	94.9	94.8	95.7	95.3	95.7	95.7	95.7	95.7	97.5	95.0	95.4	94.8	94.8	94.8	94.8	94.8
WRN	95.8	95.9	96.1	96.0	96.8	96.5	96.8	96.3	96.8	96.3	97.4	96.0	96.5	96.0	96.2	96.0	96.2	96.0
DenseNet	95.0	95.7	95.0	94.9	95.9	95.6	96.0	95.6	95.9	95.6	96.9	95.2	95.5	95.0	95.1	95.0	95.1	94.9
CLIP (ViT-B/32)	88.2	83.4	87.5	87.3	87.7	87.8	87.5	88.5	87.6	88.5	89.8	88.4	89.7	87.9	89.2	87.9	88.8	87.9
CLIP (ViT-B/16)	90.2	84.8	89.8	89.5	89.8	90.1	89.4	90.4	89.5	90.5	91.7	90.9	91.9	90.2	91.3	90.2	90.8	90.2
CLIP (ViT-L/14)	95.3	90.4	95.0	88.7	94.9	95.0	94.5	94.8	94.6	94.8	95.7	96.0	96.2	95.2	95.9	95.2	95.7	95.2

(b) CIFAR-100

Model	Acc.	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
ResNet-50	76.7	82.9	77.2	77.0	80.9	76.9	81.4	77.5	81.4	77.4	80.3	78.0	81.5	76.9	74.0	76.9	75.6	76.8
ResNet-110	75.0	82.8	75.5	75.3	79.6	75.5	80.0	76.2	80.0	76.2	79.3	76.9	80.6	75.2	71.4	75.2	73.0	75.3
WRN	79.6	83.0	79.5	79.3	83.0	79.3	83.4	79.7	83.3	79.7	81.1	79.6	83.2	79.3	75.5	79.3	77.3	79.3
DenseNet	76.3	81.0	76.2	76.0	79.6	76.0	80.1	76.6	80.1	76.6	78.7	75.6	79.9	75.8	72.6	75.8	74.0	76.0
CLIP (ViT-B/32)	62.3	52.8	60.7	60.1	63.3	62.3	59.0	63.9	58.3	63.5	60.7	62.9	65.2	62.3	57.0	62.3	57.8	62.3
CLIP (ViT-B/16)	66.7	56.0	64.5	63.9	66.9	67.1	62.7	68.4	62.0	67.9	64.0	66.5	68.5	66.9	61.0	66.9	62.1	67.0
CLIP (ViT-L/14)	76.0	65.0	73.9	73.4	76.1	76.0	74.1	78.6	73.4	78.4	73.5	74.6	78.5	76.0	72.8	76.0	73.4	75.9

(c) ImageNet

Model	Acc.	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
ResNet-18	69.8	72.0	69.6	69.2	69.4	69.7	70.3	70.8	71.4	70.7	65.5	69.4	73.1	69.8	65.3	69.8	67.8	69.8
ResNet-34	73.2	76.8	73.4	72.9	73.5	73.1	74.4	74.3	75.2	74.2	68.3	73.2	76.8	73.3	70.3	73.3	72.4	73.3
ResNet-50	80.8	39.7	78.5	77.9	84.0	82.2	84.2	80.2	84.2	80.2	69.3	81.2	81.6	80.9	71.1	80.9	74.7	80.9
ResNet-101	81.9	68.3	81.7	81.4	85.5	83.4	86.0	83.1	86.0	83.1	72.5	82.2	84.5	82.0	78.2	82.0	80.6	82.1
EffNet-B7	84.2	71.6	84.2	84.0	87.8	85.7	88.3	85.5	88.3	85.5	75.0	84.2	87.0	84.0	82.2	84.1	84.0	84.1
EffNetV2-S	84.3	67.3	84.2	84.0	87.9	85.6	88.2	85.3	88.2	85.2	76.3	84.7	86.8	84.2	81.2	84.2	83.5	84.2
EffNetV2-M	85.1	60.2	84.9	84.7	88.8	86.6	89.1	85.9	89.1	85.8	76.8	84.9	87.7	85.1	81.8	85.2	84.4	85.1
EffNetV2-L	85.8	77.3	85.7	85.4	88.6	86.9	88.9	86.7	88.9	86.6	76.4	85.8	88.1	85.6	84.2	85.6	85.9	85.7
ConvNeXt-T	82.5	65.6	82.0	81.7	85.6	84.0	85.9	83.3	85.9	83.2	73.5	81.9	84.7	82.5	78.9	82.5	81.2	82.5
ConvNeXt-S	83.6	66.0	83.5	83.2	87.4	85.3	87.8	84.7	87.8	84.7	74.9	83.1	86.3	83.6	80.7	83.6	82.9	83.6
ConvNeXt-B	84.0	65.3	83.9	83.7	87.8	85.6	88.2	85.0	88.2	85.0	74.6	84.0	86.7	84.0	81.2	84.1	83.4	84.1
ConvNeXt-L	84.4	71.9	84.5	84.3	88.4	86.2	88.8	85.8	88.8	85.8	76.6	84.5	87.4	84.5	82.6	84.5	84.3	84.5
ViT-B/32	75.9	69.6	75.9	75.6	79.9	77.3	80.5	77.4	80.4	77.3	69.3	75.7	78.9	75.9	71.0	75.9	73.7	76.0
ViT-B/16	81.0	75.5	81.2	81.0	84.8	82.6	85.3	82.8	85.3	82.7	73.8	81.2	84.0	81.0	78.8	81.0	80.5	81.1
ViT-L/32	77.0	74.2	77.3	77.2	81.6	78.8	82.2	79.0	82.2	79.0	71.7	77.0	80.8	76.9	73.9	76.9	76.0	76.9
ViT-L/16	79.6	78.8	80.1	80.0	84.5	81.7	85.1	82.1	85.1	82.1	72.9	80.1	83.6	79.7	78.3	79.7	79.8	79.7
ViT-H/14	88.6	89.0	88.6	88.4	90.4	89.3	90.5	89.5	90.5	89.5	80.8	88.4	90.8	88.6	88.7	88.6	89.3	88.7
Swin-T	81.5	74.7	81.3	81.1	84.5	82.6	85.0	82.7	85.0	82.6	73.7	81.3	84.0	81.5	78.8	81.5	80.9	81.4
Swin-S	83.2	79.9	83.3	83.1	86.8	84.6	87.3	84.7	87.3	84.7	75.3	83.7	86.1	83.1	81.7	83.1	83.1	83.2
Swin-B	83.6	79.7	83.8	83.6	87.5	85.4	88.0	85.6	88.0	85.5	75.8	83.6	86.7	83.5	82.8	83.5	84.0	83.5
SwinV2-T	82.0	73.7	82.1	81.9	85.6	83.4	86.0	83.4	86.0	83.4	73.4	82.2	84.7	82.2	79.4	82.2	81.5	82.2
SwinV2-S	83.7	77.7	83.8	83.7	87.5	85.2	88.0	85.4	88.0	85.3	75.3	84.1	86.7	83.7	82.3	83.7	83.8	83.7
SwinV2-B	84.1	78.9	84.3	84.1	87.9	85.7	88.4	85.8	88.4	85.8	76.7	84.5	87.1	84.2	83.1	84.1	84.3	84.2
CLIP (ViT-B/32)	57.3	57.3	57.1	56.7	57.9	57.5	59.3	60.1	67.1	70.7	55.4	57.1	61.8	57.2	52.7	57.2	55.0	57.2
CLIP (ViT-B/16)	62.9	62.6	62.5	62.1	63.4	63.2	64.6	65.6	69.9	66.2	60.2	62.5	67.4	63.0	59.4	63.0	61.4	63.0
CLIP (ViT-L/14)	70.1	72.1	70.0	69.8	70.9	70.3	73.0	72.8	76.9	66.6	65.7	70.5	74.6	70.1	67.3	70.1	69.2	69.9

(d) ImageNet-21K

Model	Acc.	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
MN3	15.4	27.7	err.	err.	24.0	17.4	35.5	33.3	71.9	81.2	err.	15.1	35.1	15.3	err.	15.3	15.7	15.3
ViT-B/16	19.2	21.4	err.	err.	25.5	21.9	43.4	41.3	48.7	41.5	err.	19.3	42.8	19.2	err.	19.1	20.6	19.2

(e) Amazon Fine Foods

Model	Acc.	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
T5	89.8	95.0	89.8	89.8	90.2	90.3	90.8	91.5	90.7	91.6	96.0	89.5	90.5	89.8	90.4	89.8	90.2	89.8
T5-large	91.6	97.3	91.6	91.6	92.0	91.9	92.3	93.3	92.3	93.3	97.8	91.6	92.1	91.6	92.1	91.6	91.9	91.6
RoBERTa	90.0	97.8	90.0	90.0	92.2	92.2	90.6	91.7	90.7	91.7	98.0	90.0	90.5	90.0	90.4	90.0	90.2	90.0
RoBERTa-large	91.4	98.2	91.5	91.5	93.9	93.9	92.3	93.3	92.3	93.3	98.3	91.5	92.1	91.5	91.9	91.5	91.7	91.4

(f) DynaSent

Model	Acc.	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
T5	78.8	86.8	78.5	78.5	78.9	78.8	83.6	80.5	83.4	80.6	89.7	78.5	78.9	78.3	78.7	78.3	78.4	78.2
T5-large	82.2	91.8	81.8	81.7	85.0	85.0	89.5	83.9	89.3	83.9	94.1	82.2	82.2	81.6	82.0	81.6	81.8	81.6
RoBERTa	77.7	95.1	77.9	77.8	90.9	90.9	93.5	85.4	92.7	84.1	96.1	78.3	78.4	77.8	78.0	77.7	78.0	77.8
RoBERTa-large	81.2	96.0	81.2	81.2	92.1	92.1	94.6	86.7	94.0	85.9	96.9	80.5	81.7	81.0	81.4	80.9	81.3	81.0

(g) MNLI

Model	Acc.	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
T5	88.4	94.8	88.5	88.5	89.4	89.4	91.6	90.2	91.6	90.3	96.1	89.0	88.7	88.5	88.6	88.5	88.4	88.5
T5-large	90.1	97.6	90.0	89.9	94.5	94.5	95.7	91.7	95.5	91.7	98.3	90.2	90.2	89.9	90.1	89.9	90.0	89.9
RoBERTa	86.4	96.7	86.1	86.1	92.9	92.9	94.2	88.4	93.8	88.5	97.5	86.1	86.5	86.1	86.2	86.0	86.1	86.1
RoBERTa-large	88.9	97.1	88.8	88.7	93.8	93.8	95.0	90.6	94.7	90.7	97.7	88.9	89.1	88.7	88.9	88.7	88.8	88.7

(h) Yahoo Answers

Model	Acc.	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
T5	75.3	82.0	75.6	75.4	75.1	74.6	78.0	76.1	78.0	76.0	82.9	74.8	76.5	75.4	75.4	75.4	75.0	75.4
T5-large	75.6	84.6	75.8	75.6	75.4	74.8	80.3	76.3	80.5	76.4	85.8	75.9	77.0	75.6	75.8	75.6	75.3	75.6
RoBERTa	72.4	91.9	72.6	72.5	84.4	84.4	88.6	74.6	88.6	74.7	92.8	72.0	74.4	72.3	72.7	72.3	72.4	72.3
RoBERTa-large	72.8	92.5	73.1	73.0	85.6	85.6	89.5	75.6	89.3	76.1	93.2	72.6	74.8	72.9	73.2	72.9	73.0	73.0

Table 7.7: Accuracy in % (higher is better). Methods in purple impact the model prediction, potentially degrading accuracy; methods in teal do not. Because classifiers can be well calibrated when not accurate (by having low accuracy and low confidence), it is important to monitor the accuracy or having methods that preserve the accuracy by design.

