UNIVERSITÉ CLERMONT AUVERGNE
Ecole Doctorale des Sciences Pour l'Ingénieur

DOCTORAL THESIS

Dealing with confusing samples in image classification based model

Author:
Jiarui XIE

Supervisor:
Thierry CHATEAU
Violaine ANTOINE

A thesis submitted in fulfillment of the requirements
for the degree of Docteur de l'Université Clermont Auvergne
at the
Laboratory of Informatics, Modelling and Optimization of the Systems

Composition of the jury:

Pr.	Mephu Nguifo Engelbert	Université Clermont Auvergne	President
Pr.	David Mercier	Université d'Artois	Reviewer
Pr.	Didier Coquin	Université Savoie Mont Blanc	Reviewer
Pr.	Mephu Nguifo Engelbert	Université Clermont Auvergne	Examiner
Pr.	Nicolas Sutton-Charani	IMT Mines-Alès	Examiner
Pr.	Thierry Chateau	Université Clermont Auvergne	Director of thesis
Pr.	Violaine Antoine	Université Clermont Auvergne	Supervisor

Defended on July 11th 2022

UNIVERSITÉ CLERMONT AUVERGNE
Ecole Doctorale des Sciences Pour l'Ingénieur
Laboratory of Informatics, Modelling and Optimization of the Systems

Abstract

Dealing with confusing samples in image classification based model

by Jiarui XIE

Guaranteeing the model trained by in-domain samples to make correct and reliable predictions when encountering confusing samples in the open world is an inevitable component of image classification based models. Consequently, three novelty methods based on uncertainty estimation and partial classification, respectively, are proposed. The thesis scope is limited to the image classification, and the confusing sample contains out-of-domain, and imprecise samples.

Uncertainty estimation is a straight approach adopted to detect confusing samples. The uncertainty estimation method can calculate a scalar value representing the uncertainty, then detect confusing samples by comparing to a pre-defined threshold. In the first part of this thesis, a Subjective Logic based Uncertainty Estimation method (SLUE) is proposed by extending the existing method named Evidential Deep Learning (EDL). Compared with the EDL that overlooked the advantages of the base rate, the SLUE method takes the base rates explicitly into account. The base rate in SLUE is used to guide the training process in the desired direction.

Through uncertainty estimation, we can decide whether to reject or accept a prediction. However, it is more reasonable to predict a subset for cautious decision-making. Consequently, a Partial Classification derived from Model Output (PCMO) is proposed in the second part. The PCMO method can assign beliefs to nested subsets under the Dempster-Shafer Theory (DST). That can alleviate the exponential increase in the subset number brought by the increase of the class number. Compared with the existing methods, the PCMO method shows a simple and scalable framework. On the one hand, the PCMO method is fulfilled only based on the model output that can be applied to any pre-trained convolutional neural networks, without any demand to retrain the model or conduct any further modifications. On the other hand, by considering good features of the log function and analyzing the regular pattern of model output, a novel and reasonable transformation from model output to possibility distribution is proposed. One weakness of the PCMO method is that it cannot provide beliefs for the empty set and the entire set. Consequently, we rethink the possibility calculation process in the PCMO method and proposed a new method named PCBS that combines the DST and the bell-shaped function (BSF). In which, the BSF is responsible for the possibility calculation by taking the information provided by the training output. By doing that the model is able to assign beliefs to the empty set and the entire set leading to the ability to distinguish between the out-of-domain samples and imprecise samples.

Partial classification, Uncertainty estimation, Dempster-Shafer theory, Subjective logic, Confusing sample, Dirichlet distribution, Bell-shaped function

UNIVERSITÉ CLERMONT AUVERGNE
Ecole Doctorale des Sciences Pour l'Ingénieur
Laboratory of Informatics, Modelling and Optimization of the Systems

Résumé

Traitement d'échantillons confus dans un modèle basé sur la classification d'images

par Jiarui XIE

Pour les modèles basés sur la classification d'images, il est important de garantir que le modèle entraîné par des échantillons du domaine fasse des prédictions correctes et fiables lorsqu'il rencontre des échantillons confus. Par conséquent, trois nouvelles méthodes entourant l'estimation de l'incertitude et la classification partielle, respectivement, sont proposées. La portée de la thèse est limitée au modèle basé sur la classification d'images, et l'échantillon confus contient des échantillons hors-domaine et imprécis.

La stratégie d'estimation de l'incertitude peut calculer une valeur représentant l'incertitude prédictive, puis détecter les échantillons confus en les comparant à un seuil prédéfini. Dans la première partie de cette thèse, une méthode d'estimation d'incertitude basée sur la logique subjective (SLUE) est proposée en étendant la méthode existante nommée Evidential Deep Learning (EDL). Par rapport à l'EDL qui négligeait les avantages du taux de base, la méthode SLUE prend explicitement en compte les taux de base. Le taux de base dans SLUE est utilisé pour guider le processus de formation dans la direction souhaitée.

Grâce à l'estimation de l'incertitude, nous pouvons décider de rejeter ou d'accepter une prédiction. Cependant, il est plus raisonnable de prédire un sous-ensemble pour une prise de décision prudente. Par conséquent, une classification partielle dérivée de la sortie du modèle (PCMO) est proposée dans la deuxième partie. La méthode PCMO peut attribuer des croyances à des sous-ensembles imbriqués selon la théorie de Dempster-Shafer (DST). Cela peut atténuer l'augmentation exponentielle du nombre de sous-ensembles due à l'augmentation du nombre de classes. Comparée aux méthodes existantes, la méthode PCMO présente un cadre simple et évolutif. D'une part, la méthode PCMO est remplie uniquement sur la sortie du modèle qui peut être appliquée à n'importe quel réseau neuronal convolutif pré-entraîné, sans qu'il soit nécessaire de réentraîner le modèle ou d'effectuer d'autres modifications. D'autre part, en considérant les bonnes caractéristiques de la fonction log et en analysant le modèle régulier de la sortie du modèle, une transformation nouvelle et raisonnable de la sortie du modèle en distribution de possibilités est proposée. Une faiblesse de la méthode PCMO est qu'elle ne peut pas fournir de croyances pour l'ensemble vide et l'ensemble entier. Par conséquent, nous avons repensé le processus de calcul des possibilités et proposé une nouvelle méthode appelée PCBS qui combine la DST et la fonction en forme de cloche. Dans ce cas, la fonction en forme de cloche est responsable du calcul des possibilités en prenant les informations fournies par la sortie de formation. Ce faisant, le modèle est capable d'attribuer des croyances à l'ensemble vide et à l'ensemble entier, ce qui permet de distinguer les échantillons hors-domaine des échantillons imprécis.

Classification partielle, estimation de l'incertitude, théorie de Dempster-Shafer, logique subjective, échantillon confus, distribution de Dirichlet, fonction en forme de cloche.

This thesis is dedicated to my parents.

Acknowledgements

This work is fully funded by the Auvergne Rhône Alpes region: project AUDACE2018. I appreciate having the opportunity to work on this thesis.

I would like to thank my supervisors Mrs. Violaine Antoine and Mr. Thierry Chateau for their encouragement, enthusiasm, and patience in the last three years. And this is the experience that I will cherish my whole life. The first time I met with Mrs. Antoine was when I applied for the master's internship, we worked together on the semi-supervised evidence clustering. We end the research by publishing my first paper. That was significant for me. At that time, I wanted to publish a paper. I asked many friends and teachers, but I couldn't find a way. Finally, it was Mrs. Antoine who taught me how to write an article step by step. I learned a lot from that process. Mrs. Antoine also took the initiative to teach me French, we would practice for several minutes in the office after the meeting. I appreciate that. The first time I met Mr. Chateau was when I applied for the Ph.D. position. Doing research together, I found Mr. Chateau to be approachable and of immense knowledge. I would ask a lot of questions, and he would be very patient in answering them. Having the opportunity to work with him, I was able to learn how to become a good scientific researcher. What impressed me most was that he would personally write the code and send it to me on Saturday morning.

I would like to thank all the administrative staff in LIMOS, IP, CROUS, and the doctoral school. When I encountered problems, I would send them emails, then get their immediate replies and help. I am a person who is easily influenced by the environment, without a good environment, I could not concentrate on my research. I would like to thank them for their dedication allowing me to be in a good and stable research environment.

I would like to thank all the members of my committee thesis for their feedback that helped me to improve my work.

My thanks also go to my friends at Clermont Ferrand, Chao Zhang, YongZhe Yan, Chen Xu, ZhengZe Zhu, for enriching my spare time.

Finally, I would like to express my infinite gratitude to my parents and Jie Yang. Every time I can't keep going and want to give up, or when I encounter something unpleasant, you comfort me in your own way, encourage me and show me the way forward. You have helped me to pick up my spirit again and to move forward with full motivation. Thanks a lot, love you forever!

Contents

List of Figures

List of Tables

Glossary

AI Artificial Intelligence. 1

AUC Area Under the Curve. 36

AUPRC Area Under the Precision Recall Curve. 36, 38, 83, 84

AUROC Area Under the Receiver Operating Characteristic. 36, 38, 83, 84

BBA Basic Belief Assignment. 13–16

BNN Bayesian Neural Network. 30, 32

BPA Basic Probability Assignment. 13, 19

BSF Bell Shaped Function. 9, 11, 23, 25, 67–71, 75, 80, 82–85, 88, 89

CDF Cumulative Distribution Function. 42, 44, 46

CHI Calinski-Harabasz Index. 89

CNN Convolutional Neural Network. 6, 8, 11, 12, 24, 29, 33, 39, 41, 44, 52, 87

DST Dempster-Shafer Theory. 3, 8, 9, 11, 13, 15, 16, 19, 24, 25, 40, 52, 53, 67, 68, 87

ECE Expected Calibration Error. 37, 38

EDL Evidential Deep Learning. 8, 9, 34, 39–41, 45, 47, 87, 88

FPR False Positive Rate. 36

FPR95 False Positive Rate 95. 35, 36, 38, 80, 83, 84

HAC Hierarchical Agglomerative Clustering. 89

ID In-Domain. 2, 8, 18, 34–36, 42, 44, 70, 72

IM Imprecise. 2, 6, 63, 70, 72, 74

KL Kullback-Leibler. 34, 40

MI Mutual Information. 35

MLP Multilayer Perceptron. 6, 11, 12, 24, 68, 73, 83

NN Neural Network. 1, 3, 6, 11, 24, 29, 30, 32–34, 40, 43, 51, 75, 77, 87

OOD Out-of-Domain. 2, 5, 6, 8, 18, 34–37, 40, 42, 44–47, 51, 67, 69, 70, 72, 74, 79, 80, 83, 87

OWA Ordered Weighted Average. 52

PDF Probability Density Function. 17, 22, 39, 40

PR Precision Recall. 35, 36

ReLU Rectified Linear Unit. 39, 41

ROC Receiver Operating Characteristic. 35, 36

SL Subjective Logic. 8, 9, 11, 16, 17, 19, 24, 25, 39, 40, 87, 89

SVM Support Vector Machine. 36, 51, 59

TPR True Positive Rate. 35, 36

VI Variational Inference. 30, 32

Nomenclatures

Symbol	Meaning
$(\boldsymbol{x} \in \mathbb{R}^d, y)$	the training sample and corresponding ground-truth label
$(\boldsymbol{x}^* \in \mathbb{R}^d, y^*)$	the testing sample and corresponding ground-truth label
$\hat{y}$	the observed label or predicted label
$(\mathcal{X}, \mathcal{Y})$	the sample domain and corresponding ground-truth label domain
N	the sample number
K	the class number
M	the number, e.g., the number of models
i, j, k, n	the index of a stochastic entries in a list or state space
$\mathcal{D}$	the dataset, e.g., $\mathcal{D}^{train}$ represents the training dataset
Ω	the state or hypothesis space
ω	the state or hypothesis
Ω	the entire set
$\emptyset$	the empty set
$\mathcal{P}$	the power set of Ω
$\boldsymbol{\theta}$	the set of parameters
$f_{\boldsymbol{\theta}}(\cdot)$	the neural network with parameters $\boldsymbol{\theta}$
$\ell(\cdot)$	the loss function, e.g., the cross entropy loss
$\mathcal{L}(\cdot)$	the entire loss function
u	the uncertainty value
h	the last hidden layer output number
$\boldsymbol{o} \in \mathbb{R}^K$	the model output or logit vector
$\boldsymbol{\delta} \in \mathbb{R}^h$	the last hidden layer output vector
$\boldsymbol{W} \in \mathbb{R}^{h \times K}$	the weight vector between the last hidden layer and the output layer
m	the mass function
$\boldsymbol{b}$	the belief vector
$\boldsymbol{a}$	the base rate or prior probability vector
$\boldsymbol{r}$	the evidence vector
P	the probability distribution
$\boldsymbol{p}$	the probability vector
Π	the possibility distribution
$\boldsymbol{\pi}$	the possibility vector
$\mathcal{C}$	the prior constant in subjective logic
S	the opinion in subjective logic
$\boldsymbol{\alpha}$	the Dirichlet distribution parameters
th	the threshold
bs	the batch size
ϵ	the perturbation in the fast gradient sign method
t	the training iteration
T	the temperature scaling
η	the learning rate
λ	the balance weight
β	the coefficient
a, b, c, d	the parameter used in membership function and beta PDF

Symbol	Meaning
$\mathcal{N}(\cdot)$	the Gaussian distribution
$\sigma(\cdot)$	the activation function
$\mathbb{E}(\cdot)$	the expectation function
$\mathbb{V}(\cdot)$	the variation function
$\mathbb{1}(\cdot)$	the indicator function
$\mathcal{H}(\cdot)$	the entropy function
$\mathcal{I}(\cdot)$	the mutual information function
$\mathcal{R}(\cdot)$	the decision-making risk
$\mathcal{K}(\cdot)$	the expected Kullback-Leibler divergence function
$\mathcal{M}(\cdot)$	the decision making function
$\mu(\cdot)$	the membership function used in the fuzzy theory
$acc(\cdot)$	the accuracy function
$g(\cdot)$	the utility function

Chapter 1

Introduction

In this chapter, we first introduce real-life problems caused by confusing samples and show motivations for dealing with them. The uncertainty caused by confusing samples is the root of these problems. Consequently, we give an introduction of uncertainty including two uncertainty types, six uncertainty sources, and the classical uncertainty quantification methods. Two main approaches, i.e., uncertainty estimation, and partial classification are introduced to cope with confusing samples in this thesis. In addition, the study subjects, existing challenges, contributions, and organization of this thesis are presented.

1.1 Confusing sample introduction

1.1.1 The real-life problems caused by confusing samples

Over the last decades, Artificial Intelligence (AI) has reached almost every science field and become a crucial part of various real-life applications. Generally, Neural Network (NN)s or models (the Neural Network and model are interchangeable in this thesis) are trained and tested in the closed world. The closed world hypothesis claims that all testing classes have been presented to the model during the training process, i.e., the closed world represents a predefined, closed set of classes. In contrast, the open world [BB15] delegates a continuous and evolving environment, that is to say, the training dataset is incomplete. The open world hypothesis requires the model can continuously update and be robust to unknown or unseen classes. It is challenging when the trained model exposure to the open world. In particular, for safety-critical applications, e.g., autonomous driving, it is significant for the model to detect unknown samples instead of wrongly classifying them to one of the training classes. The first autonomous driving accident in human history occurred in 2016 when a Tesla vehicle hit a white truck turning left at full speed [Ele16]. The post-incident investigation revealed several reasons, one of which was that the autopilot system recognized the white truck as a cloud in the sky [Ele16].

Similar to the Tesla accident, a Uber vehicle collision happened because the autopilot system failed to recognize a walking pedestrian [eco18]. In these two examples, both the white truck and the walking pedestrian were confusing samples. The autopilot system did not recognize the potential risk, which in turn led to the accidents. Another well-known issue caused by AI is that an African Americans' photo is misclassified as gorillas have led to racism [KG17] over the world.

From the above examples, we can see it is a catastrophe "when the model does not know when it does not know". The main motivation of this thesis is to enable the model to identify confusing samples, improve the model's robustness, and reduce the classification risk.

FIGURE 1.1: An example of a white trunk, which might be classified as a cloud by the autopilot system.

1.1.2 Two confusing sample categories

In this thesis, we focus on three type samples, the In-Domain (ID), Imprecise (IM), and Out-of-Domain (OOD) sample. The definition of the domain is similar to the state space indicated in Section 2.2. It is composed of the training dataset. The ID sample represents the training sample. In addition, datasets always contain confusing samples, as remarked by [DM93, Den00, Den97], there are two branches, i.e., the IM and OOD sample, as demonstrated in Fig. 1.2.

1. The IM sample locates at the intersection of several classes occupies a high risk of misclassification due to the equal probabilities for several classes.
2. The OOD sample represents the samples that are different from the training samples, which have two typical sources.
   - The OOD sample corresponds to an outlier situated at a large distance from each of the training classes.
   - The OOD sample might also come from samples that are not represented in the training dataset. This may happen for various reasons. For example, certain faults in technological systems, some classes correspond to states of nature that are too dangerous or too costly to obtain, or some classes have never been observed and are not even conceived by the user.

1.1.3 Dealing with confusing samples

Suppose the state space $\Omega = \{\omega_1, \cdots, \omega_K\}$ with K training classes. Under the classical context, i.e., the precise or determinable classification, the trained model f_θ needs to classify a testing sample $\boldsymbol{x}^*$ into one of the K training classes. In this thesis, the bold type corresponds to vectors, and the normal type relates to scalar variables. The model outputs are fed into the softmax layer to get the probability distribution that satisfies the additivity principle Eq. 1.1. Then, the model make classification based on the predefined decision rule $\mathcal{M}(\cdot)$ so that $\mathcal{M}(\boldsymbol{x}^*) = \hat{y}$ means that based on the decision rule a testing sample $\boldsymbol{x}^*$ is classified into class $\hat{y}$.

$$\sum_{\omega \in \Omega} P(\omega) = 1, \tag{1.1}$$

where P represents the probability distribution over Ω.

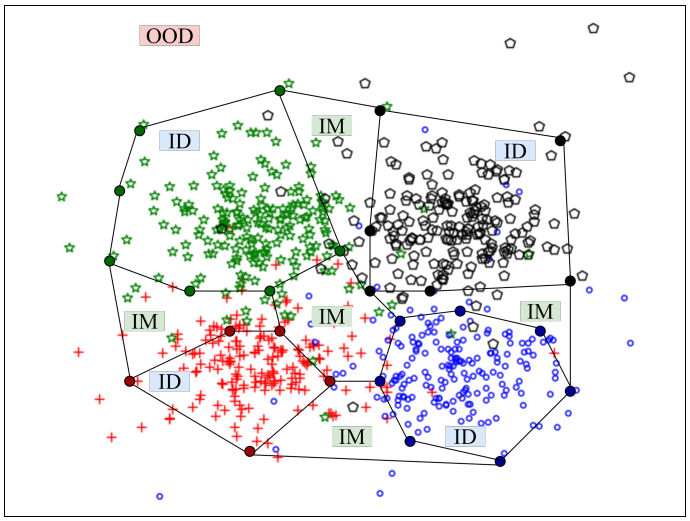


FIGURE 1.2: An example of various samples. The different samples are assigned to different partitions.

For decision-making, the most familiar and widespread method is to choose the class with the maximum probability Eq. 1.2.

$$\hat{y} = \max(P). \tag{1.2}$$

In addition, we can define the decision risk $\mathcal{R}$ for each ground-truth label or "act" which means picking one class from the training classes. Then choose the label or act with the minimum risk as the prediction.

$$\hat{y} = \min(\{\mathcal{R}(y_1), \cdots, \mathcal{R}(y_K)\}), \quad \mathcal{R}(y) = \sum_{\omega \in \Omega} \ell(y \mid \omega) P(\omega), \tag{1.3}$$

where $\ell(y \mid \omega)$ represents the loss given the ground-truth label y and chooses the ω as the prediction. The commonly used loss is "0-1" loss.

The situation is more complicated when the confusing sample participates because there is even not a proper class for them to settle. To deal with confusing samples, we adopted two methods, i.e., uncertainty estimation and partial classification. The Fig. 1.3 shows different actions for the two methods when feeding a confusing sample, i.e., a car image, into the trained NN based on the dataset containing cat, dog, and bird classes.

In the view of uncertainty estimation, if the obtained uncertainty u is greater than a predefined threshold δ, the input sample will be classified into the empty set. Otherwise, execute the classification based on the probability distribution.

On the other hand, partial classification means classifying an input sample into a class subset. It needs to calculate a value called belief or utility for each subset. Then a subset is selected as prediction based on different decision-making strategies. In this thesis, we calculate belief based on Dempster-Shafer Theory (DST) for class subsets and choose the one with maximum belief as the prediction.

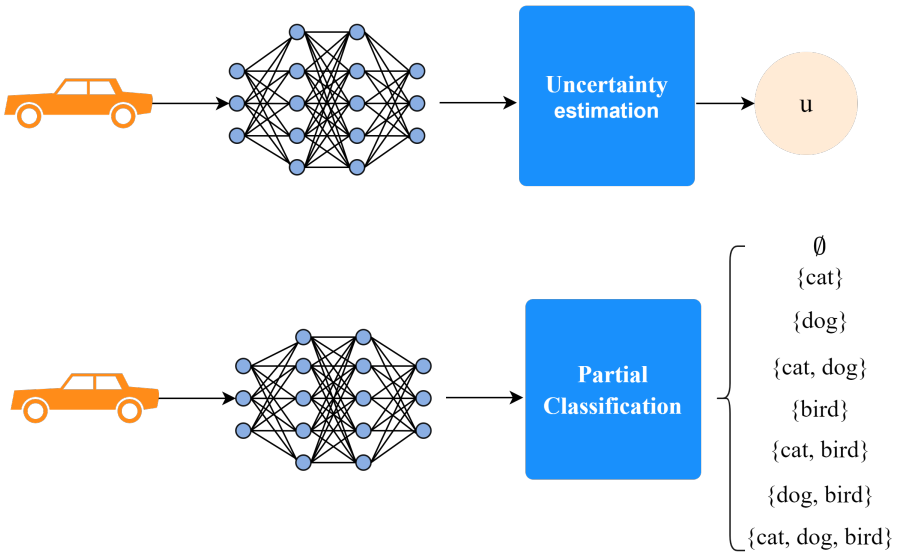


FIGURE 1.3: An example of dealing with confusing samples. The model is trained based on a dataset containing cat, dog and bird classes. Suppose, during the testing phase, the model encounters a car image. For uncertainty estimation, the model will calculate a value to determine whether to reject the sample or not. For partial classification, it can assign belief to the class subset, then choose the subset with maximum belief as the prediction.

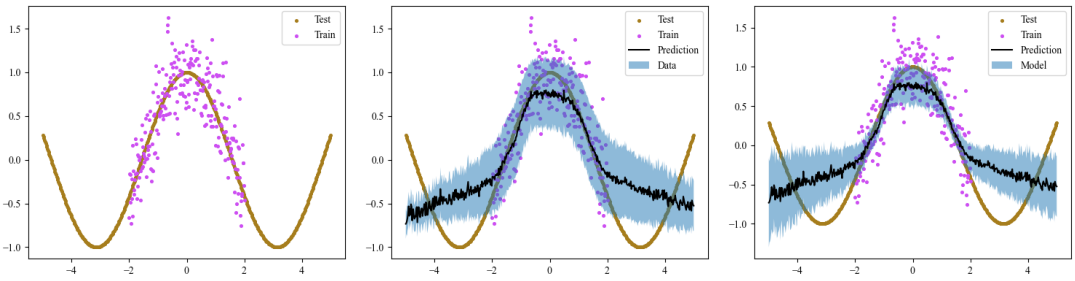
1.2 Uncertainty introduction

1.2.1 Two uncertainty categories

The predictive uncertainty is in general separated into data (statistical or aleatoric) uncertainty and model (systemic, distributional, or epistemic) uncertainty. The data uncertainty is inherent in the training dataset and describes the confidence in the training dataset. The noise in the dataset or the overlapping among classes can lead to data uncertainty. It cannot be reduced by adding more data. The model uncertainty rises due to the model itself, which describes the confidence of the prediction. The model structure, overfitting, or underfitting is the reason for this type of uncertainty. The model uncertainty can be reduced by adding more training data and optimizing the model sufficiently. The uncertainty brought by the confusing samples belongs to the data uncertainty. This is because the model is assumed to be sufficiently trained in a closed world and is not exposed to all possible samples. The Fig. 1.4 gives examples based on regression to help understand each type of uncertainty.