(a) CIFAR-10

Model	Uncal.	IRM	I-Max	TS	TS_{TVA}	VS	$VS_{reg,TVA}$	DC	$DC_{reg,TVA}$	Beta	$Beta_{TVA}$	Iso	Iso_{TVA}	BBQ	BBQ_{TVA}	HB	HB_{TVA}
ResNet-50	94.89	**94.92**	94.89	94.89	94.89	94.84	94.83	94.84	94.83	94.84	94.89	94.83	94.89	94.72	94.89	94.70	94.89
ResNet-110	**94.55**	94.53	94.51	**94.55**	**94.55**	94.50	94.52	94.51	94.52	**94.55**	**94.55**	94.42	**94.55**	94.40	**94.55**	94.32	**94.55**
WRN	95.79	**95.80**	95.76	95.79	95.79	95.68	95.73	95.68	95.74	**95.80**	95.79	95.63	95.79	95.65	95.79	95.59	95.79
DenseNet	94.99	94.98	94.97	94.99	94.99	**95.05**	**95.05**	**95.05**	**95.05**	94.99	94.99	94.86	94.99	94.74	94.99	94.80	94.99
CLIP (ViT-B/32)	88.17	88.16	87.92	88.17	88.17	90.29	90.28	**90.38**	90.33	90.18	88.17	90.36	88.17	90.29	88.17	90.18	88.17
CLIP (ViT-B/16)	90.23	90.16	90.01	90.23	90.23	92.33	92.30	**92.40**	92.33	92.10	90.23	92.21	90.23	92.19	90.23	91.90	90.23
CLIP (ViT-L/14)	95.28	95.22	88.91	95.28	95.28	96.48	**96.50**	96.49	96.48	96.31	95.28	96.31	95.28	96.39	95.28	96.25	95.28

(b) CIFAR-100

Model	Uncal.	IRM	I-Max	TS	TS_{TVA}	VS	$VS_{reg,TVA}$	DC	$DC_{reg,TVA}$	Beta	$Beta_{TVA}$	Iso	Iso_{TVA}	BBQ	BBQ_{TVA}	HB	HB_{TVA}
ResNet-50	76.71	76.64	**76.72**	76.71	76.71	76.63	76.50	76.65	76.50	76.67	76.71	76.24	76.71	76.01	76.71	74.89	76.71
ResNet-110	**75.00**	**75.00**	74.91	**75.00**	**75.00**	**75.00**	74.94	**75.00**	74.94	74.96	**75.00**	74.66	**75.00**	74.14	**75.00**	73.12	**75.00**
WRN	**79.57**	79.52	79.42	**79.57**	**79.57**	79.36	79.34	79.36	79.36	79.51	**79.57**	79.08	**79.57**	78.57	**79.57**	77.35	**79.57**
DenseNet	76.26	76.31	76.21	76.26	76.26	76.31	76.31	**76.33**	76.31	76.30	76.26	76.07	76.26	75.48	76.26	74.42	76.26
CLIP (ViT-B/32)	62.33	62.18	61.60	62.33	62.33	**67.77**	67.33	66.31	63.95	66.40	62.33	66.48	62.33	63.85	62.33	64.06	62.33
CLIP (ViT-B/16)	66.66	66.62	66.10	66.66	66.66	**71.35**	71.02	70.18	68.48	70.04	66.66	70.21	66.66	67.38	66.66	67.55	66.66
CLIP (ViT-L/14)	75.96	75.87	75.67	75.96	75.96	**80.09**	79.87	79.96	79.02	79.38	75.96	79.52	75.96	77.29	75.96	77.05	75.96

(c) ImageNet

Model	Uncal.	IRM	I-Max	TS	TS_{TVA}	VS	$VS_{reg,TVA}$	DC	$DC_{reg,TVA}$	Beta	$Beta_{TVA}$	Iso	Iso_{TVA}	BBQ	BBQ_{TVA}	HB	HB_{TVA}
ResNet-18	69.77	69.76	69.37	69.77	69.77	**69.81**	69.21	68.12	67.60	69.72	69.77	69.24	69.77	68.34	69.77	66.65	69.77
ResNet-34	**73.23**	73.19	72.77	**73.23**	**73.23**	73.19	72.70	71.96	71.53	73.10	**73.23**	72.81	**73.23**	72.21	**73.23**	70.46	**73.23**
ResNet-50	80.85	80.80	80.26	80.85	80.85	80.93	80.81	**80.94**	80.81	80.60	80.85	80.47	80.85	78.13	80.85	78.49	80.85
ResNet-101	**81.86**	81.83	81.54	**81.86**	**81.86**	81.77	81.65	81.80	81.66	81.74	**81.86**	81.44	**81.86**	80.82	**81.86**	79.80	**81.86**
EffNet-B7	84.16	84.18	84.06	84.16	84.16	**84.45**	84.31	**84.45**	84.30	84.27	84.16	84.09	84.16	83.71	84.16	82.72	84.16
EffNetV2-S	84.27	84.19	83.99	84.27	84.27	84.33	84.23	**84.34**	84.22	84.26	84.27	83.88	84.27	83.52	84.27	82.55	84.27
EffNetV2-M	85.06	85.05	84.90	85.06	85.06	**85.28**	85.19	**85.28**	85.17	85.10	85.06	84.87	85.06	84.19	85.07	83.72	85.06
EffNetV2-L	85.80	85.78	85.60	85.80	85.80	85.88	85.83	**85.89**	85.87	**85.89**	85.80	85.58	85.80	85.23	85.80	84.25	85.80
ConvNeXt-T	**82.50**	82.49	82.18	**82.50**	**82.50**	82.44	82.29	82.45	82.28	82.45	**82.50**	82.10	**82.50**	81.51	**82.50**	80.37	**82.50**
ConvNeXt-S	**83.65**	83.59	83.36	**83.65**	**83.65**	83.63	83.55	83.63	83.55	83.64	**83.65**	83.28	**83.65**	82.89	**83.65**	81.84	**83.65**
ConvNeXt-B	84.04	84.01	83.78	84.04	84.04	84.09	83.98	**84.10**	83.96	84.04	84.04	83.68	84.04	83.22	84.04	82.34	84.04
ConvNeXt-L	84.38	84.37	84.23	84.38	84.38	84.41	84.32	84.41	84.34	**84.44**	84.38	84.12	84.38	83.98	84.38	82.92	84.38
ViT-B/32	**75.95**	75.91	75.69	**75.95**	**75.95**	75.81	75.65	75.83	75.66	75.85	**75.95**	75.36	**75.95**	74.59	**75.95**	73.13	**75.95**
ViT-B/16	**81.04**	81.01	80.90	**81.04**	**81.04**	81.00	80.88	81.01	80.87	80.96	**81.04**	80.63	**81.04**	80.38	**81.04**	79.06	**81.04**
ViT-L/32	**76.96**	76.94	76.84	**76.96**	**76.96**	76.79	76.73	76.79	76.72	76.88	**76.96**	76.37	**76.96**	76.04	**76.96**	74.55	**76.96**
ViT-L/16	79.64	79.64	79.59	79.64	79.64	79.80	79.67	**79.81**	79.68	79.66	79.64	79.47	79.64	79.20	79.64	77.82	79.64
ViT-H/14	88.62	88.61	88.48	88.62	88.62	88.62	88.50	88.59	88.46	**88.63**	88.62	88.34	88.62	88.33	88.62	87.24	88.62
Swin-T	81.49	81.45	81.28	81.49	81.49	**81.55**	81.42	**81.55**	81.42	81.44	81.49	81.07	81.49	80.77	81.49	79.43	81.49
Swin-S	**83.21**	83.20	83.04	**83.21**	**83.21**	83.13	83.02	83.13	83.03	**83.21**	**83.21**	82.79	**83.21**	82.74	**83.21**	81.40	**83.21**
Swin-B	83.60	83.57	83.53	83.60	83.60	83.75	83.61	**83.76**	83.59	83.63	83.60	83.39	83.60	83.40	83.60	82.16	83.60
SwinV2-T	82.02	82.01	81.83	82.02	82.02	82.12	81.98	**82.13**	82.00	82.05	82.02	81.66	82.02	81.27	82.02	80.08	82.02
SwinV2-S	83.74	83.73	83.64	83.74	83.74	**83.81**	83.71	83.80	83.72	83.72	83.74	83.56	83.74	83.34	83.74	82.31	83.74
SwinV2-B	84.10	84.12	84.03	84.10	84.10	84.14	84.06	**84.16**	84.08	**84.16**	84.10	83.81	84.10	83.79	84.10	82.53	84.10
CLIP (ViT-B/32)	57.34	57.32	56.94	57.34	57.34	**59.80**	59.57	31.10	0.18	58.93	57.34	59.54	57.34	57.25	57.34	56.05	57.34
CLIP (ViT-B/16)	62.89	62.91	62.43	62.89	62.89	**65.61**	65.28	35.86	0.18	64.62	62.89	65.07	62.89	63.24	62.89	62.18	62.89
CLIP (ViT-L/14)	70.15	70.14	69.89	70.15	70.15	**72.66**	72.25	50.88	0.17	71.57	70.15	72.30	70.15	70.59	70.15	69.36	70.15

(d) ImageNet-21K

Model	Uncal.	IRM	I-Max	TS	TS_{TVA}	VS	$VS_{reg,TVA}$	DC	$DC_{reg,TVA}$	Beta	$Beta_{TVA}$	Iso	Iso_{TVA}	BBQ	BBQ_{TVA}	HB	HB_{TVA}
MN3	15.36	err.	err.	15.36	15.36	**37.26**	34.95	13.07	0.02	err.	15.36	35.62	15.36	err.	15.36	21.17	15.36
ViT-B/16	19.18	err.	err.	19.18	19.18	**45.42**	42.70	40.54	40.70	err.	19.18	43.72	19.18	err.	19.18	28.46	19.18

(e) Amazon Fine Foods

Model	Uncal.	IRM	I-Max	TS	TS_{TVA}	VS	$VS_{reg,TVA}$	DC	$DC_{reg,TVA}$	Beta	$Beta_{TVA}$	Iso	Iso_{TVA}	BBQ	BBQ_{TVA}	HB	HB_{TVA}
T5	89.81	89.83	89.83	89.81	89.81	90.04	90.18	90.31	90.34	90.59	89.81	90.60	89.81	90.58	89.81	**90.64**	89.81
T5-large	91.57	91.58	91.61	91.57	91.57	91.67	91.82	92.04	91.98	92.10	91.57	**92.14**	91.57	**92.14**	91.57	92.11	91.57
RoBERTa	89.95	89.95	89.98	89.95	89.95	89.99	90.15	90.29	90.21	90.56	89.95	**90.59**	89.95	**90.59**	89.95	90.54	89.95
RoBERTa-large	91.42	91.50	91.48	91.42	91.42	91.42	91.64	91.74	91.69	91.94	91.42	**91.95**	91.42	91.90	91.39	91.91	91.42

(f) DynaSent

Model	Uncal.	IRM	I-Max	TS	TS_{TVA}	VS	$VS_{reg,TVA}$	DC	$DC_{reg,TVA}$	Beta	$Beta_{TVA}$	Iso	Iso_{TVA}	BBQ	BBQ_{TVA}	HB	HB_{TVA}
T5	78.83	**78.88**	78.82	78.83	78.83	78.84	78.83	78.82	78.86	78.81	78.83	78.70	78.83	78.71	78.83	78.73	78.83
T5-large	82.20	82.16	82.19	82.20	82.20	82.20	82.23	82.32	82.25	**82.35**	82.20	82.31	82.20	82.19	82.20	82.29	82.20
RoBERTa	77.72	77.67	77.71	77.72	77.72	77.72	77.74	77.76	77.70	77.77	77.72	77.75	77.72	77.86	77.72	**77.97**	77.72
RoBERTa-large	81.19	81.29	81.33	81.19	81.19	81.19	81.19	81.24	81.44	81.28	81.19	81.50	81.19	81.51	81.19	**81.66**	81.19

(g) MNLI

Model	Uncal.	IRM	I-Max	TS	TS_{TVA}	VS	$VS_{reg,TVA}$	DC	$DC_{reg,TVA}$	Beta	$Beta_{TVA}$	Iso	Iso_{TVA}	BBQ	BBQ_{TVA}	HB	HB_{TVA}
T5	88.35	88.35	88.38	88.35	88.35	88.36	88.39	88.39	88.39	**88.41**	88.35	88.34	88.35	88.28	88.35	88.37	88.35
T5-large	90.07	90.10	90.14	90.07	90.07	90.07	90.15	90.20	**90.21**	90.07	90.07	90.15	90.07	90.06	90.07	90.16	90.07
RoBERTa	86.41	86.40	86.43	86.41	86.41	86.41	86.42	86.45	**86.48**	86.44	86.41	86.37	86.41	86.41	86.41	86.38	86.41
RoBERTa-large	88.89	88.90	88.92	88.89	88.89	88.90	88.96	88.98	**89.02**	88.90	88.89	88.95	88.89	88.86	88.89	88.92	88.89

(h) Yahoo Answers

Model	Uncal.	IRM	I-Max	TS	TS_{TVA}	VS	$VS_{reg,TVA}$	DC	$DC_{reg,TVA}$	Beta	$Beta_{TVA}$	Iso	Iso_{TVA}	BBQ	BBQ_{TVA}	HB	HB_{TVA}
T5	75.35	75.35	75.26	75.35	75.35	75.34	75.34	**75.36**	75.35	75.24	75.35	75.30	75.35	75.22	75.35	75.25	75.35
T5-large	75.57	75.60	75.56	75.57	75.57	75.59	75.55	75.69	75.66	75.53	75.57	**75.70**	75.57	75.62	75.57	75.67	75.57
RoBERTa	72.38	72.37	72.35	72.38	72.38	72.39	72.43	72.74	**73.05**	72.72	72.38	73.00	72.38	72.87	72.38	72.91	72.38
RoBERTa-large	72.84	72.83	72.81	72.84	72.84	72.83	72.88	73.01	73.40	72.98	72.84	**73.48**	72.84	73.29	72.84	73.38	72.84

Table 7.8: ECE with 15 equal mass bins in % (lower is better). Methods in purple impact the model prediction, potentially degrading accuracy; methods in teal do not.

(a) CIFAR-10

Model	Uncal.	IRM	I-Max	TS	TS_{TVA}	VS	$VS_{ng,TVA}$	DC	$DC_{ng,TVA}$	Beta	$Beta_{TVA}$	Iso	Iso_{TVA}	BBQ	BBQ_{TVA}	HB	HB_{TVA}
ResNet-50	1.81	0.71	0.62	1.33	1.24	1.28	1.29	1.28	1.29	2.03	1.55	0.93	0.70	2.45	0.47	2.33	**0.46**
ResNet-110	2.58	0.48	0.51	1.78	1.72	1.74	1.74	1.72	1.75	2.91	1.56	0.99	0.43	2.59	0.34	2.71	**0.23**
WRN	1.80	0.58	0.51	1.74	1.75	1.48	1.52	1.49	1.52	1.74	2.09	0.98	0.62	2.07	0.54	2.15	**0.46**
DenseNet	2.04	0.56	0.46	2.00	2.07	1.48	1.70	1.47	1.70	2.03	2.45	0.78	0.56	2.71	0.69	2.75	**0.29**
CLIP (ViT-B/32)	4.73	1.33	1.30	0.93	0.91	2.76	1.79	2.80	1.80	1.66	1.34	1.06	1.05	2.15	1.47	2.47	**0.90**
CLIP (ViT-B/16)	5.38	1.12	1.01	0.59	**0.48**	2.91	1.91	2.87	1.87	1.21	1.80	1.05	0.87	1.63	0.91	2.41	0.63
CLIP (ViT-L/14)	4.90	0.54	0.50	0.56	0.48	1.98	1.74	1.93	1.72	0.96	0.94	0.55	0.46	0.98	0.51	0.96	**0.37**

(b) CIFAR-100

Model	Uncal.	IRM	I-Max	TS	TS_{TVA}	VS	$VS_{ng,TVA}$	DC	$DC_{ng,TVA}$	Beta	$Beta_{TVA}$	Iso	Iso_{TVA}	BBQ	BBQ_{TVA}	HB	HB_{TVA}
ResNet-50	6.53	1.55	1.26	5.00	3.54	5.15	2.31	5.14	2.30	5.54	3.90	5.60	1.44	10.10	2.08	10.97	**1.19**
ResNet-110	7.83	1.27	1.18	5.31	4.24	5.11	2.86	5.07	2.90	5.97	4.95	6.49	1.30	9.72	2.11	10.89	**1.13**
WRN	4.33	1.07	0.95	4.36	2.79	4.46	2.38	4.45	2.37	4.40	2.85	4.28	1.22	10.29	0.94	9.99	**0.83**
DenseNet	5.16	1.12	**0.86**	4.25	2.30	4.47	2.26	4.48	2.25	4.83	2.94	4.50	1.37	10.46	1.27	10.60	1.07
CLIP (ViT-B/32)	9.51	2.10	1.74	1.86	1.78	8.72	3.49	7.97	1.99	6.51	2.75	2.32	**1.25**	7.95	2.13	7.99	1.35
CLIP (ViT-B/16)	10.64	3.35	3.04	2.66	2.72	8.68	3.15	8.20	1.72	7.03	2.55	2.68	1.77	7.14	2.12	7.13	**1.52**
CLIP (ViT-L/14)	10.96	3.14	2.94	2.47	2.53	6.01	2.05	6.57	1.74	6.52	1.93	2.39	1.64	6.65	1.66	6.48	**1.44**

(c) ImageNet

Model	Uncal.	IRM	I-Max	TS	TS_{TVA}	VS	$VS_{ng,TVA}$	DC	$DC_{ng,TVA}$	Beta	$Beta_{TVA}$	Iso	Iso_{TVA}	BBQ	BBQ_{TVA}	HB	HB_{TVA}
ResNet-18	2.59	0.78	**0.55**	1.86	1.82	1.71	2.07	3.36	3.35	4.36	1.39	3.85	0.75	9.84	0.65	9.45	0.56
ResNet-34	3.61	0.74	**0.52**	1.75	1.75	1.81	2.02	3.40	2.91	4.86	1.11	4.04	0.71	9.17	0.68	9.29	0.64
ResNet-50	41.15	2.75	2.60	3.19	1.76	3.23	1.10	3.22	1.13	11.30	2.22	1.29	0.76	8.24	0.98	5.45	**0.52**
ResNet-101	13.55	0.79	0.56	3.69	2.34	4.21	1.60	4.18	1.56	9.21	2.10	3.01	0.63	7.84	1.00	6.68	**0.46**
EffNet-B7	12.60	0.51	0.48	3.84	2.94	3.82	1.58	3.81	1.56	9.31	2.32	2.93	0.56	6.91	0.71	6.06	**0.39**
EffNetV2-S	16.92	0.63	**0.40**	4.04	3.32	3.91	1.69	3.89	1.67	7.98	2.52	2.96	0.65	7.58	0.92	6.50	0.58
EffNetV2-M	24.88	0.90	0.67	3.78	2.66	3.83	1.35	3.82	1.34	8.31	1.83	2.88	0.73	6.68	1.08	5.38	**0.54**
EffNetV2-L	8.48	0.60	0.44	2.84	1.49	3.06	0.94	3.05	0.90	9.45	1.18	2.51	0.62	6.03	0.78	5.30	**0.37**
ConvNeXt-T	16.95	1.17	0.87	3.08	1.67	3.47	1.21	3.46	1.17	8.95	1.83	2.55	0.82	7.64	0.99	6.01	**0.67**
ConvNeXt-S	17.60	0.76	0.56	3.80	2.56	4.19	1.44	4.18	1.41	8.77	2.02	3.06	0.71	7.36	0.79	6.07	**0.51**
ConvNeXt-B	18.77	0.68	**0.44**	3.81	2.67	4.09	1.44	4.07	1.44	9.44	2.18	3.03	0.73	7.48	1.04	6.04	0.58
ConvNeXt-L	12.51	0.65	0.43	4.02	2.90	4.42	1.82	4.41	1.77	7.89	1.59	3.26	0.63	7.05	0.78	6.27	**0.41**
ViT-B/32	6.37	0.71	0.61	4.10	2.49	4.64	1.90	4.62	1.84	6.53	1.75	3.58	0.71	9.24	0.74	8.30	**0.58**
ViT-B/16	5.56	0.75	**0.53**	4.19	3.18	4.27	2.09	4.25	2.07	7.24	2.39	3.38	0.70	7.68	0.93	7.12	0.57
ViT-L/32	4.13	0.85	0.75	5.30	4.20	5.37	2.67	5.37	2.64	6.19	2.80	4.42	0.72	9.18	1.03	8.64	**0.63**
ViT-L/16	5.17	1.02	**0.59**	5.92	5.20	5.28	2.74	5.26	2.69	7.36	3.59	4.10	0.76	7.82	1.29	8.32	0.60
ViT-H/14	0.61	0.56	**0.35**	1.75	0.83	1.88	1.08	1.92	1.05	8.06	0.70	2.46	0.60	3.96	0.53	4.64	0.41
Swin-T	6.82	0.77	0.49	3.10	1.82	3.43	1.33	3.43	1.27	7.70	1.67	2.94	0.71	7.52	0.90	6.87	**0.48**
Swin-S	3.57	0.59	0.52	3.92	2.98	4.17	1.84	4.17	1.82	7.88	2.33	3.29	0.59	6.78	0.83	6.97	**0.50**
Swin-B	4.65	0.60	**0.35**	4.36	3.71	4.22	2.04	4.21	2.04	7.89	2.81	3.33	0.63	6.50	0.78	6.64	0.41
SwinV2-T	8.31	0.72	**0.46**	3.62	2.21	3.92	1.60	3.90	1.59	8.66	1.83	3.07	0.71	7.97	0.67	6.84	0.52
SwinV2-S	6.06	0.64	**0.45**	4.18	3.32	4.24	1.88	4.23	1.83	8.46	2.34	3.15	0.56	7.04	0.79	6.75	0.50
SwinV2-B	5.27	0.61	0.46	4.42	3.68	4.25	1.95	4.22	1.88	7.47	2.74	3.33	0.57	6.56	0.75	6.53	**0.42**
CLIP (ViT-B/32)	1.51	1.00	**0.73**	1.62	1.56	1.42	0.81	36.01	70.52	3.59	0.84	2.25	0.98	7.88	1.04	6.66	0.79
CLIP (ViT-B/16)	1.78	1.25	0.74	1.90	1.88	1.60	0.80	34.02	66.04	4.51	1.07	2.31	0.95	8.17	0.83	6.96	**0.71**
CLIP (ViT-L/14)	2.54	1.17	**0.64**	2.03	2.04	1.79	1.32	26.06	66.39	5.93	1.50	2.39	1.10	9.03	0.88	7.87	0.84

(d) ImageNet-21K

Model	Uncal.	IRM	I-Max	TS	TS_{TVA}	VS	$VS_{ng,TVA}$	DC	$DC_{ng,TVA}$	Beta	$Beta_{TVA}$	Iso	Iso_{TVA}	BBQ	BBQ_{TVA}	HB	HB_{TVA}
MN3	12.34	err.	err.	8.69	4.43	2.51	2.38	58.84	81.16	err.	1.14	1.93	0.28	err.	0.31	5.53	**0.19**
ViT-B/16	6.54	err.	err.	9.03	6.86	2.34	1.55	8.18	3.20	err.	3.72	2.14	0.21	err.	0.43	7.90	**0.17**

(e) Amazon Fine Foods

Model	Uncal.	IRM	I-Max	TS	TS_{TVA}	VS	$VS_{ng,TVA}$	DC	$DC_{ng,TVA}$	Beta	$Beta_{TVA}$	Iso	Iso_{TVA}	BBQ	BBQ_{TVA}	HB	HB_{TVA}
T5	5.17	0.28	0.22	1.35	1.36	1.35	1.43	0.99	1.30	5.44	0.87	0.47	0.29	1.14	**0.21**	2.51	**0.21**
T5-large	5.76	0.27	0.22	1.82	1.84	1.71	1.67	1.37	1.35	5.71	1.81	0.75	0.26	1.97	**0.17**	3.18	**0.17**
RoBERTa	7.89	0.33	0.23	2.27	2.21	1.46	2.11	1.60	1.90	7.47	4.29	0.45	0.32	2.73	0.22	3.76	**0.21**
RoBERTa-large	6.82	0.34	0.18	2.53	2.45	1.47	2.13	1.29	1.93	6.35	4.21	0.56	0.28	2.81	0.24	3.68	**0.16**

(f) DynaSent

Model	Uncal.	IRM	I-Max	TS	TS_{TVA}	VS	$VS_{ng,TVA}$	DC	$DC_{ng,TVA}$	Beta	$Beta_{TVA}$	Iso	Iso_{TVA}	BBQ	BBQ_{TVA}	HB	HB_{TVA}
T5	7.92	1.76	1.33	1.60	1.60	4.78	2.12	4.58	2.20	10.85	2.83	1.46	1.89	2.00	1.51	1.58	**1.21**
T5-large	9.62	1.56	**1.02**	3.34	3.33	7.26	2.00	6.98	2.07	11.71	4.67	1.79	1.64	1.50	1.36	1.75	1.09
RoBERTa	17.34	2.38	1.59	13.14	13.14	15.80	7.68	14.96	6.35	18.34	10.36	1.69	2.36	2.45	1.27	2.15	**1.08**
RoBERTa-large	14.80	1.51	1.14	10.88	10.87	13.43	5.53	12.71	4.53	15.63	9.09	1.93	1.41	2.78	1.09	2.90	**0.61**

(g) MNLI

Model	Uncal.	IRM	I-Max	TS	TS_{TVA}	VS	$VS_{ng,TVA}$	DC	$DC_{ng,TVA}$	Beta	$Beta_{TVA}$	Iso	Iso_{TVA}	BBQ	BBQ_{TVA}	HB	HB_{TVA}
T5	6.46	0.90	0.64	1.22	1.20	3.21	1.91	3.20	1.98	7.73	2.11	0.86	0.90	1.55	0.58	1.83	**0.45**
T5-large	7.58	0.77	0.51	4.40	4.39	5.63	1.74	5.31	1.69	8.20	4.43	1.11	0.79	1.77	0.40	2.18	**0.35**
RoBERTa	10.25	0.98	0.76	6.47	6.47	7.80	2.34	7.33	2.40	11.02	6.07	1.00	1.02	2.56	0.77	3.07	**0.64**
RoBERTa-large	8.17	1.00	0.56	4.91	4.90	6.13	1.85	5.74	1.84	8.80	5.26	1.24	0.95	2.02	**0.48**	2.50	0.49

(h) Yahoo Answers

Model	Uncal.	IRM	I-Max	TS	TS_{TVA}	VS	$VS_{ng,TVA}$	DC	$DC_{ng,TVA}$	Beta	$Beta_{TVA}$	Iso	Iso_{TVA}	BBQ	BBQ_{TVA}	HB	HB_{TVA}
T5	6.64	0.83	0.90	**0.71**	0.98	2.66	0.92	2.64	0.88	7.68	1.68	1.47	0.86	3.52	0.86	4.78	0.72
T5-large	9.04	0.88	0.68	1.46	1.71	4.67	1.26	4.82	1.27	10.32	2.36	1.81	0.86	3.72	0.89	4.89	**0.65**
RoBERTa	19.53	1.06	0.79	12.02	12.00	16.26	2.18	15.85	1.70	20.13	9.36	1.94	0.99	5.15	0.72	6.48	**0.62**
RoBERTa-large	19.65	1.00	0.92	12.76	12.75	16.67	2.74	16.29	2.68	20.18	10.11	1.87	0.94	6.53	0.74	6.70	**0.60**

Table 7.9: Brier score of the predicted class in 10^{-2} (lower is better). Methods in purple impact the model prediction, potentially degrading accuracy; methods in teal do not.