1.2.2 Six uncertainty sources

Understanding the uncertainty source is a prerequisite to dealing with uncertainty. Several literatures [KV18, LW21, GTA+21] propose diverse classifications of the sources. In [KV18], authors propose separating the uncertainty source into three major branches, model fit, data quality, and scope compliance as shown in Fig. 1.5. Whereas model fit focuses on the uncertainty caused by the error in model outputs. The data quality covers the uncertainty caused by dealing with input data obtained in suboptimal conditions. And, the scope of compliance covers situations where the model is likely applied outside the scope for which it was trained. Authors in [LW21] start with the posterior probability distribution $P(y \mid \boldsymbol{x}, \boldsymbol{\theta})$ claim that the uncertainty source resides in the data $(\boldsymbol{x}, y)$ and the model $f_{\boldsymbol{\theta}}$, sumarized seven sources, e.g, the data, hyperparameters.



(A) The dataset. (B) The data uncertainty. (C) The model uncertainty.

FIGURE 1.4: Visualization of the data and model uncertainties based on regression [KG17]. The shadow blue region represents the data or model uncertainty. As we can see, the OOD testing samples and the scatted imprecise training samples lead to high data and model uncertainty.

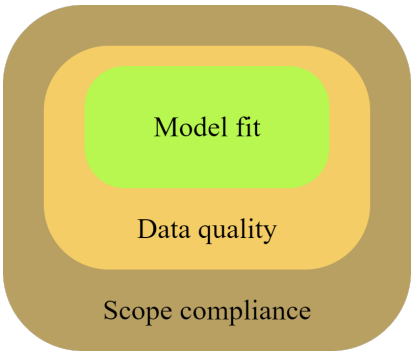


FIGURE 1.5: Onion layer of uncertainty sources proposed in [KV18].

Similar to [KV18] and based on [GTA+21], we summarize three different steps from data acquisition to confusing sample classification for the image classification based model.

1. **The data acquisition process.** Acquiring raw data with artificial measurements or equipment from the open world. Then process the raw data, e.g., resize the image, annotation, and make it ready for further model training.

2. **The model training process.** Designing a model according to the practical scenario. Training the model and decide the optimum model configurations, e.g., parameters.

3. **The confusing sample classification process.** Feeding testing samples into the trained model, and execute uncertainty estimation or partial classification based on the outputs and information provided by the model.

In general, these three steps contain six potential uncertainty sources, which again affect the final prediction. The first three sources lead to data uncertainty while the last three sources result in model uncertainty.

Source 1 (Uncertainty inherits to the open world) *It is well known that the environment is complicated that can be reflected in temperature, humidity, light, etc. These factors, in turn, affect the accuracy and reliability of sample representations. For example, a photo taken in a dimly lit state is completely different from a photo taken in a well-lit state. At the same time, the harsh environment might affect the equipment, so the quality of the acquired data becomes poor and contains noise. Finally, the environment might make it too dangerous or costly to acquire data. This might lead to an imbalance dataset, and influence the model training further.*

Source 2 (Uncertainty inherits to the data acquisition) *The data acquisition requires two steps, generally. The first is to collect information about the samples, and the second is to annotate the samples. Most of the sample information is collected using sensors, such as sound, humidity, and temperature. These artificially manufactured sensors generate more or less noise, which is difficult to distinguish and difficult to reduce. In addition, the mistake in the annotation can also lead to noise.*

Source 3 (Uncertainty caused by confusing samples) *Classification uncertainty occurs when multiple classes have similar probabilities for IM samples, or testing samples are OOD. It is due to the mismatch between the training and testing dataset. For example, a car image is an OOD sample to a cat-dog model while a dog image that comes from a new breed might be an IM sample.*

Source 4 (Uncertainty inherits to the model's architecture) *The model architecture has a distinct impact on prediction accuracy and uncertainty. For example, deeper models bring higher accuracy, but also face the risk of overfitting which is a cause of uncertainty. Also, various neural networks, such as Convolutional Neural Network (CNN), and Multilayer Perceptron (MLP), have their own characteristics and use context. An inappropriate architecture can affect the accuracy of the prediction and result in high uncertainty. Besides, the inductive assumptions about the NN might also lead to uncertainty.*

Source 5 (Uncertainty inherits to the training process) *Model training is a stochastic process since it can bring different models due to different configurations. There are many parameters that affect training, such as learning rate, dropout rate, and batch size. It is difficult to guarantee the model reaches the global optimum instead of a local one.*

Source 6 (Uncertainty inherits to the inference process) *Uncertainty also arises during the inference process. These uncertainties include the migration of the platform, for example, from Windows to Linux. The second might be due to improper usage, e.g., a hyperparameter is forgotten to be turned off or on. The third may be due to the trained model being applied in a context for which it was not intended. These errors are mostly human controllable and in this thesis, it is assumed that such errors will not occur.*

1.2.3 Uncertainty modeling

This section gives standard uncertainty modeling approaches for two introduced uncertainties, i.e., model uncertainty and data uncertainty. The expansion of these approaches and their applications will be discussed in Section 3.1.

Both data uncertainty and model uncertainty are reflected in the model prediction which is so-called predictive uncertainty Eq. 1.6. The model uncertainty is formalized as a probability distribution over the model parameters $\boldsymbol{\theta}$, while the data uncertainty is formalized as a probability distribution over the model outputs $\boldsymbol{o} = f_{\boldsymbol{\theta}}(\boldsymbol{x})$. Define a traning dataset $\mathcal{D}^{train} = \{\boldsymbol{x}_i, y_i\}_{i=1}^{N}$ with K classes. The aim is to optimize the parameters $\boldsymbol{\theta}$, to ensure the model can produce the desired output. To achieve this, the Bayesian approach defines a model likelihood $P(y \mid \boldsymbol{x}, \boldsymbol{\theta})$. For classification, the softmax likelihood can be used:

$$P(y \mid \boldsymbol{x}, \boldsymbol{\theta}) = \frac{\exp\left(f_{\boldsymbol{\theta}}(\boldsymbol{x})\right)}{\sum_{i=1}^{K} \exp\left(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)\right)}. \tag{1.4}$$

The posterior distribution $P(\boldsymbol{\theta} \mid \boldsymbol{x}, y)$, for a given dataset by applying Bayes' theorem can be written as:

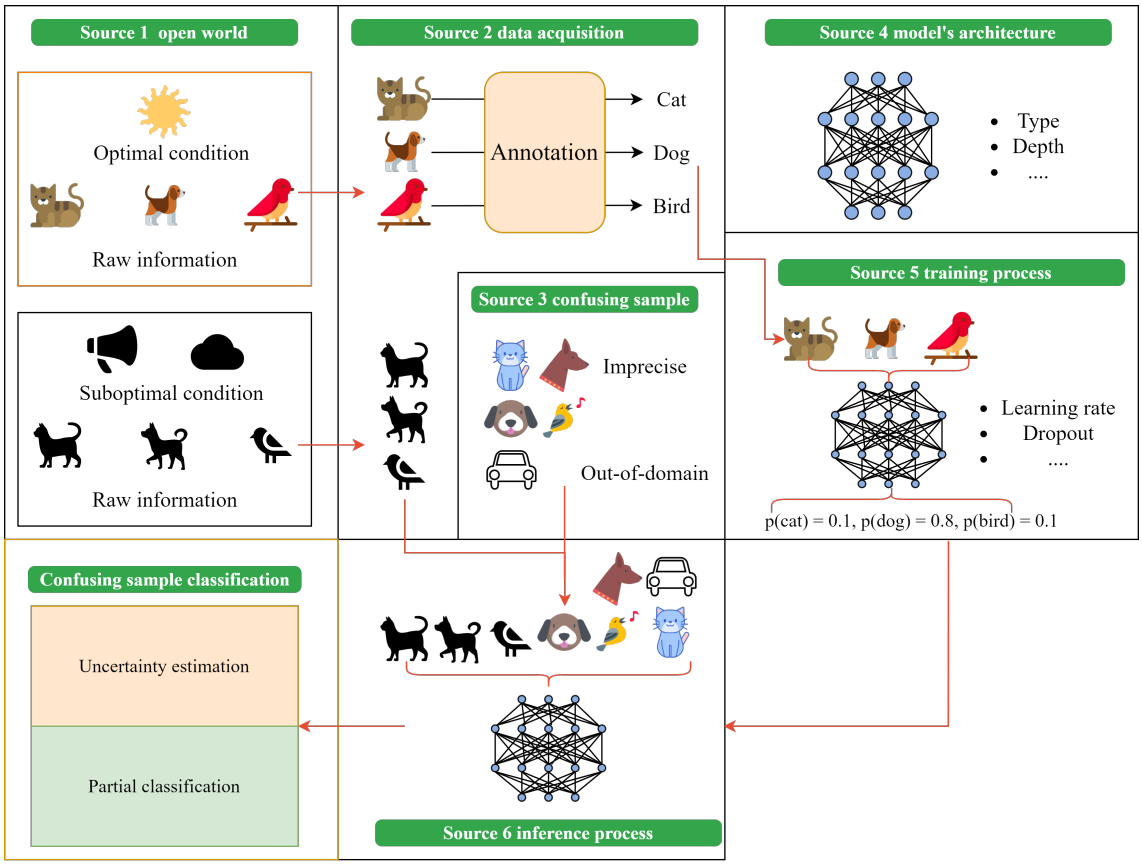


FIGURE 1.6: The confusing sample classification pipeline and underlying uncertainty sources. The blue boxes indicate the six possible sources of uncertainty in each step. The red line represents the data flow among each source.

$$P(\boldsymbol{\theta} \mid \boldsymbol{x}, y) = \frac{P(y \mid \boldsymbol{x}, \boldsymbol{\theta}) P(\boldsymbol{\theta})}{P(y \mid \boldsymbol{x})}. \tag{1.5}$$

For a given testing sample $\boldsymbol{x}^*$, the predictive class label $\hat{y}$ with regard to the $P(\boldsymbol{\theta} \mid \boldsymbol{x}, y)$ can be predicted:

$$P(\hat{y} \mid \boldsymbol{x}^*, \boldsymbol{x}, y) = \int \underbrace{P(\hat{y} \mid \boldsymbol{x}^*, \boldsymbol{\theta})}_{Data} \underbrace{P(\boldsymbol{\theta} \mid \boldsymbol{x}, y)}_{Model} \mathrm{d}\boldsymbol{\theta}. \tag{1.6}$$

1.3 Research subjects and challenges

In this thesis, we focus on the problem of how to detect confusing samples and make the right classification to improve model robustness and reduce the misclassification risk. Overall, we consider two categories:

- **Uncertainty estimation** which has an enormous need in various industries for different applications and model architectures. Based on it, we propose to compute a scalar value representing uncertainty and decide whether to reject the input sample. By rejecting the confusing samples, it is possible to reduce the classification error and the consequent risk of misclassification.

- **Partial classification** is another perspective to deal with confusing samples, which can classify the confusing samples into subsets. It is valuable since rejecting a sample according to a scalar value sometimes is undesirable in many cases, e.g., autonomous during, and medical images classification. This thesis aims to propose simpler and more efficient methods to fulfill partial classification only based on the model output. These methods can provide evidence for subsequent manual classification and make more cautious predictions.

We introduce difficulties [Man20] and challenges of the two introduced research subjects in the context of image classification based models.

1. **Insufficient of the point estimation.** The CNN based image classification model merely produces point estimations as outputs without any measure of uncertainty.
2. **Overfitting inherits to the softmax function.** Due to the additivity principle of the softmax function, even feeding an OOD sample, the model will still generate a probability for each training class. These probabilities will lead to overfitting and misclassification.
3. **Insufficient usage of the base rate.** The Evidential Deep Learning (EDL) [SKK18] is an approach of uncertainty estimation published on NIPS 2018, which is based on the Subjective Logic (SL). It constructs a loss function by the mapping between model parameters and the Dirichlet distribution. The model output is regarded as evidence for uncertainty calculation. The advantage of SL compared to DST is the introduction of the base rate, but the EDL method does not explicitly use the base rate.
4. **Requirement of the additional OOD dataset.** Generally, the easiest way to detect the OOD samples is to train the model by the ID and OOD samples. For instance, energy socre [LWOL20] is one of the existing uncertainty estimation methods, which needs the additional OOD dataset to train the model. However, sometimes, the additional OOD dataset is not reachable.
5. **Improper rejection.** It is simple to reject a sample according to the calculated uncertainty value. Sometimes despite the high uncertainty of a sample, direct rejection will have the potential for unpredictable losses. For example, in situations such as medical image classification and autonomous driving, it is proper to raise the high uncertainty upwards and leave it to manual processing.
6. **The inability the partial classification to distinguish between sample with "total ignorance" about its class membership, which needs to be classified into the empty set, and sample with "total imprecision" about its class membership, which needs to be classified into the entire set.** The existing methods [MD21, TXD21], tend to produce belief for the entire set, which will lose effectiveness when the "total ignorance" and the "total imprecision" appear simultaneously.
7. **Lack of interchangeability between the partial classification and the uncertainty estimation.** In reality, different scenarios have different requirements. For example, in the task of identifying cats from dogs, it is sufficient to detect the OOD sample, e.g., a car image. Whereas, for tumor image classification, tumor images need to be partially classified into different class subsets in order to make cautious decisions. Consequently, there is a demand for the combination of these two kinds of methods to cope with various scenarios. Although, some methods based on the DST, e.g., [MD21, TXD21] have the ability, but they did not mention nor fulfill it yet.

8. **Practicality and conveniently.** Both uncertainty estimation and partial classification methods should be fast and packageable as an auxiliary module which can be integrated on top of existing models. Consequently, these methods should not introduce major architectural changes that would cause the retraining and hyperparameter tuning of the pre-trained model.

1.4 Contributions and organization

The outlines of this thesis are shown in Fig. 1.7 consisting of two parts. In part I, we present one contribution for uncertainty estimation. In part II, we present two contributions for partial classification.

- **Part I:**
  1. **Uncertainty estimation based on subjective logic – SLUE [XCA21]** Based on the EDL method and taking the base rates explicitly into account, we proposed the SLUE method which has been presented at the *IJCNN2021* conference. The initialization of the base rate is evaluated. A comprehensive analysis with experiments is carried out. In addition, we explore and find the optimum configuration of the hyperparameter $\mathcal{C}$.
- **Part II:**
  1. **Partial classification based on DST – PCMO [XAC21]** We fulfilled Partial Classification only based on pre-trained Model Outputs (PCMO), by transforming the model outputs to beliefs for predicted sets under the DST. The most striking achievement is that the proposed method is fulfilled only based on model outputs that can be applied to any model without any demand to retrain the model or conduct any further modifications. By considering good features of log function and analyzing the regular pattern of model outputs, a novel and reasonable transformation from model outputs to possibility distribution is proposed. This led to a paper that has been presented at the *ICONIP2021* conference.
  2. **Partial classification based on DST and Bell Shaped Function (BSF) – PCBS.** We proposed a new partial classification method called PCBS, which combines the BSF and the DST. First, generate the BSF based on the training output. Second, the testing output is converted into the possibility under the obtained BSFs. Finally, the possibility is transformed into the belief to perform partial classification. Additionally, with the help of pignistic transformation, the calculated belief can be converted into probability to achieve the uncertainty estimation. This paper is under review by the *Pattern Recognition* journal.

The rest of this thesis is organized into five chapters:

- Chapter 2: the relative prerequisites to understand the proposed approaches, e.g., SL, DST, and BSF.
- Part I Chapter 3: A literature review concerning the recent advances in uncertainty estimation approaches.
- Part I Chapter 4: Description of the proposed uncertainty estimation method.
- Part II Chapter 5: A literature review concerning the recent advances in partial classification approaches.

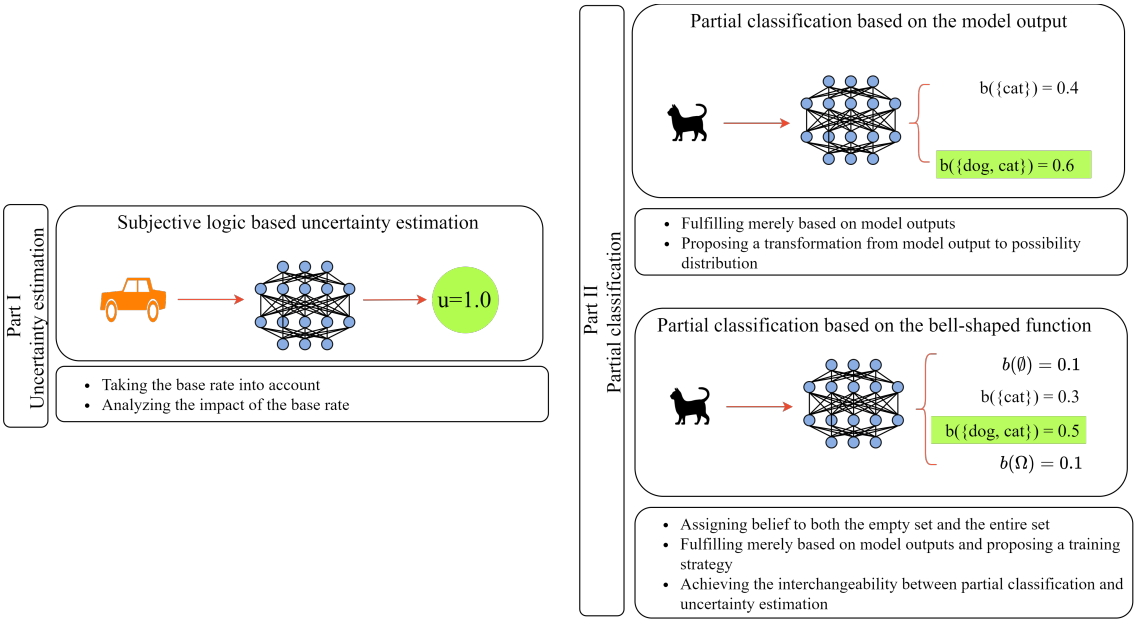


FIGURE 1.7: The contributions and organization of this thesis. The contributions of each chapter are listed in bullet points.

- Part II Chapter 6: Description of the proposed partial classification methods.

Chapter 2

Backgrounds

This chapter start with an introduction of Neural Network (NN)s, follows several uncertainty reasoning frameworks including Dempster-Shafer Theory (DST), Subjective Logic (SL), probability theory, fuzzy theory [Zad65], possibility theory [Zad99], and imprecise probability [Wal91]. Then the Dirichlet distribution, membership function, e.g., Bell Shaped Function (BSF), and contrastive-center loss are presented. The Dirichlet distribution or multivariate beta distribution is a continuous distribution over another distribution, it is used to approximate the distribution of the model outputs. The DST is adopted to calculate the belief of predicted sets during the partial classification process. We show the belief, plausibility, pignistic probability calculation, and belief fusion for multi-agents. The SL is another uncertainty reasoning framework, which focuses on the usage of the base rate, i.e., prior probability. Finally, the BSF is presented, which can perform the output to possibility transformation by mapping high-frequency values to one, and low-frequency values to zero.

2.1 Multilayer perceptron and convolutional neural network

In this section, we introduced two NN architectures used in this thesis, i.e., Multilayer Perceptron (MLP) and Convolutional Neural Network (CNN). Both two NNs are deterministic NNs, the parameters are deterministic and each repetition of a forward pass delivers an identical outcome. The NN in machine learning imitates the characteristics of animal NNs. It relies on the complexity of the system to process information by adjusting the relationship between a large number of internal neurons. The perceptron is the basis of NNs. The MLP or CNN extends the perceptron by adding and redesigning multiple hidden layers between the input and output layers. Let us start with an introduction of the perceptron.

A perceptron as shown in Fig. 2.1 has d inputs corresponds to the input feature dimension, each of which associates with a weight $\boldsymbol{\theta}_i$. The inputs are summed within the neuron after multiplying them with the weights. The summation result is subtracted with the bias $\boldsymbol{\theta}_0$. The result is eventually put into an activation function, which gives the final output with 0 representing inhibition and 1 representing activation. A perceptron can be trained by starting with random weights and iteratively applying the perceptron to each training sample. Modifying the perceptron's weights when it misclassifies the sample Eq. 2.1. This process is repeated until reach the given iteration number.

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \Delta\boldsymbol{\theta}, \quad \Delta\boldsymbol{\theta} = \eta(y - \hat{y})\boldsymbol{x}, \tag{2.1}$$

where η represents the learning rate.

The multilayer perceptron as shown in Fig. 2.2 is a NN that contains at least one hidden layer and the output of each hidden layer is transformed by an activation function. In

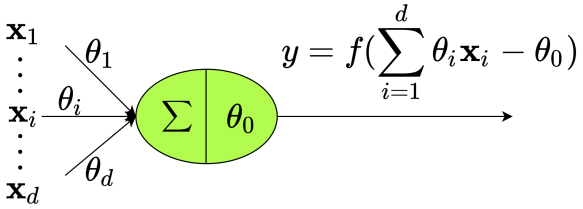


FIGURE 2.1: An example of the perceptron.

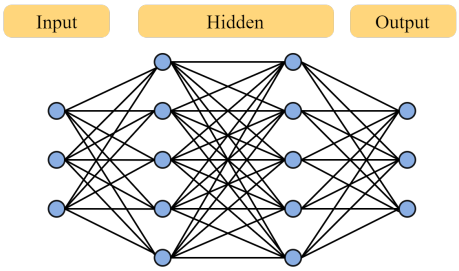


FIGURE 2.2: An example of the multilayer perceptron.

the classification problem, the cross-entropy function is usually used as the loss function. And the optimum problem is solved using gradient descent or its variations.

Different from the MLP which can only process a set of discrete values. The CNN is invented for image processing. It can effectively map the high dimensional images into low dimensional discrete values while preserving image features. The Fig. 2.3 shows an example of convolution process. Given an image sized 8×8 and a need to perform edge detection. We can design a filter or kernel of size 3×3. Then, using this filter to cover the image with a region as large as the filter. Summing up the values after multiplying the corresponding values in the region as the extracted feature. Moving the filter by a predefined stride and continue computing until every corner of the original image is covered. The prevelant CNNs are LeNet [LBBH98], VGG [SZ15], GoogLeNet [SLJ+15], ResNet [HZRS16], MobileNet [HZC+17]. LeNet [LBBH98] can be considered as the pioneer of CNN with a total of seven layers. LeNet using a combination of three layers, i.e., convolution, pooling, and nonlinear mapping to extract the spatial features. VGG [SZ15] was proposed in 2014 by K. Simonyan et al, and the main feature is “very deep". The VGG use the same size convolutional kernel (3×3) and pooling (2×2) throughout the network. This connection of 3×3 convolutional layers allows the network to have a smaller number of parameters and the multi-layer activation function allows the network to learn more about the features. GoogLeNet [SLJ+15] changes the previous model of sequential computation by allowing multiple layers to operate in parallel then integrate them. The use of different sized convolutional kernels implies different sized receptive fields and finally the fusion of features at different scales is achieved by stitching. The reason why the convolution kernel sizes of 1, 3 and 5 are used is mainly to facilitate alignment. ResNet [HZRS16] introduces a residual block used to solve the gradient explosion or gradient disappearance. This residual block takes the activation value of one layer and directly grabs it before the next layer. In order to deepen the structure of the network so that each time finer features can be learned and thus improve the accuracy of the network, one thing that needs to be achieved is a constant mapping. The MobileNet [HZC+17] is a lightweight CNN proposed for embedded devices such as mobile phones. The main innovation is the use of depthwise separable convolution, which is the decomposition of

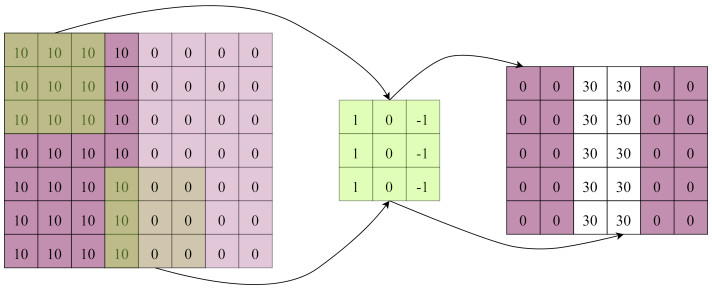


FIGURE 2.3: An example of the convolution process.

a standard convolution into a depthwise convolution and a pointwise convolution.

2.2 Dempster-Shafer theory

The DST or evidence theory [Dem67] was proposed by A.P. Dempster in the 1960s to solve multi-valued mapping problems using lower and upper probabilities. It was further expanded and improved by G. Shafer, who introduced the concept of the belief function and a set of belief fusion equations to deal with uncertainty reasoning, which is called Dempster-Shafer Theory (DST) [Sha76]. The DST can represent uncertainty and is mainly applied to information fusion [FXFL19, YCVPK21], expert systems [BCM01, MH12], and decision analysis [HAAZ21, RGP18].