(a) CIFAR-10

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
ResNet-50	3.75	3.64	3.64	3.67	3.66	3.67	3.65	3.66	3.65	3.82	3.69	3.67	**3.63**	4.05	3.76	4.00	3.78
ResNet-110	3.95	3.64	3.65	3.75	3.72	3.72	3.73	3.73	3.73	4.02	3.73	**3.62**	3.64	4.21	3.76	4.25	3.80
WRN	3.06	**3.02**	**3.02**	3.11	3.08	3.11	3.07	3.11	3.07	3.17	3.10	**3.02**	3.03	3.37	3.09	3.43	3.13
DenseNet	3.68	3.54	3.56	3.69	3.67	3.65	3.62	3.65	3.62	3.78	3.70	3.56	**3.53**	4.01	3.58	4.01	3.67
CLIP (ViT-B/32)	7.63	7.37	7.33	7.27	7.26	6.31	6.19	6.24	**6.16**	6.43	7.33	6.35	7.35	6.42	7.37	6.62	7.36
CLIP (ViT-B/16)	6.48	6.08	6.07	6.02	6.01	5.15	5.05	5.12	**5.03**	5.21	6.15	5.20	6.09	5.24	6.08	5.39	6.13
CLIP (ViT-L/14)	3.62	3.20	3.22	3.15	3.15	2.65	**2.61**	2.63	**2.61**	2.74	3.20	2.64	3.19	2.76	3.18	2.74	3.19

(b) CIFAR-100

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
ResNet-50	12.75	11.96	12.02	12.44	12.14	12.39	11.93	12.39	**11.92**	12.50	12.16	12.08	11.99	14.26	12.12	13.65	12.01
ResNet-110	13.86	**12.69**	12.74	13.29	12.94	13.27	12.82	13.26	12.82	13.37	13.11	13.07	12.70	14.88	12.80	14.24	12.75
WRN	11.05	10.72	10.74	11.05	10.85	11.03	**10.71**	11.03	10.72	11.08	10.82	10.77	10.75	12.65	10.82	12.07	10.86
DenseNet	12.49	12.09	12.10	12.33	12.16	12.30	**12.04**	12.30	**12.04**	12.47	12.16	12.11	12.07	14.06	12.13	13.41	12.11
CLIP (ViT-B/32)	17.25	16.20	16.12	15.88	15.90	15.98	**14.72**	16.03	15.09	15.92	16.28	14.97	16.23	15.55	16.31	15.36	16.25
CLIP (ViT-B/16)	17.21	16.01	15.94	15.42	15.41	15.37	**13.98**	15.44	14.06	15.49	15.87	14.39	15.85	14.69	15.92	14.52	15.88
CLIP (ViT-L/14)	14.78	13.33	13.27	12.70	12.71	11.53	**10.67**	11.69	10.90	12.59	13.17	11.00	13.16	11.56	13.18	11.38	13.15

(c) ImageNet

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
ResNet-18	13.93	13.86	**13.85**	13.94	13.93	13.92	13.92	14.48	14.53	14.50	13.87	13.89	13.87	15.56	13.90	15.02	13.93
ResNet-34	13.15	**12.97**	12.99	13.04	13.05	12.99	13.05	13.54	13.53	13.70	12.99	13.13	12.99	14.92	13.01	14.43	13.08
ResNet-50	29.79	12.25	12.33	10.95	10.88	10.98	10.92	10.98	10.92	13.14	12.07	**10.71**	12.02	11.90	12.08	11.08	12.12
ResNet-101	12.65	10.72	10.79	10.65	10.51	10.70	10.51	10.70	10.51	11.97	10.77	**10.35**	10.71	11.72	10.74	11.40	10.75
EffNet-B7	11.28	9.59	9.69	9.71	9.55	9.72	9.51	9.72	9.50	11.02	9.70	**9.41**	9.60	10.71	9.62	10.35	9.65
EffNetV2-S	12.39	9.40	9.45	9.66	9.48	9.71	9.50	9.71	9.50	10.61	9.55	**9.37**	9.43	10.64	9.46	10.37	9.50
EffNetV2-M	16.05	9.72	9.83	9.59	9.44	9.55	9.32	9.54	9.31	10.64	9.78	**9.18**	9.72	10.40	9.76	10.02	9.78
EffNetV2-L	9.80	8.99	9.08	8.90	8.82	8.93	8.83	8.93	8.84	10.28	9.01	**8.81**	9.00	9.96	9.01	9.70	9.03
ConvNeXt-T	14.02	10.87	10.96	10.39	10.33	10.40	10.33	10.40	10.32	11.79	10.89	**10.13**	10.87	11.61	10.90	11.14	10.91
ConvNeXt-S	13.62	10.30	10.35	10.16	10.02	10.16	9.95	10.15	9.95	11.32	10.36	**9.78**	10.32	11.27	10.35	10.83	10.37
ConvNeXt-B	13.86	10.14	10.20	10.05	9.89	10.01	9.79	10.01	9.79	11.36	10.20	**9.60**	10.14	11.11	10.18	10.63	10.19
ConvNeXt-L	11.58	9.88	9.99	9.92	9.76	9.99	9.70	9.99	9.71	10.97	9.92	**9.46**	9.88	10.96	9.90	10.46	9.94
ViT-B/32	12.68	12.22	12.26	12.35	12.17	12.53	12.33	12.54	12.32	13.28	12.26	**12.11**	12.24	13.72	12.26	13.40	12.30
ViT-B/16	11.02	10.65	10.71	10.89	10.72	11.01	10.84	11.01	10.83	11.98	10.73	**10.59**	10.66	12.09	10.67	11.84	10.72
ViT-L/32	12.15	**11.92**	12.02	12.35	12.11	12.49	12.21	12.49	12.21	13.20	12.02	**11.92**	11.93	13.87	11.96	13.45	12.04
ViT-L/16	11.39	11.13	11.22	11.73	11.46	11.88	11.57	11.88	11.57	12.82	11.28	11.13	**11.11**	13.21	11.15	12.71	11.22
ViT-H/14	**7.46**	7.47	7.51	7.49	**7.46**	7.57	7.59	7.57	7.59	8.79	**7.46**	7.58	7.47	8.60	7.48	8.46	7.52
Swin-T	11.09	10.59	10.64	10.64	10.53	10.71	10.64	10.71	10.63	11.68	10.64	**10.51**	10.61	11.89	10.63	11.66	10.66
Swin-S	10.14	9.98	10.04	10.23	10.06	10.32	10.12	10.32	10.13	11.40	10.06	**9.95**	9.98	11.48	10.00	11.14	10.06
Swin-B	10.18	9.90	10.00	10.21	10.05	10.22	10.10	10.23	10.10	11.42	10.01	**9.82**	9.91	11.46	9.93	11.06	10.00
SwinV2-T	11.06	10.33	10.39	10.45	10.29	10.54	10.38	10.54	10.38	11.65	10.38	**10.25**	10.34	11.69	10.36	11.43	10.37
SwinV2-S	10.05	9.61	9.67	9.91	9.72	10.01	9.78	10.01	9.77	11.13	9.70	**9.57**	9.62	11.13	9.64	10.70	9.69
SwinV2-B	10.00	9.64	9.70	9.97	9.79	10.04	9.82	10.03	9.83	11.05	9.76	**9.59**	9.64	11.21	9.66	10.78	9.73
CLIP (ViT-B/32)	17.76	17.75	17.72	17.75	17.76	17.05	**16.37**	31.17	50.78	17.52	17.74	17.24	17.76	18.05	17.79	17.38	17.81
CLIP (ViT-B/16)	16.99	16.99	16.89	16.98	16.98	16.12	**15.54**	30.53	44.73	16.74	16.97	16.22	16.98	17.14	16.99	16.54	17.03
CLIP (ViT-L/14)	14.96	14.91	14.93	14.97	14.99	14.17	**13.67**	26.10	47.35	14.95	14.91	14.32	14.92	15.39	14.93	14.87	14.99

(d) ImageNet-21K

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
MN3	14.08	err.	err.	13.35	12.71	17.17	16.26	49.94	69.66	err.	11.95	17.06	11.93	err.	11.93	**10.70**	11.95
ViT-B/16	13.51	err.	err.	13.54	13.49	18.14	17.11	20.10	18.22	err.	12.96	18.12	**12.76**	err.	12.77	12.91	12.96

(e) Amazon Fine Foods

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
T5	7.78	7.27	7.30	7.27	7.27	6.93	6.85	6.73	6.75	7.32	7.30	**6.69**	7.28	6.77	7.29	7.19	7.33
T5-large	7.02	6.34	6.36	6.40	6.40	6.11	5.94	**5.76**	5.82	6.57	6.46	5.80	6.34	5.92	6.38	6.42	6.41
RoBERTa	8.66	7.30	7.35	7.42	7.42	7.22	7.02	6.87	6.99	8.12	7.85	**6.86**	7.30	7.11	7.48	7.70	7.37
RoBERTa-large	7.43	6.15	6.23	6.30	6.29	6.18	6.01	5.89	6.00	6.96	6.85	**5.86**	6.14	6.21	6.49	6.74	6.22

(f) DynaSent

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
T5	14.68	13.96	13.96	13.89	13.89	14.15	13.88	14.09	13.87	15.40	14.02	**13.84**	13.98	14.00	14.05	14.21	13.96
T5-large	13.67	12.48	12.52	12.52	12.52	13.08	12.21	12.82	12.16	14.16	12.80	**12.14**	12.50	12.43	12.62	12.78	12.54
RoBERTa	18.98	14.96	15.09	17.03	17.03	18.16	15.53	17.66	15.15	19.39	16.62	**14.67**	14.97	15.60	15.64	15.96	15.01
RoBERTa-large	16.14	13.21	13.35	14.65	14.65	15.54	13.51	15.29	13.37	16.66	14.70	**13.09**	13.20	13.91	13.83	14.20	13.25

(g) MNLI

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
T5	8.95	8.21	8.25	8.21	8.21	8.35	8.19	8.29	8.17	9.33	8.28	**8.14**	8.22	8.28	8.29	8.65	8.24
T5-large	8.56	7.49	7.52	7.76	7.76	7.98	7.46	7.88	7.41	8.80	7.98	**7.40**	7.49	7.80	7.78	8.07	7.54
RoBERTa	11.50	9.60	9.67	10.24	10.23	10.57	9.57	10.36	9.54	11.80	10.48	**9.44**	9.60	10.06	10.03	10.48	9.60
RoBERTa-large	9.30	8.04	8.12	8.40	8.40	8.65	8.04	8.56	8.03	9.57	8.72	**7.98**	8.05	8.40	8.39	8.71	8.12

(h) Yahoo Answers

Model	Uncal.	IRM	I-Max	TS	TS_{TvA}	VS	$VS_{reg,TvA}$	DC	$DC_{reg,TvA}$	Beta	$Beta_{TvA}$	Iso	Iso_{TvA}	BBQ	BBQ_{TvA}	HB	HB_{TvA}
T5	14.70	14.18	14.21	14.14	14.15	14.22	14.14	14.21	**14.13**	14.98	14.20	14.14	14.18	14.41	14.24	14.75	14.24
T5-large	15.23	14.25	14.29	14.23	14.25	14.44	14.21	14.41	**14.12**	15.66	14.28	14.18	14.24	14.49	14.33	14.83	14.29
RoBERTa	20.95	15.94	16.12	17.57	17.56	19.20	15.92	18.84	**15.54**	21.23	17.15	15.64	15.94	16.75	16.55	17.20	15.99
RoBERTa-large	20.94	15.68	15.86	17.57	17.57	19.23	15.59	18.96	**15.30**	21.21	17.14	15.38	15.68	16.71	16.42	17.15	15.72

Bibliography

Momin Abbas, Yi Zhou, Parikshit Ram, Nathalie Baracaldo, Horst Samulowitz, Theodoros Salonidis, and Tianyi Chen. Enhancing in-context learning via linear probe calibration. *arXiv preprint arXiv:2401.12406*, 2024. 63, 65

Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U Rajendra Acharya, and others. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information fusion*, 76:243–297, 2021. Publisher: Elsevier. 22

Morayo Adedjouma, Bernard Botella, Javier Ibanez-Guzman, Kevin Mantissa, Chauk-Mean Proum, and Asma Smaoui. Defining operational design domain for autonomous systems: A domain-agnostic and risk-based approach. In *SOSE 2024 - 19th annual system of systems engineering conference*, Tacoma, United States, June 2024. URL https://hal.science/hal-04613329. tex.hal_id: hal-04613329 tex.hal_local_reference: Confiance.ai tex.hal_local_reference+duplicate-1: EC6 tex.hal_version: v1. 5, 13

Yuval Alaluf, Or Patashnik, and Daniel Cohen-Or. Restyle: A residual-based stylegan encoder via iterative refinement. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6711–6720, 2021. 26

Yuval Alaluf, Or Patashnik, Zongze Wu, Asif Zamir, Eli Shechtman, Dani Lischinski, and Daniel Cohen-Or. Third time's the charm? image and video editing with stylegan3. In *European conference on computer vision*, pages 204–220. Springer, 2022a. 25

Yuval Alaluf, Omer Tov, Ron Mokady, Rinon Gal, and Amit Bermano. Hyperstyle: Stylegan inversion with hypernetworks for real image editing. In *Proceedings of the IEEE/CVF conference on computer Vision and pattern recognition*, pages 18511–18521, 2022b. 26

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, and others. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022. 96

allpairspy. 79, 103

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*, 2016. 12

Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, Geeta Chauhan, Anjali Chourdia, Will Constable, Alban Desmaison, Zachary DeVito, Elias Ellison, Will Feng, Jiong Gong, Michael Gschwind, Brian Hirsh, Sherlock Huang, Kshiteej Kalambarkar, Laurent Kirsch, Michael Lazos, Mario Lezcano, Yanbo Liang, Jason Liang, Yinghai Lu, CK Luk, Bert Maher, Yunjie Pan, Christian Puhrsch, Matthias Reso, Mark Saroufim, Marcos Yukio Siraichi, Helen Suk, Michael Suo, Phil Tillet, Eikan Wang, Xiaodong Wang, William Wen, Shunting Zhang, Xu Zhao, Keren Zhou, Richard Zou, Ajit Mathews, Gregory Chanan, Peng Wu, and Soumith Chintala. PyTorch 2: Faster machine learning through dynamic python bytecode transformation and graph compilation. In *29th ACM international conference on architectural support for programming languages and operating systems, volume 2 (ASPLOS '24)*. ACM, April 2024. doi: 10.1145/3620665.3640366. URL https://pytorch.org/assets/pytorch2-2.pdf. 103

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017. 24, 30

Alejandro Barredo Arrieta, Natalia Diaz-Rodrıguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador Garcia, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, and others. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information fusion*, 58:82–115, 2020. Publisher: Elsevier. 13, 29

Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34 (6):26–38, 2017. Publisher: IEEE. 2

Maximilian Augustin, Yannic Neuhaus, and Matthias Hein. Analyzing and explaining image classifiers via diffusion guidance. *arXiv preprint arXiv:2311.17833*, 2023. 13

Randall Balestriero, Jerome Pesenti, and Yann LeCun. Learning in high dimension always amounts to extrapolation. *arXiv preprint arXiv:2110.09485*, 2021. 45

BBC. Google Apologises for Photos App's Racist Blunder, July 2015. URL https://www.bbc.com/news/technology-33347866. Publisher: BBC News. 3

Gilles Blanchard, Gyemin Lee, and Clayton Scott. Generalizing from several related classification tasks to a new unlabeled sample. *Advances in neural information processing systems*, 24, 2011. 17

J. Blank and K. Deb. pymoo: Multi-objective optimization in python. *IEEE access : practical innovations, open solutions*, 8:89497–89509, 2020. 79, 103

Jan-Aike Bolte, Andreas Bar, Daniel Lipinski, and Tim Fingscheidt. Towards corner case detection for autonomous driving. In *2019 IEEE Intelligent vehicles symposium (IV)*, pages 438–445. IEEE, 2019. 18, 19

Florian Bordes, Shashank Shekhar, Mark Ibrahim, Diane Bouchacourt, Pascal Vincent, and Ari Morcos. Pug: Photorealistic and semantically controllable synthetic data for

representation learning. *Advances in Neural Information Processing Systems*, 36, 2024. 86, 102

Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, 1992. 1

Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001. Publisher: Springer. 79

Jasmin Breitenstein, Jan-Aike Termöhlen, Daniel Lipinski, and Tim Fingscheidt. Systematization of corner cases for visual perception in automated driving. In *2020 IEEE intelligent vehicles symposium (IV)*, pages 1257–1264. IEEE, 2020. 19

Jasmin Breitenstein, Jan-Aike Termöhlen, Daniel Lipinski, and Tim Fingscheidt. Corner cases for visual perception in automated driving: some guidance on detection approaches. *arXiv preprint arXiv:2102.05897*, 2021. 19

Glenn W Brier. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3, 1950. 23

Tom B Brown, Dandelion Mané, Aurko Roy, Martin Abadi, and Justin Gilmer. Adversarial patch. *arXiv preprint arXiv:1712.09665*, 2017. 18

Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD workshop: Languages for data mining and machine learning*, pages 108–122, 2013. 79, 103

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, and others. Are we on the right way for evaluating large vision-language models? *CoRR*, 2024. 99

Muxi Chen, YU LI, and Qiang Xu. HiBug: On human-interpretable model debug. In *Thirty-seventh conference on neural information processing systems*, 2023a. URL https://openreview.net/forum?id=4sDHLxKb1L. 20

Yangyi Chen, Lifan Yuan, Ganqu Cui, Zhiyuan Liu, and Heng Ji. A Close Look into the Calibration of Pre-trained Language Models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1343–1367, Toronto, Canada, 2023b. Association for Computational Linguistics. 23, 62, 101, 103

Jiacheng Cheng and Nuno Vasconcelos. Calibrating Deep Neural Networks by Pairwise Constraints. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13709–13718, 2022. 23, 24

Kamal Choudhary, Brian DeCost, Chi Chen, Anubhav Jain, Francesca Tavazza, Ryan Cohn, Cheol Woo Park, Alok Choudhary, Ankit Agrawal, Simon JL Billinge, and others. Recent advances and applications of deep learning methods in materials science.

npj Computational Materials, 8(1):59, 2022. Publisher: Nature Publishing Group UK London. 1

C. Chow. On optimum recognition error and reject tradeoff. *IEEE Transactions on Information Theory*, 16(1):41–46, 1970. doi: 10.1109/TIT.1970.1054406. 20

C. K. Chow. An optimum character recognition system using decision functions. *IRE Transactions on Electronic Computers*, EC-6(4):247–254, 1957. doi: 10.1109/TEC.1957.5222035. 20

Sylvain Christin, Éric Hervet, and Nicolas Lecomte. Applications for deep learning in ecology. *Methods in Ecology and Evolution*, 10(10):1632–1644, 2019. Publisher: Wiley Online Library. 1

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *international conference on machine learning*, pages 1310–1320. PMLR, 2019. 18

Charles Corbière, Nicolas Thome, Avner Bar-Hen, Matthieu Cord, and Patrick Pérez. Addressing failure prediction by learning model confidence. *Advances in Neural Information Processing Systems*, 32, 2019. 21, 51, 53

Charles Corbière, Nicolas Thome, Antoine Saporta, Tuan-Hung Vu, Matthieu Cord, and Patrick Perez. Confidence estimation via auxiliary models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):6043–6055, 2021. Publisher: IEEE. 21, 39, 51

Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. Robustbench: a standardized adversarial robustness benchmark. *arXiv preprint arXiv:2010.09670*, 2020. 100

Audrey Cui, Ali Jahanian, Agata Lapedriza, Antonio Torralba, Shahin Mahdizadehaghdam, Rohit Kumar, and David Bau. Local relighting of real scenes. *arXiv preprint arXiv:2207.02774*, 2022. 25

Krzysztof Czarnecki. Operational design domain for automated driving systems. *Taxonomy of Basic Terms ", Waterloo Intelligent Systems Engineering (WISE) Lab, University of Waterloo, Canada*, 1, 2018. 13

Jeffrey Dastin. Amazon Scraps Secret AI Recruiting Tool that Showed Bias Against Women. *Reuters*, 2018. URL https://www.reuters.com/article/us-amazon-com-jobs-automation-insight/amazon-scraps-secret-ai-recruiting-tool-that-showed-bias-against-women-idUSKCN1MK08G. 3

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 2, 102

Li Deng. A tutorial survey of architectures, algorithms, and applications for deep learning. *APSIPA transactions on Signal and Information Processing*, 3:e2, 2014. Publisher: Cambridge University Press. 1

Greg d'Eon, Jason d'Eon, James R Wright, and Kevin Leyton-Brown. The spotlight: A general method for discovering systematic errors in deep learning models. In *Proceedings of the 2022 ACM conference on fairness, accountability, and transparency*, pages 1962–1981, 2022. 20

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021. 27, 28

Tan M Dinh, Anh Tuan Tran, Rang Nguyen, and Binh-Son Hua. Hyperinverter: Improving stylegan inversion via hypernetwork. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11389–11398, 2022. 26

Jeff Donahue and Karen Simonyan. Large scale adversarial representation learning. *Advances in neural information processing systems*, 32, 2019. 27

Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. In *International conference on learning representations*, 2017. URL https://openreview.net/forum?id=BJtNZAFgg. 26

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*, 2022. 63

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International conference on learning representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy. 21

Harris Drucker, Christopher J. C. Burges, Linda Kaufman, Alex Smola, and Vladimir Vapnik. Support vector regression machines. In M.C. Mozer, M. Jordan, and T. Petsche, editors, *Advances in neural information processing systems*, volume 9. MIT Press, 1996. URL https://proceedings.neurips.cc/paper_files/paper/1996/file/d38901788c533e8286cb6400b40b386d-Paper.pdf. 79

Xuefeng Du, Zhaoning Wang, Mu Cai, and Yixuan Li. Vos: Learning what you don't know by virtual outlier synthesis. *arXiv preprint arXiv:2202.01197*, 2022. 17

Xuefeng Du, Yiyou Sun, Jerry Zhu, and Yixuan Li. Dream the impossible: Outlier imagination with diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024. 17

Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Alex Lamb, Martin Arjovsky, Olivier Mastropietro, and Aaron Courville. Adversarially learned inference. In *International conference on learning representations*, 2017. URL https://openreview.net/forum?id=B1ElR4cgg. 27

Ran El-Yaniv and Yair Wiener. On the foundations of noise-free selective classification. *Journal of Machine Learning Research*, 11(53):1605–1641, 2010. URL http://jmlr.org/papers/v11/el-yaniv10a.html. 20, 47

Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1625–1634, 2018. 18

Sabri Eyuboglu, Maya Varma, Khaled Kamal Saab, Jean-Benoit Delbrouck, Christopher Lee-Messer, Jared Dunnmon, James Zou, and Christopher Re. Domino: Discovering systematic errors with cross-modal embeddings. In *International conference on learning representations*, 2022. URL https://openreview.net/forum?id=FPCMqjI0jXN. 20

Leo Feng, Mohamed Osama Ahmed, Hossein Hajimirsadeghi, and Amir H. Abdi. Towards better selective classification. In *The eleventh international conference on learning representations*, 2023. URL https://openreview.net/forum?id=5gDz_yTcst. 21, 51

Telmo Silva Filho, Hao Song, Miquel Perello-Nieto, Raul Santos-Rodriguez, Meelis Kull, and Peter Flach. Classifier Calibration: A survey on how to assess and improve predicted class probabilities. *Machine Learning*, 112(9):3211–3260, September 2023. ISSN 0885-6125, 1573-0565. doi: 10.1007/s10994-023-06336-7. arXiv: 2112.10327 [cs, stat]. 22

Adam Fisch, Tommi S. Jaakkola, and Regina Barzilay. Calibrated selective classification. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL https://openreview.net/forum?id=zFhNBs8GaV. 21

Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022. 28

Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit Haim Bermano, Gal Chechik, and Daniel Cohen-or. An image is worth one word: Personalizing text-to-image generation using textual inversion. In *The eleventh international conference on learning representations*, 2023. URL https://openreview.net/forum?id=NAQvF08TcyG. 85

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016. 21, 51

Ido Galil, Mohammed Dabbah, and Ran El-Yaniv. A framework for benchmarking Class-out-of-distribution detection and its application to ImageNet. In *The eleventh international conference on learning representations*, 2023a. URL https://openreview.net/forum?id=Iuubb9W6Jtk. 17

Ido Galil, Mohammed Dabbah, and Ran El-Yaniv. What can we learn from the selective prediction and uncertainty estimation performance of 523 imagenet classifiers? In *The eleventh international conference on learning representations*, 2023b. URL https://openreview.net/forum?id=p66AzKi6Xim. 21, 22, 48, 49, 63, 98

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of machine learning research*, 17(59):1–35, 2016. 17

Irena Gao, Gabriel Ilharco, Scott Lundberg, and Marco Tulio Ribeiro. Adaptive testing of computer vision models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4003–4014, 2023. 19

Roman Garnett. *Bayesian optimization*. Cambridge University Press, 2023. 77

Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, and others. A survey of uncertainty in deep neural networks. *Artificial Intelligence Review*, 56(Suppl 1):1513–1589, 2023. Publisher: Springer. 15, 22

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. *Advances in neural information processing systems*, 30, 2017. 6, 20, 21, 22, 47, 51, 65, 100