2.2.1 Basic concepts

Concept 1 (State space) *The state space Ω is a domain consisting of a set of values $\boldsymbol{\omega}$ which can also be called states, events, outcomes, hypotheses, or propositions. The different values of a state space are assumed to be exclusive and exhaustive. That means the state can only be in one of all possible states included in the domain at any moment in time. A state space can be binary (with exactly two values) or K-ary (with $K > 2$ values). For example, $\Omega = \{\omega_1, \ldots, \omega_i, \ldots, \omega_K\}$ denotes the state space of K training classes, $\mathcal{P} = 2^{\Omega}$ represents the powerset of Ω composed by all the subset of Ω including the empty set $\emptyset$, and $\mathcal{R} = 2^{\Omega} \setminus \{\emptyset, \Omega\}$ represents the hyperdomain of Ω.*

Concept 2 (Basic Belief Assignment (BBA)) *The BBA or Basic Probability Assignment (BPA) $m : \mathcal{P} \rightarrow [0, 1]$ applied on a sample $\boldsymbol{x}$ measures the degree of belief that the ground-truth label of $\boldsymbol{x}$ belongs to a subset $A \in \mathcal{P}$. It satisfies the following constraint.*

$$\sum_{A \in \mathcal{P}} m(A) = 1, \tag{2.2}$$

where for any $A \in \mathcal{P}$, $m(A)$ represents the belief that one of the classes in A is true. $m(\emptyset) = 0$ represents normalized BBA, otherwise denotes unnormalized BBA.

If m is unnormalized, two preliminary methods can apply:

1. The Dempster's normalization consists in dividing all the masses given to nonempty sets by $m(\emptyset)$ [MD08].

$$m(A) = \begin{cases} \frac{m(\mathbf{A})}{1-m(\emptyset)} & \text{if} \quad A \neq \emptyset, \\ 0 & \text{otherwise.} \end{cases} \tag{2.3}$$

2. Yager's normalization in which the mass $m(\emptyset)$ is transferred to $m(\Omega)$ [Yag96].

$$m(A) = \begin{cases} 0 & \text{if} \quad A = \emptyset, \\ m(\Omega) + m(\emptyset) & \text{if} \quad A = \Omega, \\ m(A) & \text{otherwise.} \end{cases} \tag{2.4}$$

Concept 3 (The consonant BBA) *The subset A such that $m(A) > 0$ is called the focal set of m. When the focal set is nested, m is said to be* consonant *[DP82].*

Concept 4 (The belief or credibility function) *The quantity of the belief function $bel(A)$ can be interpreted as a global measure of one's belief that hypothesis A is true.*

$$bel(A) = \sum_{B \subseteq A, B \neq \emptyset} m(B), \quad m(\emptyset) = 0, \tag{2.5}$$

where B represents any subset that belongs to $\mathcal{P}$.

Concept 5 (The plausibility function) *The quantity of the plausibility function $pl(A)$ can be viewed as the amount of belief that could potentially be placed in A.*

$$pl(A) = \sum_{A \cap B \neq \emptyset} m(B). \tag{2.6}$$

Concept 6 (The pignistic transformation) *The only transformation satisfying elementary rationality requirements to be the pignistic transformation [SK94] defined as Eq. (2.7), in which $m(A)$ is distributed equally among the elements of A for all $A \in \mathcal{P}$.*

$$BetP(\omega) = \sum_{\omega \in A} \frac{m(A)}{\mid A \mid}, \quad \forall \omega \in \Omega, \tag{2.7}$$

where $\mid A \mid$ denotes the cardinality of subset A.

For decision making, as we showed previously, a straightforward way is to choose the class that occupies the maximum pignistic probability Eq. 2.8.

$$\hat{y} = \max(BetP(\omega_1), \cdots, BetP(\omega_K)). \tag{2.8}$$

At the same time we can also make the decision by minimum the decision risk:

$$\mathcal{R}_{BetP}(\hat{y}) = \sum_{\omega \in \Omega} \ell(\hat{y} \mid \omega) \, \mathrm{BetP}(\omega). \tag{2.9}$$

In order to better understand the above concepts, we give the following example. Suppose a sensor tries to recognize the color of an object at a long distance, which can be only one of $\Omega = \{red, blue, green\}$. After analysis, the sensor provides the corresponding beliefs for each subset of Ω as demonstrated in the second column in the Table 2.1. Calculating the belief and the plausibility values for each subset belonging to Ω based on the BBA gives the third and fourth columns in the Table 2.1. Pick subset $A = \{red, blue\}$, its belief value is $m(\{red\}) + m(\{blue\}) + m(\{red, blue\}) = 0.35 + 0.25 + 0.06 = 0.66$. Also, its plausibility value is $m(\{red\}) + m(\{blue\}) + m(\{red, blue\}) + m(\{red, green\}) + m(\{blue, green\}) + m(\Omega) = 0.35 + 0.25 + 0.06 + 0.05 + 0.04 + 0.1 = 0.85$. For subset A, the closed interval $[bel(A), pl(A)]$ indicating the confidence degree of A.

$\mathcal{P}$	BBA	*bel*	*pl*
$\emptyset$	0	0	0
$\{red\}$	0.35	0.35	0.56
$\{blue\}$	0.25	0.25	0.45
$\{red, blue\}$	0.06	0.66	0.85
$\{green\}$	0.15	0.15	0.34
$\{red, green\}$	0.05	0.55	0.75
$\{blue, green\}$	0.04	0.44	0.65
Ω	0.1	1	1

TABLE 2.1: An example of basic belief assignment and the derived *bel* and *pl* values. A composite subset such as $\{red, blue\}$, it means that the actual object color is either *red* or *green*, but not both at the same time.

2.2.2 Uncertainty measurements

The entropy based measures are used to measure the uncertainty of BBA, Here we just list several commons used for further reading please refer to [Den20, HK82].

$$\begin{aligned} \text{Hohle entropy:} \quad & u^{Hohle} = -\sum_{A\in\mathcal{P}} m(A)\log_2 bel(A), \\ \text{Yager's dissonance measure:} \quad & u^{Yager} = -\sum_{A\in\mathcal{P}} m(A)\log_2 pl(A). \end{aligned} \tag{2.10}$$

Beside, Yang et al. introduced in [YHD16] a new measurement based on belief intervals.

$$u^{Yang} = \frac{1}{K}\sum_{i=1}^{K}\left(pl\left(\omega_i\right) - bel\left(\omega_i\right)\right). \tag{2.11}$$

2.2.3 Belief fusion

Suppose there are n BBAs, i.e., $m_1, \ldots, m_n$. For any $A \in \mathcal{P}$, the DST belief fusion rule is:

$$\left(m_1 \oplus \cdots \oplus m_n\right)(A) = \frac{1}{1-\kappa}\sum_{A_1\cap\cdots\cap A_n = A} m_1\left(A_1\right)\cdots m_n\left(A_n\right), \tag{2.12}$$

where κ Eq. (2.13) represents the conflict among different evidence provided by BBAs, called conflict probability.

$$\kappa = \sum_{A_1\cap\cdots\cap A_n = \emptyset} m_1\left(A_1\right)\cdots m_n\left(A_n\right). \tag{2.13}$$

Taking the previous example $\Omega = \{red, blue, green\}$, but suppose there are two sensors that provide different beliefs for the object color as shown in Table 2.2.

At first, we can calculate the conflict probability $\kappa = 0.98$, then we can calculate the fused BBA, *bel*, and *pl* values for each subset. Based on the fused belief, we can infer the object's color is red (or green).

$$\begin{aligned} \kappa &= \sum_{A\cap B=\emptyset} m_1(A)\cdot m_2(B) \\ &= m_1(red)\cdot m_2(blue) + m_1(red)\cdot m_2(green) + m_1(blue)\cdot m_2(green) \\ &= 0.98\times 0.01 + 0.98\times 0.98 + 0.01\times 0.98 = 0.98. \end{aligned} \tag{2.14}$$

	m_1	m_2	m_{12}	*bel*	*pl*
$\{red\}$	0.98	0.00	**0.49**	0.49	0.495
$\{blue\}$	0.01	0.01	0.015	0.015	0.02
$\{green\}$	0.00	0.98	**0.49**	0.49	0.495
Ω	0.01	0.01	0.005	1	1

TABLE 2.2: An example of belief fusion in DST. m_1 represents the BBA of the first sensor, m_2 represents the BBA of the second sensor, and m_{12} represents the fused BBA. The *bel* and *pl* values are calculated from m_{12}.

$$\begin{aligned} m_1 \oplus m_2(\{red\}) &= \frac{1}{1-\kappa} \sum_{A \cap B = \{red\}} m_1(A) \cdot m_2(B) \\ &= \frac{1}{1-\kappa}[m_1(\{red\}) \cdot m_2(\{red\}) + m_1(\{\Omega\}) \cdot m_2(\{red\}) + m_1(\{red\}) \cdot m_2(\{\Omega\})] \\ &= \frac{1}{0.02} \times (0.98 \times 0 + 0.01 \times 0 + 0.98 \times 0.01) = 0.49. \end{aligned} \tag{2.15}$$

2.2.4 Limitations of the DST

It is clear from the concepts presented above. The BBA is defined based on the power set of Ω. On the one hand, it can give beliefs that can cover all sets without missing any potential choice. On the other hand, this is a limitation of the DST. Consider the classification as an example, the number of sets grows exponentially as the number of classes increases. This undoubtedly increases the computational burden. An available solution is to give belief to only some specific sets, e.g., sets composed of similar classes.

2.3 Subjective logic

The Subjective Logic (SL) [Jøs18] is an uncertain reasoning framework that was initially introduced by A. Jøsang to address representations of trust. It is directly compatible with probabilistic logic. The SL allows for more realistic modeling of real-world situations and the conclusions drawn can more accurately reflect the uncertainty of the input samples. The analyst's uncertainty and lack of evidence can be taken into account during the analysis and expressed in the conclusion.

2.3.1 Basic concepts

There is some overlap concepts between DST and SL, e.g., state space, belief. The presented names are identical, however, they are mathematically different. We choose to keep the names as they are defined in the book [Jøs18]. In addition, we also adopt the same example "recognize object color by sensors" as used in Section 2.2 to explain the SL concepts.

Concept 7 (Belief mass distribution) *The belief mass distribution applies to the state space* Ω *is defined as follows.*

$$u + \sum_{\omega \in \Omega} \boldsymbol{b}(\omega) = 1, \quad \boldsymbol{b}(\omega) \in [0, 1], \tag{2.16}$$

where the uncertainty u can be regarded as the belief assigned to the state space itself. For example, $(\boldsymbol{b} = (0.5, 0.1, 0.1), u = 0.3)$ *represents the belief and uncertainty for object color give by the sensor.*

Concept 8 (Base rate distribution) *The base rate or prior probability distribution* $\boldsymbol{a}$ *assigns base rates to classes of* Ω *and respects the additivity principle expressed as:*

$$\sum_{\omega \in \Omega} \boldsymbol{a}(\omega) = 1, \quad \boldsymbol{a}(\emptyset) = 0, \quad \boldsymbol{a}(\omega) \in [0, 1]. \tag{2.17}$$

Given a state space Ω of cardinality K, the default base rate of each singleton value in the state space is $\frac{1}{K}$. In contrast to default base rates, it is possible and useful to apply realistic base rates that reflect real background probabilities in practical situations. Base rates can also be dynamically updated as a function of observed evidence. From a technical point of view, base rates are simply probabilities. From a semantic point of view, base rates are non-informative prior probabilities estimated as a function of general background information for a class of variables. Base rates make it possible to define a bijective mapping between opinions and Dirichlet Probability Density Function (PDF), and are used for probability projections [Jøs18]. Another interesting question about base rate is whether base rate can be regarded as a kind of belief. It is an open question. In my opinion, the base rate is a kind of belief in mathematical expression. The reason is that the base rate respects the constraints declared by the belief. However, they have different meanings, the base rate is a kind of prior probability, it represents the evidence without any observations, while belief represents the posterior evidence with the observations.

2.3.2 Notations of opinions

Based on the basic concepts, we can introduce the composite function called "opinion" defined by the SL. This thesis merely focuses on the multinomial opinion of the SL, i.e., opinions which are based on belief mass distributions over a state space Ω. For more information about the other opinion formats, e.g., binomial opinion and hyper opinion please refer to [Jøs18].

Concept 9 (Belief notation of opinions) *A multinomial opinion over the state space* Ω *is an ordered triplet* $\mathcal{S} = \{\boldsymbol{b}, u, \boldsymbol{a}\}$.

Concept 10 (Probability expectation function) *Let* $\mathcal{S} = \{\boldsymbol{b}, u, \boldsymbol{a}\}$ *be a belief notation of opinions, then the posterior probability expectation* $\boldsymbol{p}$ *from* Ω *to* $[0, 1]$ *can be expressed as:*

$$\boldsymbol{p} = \boldsymbol{b} + \boldsymbol{a}u. \tag{2.18}$$

The belief mass on the whole state space u is the only belief mass to be distributed. The base rate $\boldsymbol{a}$ represents the relative share that each element receives. It can be seen that $\boldsymbol{p}$ satisfies the additivity principle:

$$\sum_{\omega \in \Omega} \boldsymbol{p}(\omega) = 1, \quad \boldsymbol{p}(\emptyset) = 0. \tag{2.19}$$

If we use the default base rate $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, then the probability obtained for object color is $(0.6, 0.2, 0.2)$. A weakness of the belief notation is that it does not directly reflect the probability expectation values of the various elements in the state space. An intuitive representation of multinomial opinions could be to represent the probability expectation value directly, together with the degree of uncertainty and the base rate. This will be called the probabilistic notation of opinions.

Concept 11 (Probabilistic notation of opinions) *Let $\mathcal{S} = \{\boldsymbol{b}, u, \boldsymbol{a}\}$ be a belief notation of opinions, and the probability expectation $\boldsymbol{p}$ defined according to Concept. 10. The probabilistic notation of opinions can then be expressed as the ordered tuple $\mathcal{S} = \{\boldsymbol{p}, u, \boldsymbol{a}\}$.*

Concept 12 (Probabilistic notation equivalence) *Let $\mathcal{S}_b = (\boldsymbol{b}, u_b, \boldsymbol{a}_b)$ be an opinion expressed in belief notation, and $\mathcal{S}_p = (\boldsymbol{p}, u_p, \boldsymbol{a}_p)$ be an opinion expressed in probabilistic notation, both over the same state space. Then the following equivalence holds:*

$$\boldsymbol{p} = \boldsymbol{b} + \boldsymbol{a}_b u_b, \quad u_p = u_b, \quad \boldsymbol{a}_p = \boldsymbol{a}_b. \tag{2.20}$$

The evidence notation of opinions is centered around the Dirichlet multinomial probability distribution. In order to contain the base rate in the Dirichlet parameters, we represent the Dirichlet multinomial distribution below according to Eq. (2.30).

Concept 13 (Augmented Dirichlet notation) *Let Ω be a state space, $\boldsymbol{r}$ represent the evidence vector over the elements of Ω, $\mathcal{C}$ represents a prior constant, and $\boldsymbol{a}$ represent the base rate vector over the same elements. Then the multinomial Dirichlet density function over Ω can be expressed in augmented notation as:*

$$P(\boldsymbol{p} \mid \boldsymbol{r}, \boldsymbol{a}) = \frac{\Gamma\left(\sum_{i=1}^{K} (\boldsymbol{r}_i + \mathcal{C}\boldsymbol{a}_i)\right)}{\prod_{i=1}^{K} \Gamma\left(\boldsymbol{r}_i + \mathcal{C}\boldsymbol{a}_i\right)} \prod_{i=1}^{K} \boldsymbol{p}_i^{(\boldsymbol{r}_i + \mathcal{C}\boldsymbol{a}_i - 1)}. \tag{2.21}$$

Concept 14 (Evidence notation of opinions) *Let Ω be a state space, the evidence notation of opinions can then be expressed as the ordered tuple $(\boldsymbol{r}, \boldsymbol{a})$.*

Concept 15 (Evidence notation equivalence) *Let $\mathcal{S}_b = (\boldsymbol{b}, u_b, \boldsymbol{a}_b)$ be an opinion expressed in belief notation, and $\mathcal{S}_e = (\boldsymbol{r}, \boldsymbol{a}_e)$ be an opinion expressed in evidence notation, both over the same state space. Then the following equivalence holds:*

$$\boldsymbol{b} = \frac{\boldsymbol{r}}{\mathcal{C} + \sum_{i=1}^{K} r_i}, \quad u = \frac{\mathcal{C}}{\mathcal{C} + \sum_{i=1}^{K} r_i}. \tag{2.22}$$

As we can see from the above equation, the hyperparameter $\mathcal{C}$ impacts the uncertainty calculation. For a sample, if we fix the denominator, we can get a larger uncertainty. Of course, it is not the case that a larger $\mathcal{C}$ is better. The larger the $\mathcal{C}$ for an In-Domain (ID) sample the higher the probability of classifying it as Out-of-Domain (OOD). The effect of $\mathcal{C}$ is also interpreted in Section 4.2.3.2.

Table 2.3 provides the equivalent interpretation for a set of multinomial opinions represented in belief notation, probabilistic notation, and evidence notation.

Belief notation	Probabilistic notation	Evidence notation
$(b_1, b_2, b_3), u, (a_1, a_2, a_3)$	$(p_1, p_2, p_3), u, (a_1, a_2, a_3)$	$(r_1, r_2, r_3), (a_1, a_2, a_3)$
$(1, 0, 0), 0, (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$	$(1, 0, 0), 0, (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$	$(\infty, 0, 0), (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$
$(0, 0, 0), 1, (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$	$(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}), 0, (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$	$(0, 0, 0), (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$
$(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}), \frac{1}{4}, (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$	$(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}), \frac{1}{4}, (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$	$(3, 3, 3), (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$

TABLE 2.3: An example of three equivalent notations of multinomial opinion.

2.3.3 Evidence opinions fusion

In order to provide an interpretation of opinion fusion in SL, consider the object color is observed by two sensors. A distinction can be made between the two cases.

- **Cumulative fusion:** The two sensors observe the object's color during disjoint time periods. In this case, the observations are independent and it is natural to simply add the observations from the two sensors.
- **Averaging fusion:** The two sensors observe the object's color during the same time period. In this case, the observations are dependent and it is natural to take the average of the observations by the two sensors.

Let the two observers' respective observations be expressed as $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$. The evidence opinions resulting from these separate bodies of evidence can be expressed as $(\boldsymbol{r}_1, a)$ and $(\boldsymbol{r}_2, a)$ The cumulative fusion of these two bodies of evidence simply consists of the vector addition of $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$, expressed as:

$$(\boldsymbol{r}_1, \boldsymbol{a}) \oplus (\boldsymbol{r}_2, \boldsymbol{a}) = ((\boldsymbol{r}_1 + \boldsymbol{r}_2), \boldsymbol{a}). \tag{2.23}$$

The averaging fusion of these two bodies of evidence simply consists of averaging $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$, expressed as:

$$(\boldsymbol{r}_1, \boldsymbol{a}) \oplus (\boldsymbol{r}_2, \boldsymbol{a}) = \left(\frac{(\boldsymbol{r}_1 + \boldsymbol{r}_2)}{2}, \boldsymbol{a}\right). \tag{2.24}$$

2.3.4 Comparison between SL with DST

Consider a domain Ω with its hyperdomain $\mathcal{R}(X)$ and powerset $\mathcal{P}(X)$, and ω denote a specific value of $\mathcal{R}(X)$ or $\mathcal{R}(X)$. In DST, the belief assigned to state ω is indicated by $m(\omega)$. It is possible to define a direct bijective mapping between the of DST and the belief mass distribution of the SL, e.g., $b(\omega)$ [Jøs18], as following:

$$\begin{cases} m(\omega) & = b(\omega), \quad \forall \omega \in \mathcal{R}, \\ m(\Omega) & = u, \quad \text{otherwise.} \end{cases} \tag{2.25}$$

Technically, the BPA of DST and the opinion notation of SL are somewhat equivalent. The differences exist in interpretations. The SL can not assign belief to the domain Ω itself. That is due to the Dirichlet model, where only observations of ω are counted as evidence. The domain Ω itself can not be an observation, and hence can not be counted as evidence. Another different point is that the base rate is not part of DST, so the pignistic transformation uses the default base rate to calculate the probability.

There is also the common part, for example, both frameworks is definied based on the state space. In addition, both DST and SL proposed several belief fusion operators.

2.4 Other uncertainty reasoning frameworks

In this section, we introduce the other four uncertainty reasoning frameworks, i.e., probability theory, fuzzy theory [Zad65], possibility theory [Zad99], and imprecise probability [Wal91].

2.4.1 Probability theory

Probability theory is a framework that is used to describe random events that satisfy the additivity principle. The probability of a random event is a value between 0 and 1. The magnitude of the value reflects the likelihood of the event's occurrence. Depending on the type of outcomes which can be discrete or continuous, the probability distribution is said respectively discrete or continuous. For simplicity, we limit the review to the discrete case. Based on the probability distribution, we can measure the uncertainty through Shannon's entropy [Sha01]. Table 2.4 give an example of probability distribution.

$$\begin{aligned} \text{Probability distribution:} \quad & P(\omega) : \Omega \longrightarrow [0,1], \\ \text{Additivity principle:} \quad & \sum_{i=1}^{K} \boldsymbol{p}_i = 1, \\ \text{Uncertainty measurement:} \quad & \mathcal{H}(\boldsymbol{p}) = -\sum_{i=1}^{K} \boldsymbol{p}_i \log_v (\boldsymbol{p}_i), \end{aligned} \tag{2.26}$$

where v is the base of the logarithm.

Color	red	green	blue
Probability	0.2	0.3	0.5

TABLE 2.4: An example of the probability distribution of the object color.

2.4.2 Fuzzy theory

Different from the hard partition which forces a sample must belong to or exclude from a class. The fuzzy theory [Zad65] can provide a membership measure of the degree that a sample belongs to a class. A fuzzy set A in a state space Ω is characterized by a membership function $\mu_A : \Omega \longrightarrow [0,1]$. The value $\mu_A(\omega)$ represents the degree of membership of ω in A. For example, the statement "the object is red or green" is imprecise since we do not know the exact color. To model this imprecision, a fuzzy set $\{red, green\}$ can be used by associating membership values indicating the possible color as Table 2.5.

Color	red	green	blue
Fuzziness	0.5	0.4	0.1

TABLE 2.5: An example of the fuzziness distribution of each color given the fuzzy set $\{red, green\}$.

The idea of measuring uncertainty of fuzzy set without reference to probabilities began in 1972 with the work of Deluca and Termini [DT72], who defined the entropy of μ_A using Shannon's functional form Eq. 2.27. For other measurements, we recommend reading [PB94].

$$\mathcal{H}(\mu_A) = -\beta \sum_{i=1}^{K} \mu_A(\omega_i) \log_v \mu_A(\omega_i) + (1 - \mu_A(\omega_i)) \log_v (1 - \mu_A(\omega_i)), \tag{2.27}$$

where β is a normalizing constant.

2.4.3 Possibility theory

For a state space Ω, the possibility distribution $\pi(\omega)$ Eq. 2.28 measures the possibility of each state $\omega \in \Omega$. The possibility can be crisp, i.e., 1 if a sample belongs to a class, 0 otherwise, or fuzzy, i.e., a larger value represents a larger possibility. For example, the possibility of the object color is shown in Table 2.6.

$$\text{Possibility distribution:} \quad \pi(x) : \Omega \longrightarrow [0, 1]. \tag{2.28}$$

Color	red	green	blue
Possibility	1 (0.9)	1 (0.5)	0 (0.1)

TABLE 2.6: An example of the possibility distribution of the object color. The values outside the brace indicate the crisp possibility while the values inside the brace indicate the fuzzy possibility.

2.4.4 Imprecise probability theory

The imprecise probability theory [Wal91] is also known as the upper and lower bound probability theory, which claims the personal opinion of a statement should be expressed by a bound instead of a single value. The Table 2.7 give a probability bound for each color. Based on it we can derive the probability distribution by fixing two of them to calculate the left one. And the uncertainty is measured by the difference between two bounds.

Color	red	green	blue
Possibility	[0.5, 0.9]	[0.2, 0.3]	[0.1, 0.4]

TABLE 2.7: An example of the imprecise probability distribution of bound of each color.

2.5 Other backgrounds

2.5.1 Dirichlet distribution

The Dirichlet distribution was introduced in 1973 by Johann P. G. Lejeune Dirichlet and has a variety of applications in fields such as computer vision [MG18, SKK18] and natural language processing [SCL19, MWZY20]. The conjugate nature of the Dirichlet distribution, which means the posterior distribution is the same distribution as the prior distribution, makes the Dirichlet distribution often used as the prior distribution. There are several constructions of the Dirichlet distribution, for example, the Pólya urn model [Hop84]. Since the Dirichlet distribution is the promotion of the beta distribution in the high dimensional case. Let us understand it starting with an example of the beta distribution.

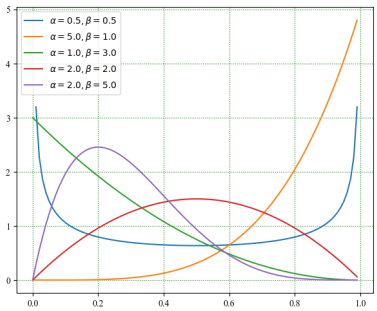
Consider a coin and we want to know whether it's fair. Thus flip it three times, and comes up heads all the time. According to the traditional probabilistic theory, the probability of head-up is $100\% = \frac{3}{3}$, which is contrary to our common sense of 50%, so we think the coin is unfair. The concern is that three tosses are a tiny experiment that might be a matter of luck. To deal with it, Bayes believes that before experimenting, one should make assumptions about the probability of a head-up. For example, if one obtains 500

heads and 500 tails from 1000 tosses, plus the three new results obtained, the probability of head-up toss is 50.3% $= \frac{503}{1000}$, we can tell the coin is fair. However, if one obtained 900 heads and 100 tails from 1000 tosses, plus the three new results obtained, the probability of head-up is 90.3% $= \frac{903}{1000}$, we can tell the coin is unfair. As we can see, different assumptions will lead to different inferences. Thus how should we give a reasonable prior distribution of head? The beta distribution determined by two parameters can give different prior distributions based on different parameters i.e., v_1 and v_2. That is to say, the beta distribution Eq. (2.29) and Fig. 2.4a is a distribution over the probability of head-up.

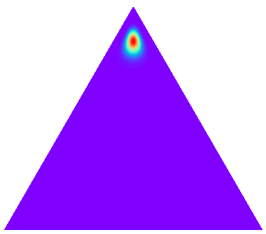
From coin game to image classification, it is interesting to know the probability distribution over the training classes. In the binary case, it is determined by the beta distribution. In high dimensional case, it is determined by the Dirichlet distribution. The Dirichlet distribution captures a sequence of observations of the K possible outcomes with K positive real parameters $\boldsymbol{\alpha}$, each corresponding to one of the possible outcomes. The Dirichlet PDF is then given by Eq. (2.30).

$$\begin{aligned} \Gamma(x) &= \int_0^{+\infty} t^{x-1} e^{-t}\, \mathrm{d}t (x > 0), \\ P(\boldsymbol{x}, a, b) &= \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} \boldsymbol{x}^{a-1} (1-\boldsymbol{x})^{b-1}. \end{aligned} \tag{2.29}$$

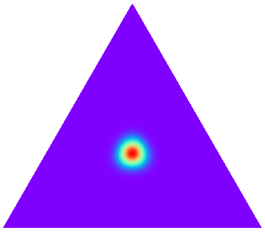
$$P(\boldsymbol{p} \mid \boldsymbol{\alpha}) = \frac{\Gamma\left(\sum_{i=1}^{K} \boldsymbol{\alpha}_i\right)}{\prod_{i=1}^{K} \Gamma\left(\boldsymbol{\alpha}_i\right)} \prod_{i=1}^{K} \boldsymbol{p}_i^{\boldsymbol{\alpha}_i - 1}, \quad \boldsymbol{\alpha} > 0. \tag{2.30}$$



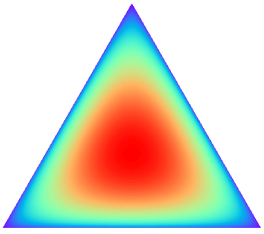
(A) The beta PDF.



(B) The Dirichlet PDF, $\boldsymbol{\alpha} = \{10, 10, 100\}$.



(C) The Dirichlet PDF, $\boldsymbol{\alpha} = \{50, 50, 50\}$.

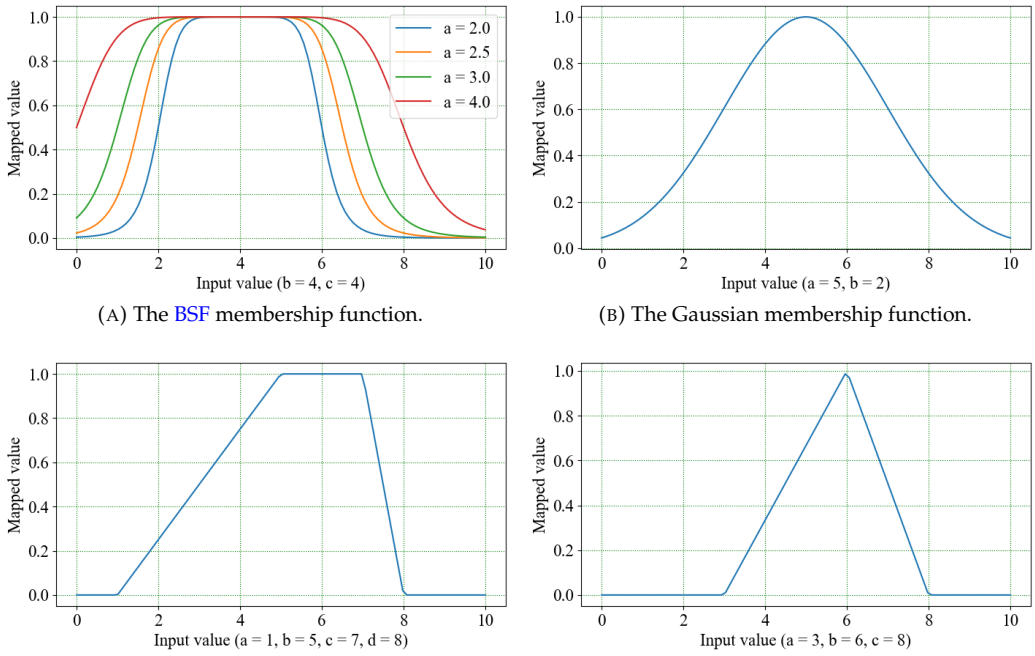


(D) The Dirichlet PDF, $\boldsymbol{\alpha} = \{1.5, 1.5, 1.5\}$.

FIGURE 2.4: An example of beta and Dirichlet probability density function.

2.5.2 Membership functions

The membership function gives the partial truth, i.e., a value between 0 and 1, to a state in the state space. The names of the membership functions are given according to the



(A) The BSF membership function.

(B) The Gaussian membership function.

(C) The trapezoidal membership function.

(D) The triangular membership function.

FIGURE 2.5: An example of four membership functions.

shape of the curve, i.e. bell-shaped, Gaussian, trapezoidal, and triangular membership functions [TSH17].

The BSF is similar to the Gaussian distribution and is typically continuous and symmetric, matching both side values to zero, and the left part to one. As illustrated in Fig. 2.5a. The BSF Eq. 2.31 depends on three parameters a, b and c. Among them, a defines the range of values that will be matched to one; b defines the slope of the curve on both sides of the central plateau, where a larger value results in a more steep transformation; c defines the center of the curve.

$$\sigma^{BSF}(\boldsymbol{x}, a, b, c) = \frac{1}{1 + \left|\frac{\boldsymbol{x}-c}{a}\right|^{2b}}. \tag{2.31}$$

The Gaussian function is one of the most widespread and well-known functions, and its variation Eq. 2.32 is used for membership calculation.

$$\sigma^{Gaussian}(\boldsymbol{x}, a, b) = \exp^{-\frac{1}{2}\left(\frac{\boldsymbol{x}-a}{b}\right)^2}. \tag{2.32}$$

Trapezoidal membership function has four parameters: a, b for feet and c, d for shoulders expressed as Eq. 2.33.

$$\sigma^{trapezoidal}(\boldsymbol{x}, a, b, c, d) = \max\left(\min\left(\frac{\boldsymbol{x}-a}{b-a}, 1, \frac{d-\boldsymbol{x}}{d-c}\right), 0\right). \tag{2.33}$$

The triangular membership function is the simplest shape among others. It is defined by three parameters: a, c for feet, and b for the top of the curve expressed as Eq. 2.34.

$$\sigma^{triangular}(\boldsymbol{x}, a, b, c) = \max\left(\min\left(\frac{\boldsymbol{x}-a}{b-a}, 1, \frac{c-\boldsymbol{x}}{c-b}\right), 0\right). \tag{2.34}$$

2.5.3 The contrastive-center loss function

Give the training dataset $\mathcal{D}^{train}$, the CNN based model $f_{\boldsymbol{\theta}}$, the last hidden layer $\boldsymbol{\delta}_n = \{\delta_1, \ldots, \delta_i, \ldots, \delta_h\}$ and the output layer $\boldsymbol{o}_n = \{o_1, \ldots, o_i, \ldots, o_K\}$. In general, the empirical loss $\mathcal{L}_{CE}(\boldsymbol{x}, y)$ over $\mathcal{D}^{train}$ has the following form:

$$\mathcal{L}_{CE}(\mathcal{D}^{train}) = \sum_{i=1}^{N} \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), y_i), \tag{2.35}$$

where $\ell(\cdot)$ is the cross-entropy loss function.

In order to improve the discriminative ability of the NN. In [QS17], introduced the contrastive-center loss function Eq. (2.36). This new loss function simultaneously considers inner-class compactness and inter-class separability by assessing the contrastive values between: (1) the distances of training samples to their corresponding class centers, and (2) the sum of the distances of training samples to their non-corresponding class centers.

$$\mathcal{L}_{CC}(\mathcal{D}^{train}) = \frac{1}{2} \sum_{i=1}^{N} \frac{||\boldsymbol{x}_i - \boldsymbol{c}_{y_i}||^2}{\left(\sum_{j=1, j \neq y_i}^{K} ||\boldsymbol{x}_i - \boldsymbol{c}_{y_j}||^2\right) + \beta}, \tag{2.36}$$

where the $\boldsymbol{c}_{y_i}$ denotes the y_i class center, and β is a coefficient used for preventing the denominator equals zero.

The final form of the loss function Eq. (2.37) is generated by integrating Eqs. (2.35) and (2.36). The calculation steps are shown in Fig. 2.6. For the convenience of description, we will refer to Eq. (2.37) as contrastive-center loss in the following.

$$\mathcal{L}(\mathcal{D}^{train}) = \mathcal{L}_{CE}(\mathcal{D}^{train}) + \lambda \mathcal{L}_{CC}(\mathcal{D}^{train}), \tag{2.37}$$

where λ is a balancing weight.

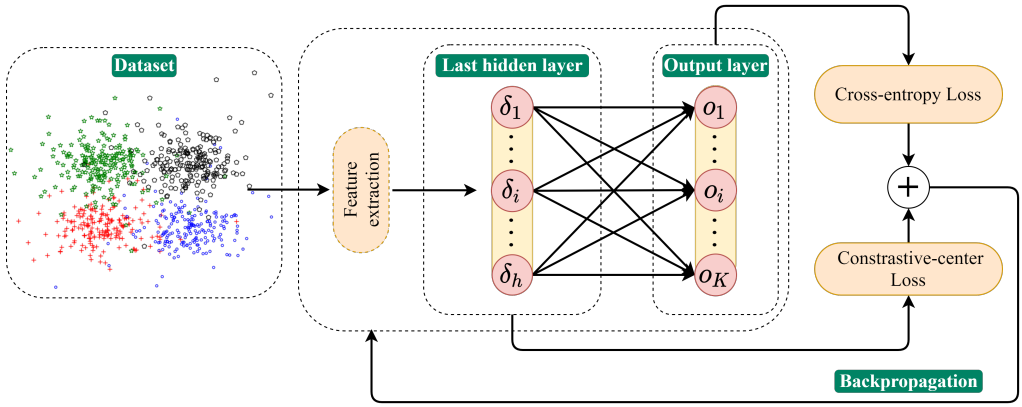


FIGURE 2.6: The framework of the contrastive-center loss function. The cross-entropy loss is calculated based on the model output, while the contrastive-center loss is calculated based on the last hidden layer output.

2.6 Conclusions

This section introduces some background knowledge that is going to be used in the following chapters. At first, the MLP and CNNs, e.g., VGG, GoogLeNet, ResNet, and MobileNet, are introduced. The uncertainty reasoning frameworks, such as SL and DST are introduced. Both SL and DST can achieve uncertainty derivation and belief fusion. The

difference is that the SL takes base rate into account, and the DST can give belief to the empty set. The Dirichlet distribution used in SL is also presented, and it is used to connect the model output with the uncertainty derivation. The BSF that converts the testing output into possibility based on the training output is also introduced.

Part I

Uncertainty estimation

Chapter 3

A survey of uncertainty estimation

There are a sea of uncertainty estimation methods and applications. In this thesis, we only focus on the uncertainty estimation for Convolutional Neural Network (CNN) based image classification. Other methods such as regression and image segmentation are not discussed. As mentioned in Section 1.2.2 the uncertainty estimation can be part into three steps, in this section the third step "the confusing sample classification process" can be part into three more concise parts, i.e., input sample, model, and output. This is inspired by the classical sample process: feed a sample into the Neural Network (NN) and perform further execution on the model outputs. Then we classify the existing uncertainty estimation methods into these three categories.

1. Uncertainty estimation based on the input sample. Augmenting a single sample into M samples. Feeding the obtained M samples into a certain model to get M outputs to calculate the variation as uncertainty.

2. Uncertainty estimation based on the model. For a certain sample, we can ensemble M different models, and use these M models to get M outputs for the same sample to calculate the variation as uncertainty. For example, using dropout for a single sample during the testing phase to get M outputs to calculate the variation. Alternately, for a certain model, we can get M sets of parameters by sampling M times from the model parameter distribution, and use these M weights to get M outputs for the same sample to calculate the variation.

3. Uncertainty estimation based on the output. The uncertainty is directly computed based on the model output.

An overview of three category methods is given in Fig. 3.1. In the following subsections, the main ideas and further extensions of the three categories are presented and discussed. Apart from an introduction of the related works, the commonly used measurements, datasets and baseline are introduced.

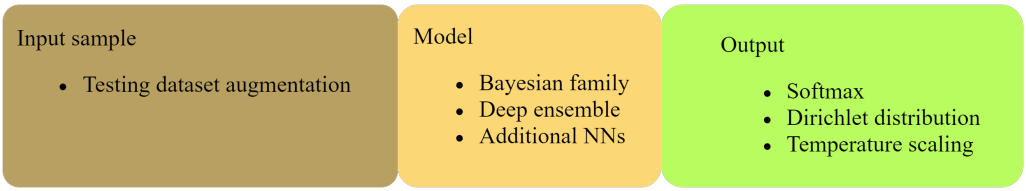


FIGURE 3.1: An overview of existing uncertainty estimation methods.

3.1 Related works

3.1.1 Uncertainty estimation based on the input sample

The testing dataset augmentation as shown in Fig. 3.2 is one of the simple uncertainty estimation techniques. The idea of this method is to create multiple testing samples from each testing sample by applying data augmentation techniques. Then test all those samples to compute a predictive distribution in order to measure uncertainty. Mostly, the testing dataset augmentations have been used in medical image processing [WLOV18, WLA+19, MMKF+20, AB18]. Moshkov et al. [MMKF+20] used the test time augmentation technique for cell segmentation tasks. Next, they used a majority voting to create the final output segmentation mask and discuss the policies of applying different augmentation techniques and how they affect the final predictive results of the deep networks.

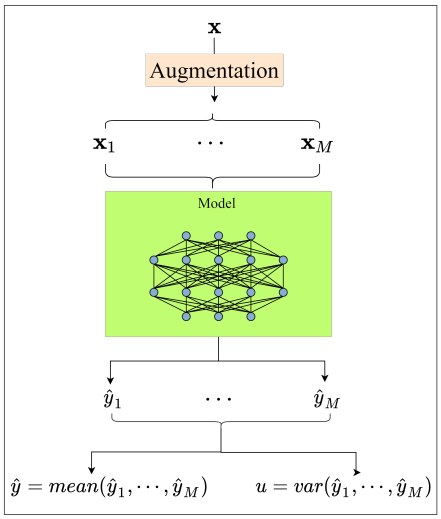


FIGURE 3.2: An example of the testing dataset augmentation method.

3.1.2 Uncertainty estimation based on the model

In a classical NN, weights or parameters are represented by single values and are often learned using back propagation. Whereas, Bayesian Neural Network (BNN) [TLS89] as shown in Fig. 3.3 represents weights in the form of distributions and use "Bayes by backprop" to learn the weight distributions. As introduced in Section 1.2.3, given a training data-label pair (x, y), the posterior distribution over parameters is modeled by assuming a prior distribution over the parameters and applying the Bayesian theorem. Once the posterior distribution over the model parameters has been estimated, the predictive label $\hat{y}$ for a testing sample $\mathbf{x}^*$ can be obtained by:

$$P(\hat{y} \mid \boldsymbol{x}^*, \boldsymbol{x}, y) = \int \underbrace{P(\hat{y}^* \mid \boldsymbol{x}^*, \boldsymbol{\theta})}_{data} \underbrace{P(\boldsymbol{\theta} \mid \boldsymbol{x}, y)}_{model} \mathrm{d}\boldsymbol{\theta}. \tag{3.1}$$

Here is a regression example that uses BNN to fit the underlying training data generator Fig. 3.4. As we can see, with the increase of the training sample, the model uncertainty decrease.

The reasons that prevented the universal use of BNN is that the posterior distribution $p(\boldsymbol{\theta} \mid x, y)$ in Eq. (3.1) is difficult to calculate. Consequently, the Variational Inference (VI), dropout methods are invented to approximate the posterior distribution.

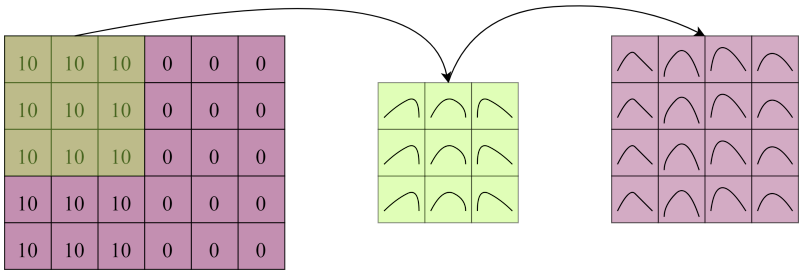
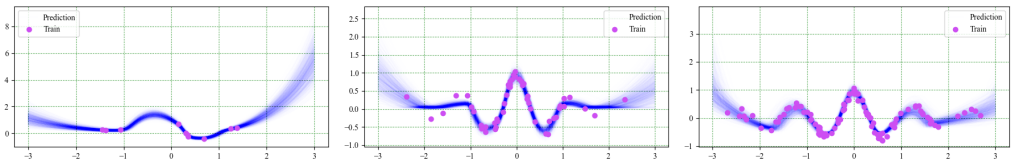


FIGURE 3.3: An example of Bayesian neural network.



(A) 20 training samples. (B) 50 training samples. (C) 100 training samples.

FIGURE 3.4: An example of model uncertainty based on Bayesian neural network.

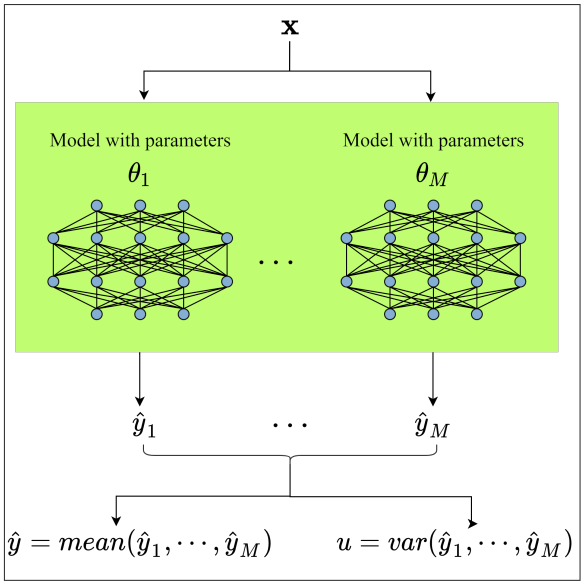


FIGURE 3.5: An example of the Bayesian family method.

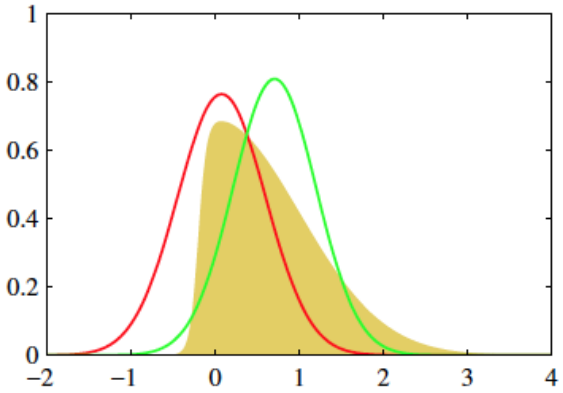


FIGURE 3.6: An example of variational inference method. Find a distribution to approximate the latent distribution.

VI is a branch of Bayesian approximate methods that transforms the posterior distribution approximation problem into the distance optimization problem. Consider a problem: given a dataset $\mathcal{D}$ and a known model form, the aim is to find the posterior distribution $P(\boldsymbol{\theta} \mid \mathcal{D})$ with latent variables $\boldsymbol{\theta}$. First, our original goal is to infer the required distribution P from the existing data. When P is not easy to express and cannot be solved directly, we try to use the variational inference method. Find a distribution P_1 that is easy to express and solve, and when the difference between P and P_1 is small, P_1 can be used as the approximate distribution P. In this process, the aim shifts from the problem of finding the distribution to the optimization problem of reducing the distance. Here is an example Fig. 3.6 used to better understand the above description. The yellow distribution is our original target P, which is not easy to find. Since it looks a bit like a Gaussian distribution, we try to find a distribution P_{red} and a distribution P_{green} from the Gaussian distribution, calculate their overlapping areas and choose the similar one as the approximate distribution of P.

The VI methods for BNNs have been pioneered by Hinton and Van Camp [HVC93] where the authors derived a diagonal Gaussian approximation of the posterior distribution. Another extension has been proposed by Barber et al. [BB98], in which the full covariance matrix was chosen as the variational. Several modern approaches can be viewed as extensions of this early works [HVC93, BB98] with a focus on how to scale the variational inference to modern NNs. Notable et al. [KSW15] introduced the local reparameterization trick to reduce the variance of the stochastic gradients. In order to make more expressive variational distributions feasible for NNs, several works proposed to infer using the matrix normal distribution [ZSDG18, SCC17] or variants where the covariance matrix is decomposed into the Kronecker products of smaller matrices or in a low rank form plus a positive diagonal matrix.

Monte Carlo (MC) [Nea12] is another approximate method that is a slow and computationally expensive method when integrated into a deep architecture. To combat this, MC dropout has been introduced, which uses dropout [SHK+14] as a regularization term to compute the uncertainty [GG16]. MC dropout makes the uncertainty estimation process simple and easy to implement by randomly sampling from the model parameter distribution. For a sample, MC dropout executes M dropouts to obtain M outputs. Then M outputs are ensemble by calculating the variation representing the uncertainty. In [ASKR18], two dropout methods, i.e. element-wise Bernoulli dropout [SHK+14] and spatial Bernoulli dropout [TGJ+15] are implemented to compute the model uncertainty in BNNs for the end-to-end autonomous vehicle control. The architecture of dropout

method is presented in Fig. 3.7.

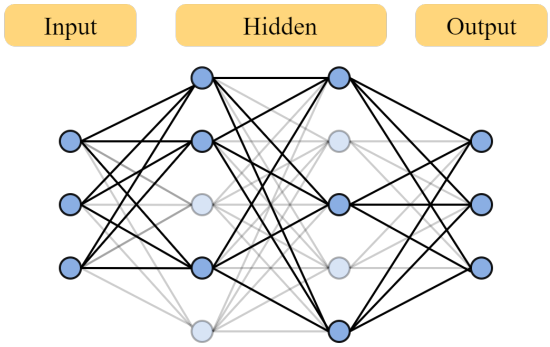


FIGURE 3.7: An example of the dropout method.

The Markov chain Monte Carlo (MCMC) method [Anz12] is a Monte Carlo integration using Markov chains. The basic idea is to construct a Markov chain so that its smooth distribution is the posterior distribution of the parameters. The construction of the Markov chain transfer kernel is crucial, and different methods of transfer kernel construction will produce different MCMC methods. Commonly used transfer kernel construction methods are Gibbs and Metropolis-Hastings sampling methods. Hamiltonian Monte Carlo or Hybrid Monte Carlo (HMC) [DKPR87] is an important variant of MCMC sampling method [Nea92]. The stochastic gradient MCMC (SG-MCMC) [CFG14] was proposed to train CNNs. It only needs to estimate the gradient on small sets of mini-batches. In addition, SG-MCMC can be converged to the true posterior by decreasing the step sizes [DFB+14].

The goal of supervised learning is to learn a stable model that performs well in all aspects. In practice, this is often not the case, but sometimes we can get multiple models that outperform in some aspects. The idea behind ensemble learning [SR18, HS90] is that even if a classifier gets a wrong prediction, the other classifiers can correct the error. Several works applying ensemble methods to different kinds of practical tasks and approaches, for example bioinformatics [CGYY20, NGB20], remote sensing [DWWZ19, MD20], or reinforcement learning [KCD+18] can be found in the literature. In addition, ensembles are widely used for modeling uncertainty on predictions of complex models. The deep ensemble [LPB17] uses the entire training dataset to train M independent randomly initialized models. After the M independent models are trained, the model predictions are fused as shown in Fig. 3.8. As for example in climate prediction [LP07, Par13].