Yonatan Geifman and Ran El-Yaniv. Selectivenet: A deep neural network with an integrated reject option. In *International conference on machine learning*, pages 2151–2159. PMLR, 2019. 21, 51, 96

Yonatan Geifman, Guy Uziel, and Ran El-Yaniv. Bias-reduced uncertainty estimation for deep neural classifiers. In *International conference on learning representations*, 2019. URL https://openreview.net/forum?id=SJfb5jCqKm. 21, 48, 98

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, and others. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023. 96, 97, 98

Chuanxing Geng, Sheng-jun Huang, and Songcan Chen. Recent advances in open set recognition: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 43(10):3614–3631, 2020. Publisher: IEEE. 16

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007. Publisher: Taylor & Francis. 23

Ethan Goan and Clinton Fookes. Bayesian neural networks: An introduction and survey. *Case Studies in Applied Bayesian Data Science: CIRM Jean-Morlet Chair, Fall 2018*, pages 45–87, 2020. Publisher: Springer. 22

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014a. 24

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014b. 18

Parikshit Gopalan, Lunjia Hu, and Guy N Rothblum. On computationally efficient multiclass calibration. *arXiv preprint arXiv:2402.07821*, 2024. 22

Yash Goyal, Ziyan Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. Counterfactual visual explanations. In *International conference on machine learning*, pages 2376–2384. PMLR, 2019. 29

Nico Grant and Kashmir Hill. Google's Photo App Still Can't Find Gorillas. And Neither Can Apple's. *The New York Times*, May 2023. URL https://www.nytimes.com/2023/05/22/technology/ai-photo-labels-google-apple.html. 3

David Gunning, Mark Stefik, Jaesik Choi, Timothy Miller, Simone Stumpf, and Guang-Zhong Yang. XAI—Explainable artificial intelligence. *Science robotics*, 4(37): eaay7120, 2019. Publisher: American Association for the Advancement of Science. 13

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017. 21, 22, 23, 24, 49, 63, 98, 102

Chirag Gupta and Aaditya Ramdas. Top-label calibration and multiclass-to-binary reductions. In *International Conference on Learning Representations*, January 2022. 22, 23, 24, 61

Kartik Gupta, Amir Rahimi, Thalaiyasingam Ajanthan, Thomas Mensink, Cristian Sminchisescu, and Richard Hartley. Calibration of Neural Networks using Splines. In *International Conference on Learning Representations*, December 2021. 22

Joris Guérin, Kevin Delmas, Raul Ferreira, and Jérémie Guiochet. Out-of-distribution detection is not all you need. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 14829–14837, 2023. Number: 12. 17, 99

Arya Ketabchi Haghighat, Varsha Ravichandra-Mouli, Pranamesh Chakraborty, Yasaman Esfandiari, Saeed Arabi, and Anuj Sharma. Applications of deep learning in intelligent transportation systems. *Journal of Big Data Analytics in Transportation*, 2:115–145, 2020. Publisher: Springer. 1

Zhixiong Han, Yaru Hao, Li Dong, Yutao Sun, and Furu Wei. Prototypical calibration for few-shot learning of language models. In *The eleventh international conference on learning representations*, 2022. 63, 65

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 52, 59, 67

Florian Heidecker, Maarten Bieshaar, Bernhard Sick, HJ Hof, M Fritz, C Krauß, and O Wasenmüller. Towards corner case identification in cyclists' trajectories. In *Proc. of CSCS*, pages 1–2, 2019. 19

Florian Heidecker, Jasmin Breitenstein, Kevin Rösch, Jonas Löhdefink, Maarten Bieshaar, Christoph Stiller, Tim Fingscheidt, and Bernhard Sick. An application-driven conceptualization of corner cases for perception in highly automated driving. In *2021 IEEE intelligent vehicles symposium (IV)*, pages 644–651. IEEE, 2021. 19

Florian Heidecker, Maarten Bieshaar, and Bernhard Sick. Corner cases in machine learning processes. *AI Perspectives & Advances*, 6(1):1, January 2024. ISSN 2948-2143. doi: 10.1186/s42467-023-00015-y. URL https://aiperspectives.springeropen.com/articles/10.1186/s42467-023-00015-y. 19

Kilian Hendrickx, Lorenzo Perini, Dries Van der Plas, Wannes Meert, and Jesse Davis. Machine learning with a reject option: A survey. *Machine Learning*, 113(5):3073–3110, 2024. Publisher: Springer. 20

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International conference on learning representations*, 2019. URL https://openreview.net/forum?id=HJz6tiCqYm. 34, 36, 40

Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International conference on learning representations*, 2017. URL https://openreview.net/forum?id=Hkg4TI9xl. 16, 22, 39, 65, 100

Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. In *International conference on learning representations*, 2019. URL https://openreview.net/forum?id=HyxCxhRcY7. 17

Dan Hendrycks, Nicholas Carlini, John Schulman, and Jacob Steinhardt. Unsolved problems in ml safety. *arXiv preprint arXiv:2109.13916*, 2021. 12

Dan Hendrycks, Steven Basart, Mantas Mazeika, Andy Zou, Joseph Kwon, Mohammadreza Mostajabi, Jacob Steinhardt, and Dawn Song. Scaling out-of-distribution detection for real-world settings. In *International conference on machine learning*, pages 8759–8773. PMLR, 2022. 17

Dan Hendrycks, Mantas Mazeika, and Thomas Woodside. An overview of catastrophic ai risks. *arXiv preprint arXiv:2306.12001*, 2023. 13

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 35, 67

Geoffrey Hinton. Distilling the Knowledge in a Neural Network. *arXiv preprint arXiv:1503.02531*, 2015. 21

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. 28

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 27, 73

John H Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992. 77

Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, and others. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1314–1324, 2019. 59

Andrew G Howard. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 15

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 59, 67

Xinyu Huang, Youcai Zhang, Jinyu Ma, Weiwei Tian, Rui Feng, Yuejie Zhang, Yaqian Li, Yandong Guo, and Lei Zhang. Tag2Text: Guiding vision-language model via image tagging. In *The twelfth international conference on learning representations*, 2024. URL https://openreview.net/forum?id=x6u2BQ7xcq. 20

Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *Learning and intelligent optimization: 5th international conference, LION 5, rome, italy, january 17-21, 2011. Selected papers 5*, pages 507–523. Springer, 2011. 77

Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls. *Advances in neural information processing systems*, 33:9841–9850, 2020. 25, 32, 38

Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine learning*, 110(3):457–506, 2021. Publisher: Springer. 14

Abdul Jabbar, Xi Li, and Bourahla Omar. A Survey on Generative Adversarial Networks: Variants, Applications, and Training. *ACM Computing Surveys*, 54(8):157:1–157:49, October 2021. ISSN 0360-0300. doi: 10.1145/3463475. 32

Ali Jahanian, Lucy Chai, and Phillip Isola. On the" steerability" of generative adversarial networks. *arXiv preprint arXiv:1907.07171*, 2019. 25

Saachi Jain, Hannah Lawrence, Ankur Moitra, and Aleksander Madry. Distilling model failures as directions in latent space. In *The eleventh international conference on learning representations*, 2023. URL https://openreview.net/forum?id=99RpBVpLiX. 20

Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1): 2, 2020. Publisher: MDPI. 1

Guillaume Jeanneret, Loic Simon, and Frédéric Jurie. Adversarial counterfactual visual explanations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16425–16435, 2023. 13, 29

Zhongtao Jiang, Yuanzhe Zhang, Cao Liu, Jun Zhao, and Kang Liu. Generative calibration for in-context learning. *arXiv preprint arXiv:2310.10266*, 2023. 63, 65

Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):4037–4058, 2020. Publisher: IEEE. 1

Nari Johnson, Ángel Alexander Cabrera, Gregory Plumb, and Ameet Talwalkar. Where does my model underperform? a human evaluation of slice discovery algorithms. In *Proceedings of the AAAI conference on human computation and crowdsourcing*, volume 11, pages 65–76, 2023. Number: 1. 20

Donald R Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of global optimization*, 21:345–383, 2001. Publisher: Springer. 77

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020. 15

Archit Karandikar, Nicholas Cain, Dustin Tran, Balaji Lakshminarayanan, Jonathon Shlens, Michael C Mozer, and Becca Roelofs. Soft calibration objectives for neural networks. *Advances in Neural Information Processing Systems*, 34:29768–29779, 2021. 23

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 25, 31

Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *Advances in neural information processing systems*, 33:12104–12114, 2020a. 25

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020b. 25, 26

Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. *Advances in neural information processing systems*, 34:852–863, 2021. 25

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 28

Durk P Kingma and Prafulla Dhariwal. Glow: Generative Flow with Invertible 1x1 Convolutions. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. 31

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, and others. Wilds: A benchmark of in-the-wild distribution shifts. In *International conference on machine learning*, pages 5637–5664. PMLR, 2021. 100

Alex Krizhevsky, Geoffrey Hinton, and others. Learning multiple layers of features from tiny images, 2009. 59, 67, 101

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012. 2

Meelis Kull, Telmo M. Silva Filho, and Peter Flach. Beyond sigmoids: How to obtain well-calibrated probabilities from binary classifiers with beta calibration. *Electronic Journal of Statistics*, 11(2):5052–5080, January 2017. ISSN 1935-7524, 1935-7524. doi: 10.1214/17-EJS1338SI. Publisher: Institute of Mathematical Statistics and Bernoulli Society. 22, 24, 60

Meelis Kull, Miquel Perello Nieto, Markus Kängsepp, Telmo Silva Filho, Hao Song, and Peter Flach. Beyond temperature scaling: Obtaining well-calibrated multi-class probabilities with dirichlet calibration. *Advances in neural information processing systems*, 32, 2019. 23, 59, 60, 62

Ananya Kumar, Percy S Liang, and Tengyu Ma. Verified uncertainty calibration. *Advances in Neural Information Processing Systems*, 32, 2019a. 23

Aviral Kumar, Sunita Sarawagi, and Ujjwal Jain. Trainable Calibration Measures for Neural Networks from Kernel Mean Embeddings. In *Proceedings of the 35th International Conference on Machine Learning*, pages 2805–2814. PMLR, July 2018. ISSN: 2640-3498. 23

Ram Shankar Siva Kumar, David O Brien, Kendra Albert, Salomé Viljöen, and Jeffrey Snover. Failure modes in machine learning systems. *arXiv preprint arXiv:1911.11034*, 2019b. 99

Fabian Küppers, Jan Kronenberger, Amirhossein Shantia, and Anselm Haselhoff. Multivariate confidence calibration for object detection. In *The IEEECVF conference on computer vision and pattern recognition (CVPR) workshops*, June 2020. 103

Pol Labarbarie, Adrien CHAN-HON-TONG, Stéphane Herbin, and Milad Leyli-abadi. Optimal transport based adversarial patch to leverage large scale attack transferability. In *The twelfth international conference on learning representations*, 2024. URL https://openreview.net/forum?id=nZP10evtkV. 18

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017. 17, 22

Oran Lang, Yossi Gandelsman, Michal Yarom, Yoav Wald, Gal Elidan, Avinatan Hassidim, William T Freeman, Phillip Isola, Amir Globerson, Michal Irani, and others. Explaining in style: Training a gan to explain a classifier in stylespace. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 693–702, 2021. 13, 14, 27, 29, 32

Adrien Le Coz, Stéphane Herbin, and Faouzi Adjed. Leveraging generative models to characterize the failure conditions of image classifiers. In *The IJCAI-ECAI-22 workshop on artificial intelligence safety (aisafety 2022)*, Vienna, Austria, July 2022. URL https://hal.science/hal-03797490. tex.hal_id: hal-03797490 tex.hal_local_reference: Confiance.ai tex.hal_local_reference+duplicate-1: EC5 tex.hal_version: v1. 9, 11

Adrien Le Coz, Stéphane Herbin, and Faouzi Adjed. Explaining an image classifier with a generative model conditioned by uncertainty. In *Uncertainty meets explainability | workshop and tutorial @ ECML-PKDD 2023*, Torino, Italy,

September 2023. URL https://hal.science/hal-04194943. tex.hal_id: hal-04194943 tex.hal_local_reference: Confiance.ai tex.hal_local_reference+duplicate-1: EC5 tex.hal_version: v1. 9, 11

Adrien Le Coz, Houssem Ouertatani, Stéphane Herbin, and Faouzi Adjed. Efficient exploration of image classifier failures with bayesian optimization and text-to-image models. In *Generative models for computer vision - CVPR 2024 workshop*, Seattle, United States, June 2024. URL https://hal.science/hal-04549384. tex.hal_id: hal-04549384 tex.hal_local_reference: Confiance.ai tex.hal_local_reference+duplicate-1: EC5 tex.hal_version: v2. 11

Adrien Le Coz, Stéphane Herbin, and Faouzi Adjed. Confidence calibration of classifiers with many classes. In *Advances in Neural Information Processing Systems*, volume 37, 2025. 10, 11

Yann LeCun, Bernhard Boser, John Denker, Donnie Henderson, Richard Howard, Wayne Hubbard, and Lawrence Jackel. Handwritten digit recognition with a back-propagation network. *Advances in neural information processing systems*, 2, 1989. 2