Besides the approaches described above, several other approaches exist. Raghu et al. [RBS+19] recommended a direct uncertainty prediction and suggested training two NNs. One for the prediction task and another one estimates the uncertainty of the first NN. Similarly, Ramalho and Miranda [RM20] introduced an additional NN for uncertainty estimation. But in contrast to [RBS+19], the representation space of the training data is considered and the density around a given test sample is evaluated. The additional NN uses this training data density in order to predict whether the main network's estimate is expected to be correct or false.

3.1.3 Uncertainty estimation based on the output

For classification tasks, the output represents class probabilities. These probabilities are a result of applying the softmax function. The classical and simplest uncertainty estimation method based on softmax is estimated by the following equation.

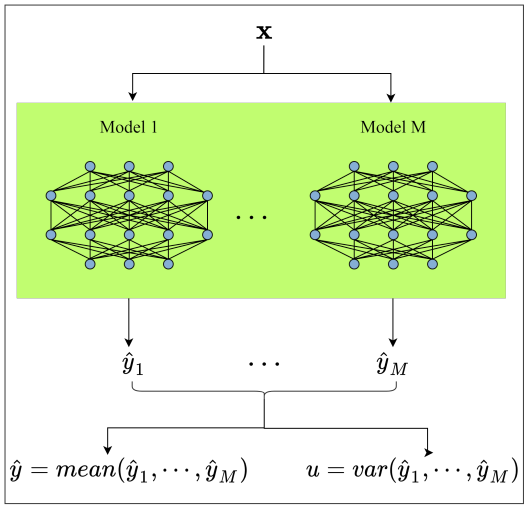


FIGURE 3.8: An example of the deep ensemble method.

$$u = 1 - \max(\sigma_{softmax}(\boldsymbol{o})). \tag{3.2}$$

Many uncertainty estimation approaches followed the idea of predicting the model output distribution. Often, the loss function of such NN takes the expected divergence between the true distribution and the predicted distribution into account. Dirichlet distribution can represent the prior distribution of the class distribution and naturally is used to interpret the uncertainty. The Dirichlet distribution is utilized in several approaches as Dirichlet prior networks [MG18] and Evidential Deep Learning (EDL) [SKK18]. Dirichlet prior NNs [MG18] are trained with the goal of minimizing the expected Kullback-Leibler (KL) divergence between the In-Domain (ID) predictions (a sharp Dirichlet distribution) and the Out-of-Domain (OOD) predictions (a flat Dirichlet distribution) [MG18]. As a follow up, [MG19] discussed the case when the data uncertainty is high, the KL-divergence can lead to an undesirable multi-model target distribution. In order to avoid this, they use the reverse KL-divergence. The experiments showed improved results in the uncertainty estimation as well as the adversarial robustness. The EDL [SKK18] optimize the NN over a Dirichlet distribution parameterized by the based rate, prior constant, and evidence. The loss function is derived by using subjective logic and interpreting the model outputs as evidence then trying to infer the multinomial opinion for uncertainty estimation. For the Dirichlet distribution based methods, the uncertainty type measured depends on the method used. It measures the model uncertainty if the Dirichlet distribution is used to approximate the posterior probability. For other cases, e.g, deriving a new loss function from the Dirichlet distribution, it contains both two uncertainty types.

Furthermore, several approaches can be applied to already trained NNs without affecting the training and predicting processes. Temperature scaling is a very simple post-processing step that helps to calibrate the model. For classification problems, the output of the last layer is further passed to the softmax function to obtain the probability of each class. This step is modified by temperature scaling Eq. 3.3. In [LLS17], a relatively simple approach based on small perturbations on the training dataset and the temperature scaling calibration is presented leading to efficient differentiation of ID and OOD data.

$$\sigma_{softmax}(\boldsymbol{o}) = \frac{\exp\left(\frac{o}{T}\right)}{\sum_{i=1}^{K} \exp\left(\frac{o_i}{T}\right)}, \tag{3.3}$$

where T represents the temperature scaling.

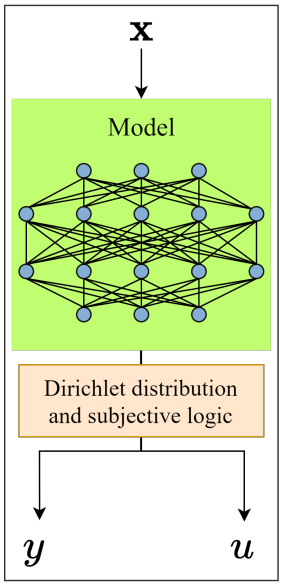


FIGURE 3.9: An example of the Dirichlet distribution based method.

3.2 Measurements

In order to evaluate the uncertainty estimation performance, we summarize several approaches.

3.2.1 Classical measurements

Consider a classification task based on a dataset $\mathcal{D}$ with K different classes and a probability vector $\boldsymbol{p}$ for an input sample $\boldsymbol{x}$. In order to evaluate the amount of uncertainty, one can apply the entropy measures, which describe the average level of information of a random variable.

$$\mathcal{H}(\boldsymbol{p}) = -\sum_{i=1}^{K} \boldsymbol{p}_i \log_2 (\boldsymbol{p}_i) \,. \tag{3.4}$$

The Mutual Information (MI) is another simple method that computes the expected divergence between the stochastic softmax output distribution P_1 and the expected softmax output distribution P_2 to measure the uncertainty.

$$\mathcal{I}(P_1, P_2) = \sum_{P_1, P_2} P_{12} \log \frac{P_{12}}{P_1 P_2}, \tag{3.5}$$

where P_{12} is the joint distribution.

3.2.2 Complete dataset measurements

While the criteria described measuring the performance of an individual sample, others evaluate the usage on a set of samples [HG17]. For that, the samples are split into two sets, for example, ID and OOD or correctly and falsely classified. They we can define precision, recall, True Positive Rate (TPR), and False Positive Rate 95 (FPR95) as shown in Eq. (3.6). The two most common approaches are the Receiver Operating Characteristic (ROC) curve and the Precision Recall (PR) curve. Both methods generate curves based on

		Prediction	
		P	N
Ground-truth label	P	TP	FN
	N	FP	TN

TABLE 3.1: The confusion matrix.

different thresholds of the underlying measure. For each considered threshold, the Receiver Operating Characteristic (ROC) curve plots the TPR against the False Positive Rate (FPR), and the Precision Recall (PR) curve plots the precision against the recall. While the ROC and PR curves give a visual idea of how well the underlying measures are suited to separate the samples, they do not give a qualitative measure. To reach this, the Area Under the Curve (AUC) can be evaluated. For the evaluation of ID and OOD examples, the FPR95, Area Under the Receiver Operating Characteristic (AUROC), and Area Under the Precision Recall Curve (AUPRC) are commonly used [NHL20, MG18, MG19].

$$P = \frac{TP}{TP + FP}, \quad R = \frac{TP}{TP + FN}, \quad TPR = \frac{TP}{TP + FN}, \quad FPR = \frac{FP}{FP + TN}. \tag{3.6}$$

The FPR95 is a measure of accuracy, which is calculated as the fraction between the number of false positives and the total number of negatives. It's the probability that a positive result will be given when the true value is negative. For example, FPR95 (the smaller the better) means the false positive rate of OOD samples when the true positive rate of ID samples is at 95%.

The AUROC (the bigger the better) tells whether the trained model can correctly rank a random positive sample before a random negative sample. For a diagnosis model, an AUROC of 0.7 means that the model has a good discriminatory ability, seventy percent of the time, the model will correctly assign a higher risk to a patient with a symptom than a patient without a symptom. The AUPRC (the bigger the better) denotes a high value for both recall and precision criteria, where high precision associates with a small false-positive rate, and high recall relates to a small false-negative rate. High scores for both represent that the model does correct classification (high precision), as well as obtain a majority of all positive results (high recall). The FPR95, AUROC, and AUPRC values are calculated with the library shared by Liu et al.[1].

3.2.3 Calibration measurements

The model calibration is used to keep the model prediction probabilities consistent with the real probabilities. For classification models, such as Support Vector Machine (SVM), their output does not represent the real probability. Thus, there is a need to use the calibration algorithm to make the output consistent with the real probability. Here is an example, according to the model output, the samples are divided into 10 bins, i.e., samples with a probability of 0 to 0.1 are grouped into one bin, those with a probability of 0.1 to 0.2 are grouped into one bin, etc. The 10 bins are used as horizontal coordinates; the percentage of positive samples in each bin as vertical coordinates, the graphs are drawn from them can be used for evaluation Fig. 3.10. The reason why it works is that according to the definition of probability, the percentage of positive samples should be 0.1 for the bin with 0.1 probability. Therefore, the reliability diagram is actually a measure of whether the output probability of the model is consistent with the true probability.

[1] https://github.com/wetliu/energy_ood

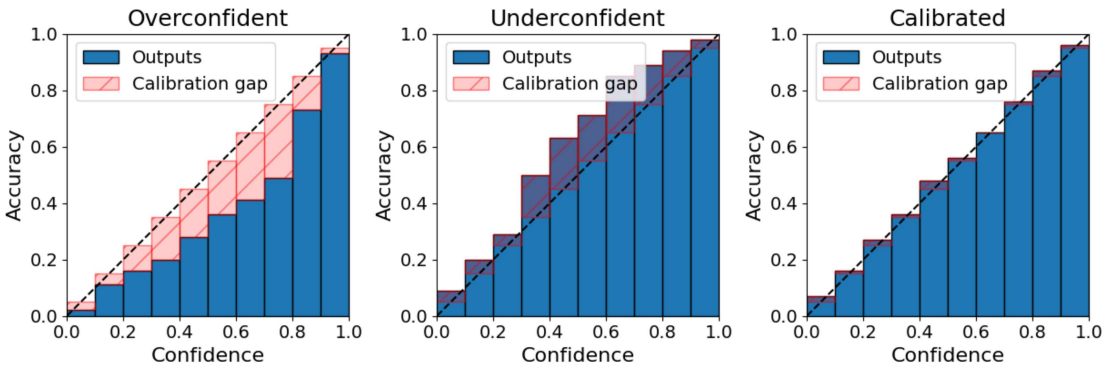


FIGURE 3.10: An example of the reliability diagram. (a) Reliability diagram showing an overconfident classifier: the bin-wise accuracy is smaller than the corresponding confidence. (b) Reliability diagram of an underconfident classifier: the bin-wise accuracy is larger than the corresponding confidence. (c) Reliability diagram of a well-calibrated classifier: the confidence fits the actual accuracy for the single bins. These figures come from [GTA+21].

Approaches that reduce the model uncertainty, also lead to a better calibrated classifier. This is because the remaining predicted data uncertainty better represents the actual uncertainty on the prediction. For classification tasks, several calibration measures are based on binning. The Expected Calibration Error (ECE) [NCH15, GPSW17, NCH15, MWT+20] is defined by Eq. (3.7) was adopted. This groups the probability interval into M bins with m_i samples inside and assigns each predicted probability to the bin that encompasses it. The calibration error is the difference between the fraction of predictions in the bin that are correct (accuracy) and the mean of the probabilities in the bin (confidence).

$$\mathrm{ECE} = \sum_{i=1}^{M} \frac{m_i}{M} \mid \mathrm{acc}_i - \mathrm{conf}_i \mid, \tag{3.7}$$

where acc_i and conf_i are the accuracy and confidence of bin i, respectively.

3.3 Datasets and baselines

In this section, we collect commonly used datasets and baselines for uncertainty estimation. In the scope of this thesis, we only focus on the OOD detection of the image classification based model. The choice of datasets is mostly consistent among all reviewed works. Most of the literature start with a toy dataset, i.e., the Blob, Moon, Circle datasets, to explain the proposed method. The most common datasets for Out-of-Domain (OOD) detection are Mnist [LCB10], Cifar10 [Kri12], Cifar100 [Kri12], Svhn [NWC+11], Lsun [YZS+15], Texture [CMK+14], Places365 [ZKL+17], ImageNet and its tiny variant are also studied frequently. When OOD detection is studied where models trained on Cifar10 [Kri12], Cifar100 [Kri12] are evaluated on Svhn [NWC+11], Lsun [YZS+15], Texture [CMK+14], Places365 [ZKL+17]. Meanwhile, MNIST [LCB10] is paired with variants of itself like notMNIST and FashionMNIST. Finally, the most commonly used baselines by far are Softmax [HG17], MC Dropout [GG16], deep ensembles [LPB17] and temperature scaling [LWOL20].

3.4 Conclusions

This chapter introduces some uncertainty estimation methods, measurements, datasets, and baselines. The commonly used methods are, for example, Bayesian based methods, deep ensemble methods, and Dirichlet distribution based methods. For Dirichlet distribution based methods, there is always a need to reconstruct a loss function and train the model, but the uncertainty can be derived directly from the model output. For Bayesian based methods, one obvious weakness is the computation burden, but there are several approximations invented to alleviate this challenge. The deep ensemble method calculates the uncertainty based on the prediction of several models. Consequently, the memory and computational consumption used for training several models can be the bottleneck. Also, we introduce some measurements. The entropy and its variations focus on the individual prediction. The FPR95, AUROC, and AUPRC focus on the evaluation of a set of predictions. And ECE quantify the predictive uncertainty through the model calibration. In addition, we also present the commonly used datasets and baselines.

Chapter 4

Novelty uncertainty estimation approach

Uncertainty estimation represents methods that can calculate an uncertainty value for an input sample. By comparing the obtained value with a predefined threshold, the model can decide whether to reject the input sample or not. The rejected sample is classified into the empty set or the entire set for further manual process. Uncertainty estimation is one of the most common and straightforward methods to deal with confusing samples. This chapter starts with a description of the existing Evidential Deep Learning (EDL) method, including the loss function used to train the model and the derivation of uncertainty from the model output. The weakness of the EDL method is that the base rate is not explicitly used. Consequently, we propose the SLUE method to train the model more effectively by updating the base rate during the training process.

4.1 A brief introduction of the evidential deep learning

Before going deeper, let us recall the Subjective Logic (SL) concepts which have been presented in the Section 2.3. Consider a state space $\Omega = \{\omega_1, \cdots, \omega_K\}$ including $K > 2$ mutually exclusive states. A belief notation of opinions over the Ω is an ordered triplet $\mathcal{S} = \{\boldsymbol{b}, u, \boldsymbol{a}\}$. The $\boldsymbol{b} = \{b_1, \cdots, b_K\}$ represents belief for each state in Ω and u denotes the overall uncertainty, respect the following constrain:

$$u + \sum_{i=1}^{K} b_i = 1, \quad u \geq 0, \quad b_i \geq 0. \tag{4.1}$$

Then the posterior probability expectation function $\boldsymbol{p} : \Omega \longrightarrow [0, 1]$ can be expressed as:

$$\boldsymbol{p} = \boldsymbol{b} + u\boldsymbol{a} = \frac{\boldsymbol{\alpha}}{\sum_{i=1}^{K} \boldsymbol{\alpha}_i}. \tag{4.2}$$

The architecture of the EDL is similar to classical Convolutional Neural Network (CNN). The only difference is that the softmax layer is replaced with an Rectified Linear Unit (ReLU) layer Eq. (4.3).

$$\delta^{ReLU}(\boldsymbol{x}) = \begin{cases} \boldsymbol{x}, & \boldsymbol{x} \geq 0, \\ 0, & otherwise. \end{cases} \tag{4.3}$$

The ReLU ensures non-negative output, which is taken as the evidence vector ($\boldsymbol{r} = \delta^{ReLU}(\boldsymbol{x})$) for the Dirichlet distribution. Taking $\boldsymbol{\alpha} = \boldsymbol{r} + \mathcal{C}\boldsymbol{a} = \boldsymbol{r} + K\frac{\mathbf{1}}{K} = \boldsymbol{r} + 1$ as the Dirichlet distribution parameters. With the mapping between belief notation of opinion and Dirichlet Probability Density Function (PDF). We can calculate the belief $\boldsymbol{b}$ and uncertainty u as show in Eq. (4.4).

$$\boldsymbol{b} = \frac{\boldsymbol{r}}{\mathcal{C} + \sum_{i=1}^{K} \boldsymbol{r}_i}, \quad u = \frac{\mathcal{C}}{\mathcal{C} + \sum_{i=1}^{K} \boldsymbol{r}_i}. \tag{4.4}$$

In EDL, the author proposed an expected mean square loss function Eq. 4.5 to train the model. In order to guarantee the total evidence should shrink to 0 when the input sample cannot be correctly classified, a Kullback-Leibler (KL) divergence term is incorporated into the loss function that regularizes the predictive distribution by penalizing those divergences from the "I do not know" state.

$$\begin{aligned} \mathcal{L}(\boldsymbol{x}_i) &= \sum_{j=1}^{K} \left(y_{ij} - p_{ij}\right)^2 + \frac{p_{ij}\left(1 - p_{ij}\right)}{\left(S_i + 1\right)}, \\ \mathcal{L}(\mathcal{D}^{train}) &= \sum_{i=1}^{N} \mathcal{L}_i + \lambda_t \sum_{i=1}^{N} KL\left[D\left(\boldsymbol{p_i} \mid \hat{\boldsymbol{\alpha}}_i\right) \| D\left(\boldsymbol{p}_i \mid (\frac{\mathcal{C}}{K}, \ldots, \frac{\mathcal{C}}{K})\right)\right], \end{aligned} \tag{4.5}$$

where K is the number of classes, N is the number of samples, $\boldsymbol{y}_i = \{y_{ij} \mid j = 1, \ldots, K\}$ is a one-hot label of sample $\boldsymbol{x}_i$, $\boldsymbol{p}_i = \{p_{ij} \mid j = 1, \ldots, K\}$ is a vector representing class assignment probabilities, $S_i = \mathcal{C} + \sum_{i=1}^{K} \boldsymbol{r}_i$, the annealing coefficient $\lambda_t = \min(1.0, \frac{t}{10}) \in [0, 1]$, t representing the current training epoch index, $D\left(\boldsymbol{p}_i \mid (\frac{\mathcal{C}}{K}, \ldots, \frac{\mathcal{C}}{K})\right)$ is the uniform Dirichlet distribution, and $\hat{\boldsymbol{\alpha}} = \boldsymbol{e}(1 - y) + 1$ represents the Dirichlet parameters after removing the non-misleading evidence from predicted parameters.

The EDL can provide a more detailed uncertainty estimation model than the standard softmax point estimation. The performance of the EDL method is evaluated through Mnist [LCB10] and Cifar10 [Kri12] datasets based on the entropy criteria. The contributions of the EDL method contain the detection of Out-of-Domain (OOD) samples and improved endurance against adversarial perturbations.

4.2 The proposed approach considering the base rate explicitly

4.2.1 Introduction

Currently, Neural Network (NN)s have a pivotal role in various applications of human endeavor. It possesses an admirable prediction accuracy but less prediction confidence. As an example, if we feed an image of car into a cat-dog NN, this image will be classified as either being a cat or a dog rather than as being inappropriate. Obviously, to a human being, this is lacking intelligence. To address this problem, we propose a new method called Subjective Logic based Uncertainty Estimation (SLUE) [XCA21]. It is derived from EDL [SKK18] based on SL by processing model outputs while giving uncertainty to each prediction. The impetus behind this is that the SL boosts the traditional Dempster-Shafer Theory (DST) in the sense that opinions take base rates into account, whereas DST ignores base rates. With base rates, we can make good use of prior knowledge. At the same time, it also makes it possible to define a bijective mapping between subjective opinions and Dirichlet PDF [Jøs18]. With a bijective mapping, the uncertainty and probability expectation formula can be easily derived.

In EDL, there is no explicit use of base rates because it just used the default base rates nor analyze the potential initialization methods. In addition, the base rates is not updated during the training process. The Dirichlet distribution parameters appeared in EDL ($\boldsymbol{\alpha} = \boldsymbol{r} + 1$) are composed of evidence and a weight of one that is allocated to all the training classes. In comparison, the SLUE used the base rates to refine the Dirichlet distribution parameters ($\boldsymbol{\alpha} = \boldsymbol{r} + \mathcal{C}\boldsymbol{a}$) to guide the training process. Within the SLUE method, the base rates is updated after each batch leading to a more flexible and precise

classification. In addition to the underutilization of base rates within the EDL method, the sum of all the weights equals the number of training classes. Intuitively, the sum of all the weights could be a hyperparameter. Consequently, in SLUE, the sum of all the weights is represented by the prior constant $\mathcal{C}$; the optimum can be explored through experiments.

Compared with the existing methods, this work makes the following contributions:

1. Taking the base rates explicitly into account, the initial choice is evaluated. A comprehensive analysis with experiments is carried out.

2. Exploring and finding the optimum configuration of the hyperparameter $\mathcal{C}$.

4.2.2 Proposed method

Suppose the state space is composed by K training class, i.e., $\{\omega_1, \cdots, \omega_K\}$. We define the evidence $\boldsymbol{r}_i$, probability $\boldsymbol{p}_i$, and base rates $\boldsymbol{a}_i$ for the current batch i, consequently, the probability $\boldsymbol{p}_{i-1}$ for past batch $i-1$. We feed the model outputs into the ReLU and take the outputs as $\boldsymbol{r}_i$, meanwhile, $\boldsymbol{p}_i$ can be calculated with Eq. (4.2).

The motivation behind updating the base rate is inspired by [Jøs18]. In which, it claims that the base rates can be dynamically updated as a function of observed evidence. In the beginning, we do not know the class proportion. However, once made a batch prediction, we can take the previous probability $\boldsymbol{p}_{i-1}$ to update the base rates $\boldsymbol{a}_i$, i.e., $\boldsymbol{p}_{i-1} \longrightarrow \boldsymbol{a}_i$ as shown in Fig. 4.1. Then, $\boldsymbol{\alpha} = \boldsymbol{r} + \mathcal{C}\boldsymbol{a}$ are used as the parameters of a Dirichlet distribution. The uncertainty can be calculated with Eq. (4.4) to determine whether to accept or reject the current prediction. The SLUE method uses Eq. (4.4) to quantify uncertainty directly and probability calculated from Eq. (4.2) to make prediction decisions. The format loss function Eq. 4.5 is adopted.

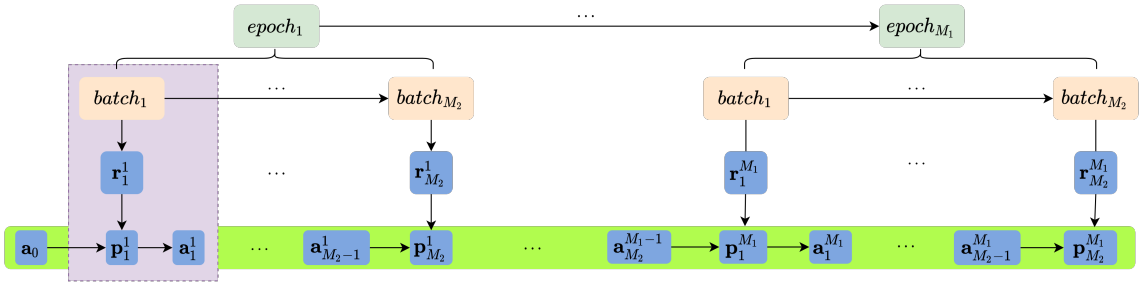


FIGURE 4.1: The base rate update process. Suppose there are M_1 epoches, each epoch contains M_2 batches, and the batch size is bs. The default base rate $\boldsymbol{a}_1 \in \mathbb{R}^{bs \times K}$ is a matrix filled by $\frac{1}{K}$. At each batch i, we use the previous probability $\boldsymbol{p}_{i-1}$ update the base rate.

4.2.3 Experiment

4.2.3.1 Experiment protocol

We use the standard CNNs with ReLU as the neural network architecture, all experiments are implemented in Pytorch. For the Mnist dataset [LCB10], a standard LeNet [LBBH98] was trained. Following the suggestion of [LW17], an augmented LeNet version that contained 192 filters at each convolutional layer and had 1000 hidden units for the fully connected layers was trained for Cifar10 [Kri12] and Cifar100 [Kri12] datasets. The characteristics of the datasets are shown in Table. 4.1. For training, we use the hold-out strategy, that is to say, 70% of samples are used for training, and 30% of samples used for testing. The Adam optimizer has been used with default settings for training.

Name	# Used classes	# Training samples	# Testing samples
Mnist	10	55000	10000
Cifar10	10	50000	10000
Cifar100	10	5000	1000
Lsun	10	-	2000
Texture	10	-	2000
Places365	10	-	2000
Mnist5	5	25000	5000
Cifar5	5	28000	4800

TABLE 4.1: An overview of datasets involved in SLUE.

Apart from the measurements introduced in Section 3.2, we also adopt the following measurements.