Yann LeCun, Corinna Cortes, and CJ Burges. MNIST handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010. 34

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553): 436–444, 2015. Publisher: Nature Publishing Group UK London. 1, 2

Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *International conference on learning representations*, 2018a. URL https://openreview.net/forum?id=ryiAv2xAZ. 17

Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. *Advances in neural information processing systems*, 31, 2018b. 17, 18, 100

Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. Domain generalization with adversarial feature learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5400–5409, 2018. 17

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International conference on machine learning*, pages 12888–12900. PMLR, 2022a. 96

Kaican Li, Kai Chen, Haoyu Wang, Lanqing Hong, Chaoqiang Ye, Jianhua Han, Yukuai Chen, Wei Zhang, Chunjing Xu, Dit-Yan Yeung, and others. Coda: A real-world road corner case dataset for object detection in autonomous driving. In *European conference on computer vision*, pages 406–423. Springer, 2022b. 19

Mingkun Li and I.K. Sethi. Confidence-based active learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(8):1251–1261, August 2006. ISSN 1939-3539. 22, 65

Shiyu Liang, Yixuan Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *International conference on learning representations*, 2018. URL https://openreview.net/forum?id=H1VGkIxRZ. 17

Thomas Liao, Rohan Taori, Inioluwa Deborah Raji, and Ludwig Schmidt. Are we learning yet? a meta review of evaluation failures across machine learning. In *Thirty-fifth conference on neural information processing systems datasets and benchmarks track (round 2)*, 2021. 100

Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. Taking a Step Back with KCal: Multi-Class Kernel-Based Calibration for Deep Neural Networks. In *The Eleventh International Conference on Learning Representations*, September 2022. 23, 59

Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. Explainable AI: a review of machine learning interpretability methods. *Entropy. An International and Interdisciplinary Journal of Entropy and Information Studies*, 23(1), 2021. ISSN 1099-4300. doi: 10.3390/e23010018. URL https://www.mdpi.com/1099-4300/23/1/18. Number: 18 tex.pubmedid: 33375658. 29

Huan Ling, Karsten Kreis, Daiqing Li, Seung Wook Kim, Antonio Torralba, and Sanja Fidler. Editgan: High-precision semantic image editing. *Advances in Neural Information Processing Systems*, 34:16331–16345, 2021. 25, 32, 38

Jin Liu, Yi Pan, Min Li, Ziyue Chen, Lu Tang, Chengqian Lu, and Jianxin Wang. Applications of deep learning to MRI images: A survey. *Big Data Mining and Analytics*, 1 (1):1–18, 2018. Publisher: TUP. 2

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, 2023. Publisher: ACM New York, NY. 101

Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. Energy-based out-of-distribution detection. *Advances in neural information processing systems*, 33:21464–21475, 2020. 17

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. 59

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021. 59

Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, and others. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12009–12019, 2022a. 59

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11976–11986, 2022b. 59

Jie Lu, Anjin Liu, Fan Dong, Feng Gu, Joao Gama, and Guangquan Zhang. Learning under concept drift: A review. *IEEE transactions on knowledge and data engineering*, 31(12):2346–2363, 2018. Publisher: IEEE. 17

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International conference on learning representations*, 2018. URL https://openreview.net/forum?id=rJzIBfZAb. 18

TorchVision maintainers and contributors. TorchVision: PyTorch's computer vision library, 2016. URL https://github.com/pytorch/vision. 77, 101

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015. 26

Markos Markou and Sameer Singh. Novelty detection: a review—part 2:: neural network based approaches. *Signal processing*, 83(12):2499–2521, 2003. Publisher: Elsevier. 16

Julian John McAuley and Jure Leskovec. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *Proceedings of the 22nd international conference on world wide web*, Www '13, pages 897–908, New York, NY, USA, 2013. Association for Computing Machinery. ISBN 978-1-4503-2035-1. Number of pages: 12 Place: Rio de Janeiro, Brazil. 59, 102

Marcel Aguirre Mehlhorn, Andreas Richter, and Yuri A.W. Shardt. Ruling the operational boundaries: a survey on operational design domains of autonomous driving systems. *IFAC-PapersOnLine*, 56(2):2202–2213, 2023. ISSN 2405-8963. doi: https://doi.org/10.1016/j.ifacol.2023.10.1128. URL https://www.sciencedirect.com/science/article/pii/S2405896323015318. 5, 13

Jan Hendrik Metzen, Robin Hutmacher, N Grace Hua, Valentyn Boreiko, and Dan Zhang. Identification of systematic errors of image classifiers on rare subgroups. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5064–5073, 2023. 19, 71, 74, 75, 79, 82

Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence*, 267:1–38, 2019. Publisher: Elsevier. 13

Matthias Minderer, Josip Djolonga, Rob Romijnders, Frances Hubis, Xiaohua Zhai, Neil Houlsby, Dustin Tran, and Mario Lucic. Revisiting the Calibration of Modern Neural Networks. In *Advances in Neural Information Processing Systems*, volume 34, pages 15682–15694. Curran Associates, Inc., 2021. 22, 23

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 24, 38

MLEAP Consortium. EASA research – machine learning application approval (MLEAP) final report. Horizon Europe research and innovation programme report, European Union Aviation Safety Agency, May 2024. URL https://www.easa.europa.eu/en/downloads/139926/en. 5, 13

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, and others. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015. Publisher: Nature Publishing Group UK London. 2

Jonas Mockus, Vytautas Tiesis, and Antanas Zilinskas. The application of Bayesian methods for seeking the extremum. *Towards Global Optimization*, 2(117-129):2, 1978. Publisher: Amsterdam: Elsevier. 77

Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *International conference on machine learning*, pages 10–18. PMLR, 2013. 17

Jishnu Mukhoti, Viveka Kulharia, Amartya Sanyal, Stuart Golodetz, Philip Torr, and Puneet Dokania. Calibrating Deep Neural Networks using Focal Loss. In *Advances in Neural Information Processing Systems*, volume 33, pages 15288–15299. Curran Associates, Inc., 2020. 101, 103

Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining Well Calibrated Probabilities Using Bayesian Binning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 29(1), February 2015. ISSN 2374-3468. tex.copyright: Copyright (c). 23, 24, 60

Harikrishna Narasimhan, Aditya Krishna Menon, Wittawat Jitkrittum, and Sanjiv Kumar. Plugin estimators for selective classification with out-of-distribution detection. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=DASh78rJ7g. 21, 51

Muzammal Naseer, Salman Khan, and Fatih Porikli. Local gradients smoothing: Defense against localized adversarial attacks. In *2019 IEEE winter conference on applications of computer vision (WACV)*, pages 1300–1307. IEEE, 2019. 18

Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021. 28

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021. 27

Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd international conference on Machine learning*, ICML '05, pages 625–632, New York, NY, USA, August 2005. Association for Computing Machinery. ISBN 978-1-59593-180-1. 61

Changhai Nie and Hareton Leung. A survey of combinatorial testing. *ACM Computing Surveys (CSUR)*, 43(2):1–29, 2011. Publisher: ACM New York, NY, USA. 74

Jeremy Nixon, Michael W Dusenberry, Linchuan Zhang, Ghassen Jerfel, and Dustin Tran. Measuring calibration in deep learning. In *CVPR workshops*, volume 2, 2019. 23

Curtis G Northcutt, Anish Athalye, and Jonas Mueller. Pervasive label errors in test sets destabilize machine learning benchmarks. *arXiv preprint arXiv:2103.14749*, 2021. 18

Philipp Oberdiek, Gernot Fink, and Matthias Rottmann. Uqgan: A unified model for uncertainty quantification of deep classifiers trained via conditional gans. *Advances in Neural Information Processing Systems*, 35:21371–21385, 2022. 29

On-Road Automated Driving (ORAD) Committee. Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles. manual, June 2018. URL https://doi.org/10.4271/J3016_201806. 5, 13

OpenAI. GPT-4 system card, 2023. URL https://cdn.openai.com/papers/gpt-4-system-card.pdf. 2, 13, 96, 97

Tinghui Ouyang, Vicent Sanz Marco, Yoshinao Isobe, Hideki Asoh, Yutaka Oiwa, and Yoshiki Seo. Corner case data description and detection. In *2021 IEEE/ACM 1st workshop on AI engineering-software engineering for AI (WAIN)*, pages 19–26. IEEE, 2021a. 18, 19

Tinghui Ouyang, Vicent Sanz Marco, Yoshinao Isobe, Hideki Asoh, Yutaka Oiwa, and Yoshiki Seo. Improved surprise adequacy tools for corner case data description and detection. *Applied Sciences*, 11(15):6826, 2021b. Publisher: MDPI. 18, 19

Gaurav Parmar, Yijun Li, Jingwan Lu, Richard Zhang, Jun-Yan Zhu, and Krishna Kumar Singh. Spatially-adaptive multilayer selection for gan inversion and editing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11399–11409, 2022. 26

Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2085–2094, 2021. 25, 32, 38

Kanil Patel, William H Beluch, Bin Yang, Michael Pfeiffer, and Dan Zhang. Multi-class uncertainty calibration via mutual information maximization-based binning. In *International conference on learning representations*, 2020. 23, 24, 58, 59, 60

Ajeet Ram Pathak, Manjusha Pandey, and Siddharth Rautaray. Application of deep learning for object detection. *Procedia computer science*, 132:1706–1717, 2018. Publisher: Elsevier. 2

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. 39

Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. Deepxplore: Automated whitebox testing of deep learning systems. In *proceedings of the 26th symposium on operating systems principles*, pages 1–18, 2017. 18, 19

Luis Perez and Jason Wang. The effectiveness of data augmentation in image classification using deep learning. *arXiv preprint arXiv:1712.04621*, 2017. 15

Stanislav Pidhorskyi, Donald A Adjeroh, and Gianfranco Doretto. Adversarial latent autoencoders. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14104–14113, 2020. 27, 45

John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999. Publisher: Cambridge, MA. 23, 54, 55

Antoine Plumerault, Hervé Le Borgne, and Céline Hudelot. Controlling generative models with continuous factors of variations. In *ICLR 2020-eighth international conference on learning representations*, 2020. 25

Christopher Potts, Zhengxuan Wu, Atticus Geiger, and Douwe Kiela. DynaSent: a dynamic benchmark for sentiment analysis. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: Long papers)*, pages 2388–2404, Online, August 2021. Association for Computational Linguistics. 59, 102

A Radford. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 24

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, and others. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 19, 28, 59, 85, 96, 99

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21 (140):1–67, 2020. 59

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022. 2, 28, 74

Keivan Rezaei, Mehrdad Saberi, Mazda Moayeri, and Soheil Feizi. PRIME: Prioritizing interpretability in failure mode extraction. In *The twelfth international conference on learning representations*, 2024. URL https://openreview.net/forum?id=QrEHs9w5UF. 20

Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. ImageNet-21K pretraining for the masses. In *Thirty-fifth conference on neural information processing systems datasets and benchmarks track (round 1)*, 2021. 59, 101, 102, 103

Nicolas La Rocco. Level-3-Fahren mit 130 km/h: Mercedes gestaltet nächste ODD für Drive Pilot aus, August 2022. URL https://www.computerbase.de/2022-08/level-3-fahren-mit-130-km-h-mercedes-gestaltet-naechste-odd-fuer-drive-pilot-aus/. tex.timestamp: 13:10. 5

Daniel Roich, Ron Mokady, Amit H Bermano, and Daniel Cohen-Or. Pivotal tuning for latent-based editing of real images. *ACM Transactions on graphics (TOG)*, 42(1):1–13, 2022. Publisher: ACM New York, NY. 26

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 28, 73, 96

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015. 73

Lukas Ruff, Jacob R Kauffmann, Robert A Vandermeulen, Grégoire Montavon, Wojciech Samek, Marius Kloft, Thomas G Dietterich, and Klaus-Robert Müller. A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, 109(5):756–795, 2021. Publisher: IEEE. 16, 18

Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22500–22510, 2023. 28

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, and others. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115: 211–252, 2015. Publisher: Springer. 14

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, and others. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022. 2, 28, 74

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 67

Axel Sauer and Andreas Geiger. Counterfactual generative networks. In *International conference on learning representations*, 2021. URL https://openreview.net/forum?id=BXewfAYMmJw. 29

Axel Sauer, Katja Schwarz, and Andreas Geiger. Stylegan-xl: Scaling stylegan to large diverse datasets. In *ACM SIGGRAPH 2022 conference proceedings*, pages 1–10, 2022. 25, 38

Axel Sauer, Tero Karras, Samuli Laine, Andreas Geiger, and Timo Aila. Stylegan-t: Unlocking the power of gans for fast large-scale text-to-image synthesis. In *International conference on machine learning*, pages 30105–30118. PMLR, 2023. 25

Divya Saxena and Jiannong Cao. Generative adversarial networks (GANs) challenges, solutions, and future directions. *ACM Computing Surveys (CSUR)*, 54(3):1–42, 2021. Publisher: ACM New York, NY, USA. 32

Ben Schoon. Google's vision for pixel 9 is great, but doesn't escape my biggest problem with AI, 2024. URL https://9to5google.com/2024/08/25/google-pixel-9-ai-problem/. Authority: 9to5Google. 4

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, and others. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems*, 35:25278–25294, 2022. 28, 73

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: visual explanations from deep networks via gradient-based localization. *International journal of computer vision*, 128:336–359, 2020. Publisher: Springer. 13

Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9243–9252, 2020. 25

Pramila P Shinde and Seema Shah. A review of machine learning and deep learning applications. In *2018 Fourth international conference on computing communication control and automation (ICCUBEA)*, pages 1–6. IEEE, 2018. 1

Faiz Siddiqui. Tesla drivers report a surge in 'phantom braking'. *The Washington Post*, February 2022. URL https://www.washingtonpost.com/technology/2022/02/02/tesla-phantom-braking/. 6

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, and others. Mastering the game of Go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016. Publisher: Nature Publishing Group. 2

K Simonyan and A Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd international conference on learning representations (ICLR 2015)*. Computational and Biological Learning Society, 2015. 52, 67

Natasha Singer. Amazon Is Pushing Facial Technology That a Study Says Could Be Biased. *The New York Times*, 2019. URL https://www.nytimes.com/2019/01/24/technology/amazon-facial-technology-study.html. 3

Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams. Scalable bayesian optimization using deep neural networks. In *International conference on machine learning*, pages 2171–2180. PMLR, 2015. 77

Jacob Snow. Amazon's Face Recognition Falsely Matched 28 Members of Congress With Mugshots, 2018. URL https://www.aclu.org/news/privacy-technology/amazons-face-recognition-falsely-matched-28. Publisher: ACLU of Northern California. 3

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013. 65, 102

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015. 27

Haorui Song, Yong Du, Tianyi Xiang, Junyu Dong, Jing Qin, and Shengfeng He. Editing out-of-domain gan inversion via differential activations. In *European conference on computer vision*, pages 1–17. Springer, 2022. 26

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International conference on learning representations*, 2021a. URL https://openreview.net/forum?id=St1giarCHLP. 28, 91

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019. 27

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International conference on learning representations*, 2021b. URL https://openreview.net/forum?id=PxTIG12RRHS. 28

Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. Beyond neural scaling laws: beating power law scaling via data pruning. *Advances in Neural Information Processing Systems*, 35:19523–19536, 2022. 52

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014. Publisher: JMLR. org. 51

Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *Computer vision–ECCV 2016 workshops: Amsterdam, the netherlands, october 8-10 and 15-16, 2016, proceedings, part III 14*, pages 443–450. Springer, 2016. 17

Richard S. Sutton. The bitter lesson, 2019. URL http://www.incompleteideas.net/IncIdeas/BitterLesson.html. 96

C Szegedy. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013. 18

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 67

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 67

Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 59

Mingxing Tan and Quoc Le. Efficientnetv2: Smaller models and faster training. In *International conference on machine learning*, pages 10096–10106. PMLR, 2021. 59

Tesla. Model 3 owner's manual. manual, Tesla, Inc., 2024. URL https://www.tesla.com/ownersmanual/model3/en_us/GUID-2CB60804-9CEA-4F4B-8B04-09B991368DC5.html. 4

Sunil Thulasidasan, Gopinath Chennupati, Jeff A Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. On Mixup Training: Improved Calibration and Predictive Uncertainty for Deep Neural Networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 23

Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In *Proceedings of the 40th international conference on software engineering*, pages 303–314, 2018. 18, 19

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288, 1996. Publisher: Oxford University Press. 79

Shengbang Tong, Erik Jones, and Jacob Steinhardt. Mass-producing failures of multimodal systems with language models. In *Thirty-seventh conference on neural information processing systems*, 2023. URL https://openreview.net/forum?id=T6iiOqsGOh. 3, 19, 28, 99

Shengbang Tong, Zhuang Liu, Yuexiang Zhai, Yi Ma, Yann LeCun, and Saining Xie. Eyes wide shut? exploring the visual shortcomings of multimodal llms. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9568–9578, 2024. 99

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and others. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 65

Omer Tov, Yuval Alaluf, Yotam Nitzan, Or Patashnik, and Daniel Cohen-Or. Designing an encoder for stylegan image manipulation. *ACM Transactions on Graphics (TOG)*, 40(4):1–14, 2021. Publisher: ACM New York, NY, USA. 26, 44

Juozas Vaicenavicius, David Widmann, Carl Andersson, Fredrik Lindsten, Jacob Roll, and Thomas Schön. Evaluating model calibration in classification. In *The 22nd international conference on artificial intelligence and statistics*, pages 3459–3467. PMLR, 2019. 22, 23

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. URL http://jmlr.org/papers/v9/vandermaaten08a.html. 35

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in neural information processing systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf. 73

Joshua Vendrow, Saachi Jain, Logan Engstrom, and Aleksander Madry. Dataset interfaces: Diagnosing model failures using controllable counterfactual generation. *arXiv preprint arXiv:2302.07865*, 2023. 19, 71, 74

Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, Dhruv Nair, Sayak Paul, William Berman, Yiyi Xu, Steven Liu, and Thomas Wolf. Diffusers: State-of-the-art diffusion models, 2022. URL https://github.com/huggingface/diffusers. 91, 103

Ellen M. Voorhees and Dawn M. Tice. Building a question answering test collection. In *Proceedings of the 23rd annual international ACM SIGIR conference on research and development in information retrieval*, Sigir '00, pages 200–207, New York, NY, USA, 2000. Association for Computing Machinery. ISBN 1-58113-226-3. doi: 10.1145/345508.345577. URL https://doi.org/10.1145/345508.345577. Number of pages: 8 Place: Athens, Greece. 65, 102

Andrey Voynov and Artem Babenko. Unsupervised discovery of interpretable directions in the gan latent space. In *International conference on machine learning*, pages 9786–9796. PMLR, 2020. 25

Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harv. JL & Tech.*, 31: 841, 2017. Publisher: HeinOnline. 29

C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 89, 102

Ben Wang and Aran Komatsuzaki. GPT-j-6B: a 6 billion parameter autoregressive language model, May 2021. URL https://github.com/kingoflolz/mesh-transformer-jax. 65

Boxin Wang, Weixin Chen, Hengzhi Pei, Chulin Xie, Mintong Kang, Chenhui Zhang, Chejian Xu, Zidi Xiong, Ritik Dutta, Rylan Schaeffer, and others. DecodingTrust: a comprehensive assessment of trustworthiness in GPT models. In *NeurIPS*, 2023. 99, 100

Cheng Wang. Calibration in deep learning: a survey of the state-of-the-art. *arXiv preprint arXiv:2308.01222*, 2023. 22

Mei Wang and Weihong Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312:135–153, 2018. Publisher: Elsevier. 6, 17

Zhengwei Wang, Qi She, and Tomas E Ward. Generative adversarial networks in computer vision: A survey and taxonomy. *ACM Computing Surveys (CSUR)*, 54(2):1–38, 2021. Publisher: ACM New York, NY, USA. 32

Yuni Wen and Matthias Holweg. A phenomenological perspective on AI ethical failures: The case of facial recognition technology. *AI & SOCIETY*, 39(4):1929–1946, August 2024. ISSN 1435-5655. doi: 10.1007/s00146-023-01648-7. URL https://doi.org/10.1007/s00146-023-01648-7. 3

David Widmann, Fredrik Lindsten, and Dave Zachariah. Calibration tests in multi-class classification: A unifying framework. *Advances in neural information processing systems*, 32, 2019. 22

Wikipedia. Operational design domain - Wikipedia, the free encyclopedia, 2024a. URL https://en.wikipedia.org/wiki/Operational_design_domain. 5

Wikipedia. List of Tesla Autopilot crashes - Wikipedia, the free encyclopedia, 2024b. URL https://en.wikipedia.org/wiki/List_of_Tesla_Autopilot_crashes. 4

Wikipedia. ImageNet - Wikipedia, the free encyclopedia, 2024c. URL https://en.wikipedia.org/wiki/ImageNet. 2

Olivia Wiles, Isabela Albuquerque, and Sven Gowal. Discovering bugs in vision models using off-the-shelf image generation and captioning. In *NeurIPS ML safety workshop*, 2022. 19, 29, 71, 74, 82

Adina Williams, Nikita Nangia, and Samuel Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In Marilyn Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long papers)*, pages 1112–1122, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. 59, 102

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: System demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6. 101, 103

Zongze Wu, Dani Lischinski, and Eli Shechtman. Stylespace analysis: Disentangled controls for stylegan image generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12863–12872, 2021. 25, 26, 31, 32, 38

Weihao Xia, Yulun Zhang, Yujiu Yang, Jing-Hao Xue, Bolei Zhou, and Ming-Hsuan Yang. Gan inversion: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 45(3):3121–3138, 2022. Publisher: IEEE. 25, 38

Jingkang Yang, Kaiyang Zhou, Yixuan Li, and Ziwei Liu. Generalized out-of-distribution detection: A survey. *arXiv preprint arXiv:2110.11334*, 2021. 6, 16, 99

Jingkang Yang, Pengyun Wang, Dejian Zou, Zitang Zhou, Kunyuan Ding, Wenxuan Peng, Haoqi Wang, Guangyao Chen, Bo Li, Yiyou Sun, and others. Openood: Benchmarking generalized out-of-distribution detection. *Advances in Neural Information Processing Systems*, 35:32598–32611, 2022. 17, 100

Bianca Zadrozny and Charles Elkan. Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 609–616, San Francisco, CA, USA, June 2001. Morgan Kaufmann Publishers Inc. ISBN 978-1-55860-778-1. 23, 50, 54, 60

Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 694–699, New York, NY, USA, July 2002. Association for Computing Machinery. ISBN 978-1-58113-567-1. 23, 24, 54, 55, 60

Martin Zaefferer. Surrogate models for discrete optimization problems. 2018. 77

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 59

Jize Zhang, Bhavya Kailkhura, and T Yong-Jin Han. Mix-n-match: Ensemble and compositional methods for uncertainty calibration in deep learning. In *International conference on machine learning*, pages 11117–11128. PMLR, 2020. 23, 24, 58, 59, 60

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3836–3847, 2023. 28

Mengshi Zhang, Yuqun Zhang, Lingming Zhang, Cong Liu, and Sarfraz Khurshid. DeepRoad: GAN-based metamorphic testing and input validation framework for autonomous driving systems. In *Proceedings of the 33rd ACM/IEEE international conference on automated software engineering*, pages 132–142, 2018. 32

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in neural information processing systems*, volume 28. Curran Associates, Inc., 2015a. URL https://proceedings.neurips.cc/paper_files/paper/2015/file/250cf8b51c773f3f8dc8b4be867a9a02-Paper.pdf. 65, 102

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in neural information processing systems*, volume 28. Curran Associates, Inc., 2015b. 59, 102

Youcai Zhang, Xinyu Huang, Jinyu Ma, Zhaoyang Li, Zhaochuan Luo, Yanchun Xie, Yuzhuo Qin, Tong Luo, Yaqian Li, Shilong Liu, and others. Recognize anything: A strong image tagging model. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1724–1732, 2024. 20

Shengjia Zhao, Michael Kim, Roshni Sahoo, Tengyu Ma, and Stefano Ermon. Calibrating predictions to decisions: a novel approach to multi-class calibration. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in neural information processing systems*, volume 34, pages 22313–22324. Curran Associates, Inc., 2021a. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/bbc92a647199b832ec90d7cf57074e9e-Paper.pdf. 22

Xujiang Zhao, Shu Hu, Jin-Hee Cho, and Feng Chen. Uncertainty-based Decision Making Using Deep Reinforcement Learning. In *2019 22th International Conference on Information Fusion (FUSION)*, pages 1–8, July 2019. 22

Zhengli Zhao, Dheeru Dua, and Sameer Singh. Generating natural adversarial examples. In *International conference on learning representations*, 2018. URL https://openreview.net/forum?id=H1BLjgZCb. 29

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models. In *International conference on machine learning*, pages 12697–12706. PMLR, 2021b. 63

Han Zhou, Xingchen Wan, Lev Proleev, Diana Mincu, Jilin Chen, Katherine Heller, and Subhrajit Roy. Batch calibration: Rethinking calibration for in-context learning and prompt engineering. *arXiv preprint arXiv:2309.17249*, 2023. 63, 65

Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain generalization: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4):4396–4415, 2022. Publisher: IEEE. 6, 17

Na Zhou, Chuan-Tao Zhang, Hong-Ying Lv, Chen-Xing Hao, Tian-Jun Li, Jing-Juan Zhu, Hua Zhu, Man Jiang, Ke-Wei Liu, He-Lei Hou, and others. Concordance study between IBM Watson for oncology and clinical practice for patients with cancer in China. *The oncologist*, 24(6):812–819, 2019. Publisher: Oxford University Press. 3

Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *Computer vision–ECCV 2016: 14th european conference, amsterdam, the netherlands, october 11-14, 2016, proceedings, part V 14*, pages 597–613. Springer, 2016. 26

Orr Zohar, Shih-Cheng Huang, Kuan-Chieh Wang, and Serena Yeung. Lovm: Language-only vision model selection. *Advances in Neural Information Processing Systems*, 36, 2024. 99