- *Extended accuracy* is the number of correctly classified In-Domain (ID) samples plus the rejected wrong classified samples divided by the total number of samples. This is different from test accuracy, which only takes correctly classified samples into account; with uncertainty, the rejected wrong classified samples should also be regarded as correctly classified samples and be taken into account. A higher value is better.
- *Entropy* is used to evaluate the prediction uncertainty as described in [LW17]. The increase in prediction uncertainty leads to an increase in entropy. Consequently, % max entropy which means the ratio of prediction entropy to the maximum prediction entropy is used for uncertainty estimation, for OOD samples, a higher % max entropy value is better. And the Cumulative Distribution Function (CDF) is also adopted, which describes the entropy distribution for each sample. For OOD samples, a curve closer to the right bottom corner is desired.

4.2.3.2 Initial stage choice

This section describes and compares several strategies to choose the prior constant $\mathcal{C}$ and initial base rates. The first five classes of the Mnist and Cifar10 datasets were extracted to generate Mnist5 and Cifar5 datasets. The model with the SLUE method was trained based on these two datasets; then it was tested with Mnist and Cifar10 testing datasets that contained all ten classes. During this process, the first five class samples played the role of ID samples, while the last five class samples acted as OOD samples.

To select the optimum prior constant $\mathcal{C}$, we examined various settings from zero to 100 at an interval of five (the number of training classes). As we can see from Fig. 4.2, the % max entropy kept increasing; meanwhile, the extended accuracy reached an optimum. To balance these two criteria and take the integer multiples value of the number of training classes, a value four times the number of training classes was used in the experiments. The optimal extended accuracy and intersection are indicated by two dashed black lines, and the chosen prior constant $\mathcal{C}$ is indicated by dash-dot blue lines.

Events that can be repeated many times are typically frequentist in nature, meaning that base rates for such events typically can be derived from statistical observations [Jøs18]. Thus, for the initial base rates, there are four candidates:

1. Use the uniform base rates (hereafter called uniform prior), i.e., $(\frac{1}{K}, \cdots, \frac{1}{K})$.

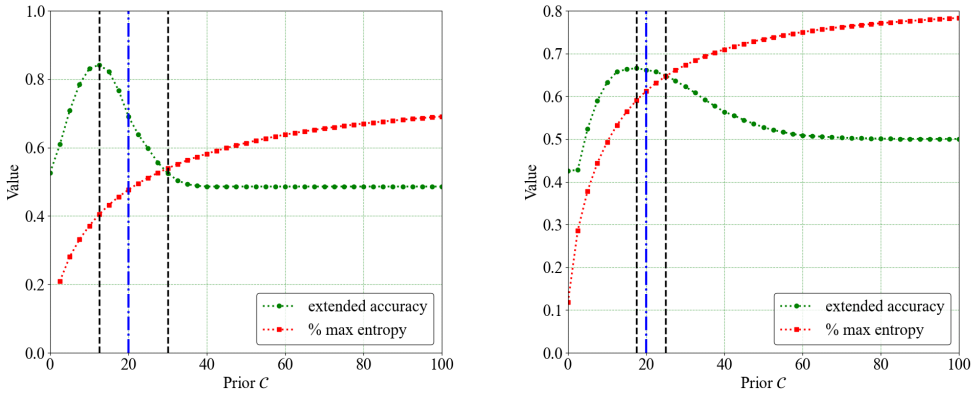


FIGURE 4.2: Estimation results based on different prior constant $\mathcal{C}$ values for Mnist5 (left) and Cifar5 (right) datasets.

Strategy	Extended accuracy	% Max Entropy
	Mnist5 / Cifar5	Mnist5 / Cifar5
Uniform Prior	**0.733 / 0.645**	**0.448 / 0.608**
Frequency Prior	0.733 / 0.645	0.448 / 0.608
Highest Frequency Prior	0.733 / 0.645	0.446 / 0.607
Lowest Frequency Prior	0.737 / 0.645	0.447 / 0.607

TABLE 4.2: The estimation results based on different base rates initial strategies.

2. Use each training class frequency as the initial base rates (hereafter called frequency prior), i.e., $(\frac{\sum_{(x_i,y_i)\in\mathcal{D}^{train}} \mathbb{1}(y_i=\omega_1)}{N}, \cdots, \frac{\sum_{(x_i,y_i)\in\mathcal{D}^{train}} \mathbb{1}(y_i=\omega_K)}{N})$.
3. Assign the whole base rates to the training class that has the highest frequency (hereafter called highest frequency prior), i.e., $(1, 0, \cdots, 0)$, where the first class has the highest frequency.
4. Assign the whole base rates to the training class that has the lowest frequency (hereafter called lowest frequency prior), i.e., $(1, 0, \cdots, 0)$, where the first class has the lowest frequency.

Prior constant $\mathcal{C}$ is set to equal four times the number of training classes, then each base rate initial strategy is verified in turn. The training and testing period operations are the same as those for choosing optimum prior constant $\mathcal{C}$. As can be seen from Table 4.2, there was not an obvious difference between the four strategies; because the initial base rate is just initialization, the final classification is determined by all the base rates in every iteration step, and the initial base rate does not dominate. As a result, the easiest achieved strategy (uniform prior) was chosen for determining the initial base rates.

4.2.3.3 Evolution of base rate

Since the base rate is updated after each batch, we show the evolution of the base rate. Based on the Cifar10 dataset, four different NNs are trained. Each training process contains 10 epochs and each epoch has many batches. The initial base rate is set to the uniform prior, and the average of the base rate of each class of the last batch is calculated as shown in Fig. 4.3. As shown in the Fig. 4.4, from the results obtained, some of the base

Method	Mnist	Cifar5
CNN	0.994	0.764
EDL	0.993	0.843
SLUE	**0.997**	**0.843**

TABLE 4.3: The testing accuracies for Mnist and Cifar5 datasets in SLUE.

rates fluctuate around a central mean, while others maintain an upward or downward trend. It can be seen that the base rate plays a role in guiding the training toward the desired direction.

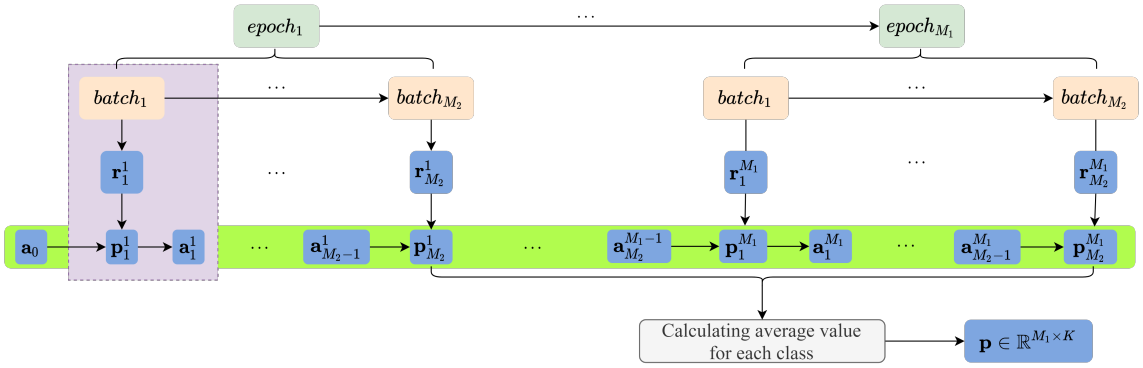
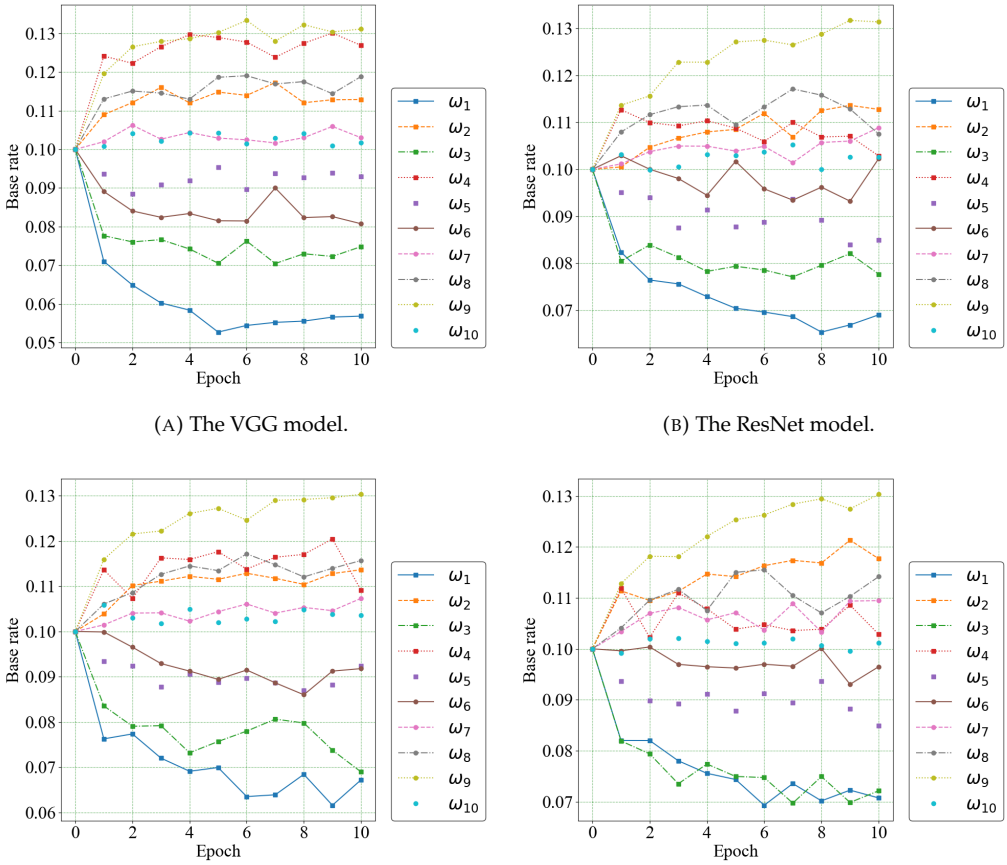


FIGURE 4.3: The process of obtaining base rate evolution. Suppose there are M_1 epoches, each epoch contains M_2 batches, the batch size is bs, and the base rate at end of i^{th} epoch $\boldsymbol{p}_{M_2}^{i} \in \mathbb{R}^{bs \times K}$. Then, calculating the average probability in the batch dimension for each class getting $\boldsymbol{p}_{M_2}^{i'} \in \mathbb{R}^{1 \times K}$. Finally, stacking all the average probability vertically, we can get the base rate evolution matrix $\boldsymbol{p} \in \mathbb{R}^{M_1 \times K}$.

4.2.3.4 Synthetic dataset

First and foremost, SLUE accuracy performance was evaluated. Following the suggestion of [LW17], the Cifar10 dataset was reduced by selecting the first five classes; it is referred to as Cifar5. Since with CNNs method the model cannot calculate uncertainty for OOD samples, to be fair, classical test accuracy was adopted instead of extended accuracy. Table 4.3 demonstrates the main characteristics for comparing the SLUE method and the existing methods. The SLUE method achieved a match and better performance in test accuracy. Hence, the uncertainty estimation extensions of SLUE do not decline the model performance on ID sample classification.

There is one more point that should be touched upon, the model uncertainty performance. The models were trained with the Cifar10 and Cifar100 datasets separately. The trained models were tested with the OOD datasets (e.g., Texture, Places365, Lsun, Cifar10, and Cifar100 datasets). To be homogeneous between testing datasets and training datasets, for Texture, Places365, and Cifar100 datasets, the first ten classes were extracted and then used for the experiments. The results of the prediction entropy CDF on the OOD datasets are shown in Fig. 4.5. Since the predictions for these samples are all wrong since it is OOD, predictions with maximum entropy are expected. A high start entropy value will be observed, and after this beginning point, there will be a dramatic increase in probability. Regarding the figure, the curves closer to the bottom right corner of the plot are wanted, which demonstrates maximum entropy in all predictions [LW17]. What is



(A) The VGG model.

(B) The ResNet model.

(C) The WideResNet model.

(D) The MobileNet model.

FIGURE 4.4: The evolution of the base rate based on four models trained by the Cifar10 dataset.

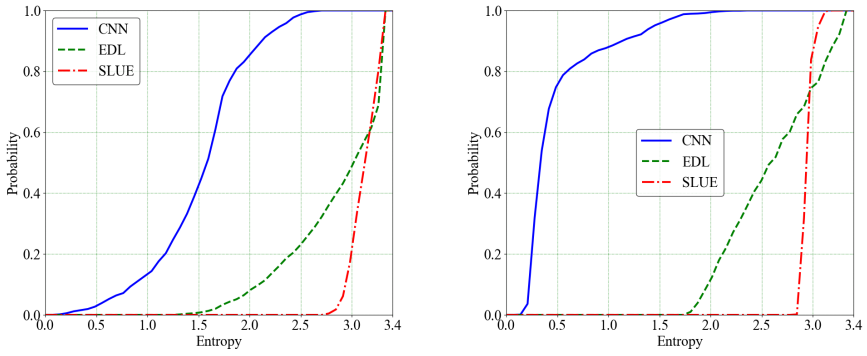
D^{train}	Method	% max entropy ↑
Cifar10	EDL	0.78
	SLUE	**0.93**
Cifar100	EDL	0.76
	SLUE	**0.92**

TABLE 4.4: The comparison between different estimation methods. ↑ indicates larger values are better, and bold numbers are superior results. All values are percentages and are averaged over the four OOD datasets described in section 4.2.3.4.

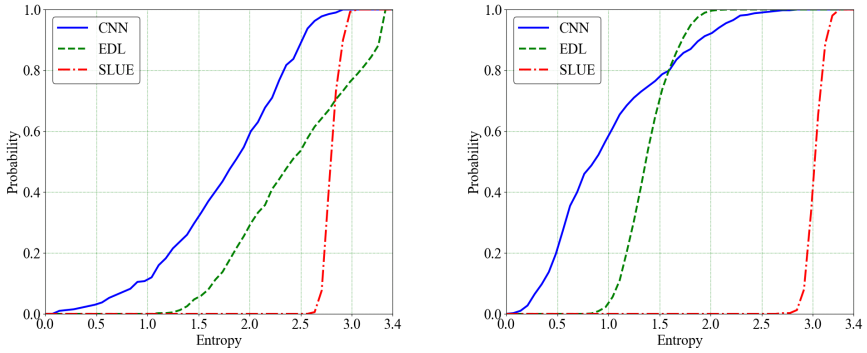
striking about the curves in these figures is that the SLUE method associates much more uncertainty with its predictions than other methods. Compared to the decentralized entropy distribution on the EDL method, the SLUE method is more concentrated. This attribution makes it easier to distinguish OOD samples. As Table. 4.4 demonstrated that SLUE reached better uncertainty assessment performance 15% improvement in terms of % max entropy. It is apparent that the uncertainty estimates of the SLUE method are better than the EDL method.

4.2.3.5 Adversarial dataset

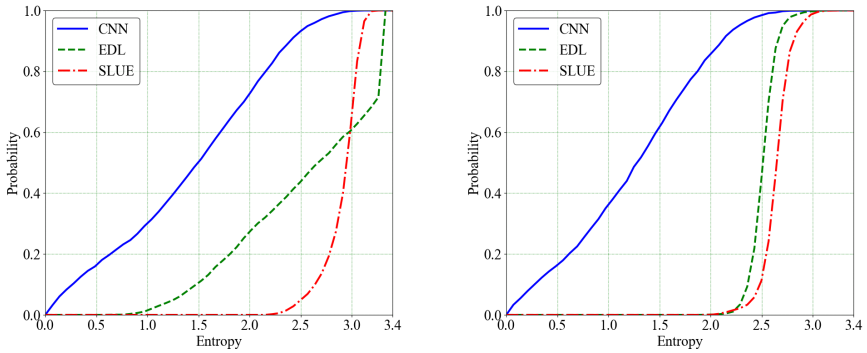
Last but not least, the different methods were also evaluated against adversarial samples [SKK18, LW17, MG19]. Using the fast gradient sign method, adversarial Mnist and



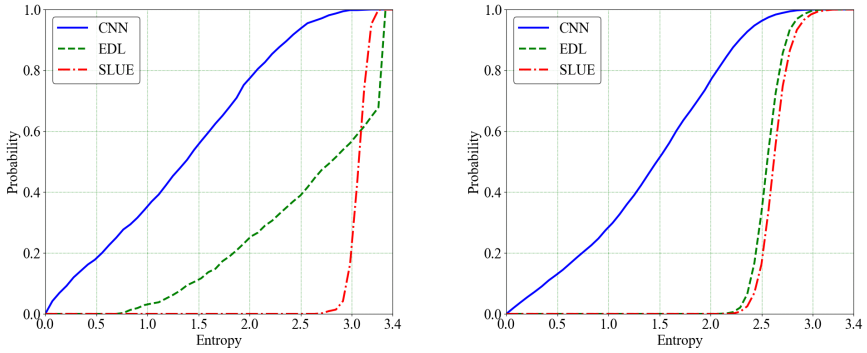
(A) Texture dataset.



(B) Places365 dataset.

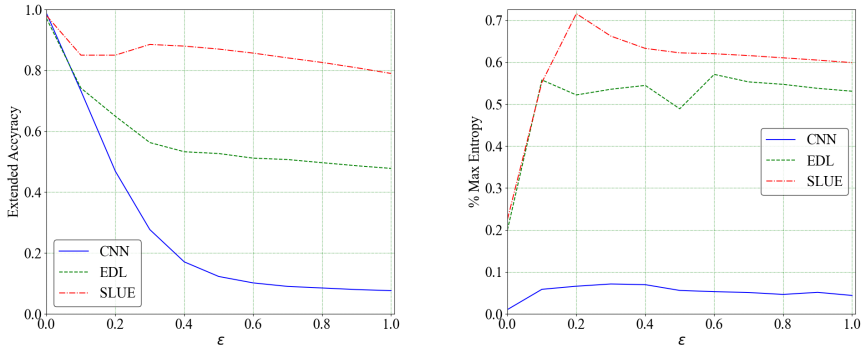


(C) Lsun dataset.

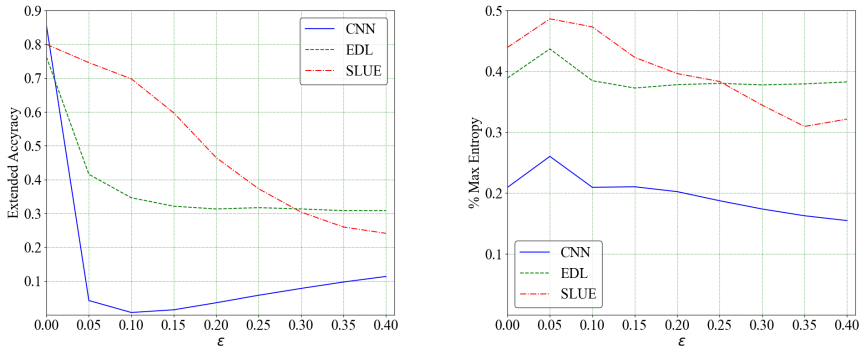


(D) Cifar100 dataset (left) and Cifar10 dataset (right).

FIGURE 4.5: The performance of SLUE against out-of-domain datasets. Empirical CDF for the entropy of the predictive distributions on the OOD datasets based on a model trained by Cifar10 (left column) and Cifar100 (right column) datasets.



(A) Mnist dataset.



(B) Cifar10 dataset.

FIGURE 4.6: extended accuracy and % max entropy as a function of the adversarial perturbation ϵ.

Cifar10 datasets are generated. The feature is that the bigger perturbation ϵ is, the generated datasets were closer to the OOD datasets. Because it becomes harder to make correct predictions, bigger % max entropy would be observed. Fig. 4.6 presents an overview of the performance of extended accuracy and % max entropy for the SLUE method against adversarial datasets. These figures are quite revealing in several ways. First, the figure indicates that the SLUE method has the highest extended accuracy for the adversarial datasets as shown in the left column of the figure. Second, with the comparable % max entropy on all of its predictions as indicated by the right column of the figure, the SLUE method can be used for the identification of OOD samples. The SLUE method represents a good balance between prediction uncertainty and extended accuracy criteria. It associates high uncertainty with the wrong predictions, which can be used to reject OOD samples, improving model robustness.

4.3 Conclusions

The aim of the present research was to extend the existing EDL method by taking base rates into account. It verified the SLUE method uncertainty performance through OOD and adversarial datasets. It introduced how to select the initial parameters and use extended accuracy as one of the measurement. Meanwhile, we verified the performance of the SLUE method against OOD and adversarial datasets. From the obtained results, we can see that guided by base rates, the SLUE method works better not only in terms of extended accuracy but also on the uncertainty estimation performance. As it can reject OOD samples, this approach will prove useful in improving model robustness.

Part II

Partial classification

Chapter 5

A survey of partial classification

Partial classification is fulfilled by classifying confusing samples into subsets, thus making more cautious decisions and avoiding misclassification as well as potential risks. This chapter summarizes common partial classification methods, which we classify into two main categories that can be used directly for pre-trained models and those that require fine-tuned processes. In addition, we introduce the commonly used datasets and measurements.

5.1 Related works

Partial, indeterminate or set-valued classification, which is able to predict more than one class in case of high uncertainty. At the first glance partial classification seems to be linked to multi-label classification [DWCH12, Vov12]. The confusion comes from the fact that both methods produce a class subset as the prediction. However, the crucial dissimilarity comes in that an input sample is labeled by a subset of classes for multi-label classification, whereas for partial classification, only a single class. The goal of the partial classification is to build a classifier $f_\theta(\boldsymbol{x})$; which has two desired properties: the predicted set cardinality $\mid f_\theta(\boldsymbol{x}) \mid$ is not too large and it is trend to make the correctly classification $y \in f_\theta(\boldsymbol{x})$. The uncertainty estimation that calculates a value used to reject the confusing sample or classify the confusing sample into a Out-of-Domain (OOD) class is also a kind of partial classification. In this section, we will only present the unique parts different from the uncertainty estimation. The prevalent partial classification methods are split into two categories, i.e., used for pre-trained and used for the fine-tuned model.

5.1.1 Methods based on the pre-trained model

For dealing with confusing samples through partial classification, a straightforward way is to always predict classes subsets with a fixed cardinality, e.g., a class subset that contains the top five most proper classes [RDS+15]. The top-k strategy can also be used to construct the loss function to train the classifiers. Lapin et al [LHS16] proposed a new surrogate hinge loss as well as a modified cross-entropy loss. Berrada et al [BZK18] modified the loss proposed by [LHS15] in order to provide a smooth loss function better adapted for Neural Network (NN)s and showed that the resulting method achieves state-of-the-art performance on classical image datasets. [LHS16] proposed an novel top-k loss function as modification of the softmax and the multiclass Support Vector Machine (SVM) loss and provide efficient optimization schemes for them. However, there is no reason to predict exactly five or any other a prior fixed number of classes all the time. Consequently, [LJS21] proposed an alternative sensible strategy is to use an adaptive approach in which the number of classes returned varies, but must average to a particular value over all the samples.

Another plain approach is performing thresholding on class probability [MWD+21]. One can return classes whose probability exceeds a fixed threshold or can define a threshold on the cumulative probability of the class subset. In the cumulative probability case, one first sorts the classes in decreasing order of class probabilities. For a threshold, one then returns the top-k classes. This strategy can also be used to build loss function, for example, the conformal predictor.

[MD21] proposed a new partial classifier combines the Dempster-Shafer Theory (DST) and the Convolutional Neural Network (CNN). It introduced a method to define the utilities of predicted sets using Ordered Weighted Average (OWA) operators [Yag92] based on the utility of precise classification and a control parameter. With the extended utility matrix, a generalized maximum expected utility principle is used to make the partial classification. The predicted set amount increases exponentially, as the increase of the class number. In order to alleviate the computation burden, [TXD21] proposed an approach for selecting the interesting predicted set. In [LHZD20], a decision-level combination method is proposed for the multisource domain adaptation based on Dempster-Shafer Theory (DST). The sample is assigned to a singleton class if its neighborhoods can be correctly classified. Otherwise, it is committed to the subset of several possible classes.

5.1.2 Methods based on the fine-tuned model

Inspired by the reject option, an approach integrates costs of indeterminacy in the decision-making [BW08, DCDB09]. One drawback of this approach is that it does not differentiate between rejection due to ambiguity and rejection due to lack of information. [YDM14, YDM17, DY14] introduced a nested dichotomies strategy, which offers computational advantages in both the training and testing processes. Nested dichotomies are binary decomposition methods that transform a multiclass problem into a set of two-class problems easier to solve than the original one. They are so-called meta-classifiers, as each two-class problem can be solved by any classifier. In [DKS19], the authors proposed a semi-supervised procedure based on empirical risk minimization. They derived rates of convergence under smoothness conditions on the conditional probabilities. Authors in [DKS19] also adopt the idea to use unlabeled data to build the partial classifier, where the authors proposed a two-step empirical risk minimization procedure and derived rates of convergence. Later, [RV09] developed a minimax analysis of this framework and derived optimal rates of convergence for a semi-supervised approach based on plug-in under smoothness conditions.

Given an input sample $(\mathbf{x}, y)$, and a confidence level or threshold δ the goal in conformal prediction theory is to provide a set $f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \{y_i, \cdots, y_j\}$, which satisfies

$$p\left(y \notin f_{\boldsymbol{\theta}}(\boldsymbol{x})\right) \leq \delta. \tag{5.1}$$

The conformal prediction theory could be viewed as a way to build partial classifiers. The disadvantages include high computational complexity and randomized nature of the constructed classifiers. For a broad review of conformal prediction theory with main theoretical and practical advances could refer to [VGS05, VNF+17].

In practice, reliability is also needed when representing uncertainty. This is one of the core ideas of the imprecise probability framework [Wal91], where uncertainty is modeled by a set of possible probability distributions instead of a single one. This set, also called credal set [Kyb84], represents the uncertainty about the true distribution that cannot be perfectly identified. Consider a standard probabilistic classifier could yield precise estimation such as $\{p(\omega_1) = 0.1, p(\omega_2) = 0.3, p(\omega_3) = 0.6\}$. In the imprecise probability framework, a classifier could return interval-valued estimation such as

$\{p(\omega_1) = [0, 0.2], p(\omega_2) = [0.3, 0.4], p(\omega_3) = [0.4, 0.6]\}$. The probabilities are interval-valued, the width of which represents the uncertainty about the estimations. A large interval means that we have poor or inconsistent information about the class.

5.2 Measurements

Different from the various measurements used for uncertainty estimation, the measurement of partial classification are surrounded by two terms, i.e., the accuracy and the informativeness. The accuracy is literally what it means, the informativeness means the predicted set cardinality and the informative prediction should not contain too many candidates. The classification problem is about assigning a prediction $\hat{y}$ to a sample $\mathbf{x}$ issued from the input feature space. When the prediction $\hat{y}$ is a single label of the output space Ω, it is what we call precise prediction. For accuracy, we can evaluate the performance of the precise prediction, as an example, by the "0/1" accuracy. As shown in Table 5.1, take the same example that was used to explain the DST.

$acc_{\hat{y}}(y)$	Ground-truth label		
	y = r	y = g	y = b
$\hat{y} = r$	1	0	0
$\hat{y} = g$	0	1	0
$\hat{y} = b$	0	0	1

TABLE 5.1: The accuracy matrix is defined according to the prediction and the ground-truth label. The accuracy equals 1 if $\hat{y} = y$; 0 otherwise. "r" represents "red", "g" represents "green", and "b" represents "blue".

The classical method to evaluate classifier performance is the expected accuracy

$$\mathbb{E}\left(acc_{\hat{y}}\right) = \frac{1}{N} \sum_{\mathbf{x} \in \mathcal{X}} acc_{\hat{y}}(f_{\boldsymbol{\theta}}(\mathbf{x})). \tag{5.2}$$

5.2.1 The informativeness measurements

As mentioned above, partial classification is trying to find a trade-off between informativeness and accuracy. Consequently, the predicted set averaged cardinality (AC) is adopted to measure the predictive informativeness.

$$AC = \frac{1}{N} \sum_{x \in \mathcal{X}} \mid f_{\boldsymbol{\theta}}(\boldsymbol{x}) \mid . \tag{5.3}$$

5.2.2 The accuracy measurements

When dealing with partial classification, since the prediction is a subset, "0/1" accuracy is improper. There is a need to derive accuracy function $acc_{\hat{y}}$ for partial classification as shown in Table 5.2. The existing methods can be classified into the following three main branches, i.e., discounted accuracy, utility discounted accuracy, and F_β measure.

A common proposal is the so-called discounted accuracy. As shown in Table 5.3, such that $acc_{\hat{y}}(y) = 1/|\hat{y}|$ if $y \in \hat{y}$; $acc_{\hat{y}}(y) = 0$ otherwise. Once get the discounted accuracy for each sample, we can calculate averaged discounted accuracy (ADA) to measure the prediction performance.

$acc_{\hat{y}}(y)$	Ground-truth label		
	y = r	y = g	y = b
$\hat{y} = r$	1	0	0
$\hat{y} = g$	0	1	0
$\hat{y} = b$	0	0	1
$\hat{y} = \{r, g\}$	$acc_{\{r,g\}}(r)$	$acc_{\{r,g\}}(g)$	$acc_{\{r,g\}}(b)$
$\hat{y} = \{g, b\}$	$acc_{\{g,b\}}(r)$	$acc_{\{g,b\}}(g)$	$acc_{\{g,b\}}(b)$
$\hat{y} = \{r, b\}$	$acc_{\{r,b\}}(r)$	$acc_{\{r,b\}}(g)$	$acc_{\{r,b\}}(b)$
$\hat{y} = \{r, g, b\}$	$acc_{\{r,g,b\}}(r)$	$acc_{\{r,g,b\}}(g)$	$acc_{\{r,g,b\}}(b)$

TABLE 5.2: The accuracy matrix for the partial classification.

$$acc_{\hat{y}}(y) = \frac{1}{\mid \hat{y} \mid} \mathbb{1}(y \in \hat{y}), \quad ADA = \frac{1}{N} \sum_{\mathbf{x} \in \mathcal{X}} acc_{\hat{y}}(f_{\boldsymbol{\theta}}(\mathbf{x})). \tag{5.4}$$

$acc_{\hat{y}}(y)$	Ground-truth label		
	y = r	y = g	y = b
$\hat{y} = r$	1	0	0
$\hat{y} = g$	0	1	0
$\hat{y} = b$	0	0	1
$\hat{y} = \{r, g\}$	$\frac{1}{2}$	$\frac{1}{2}$	0
$\hat{y} = \{g, b\}$	0	$\frac{1}{2}$	$\frac{1}{2}$
$\hat{y} = \{r, b\}$	$\frac{1}{2}$	0	$\frac{1}{2}$
$\hat{y} = \{r, g, b\}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$

TABLE 5.3: The accuracy matrix is defined according to the discount accuracy for partial classification.

Rather than adopting the discounted accuracy, Zaffalon et al. [ZCM12] propose to keep $acc_{\hat{y}}(y) = 0$ if $y \notin \hat{y}$, but to take $acc_{\hat{y}}(y) = g(\frac{1}{|\hat{y}|})$ if $y \in \hat{y}$ where g is a utility function on $[0, 1]$ such that $g(\frac{1}{|\hat{y}|}) \geq \frac{1}{|\hat{y}|}, g(1) = 1$ and $g(0) = 0$. They interpret g as a concave function modeling the risk-aversion, i.e., the utility or the cautiousness-seeking attitude of the decision-maker. In particular, they propose specific quadratic forms of g as:

$$g(x) = -0.6 \left(\frac{1}{\mid \hat{y} \mid} \right)^2 + 1.6 \frac{1}{\mid \hat{y} \mid}. \tag{5.5}$$

This specific utility keeps some appealing properties of the discounted accuracy. When the correct class is not in $\hat{y}$ the accuracy should stay at 0, when the prediction is both precise and accurate then the accuracy should be 1. Moreover, this utility function can express the risk-aversion by giving a smaller cost or higher accuracy to imprecise but correct predictions compared to the discounted accuracy. As shown in Table 5.4.

F_β measure, which computes the harmonic mean weighted by the coefficient β between precision P and recall R, can be use to fulfill $acc_{\hat{y}}(y)$, as shown in Table 5.5.

$$F_\beta = \frac{\left(1 + \beta^2\right) PR}{\beta^2 P + R}, \quad P = \frac{\mathbb{1}_{\hat{y}}(y)}{|\hat{y}|}, \quad R = \mathbb{1}_{\hat{y}}(y). \tag{5.6}$$

$acc_{\hat{y}}(y)$	Ground-truth label		
	y = r	y = g	y = b
$\hat{y} = r$	1	0	0
$\hat{y} = g$	0	1	0
$\hat{y} = b$	0	0	1
$\hat{y} = \{r, g\}$	0.65	0.65	0
$\hat{y} = \{g, b\}$	0	0.65	0.65
$\hat{y} = \{r, b\}$	0.65	0	0.65
$\hat{y} = \{r, g, b\}$	0.46	0.46	0.46

TABLE 5.4: The accuracy matrix is defined according to the utility discount accuracy for the partial classification.

$acc_{\hat{y}}(y)$	Ground-truth label		
	y = r	y = g	y = b
$\hat{y} = r$	1	0	0
$\hat{y} = g$	0	1	0
$\hat{y} = b$	0	0	1
$\hat{y} = \{r, g\}$	$\frac{2}{3}$	$\frac{2}{3}$	0
$\hat{y} = \{g, b\}$	0	$\frac{2}{3}$	$\frac{2}{3}$
$\hat{y} = \{r, b\}$	$\frac{2}{3}$	0	$\frac{2}{3}$
$\hat{y} = \{r, g, b\}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$

TABLE 5.5: The accuracy matrix is defined according to the F_β function for the partial classification.

5.2.3 The hybrid

The hybrid approach takes into account both accuracy and predicted set cardinality. This framework is proposed and analyzed in the literature [Ha97]. Its goal is to find the optimal error-reject trade-off using a specific loss structure defined as follows:

$$\ell_{\hat{y}}(y) = \ell_1(y, \hat{y}) + \ell_2(\mid \hat{y} \mid -1), \tag{5.7}$$

where ℓ_1 modelling the loss of missing the true class y. ℓ_2 representing the cost of being imprecise, with the condition that $\ell_2 < \frac{1}{2\ell_1}$ for all y. The obtained cost matrix is given by Table 5.6.

$c_{\hat{y}}(y)$	Ground-truth label		
	y = r	y = g	y = b
$\hat{y} = r$	0	ℓ_1	ℓ_1
$\hat{y} = g$	ℓ_1	0	ℓ_1
$\hat{y} = b$	ℓ_1	ℓ_1	0
$\hat{y} = \{r, g\}$	ℓ_2	ℓ_2	$\ell_1 + \ell_2$
$\hat{y} = \{g, b\}$	ℓ_2	$\ell_1 + \ell_2$	ℓ_2
$\hat{y} = \{r, b\}$	$\ell_1 + \ell_2$	ℓ_2	ℓ_2
$\hat{y} = \{r, g, b\}$	$2\ell_2$	$2\ell_2$	$2\ell_2$

TABLE 5.6: The accuracy matrix is defined according to the class-selective rejection rule for the partial classification.

5.3 Datasets and baselines

As we have done in the uncertainty estimation part, we summarize the often used datasets and baseline from different partial classification methods. The commonly used datasets are synthetic, e.g, datasets generated based on the Gaussian distribution and UCI dataset. The most prevalent measurements used by various methods are the utility discounted accuracy and predicted set cardinality. For the baseline, different from the uniform and widespread admit methods of the uncertainty estimation, different partial classification methods have their preferences. In this thesis, we take PCBF [MD21] method as the baseline.

5.4 Conclusions

The related works presented in this chapter include two categories that can be directly applied to pre-trained models and methods that need to accompany the new loss function and new model architecture. The former can be directly applied to pre-trained models, but also requires a specific loss function to train the model. The latter needs to construct a new loss function or even a change to the original model architecture. In addition, some commonly used measurements are introduced, e.g., average cardinality and average discounted accuracy.

Chapter 6

Novelty partial classification approaches

As mentioned, both uncertainty estimation and partial classification should be fast and packageable as an auxiliary module that can be integrated on the top of pre-trained models without retraining or major architecture changes. In this chapter, we introduced two novelty methods that are fulfilled only based on the model output. The kernel idea is to assign belief to nested subsets. And choose the subset with the maximum belief as the prediction.

6.1 The proposed approach based on the model output

6.1.1 Introduction

The precise or certainty classification [LLY+16, WJC+20] is a well-known issue in which a sample is classified into one and only one of the training classes. Unfortunately, such a strict classification sometimes results in misclassification when the input sample does not contain sufficient evidence to identify a certain class. The partial classification [Ha97, MD21, MWD+21] is one of the more practical ways to solve this problem. It is defined as the assignment of a sample into a class subset. For example, let us consider a class set $\Omega = \{\omega_1, \omega_2, \omega_3\}$. Here, we cannot manage to reliably classify a sample into a single class, but it is almost sure that it does not belong to ω_1. Consequently, it is more reasonable to assign it to the subset $\{\omega_2, \omega_3\}$. In practice, high ambiguity emerges in numerous applications, and large-scale datasets contain a fair amount of confusing samples, these are the bedrock of the usage of partial classification. For instance, the goal of road surfaces classification [ZYZZ16] is to produce a prediction with almost null errors which can be expected from partial classification.

A considerable amount of literature has been published on partial classification and has always led to different classification strategies. On the one hand, researchers attempted to predict a subset with prior fixed cardinality [RDS+15] or with a rejection option [KSW15, LPB17, LWOL20]. They can be seen as a special case of partial classification by classifying the sample into one specific class subset. On the other hand, a number of authors attempted to modify the loss function [DCDB09, Ha97, MWD+21] or build a new classifier [SLW19, VGS05, Zaf02] to provide beliefs for predicted sets. Usually, such algorithms are time-consuming. To this end, it is essential to reduce the computation and time complexity by efficiently and sufficiently leveraging the information provided by the pre-trained neural network.

We presented a new Partial Classification method based on pre-trained CNN-based Model Outputs (PCMO) [XAC21]. Different from the existing methods, the PCMO method simply and efficiently fulfilled partial classification only based on pre-trained CNN-based model outputs, and provided beliefs to predicted sets for further prediction.

As shown in Fig. 6.1, at first, the CNN-based model extracts features from the input layer through the combination of the feature extraction process and the fully connected layer between the last hidden layer and the output layer. Second, the received features are converted into beliefs under the Dempster-Shafer theory (DST) [Sha76] through the output to possibility and the possibility to belief processes. Finally, the PCMO method performs partial classification based on the produced beliefs by choosing the maximum belief and generating the corresponding class subset as the prediction.

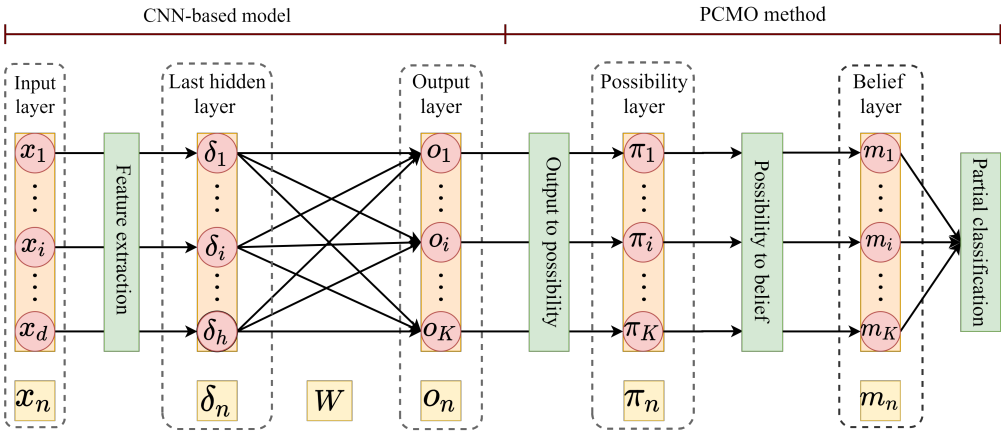


FIGURE 6.1: The framework of the PCMO method. The feature extraction process is demonstrated simply, it can be any kind of CNN-based architecture, e.g., fully connected layer, LeNet [LBBH98], GoogLeNet [SLJ+15], or ResNet [HZRS16]. The detailed output to possibility, possibility to belief, and partial classification processes are presented in Section 6.2.2.

The contributions can be summarized as follows:

- The most striking achievement is that the proposed method is fulfilled only based on model outputs that can be applied to any pre-trained CNN-based model without any demand to retrain the model or conduct any further modifications.
- By considering good features of log function and analyzing the regular pattern of model outputs, a novel and reasonable transformation from model outputs to possibility distribution is proposed.

6.1.2 The pattern of the CNN-based model outputs

The convolutional neural network (CNN) [LBBH98] is a machine learning method that uses multiple layers to progressively extract features from raw data as sample representation. Define a training dataset $\mathcal{D}^{train} = \{\boldsymbol{x}_n, y_n\}_{n=1}^{N}$ has K classes, where $\boldsymbol{x}_n \in \mathbb{R}^d$, and $y_n \in \{1, \ldots, i, \ldots, K\}$. A CNN-based model $f_{\boldsymbol{\theta}}(\boldsymbol{x})$, with the entire model parameter $\boldsymbol{\theta}$. From the last hidden layer $\boldsymbol{\delta}_n = \{\delta_1, \ldots, \delta_i, \ldots, \delta_h\}$ to the output layer $\boldsymbol{o}_n = \{o_1, \ldots, o_i, \ldots, o_K\}$, the weight $\boldsymbol{W} \in \mathbb{R}^{h \times K}$ defines a transformation, i.e., $\boldsymbol{o}_n = \boldsymbol{W}\boldsymbol{\delta}_n$. In general, the empirical loss $\mathcal{L}_{\boldsymbol{\theta}}$ over $\mathcal{D}^{train}$ has the following form:

$$\mathcal{L}_{\boldsymbol{\theta}}(\mathcal{D}^{train}) = \sum_{n=1}^{N} \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}), y_n), \tag{6.1}$$

where $\ell(\cdot)$ is the cross-entropy loss.

The CNN-based model mentioned in this thesis respects two usual and reasonable assumptions. The model loss Eq. (6.1) converges to zero when iteration t approaches infinity, i.e., $\lim_{t \to \infty} \mathcal{L}_{\boldsymbol{\theta}_t} = 0$, and the model's last hidden layer and the output layer are

fully connected. Based on the two assumptions, [ZL20] shows, both theoretically and empirically, that the last weight layer $\boldsymbol{W}$ of a neural network converges to a Support Vector Machine (SVM) trained on the last hidden layer output with the commonly used cross-entropy loss.

Since $\boldsymbol{W}$ represents a hyperplane, the farther the input sample is from the hyperplane, the greater the corresponding class output will be. As illustrated in Fig. 6.2, the model output contours of the CNN-based model are radiated, becoming larger as the distance from the hyperplane increases.

6.1.3 Proposed Method

As it can be seen from Section 6.1.2, a sample that is far from the training dataset occupies high outputs for several classes leading to high probabilities for the corresponding classes, resulting in the improper execution of precise classification. Consider, from another angle, the high outputs for multiple classes can be regarded as evidence to classify a sample into a class subset. From this point, we proposed to calculate beliefs only based on pre-trained CNN-based model outputs to fulfill partial classification. Moreover, we chose the possibility as the bridge between model outputs and beliefs, then proposed the following transformations.

Sorting $\boldsymbol{o}_n$ by descending order to get $\boldsymbol{o}'_n = \{o'_1 \geq \cdots \geq o'_i \geq \cdots \geq o'_K\}$, where o'_i is the i^{th} largest element in $\boldsymbol{o}_n$. Then, a prerequisite step is to prepare a temporary vector $\boldsymbol{v}_n = \{v_1, \ldots, v_i, \ldots, v_K\}$ based on Eq. (6.2) that coordinates with Eq. (6.3) to calculate the target possibility distribution.

$$v_i = \frac{1}{|A_i|} \sum_{k=1}^{i} \log_2(1 + \max(0, o'_k)), \tag{6.2}$$

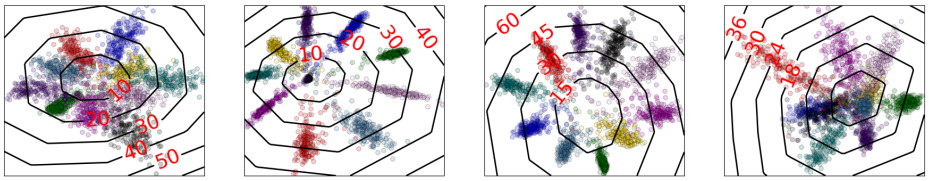
where $\frac{1}{|A_i|}$ is used to penalize the ambiguity caused by classifying $\boldsymbol{x}_n$ to subset A_i. If we consider a reasonable assumption that the desired possibility transformation should keep the original pattern of outputs, escalating the difference for small values while narrowing the difference for bigger values. The $\log_2$ function should be chosen, which tends to be flat after the initial rapid growth. At the same time, in order to avoid the negative possibility, use $\max(0, o'_k)$ to clamp the outputs and move the $\log_2$ function to the left by one unit.

After min-max normalization by Eq. (6.3), we can get the possibility distribution $\boldsymbol{\pi}_n = \{\pi_1, \ldots, \pi_i, \ldots, \pi_K\}$. As claimed in [DP82] that any possibility distribution is a plausibility function corresponding to a consonant m. Our possibility distribution $\boldsymbol{\pi}_n$ can be transformed to belief function m according to Eq. (6.4), the detailed calculation is presented in Fig. 6.3. In our case, π_K equals zero, which implies that $m(\Omega)$ equals zeros.

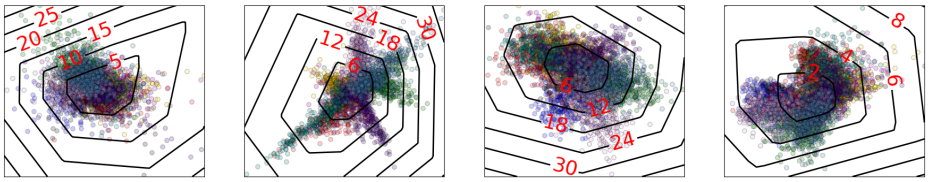
$$\boldsymbol{\pi}_n = \frac{\boldsymbol{v}_n - \min(\boldsymbol{v}_n)}{\max(\boldsymbol{v}_n) - \min(\boldsymbol{v}_n)}. \tag{6.3}$$

$$m(A_i) = \begin{cases} \pi_j - \pi_{j+1} & \text{if } A_i = \{\omega_1, \cdots, \omega_j\} \text{ for some } j \in \{1, \cdots, K-1\}, \\ 0 & \text{otherwise.} \end{cases} \tag{6.4}$$

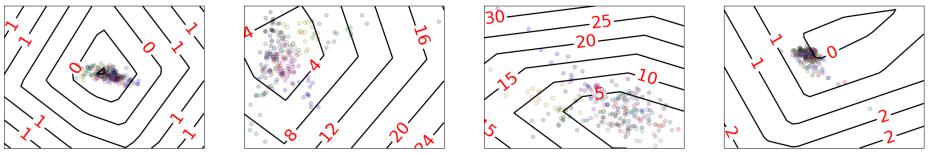
The PCMO classification algorithm is detailed in Algorithm 1. Based on the beliefs calculated through Eqs. (6.2), (6.3), and (6.4), we chose the subset with the maximum belief as the prediction. Suppose, the maximum belief is $m(A_i)$, the PCMO method will generate the predicted set $\{\omega_1, \ldots, \omega_i\}$ corresponding to the top i maximum outputs. The Table 6.1 shows a comparison between the belief calculated with log function (PCMO)



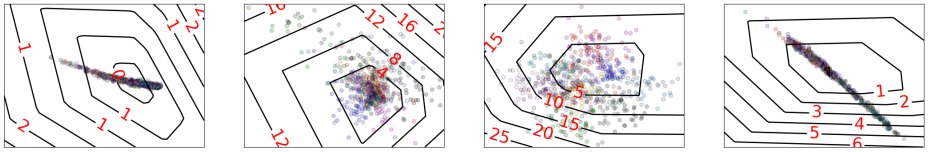
(A) MNIST dataset.



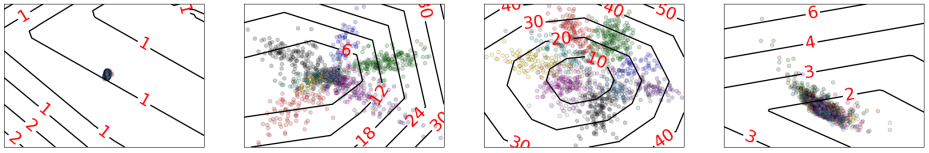
(B) Cifar10 dataset.



(C) Cifar100 dataset.



(D) Lsun dataset.



(E) Svhn dataset.

FIGURE 6.2: The model output contours on MNIST [LCB10], Cifar10 [Kri12], Cifar100 [Kri12], Lsun [YZS+15], and Svhn [NWC+11] datasets. Different prevalent CNN-based models are verified. From left to right are LeNet [LBBH98], GoogLeNet [SLJ+15], ResNet [HZRS16], and MobileNet [HZC+17]. For visualization purposes, the h that appears in the last hidden layer is set as two. Meanwhile, according to the minimum and maximum column values of $\boldsymbol{\delta} \in \mathbb{R}^{N\times 2}$ a 2D mesh can be generated. Feed this mesh into the last hidden layer to get the outputs which can be regarded as contours. As we can see, the pattern of the CNN-based model is that a sample far from the training dataset can bring high outputs and lead to high probabilities for several classes. Under this context, partial classification rather than precise classification should be used.

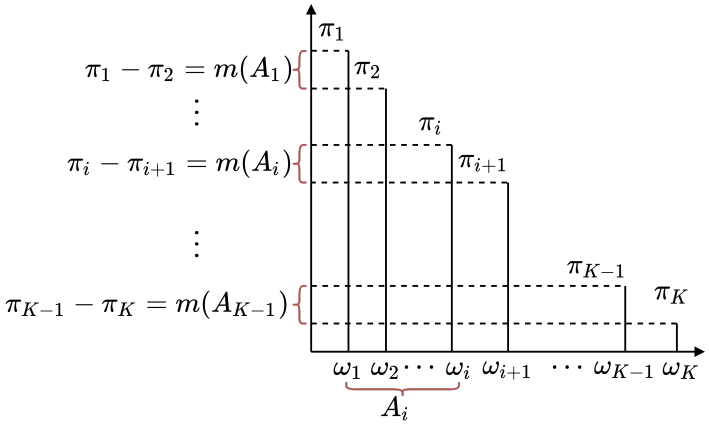


FIGURE 6.3: The belief calculation process in PCMO.

Algorithm 1 Classification process for a sample $\boldsymbol{x}_n$

Require:
Model outputs $\boldsymbol{o}_n \in \mathbb{R}^K$
Ensure:
Predicted set *predSet*
1: Sorting $\boldsymbol{o}_n$ in descending order to get the sorted index *index* and sorted outputs $\boldsymbol{o}'_n$
2: Calculating vector $\boldsymbol{v}_n$ base on $\boldsymbol{o}'_n$ according to Eq. (6.2)
3: Calculating possibility distribution $\boldsymbol{\pi}_n$ base on $\boldsymbol{v}_n$ according to Eq. (6.3)
4: Calculating belief $\boldsymbol{m}_n$ base on $\boldsymbol{\pi}_n$ according to Eq. (6.4)
5: Obtaining the maximum belief index $idx = argmax(\boldsymbol{m}_n)$ for the sample $\boldsymbol{x}_n$
6: Generating the predicted set $predSet = list(index[1 : idx])$, which contains the candidate classes
7: return *predSet*

and belief calculated without log function. Noteworthy, the kernel ideal of partial classification is to assign a value to each predicted subset. This value can be called belief or possibility. If we focus on the possibility, then the possibility of each predicted subset has to be constructed based on the possibility of each state (class). This is theoretically possible to execute partial decisions in the possibilistic framework, but further verification is needed.

6.1.4 Experiments

6.1.4.1 Experiment protocol

There are four prevalent datasets involved, i.e., modified national institute of standards and technology (MNIST) [LCB10], canadian institute for advanced research 10 (CIFAR10) [Kri12], street view house number (SVHN) [NWC+11], and large-scale scene understanding challenge (LSUN) [YZS+15]. The characteristics of the datasets are shown in Table 6.8. Four classical CNN-based models, i.e., LeNet [LBBH98], GoogLeNet [SLJ+15], residential energy services network (ResNet) [HZRS16], and MobileNet [HZC+17], are adopted to prove the efficiency of the PCMO method. We used the cross-entropy loss as the loss function and the rectified linear unit (ReLU) as the activation function. For training, we use the holdout strategy, that is to say, 70% of samples are used for training, and 30% of samples used for testing. The Adam optimizer has been used with default settings for training. Since the proposed method can be used for any pre-trained neural

Id	Type	Output $(\omega_1, \omega_2, \omega_3, \omega_4, \omega_5)$	Belief with log (PCMO) $(m(A_1), m(A_2), m(A_3), m(A_4))$	Belief without log
1	ID	(10.0, 1.0, 1.0, 1.0, 1.0)	(**0.62**, 0.20, 0.10, 0.06)	(**1.0**, 0.0, 0.0, 0.0)
2	OOD	(10.0, 10.0, 10.0, 10.0, 10.0)	(0.0, 0.0, 0.0, **1.0**)	(nan, nan, nan, nan)
3	IM	(10.0, 10.0, 10.0, 10.0, 1.0)	(0.0, 0.0, 0.0, **1.0**)	(0.0, 0.0, 0.0, **1.0**)
4	IM	(10.0, 10.0, 10.0, 1.0, 1.0)	(0.0, 0.0, **0.62**, 0.37)	(0.0, 0.0, **1.0**, 0.0)
5	IM	(10.0, 10.0, 1.0, 1.0, 1.0)	(0.0, **0.55**, 0.27, 0.16)	(0.0, **1.0**, 0.0, 0.0)
6	IM	(10.0, 9.0, 8.0, 7.0, 6.0)	(0.22, 0.23, 0.25, **0.27**)	(**0.25**, 0.25, 0.25, 0.25)
7	IM	(10.0, 9.0, 8.0, 6.0, 6.0)	(0.19, 0.21, **0.36**, 0.22)	(0.25, 0.25, **0.5**, 0.0)
8	IM	(10.0, 10.0, 10.0, 9.5, 9.5)	(0.0, 0.0, **0.62**, 0.37)	(0.0, 0.0, **1.0**, 0.0)
9	IM	(5.0, 4.0, 3.0, 2.0, 1.0)	(0.19, 0.22, 0.26, **0.32**)	(**0.25**, 0.25, 0.25, 0.25)
10	IM	(5.0, 5.0, 4.0, 4.0, 4.0)	(0.0, **0.55**, 0.27, 0.16)	(0.0, **1.0**, 0.0, 0.0)
11	IM	(3.0, 2.0, 2.0, 1.0, 1.0)	(**0.36**, 0.12, 0.31, 0.19)	(**0.5**, 0.0, 0.5, 0.0)

TABLE 6.1: An example to show the role played by the log function in PCMO. Suppose the training dataset contains five classes, i.e., $\Omega = (\omega_1, \omega_2, \omega_3, \omega_4, \omega_5)$, and 11 synthetic model outputs. Different beliefs are calculated according to whether use the log function. The reason for using the log function is to take advantage of its growth trend, which is sharp at the beginning then smooth. Consequently, the PCMO can map the larger output values to be smaller, thus maintaining the power of large values while keeping the weight of small output values. As mentioned above, the PCMO chooses the subset with the maximum belief as the prediction, shown in bold in the table. The first thing can be seen is that the PCMO is able to make predictions correctly. For obtained beliefs without using the log function, when all the outputs are the same, although it cannot happen in practice, it will make the divisor equal to zero leading to nan. In addition, we can summarize another two points. First, when the log function is not used, the belief will be completely assigned to a certain set, and completely exclude the possibility of other sets to be predicted, such as samples 3, 4, 5, 8, and 10. Second, when the difference between the output values is the same, the belief is assigned equally without using the log function. And, the effect of the output range is not taken into account, for example, sample 6 belongs to $[1, 5]$, and sample 9 belongs to $[6, 10]$ are assigned with the same belief $(0.25, 0.25, 0.25, 0.25)$.

network, we compare the PCMO method with the same type of method. Consequently, we chose energy score (based on pre-trained model outputs) [LWOL20], dropout score (based on several executions of the pre-trained model on the testing dataset) [KSW15], and ensemble score (based on executions of several pre-trained models on the testing dataset) [LPB17]. For criteria, we adopt ADA and AC as introduced in Section 5.2.

Name	# Used class	# Training samples	# Testing samples
MNIST	10	55000	10000
CIFAR10	10	50000	10000
SVHN	10	4000	2000
LSUN	10	2400	2000

TABLE 6.2: An overview of datasets involved in PCMO.

6.1.4.2 Evaluation of the PCMO method

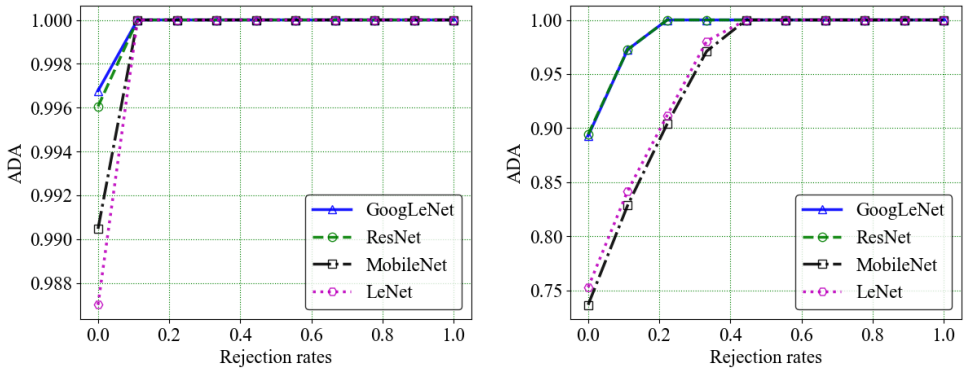
The PCMO method performs partial classification by choosing the predicted set that occupies the maximum belief. Naturally, the bigger cardinality of the predicted set indicates a more confusing input sample. Thus, in order to verify the efficiency of partial classification and the capacity of reducing the classification risk under the PCMO method. We rejected the most confusing samples according to different rejection rates [NZH09].

On the one hand, we executed the PCMO method for different CNN-based models when rejection rates change from 0.0 to 1.0. Fig. 6.4 is quite revealing in two ways. First, the ADA increases along with the increase in rejection rates. Second, the selected four classical CNN-based models achieved good ADA values, except for the slightly worse initial accuracy of LeNet and MobileNet due to their simple model architecture. This indicates that the PCMO method performed partial classification based on the calculated beliefs. For AC, which has a high value for Imprecise (IM) samples because PCMO trend to make caution predictions. Then decrease along with the reduction of IM sample quantity. The performance of the different CNN-based models further proves that partial classification can be achieved only based on the CNN-based model outputs.

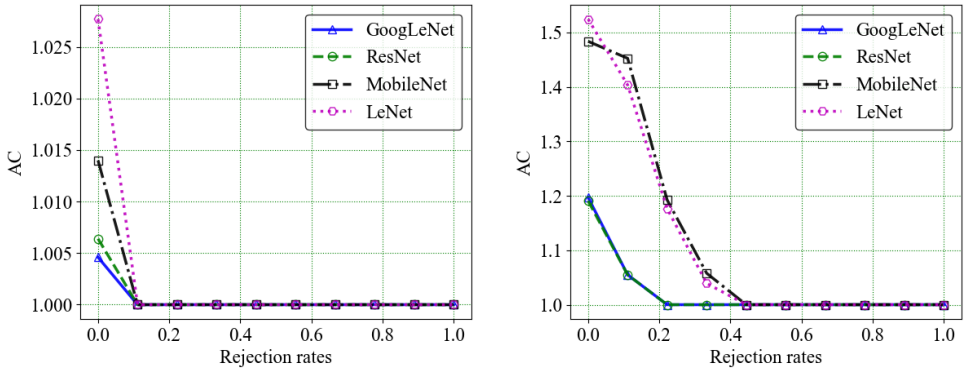
On the other hand, we verified different methods based on different neural networks and methods, as shown in Fig. 6.5. The ADA of the PCMO method increases significantly when the rejection rate increases. In contrast, the ADA of the other methods performed fluctuation or insensitivity when the rejection rate increases. The striking performance is evidence that the PCMO method makes a well-distributed partial classification while the others only classified samples to a class subset when the rejection rate is large.

6.1.4.3 Comparison among different normalization strategies

The normalization step is included in the process of transforming the model output into possibility. Since we need to get the possibility between 0 and 1, in PCMO, we use the max-min normalization. In this section, we evaluated some other options, such as dividing by the maximum value and softmax as shown in Table 6.3. The performance of the three normalizations based on different models trained on different datasets is shown in the Fig. 6.6. From the results, we can see the performance of these three strategies is similar, the "divide maximum" is a little better. That is because it can assign belief to the entire set, and classify the confusing samples into the entire set. Based on the comparison, the normalization strategy based on the knowledge provided by the training dataset can be imaged.



(A) The ADA curves.



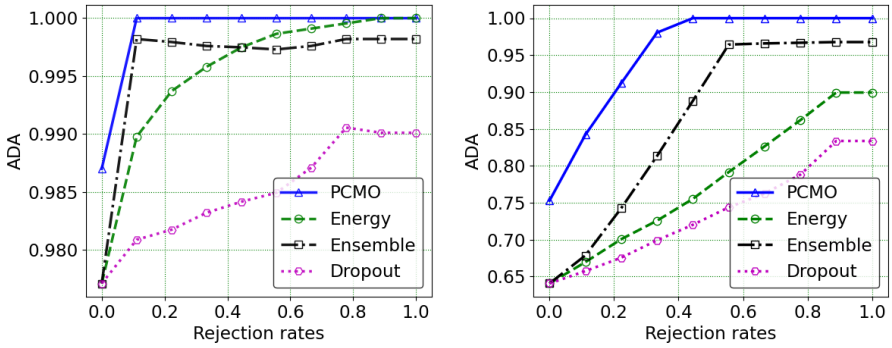
(B) The AC curves.

FIGURE 6.4: The performance in terms of ADA (the first row) and AC (the second row) values of different CNN-based models with different rejection rates based on Mnist (left) and Cifar10 (right) datasets.

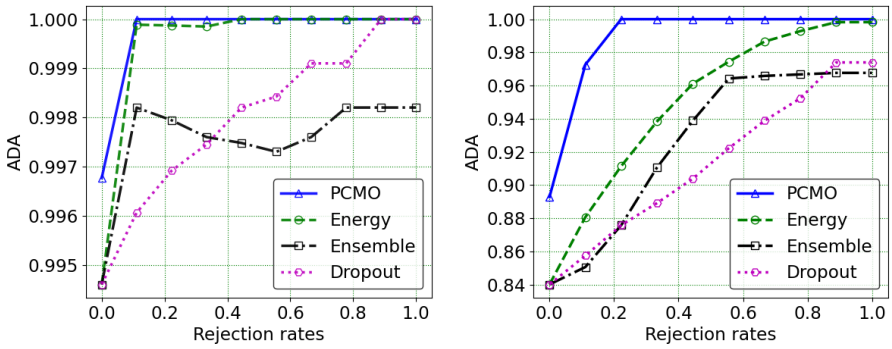
$$\begin{aligned} \text{Max-min:} \quad & \frac{\boldsymbol{x} - \min(\boldsymbol{x})}{\max(\boldsymbol{x}) - \min(\boldsymbol{x})}, \\ \text{Divide maximum:} \quad & \frac{\boldsymbol{x}}{\max(\boldsymbol{x})}, \\ \text{Softmax:} \quad & \frac{\exp(\boldsymbol{x})}{\sum_{i=1}^{K} \exp(\boldsymbol{x}_i)}. \end{aligned} \tag{6.5}$$

6.1.5 Conclusions

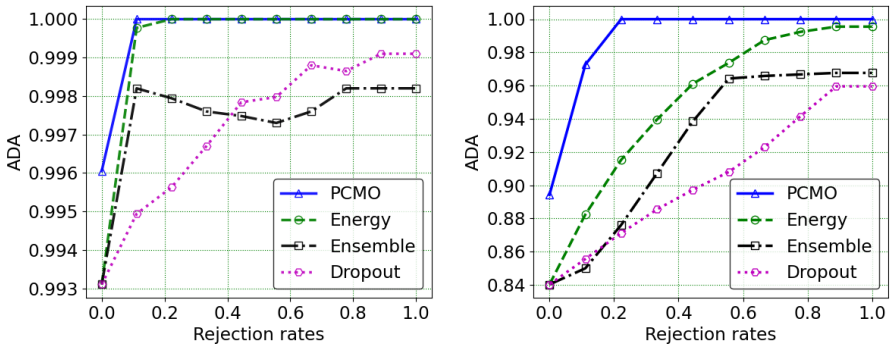
In this section, we present a new partial classification method named PCMO, which is fulfilled based on pre-trained CNN-based model outputs. From this point of view, the time complexity of PCMO method is $O(1)$. At first, we theoretically and empirically proved our hypothesis that a sample far from the training dataset can provide high outputs and lead to high probabilities for several classes. Second, we adopted possibility as the bridge fulfilling the transformation from model outputs to beliefs for the predicted sets. Then, we verified the PCMO method with different CNN-based models, as well as different methods based on four datasets. From the production of ADA and AC criteria, we can tell that the PCMO method performs better than the existing methods, as it can provide a high belief to a certain sample, as well as a high uncertainty to a confusing



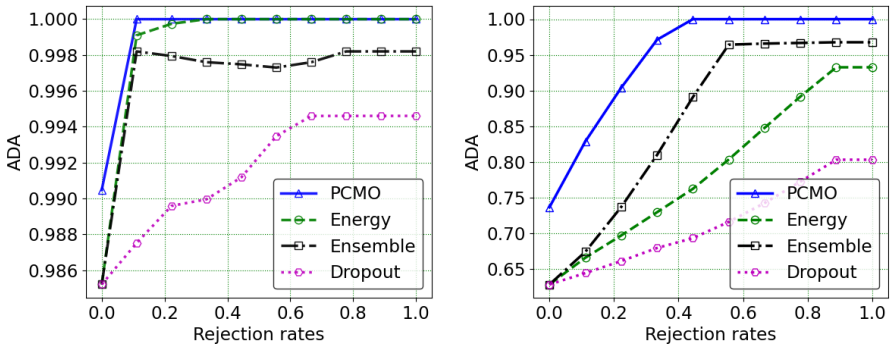
(A) The curves for LeNet.



(B) The curves for GoogLeNet.

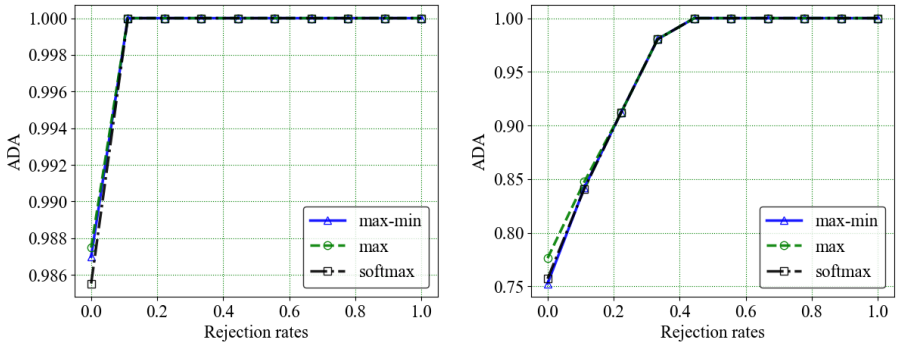


(C) The curves for ResNet.

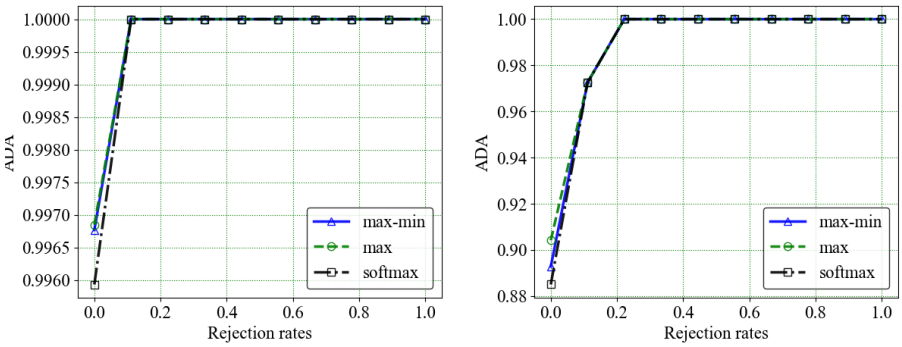


(D) The curves for MobileNet.

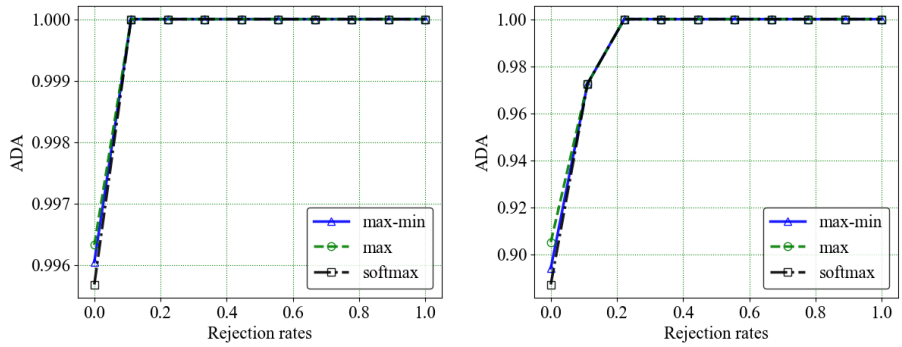
FIGURE 6.5: The performance in terms of ADA values of different methods with different rejection rates based on Mnist (left) and Cifar10 datasets, and LeNet (the first row), GoogLeNet (the second row), ResNet (the third row), and MobileNet (the fourth row).



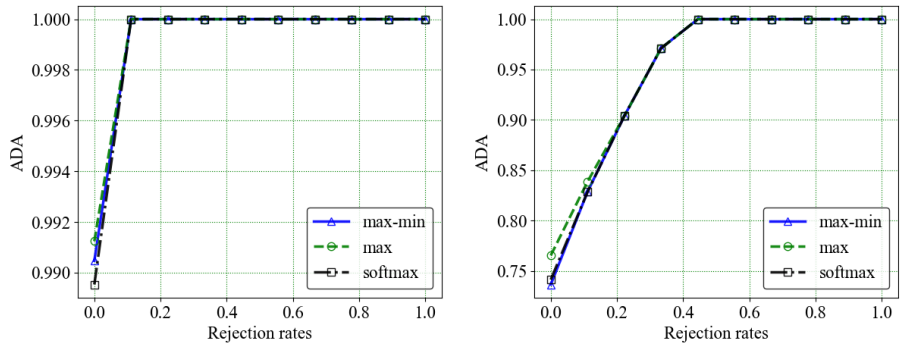
(A) The curves for LeNet.



(B) The curves for GoogLeNet.



(C) The curves for ResNet.



(D) The curves for MobileNet.

FIGURE 6.6: The performance in terms of ADA values of different methods with different rejection rates based on Mnist (left) and Cifar10 (right) datasets, and LeNet (the first row), GoogLeNet (the second row), ResNet (the third row), and MobileNet (the fourth row).

Name	The value range after normalization	$m(\emptyset)$	$m(\Omega)$
Max-min	$\min(\cdot) = 0,\ \max(\cdot) = 1$	0	0
Divide maximum	$0 \leq \min(\cdot) \leq 1, \max(\cdot) = 1$	0	[0, 1]
Softmax	$0 \leq \min(\cdot),\ \max(\cdot) \leq 1$	[0, 1]	[0, 1]

TABLE 6.3: The belief assignment comparison among different normalization strategies in PCMO. As can be seen from Eq. 6.4, the beliefs for empty set and entire are calculated based on the minimum and maximum value, respectively. Taking the max-min normalization strategy as an example. Since, the maximum value and the minimum value provided by the max-min normalization are 0 and 1, consequently, the $m(\emptyset)$ and $m(\Omega)$ are 0.

sample. The PCMO method proved effective in increasing prediction accuracy and ultimately reducing the classification risk. The partial classification process based on PCMO involves many steps, such as computing a temporary variable and performing normalization. Those steps might lead to a loss of information. At the same time, the PCMO method is not able to provide belief for the empty set or the entire set. Therefore, the next step is to propose a new way to fulfill the transformation from output to possibility and assign belief to the empty set and the entire set.

6.2 The proposed approach rethink the possibility calculation

6.2.1 Introduction

Although both the partial classification method, and the uncertainty estimation method, have high potential in high-risk fields such as medical image analysis or autonomous driving, the following challenges remain.

1. Requirement of the additional Out-of-Domain (OOD) dataset. For instance, energy score [LWOL20], which needs the additional OOD dataset to train the model. However, sometimes, the additional OOD dataset is not reachable.

2. Inability of the partial classification to distinguish between sample with "total ignorance" about its class membership, which needs to be classified into the empty set, and sample with "total imprecision" about its class membership, which needs to be classified into the entire set. The existing methods, tend to produce belief for the entire set, which will lose effectiveness when the "total ignorance" and the "total imprecision" appear simultaneously.

3. Lack of interchangeability between the partial classification and the uncertainty estimation. In reality, different scenarios have different requirements. For example, in the task of identifying cats from dogs, it is sufficient to detect the noise sample, e.g., a car image. Whereas, for tumor image classification, tumor images need to be partially classified into different class subsets in order to make cautious decisions. Consequently, there is a demand for the combination of these two kinds of methods to cope with various scenarios. Although, some methods based on the Dempster-Shafer Theory (DST) [MD21, TXD21] have the interchangeability, but they did not mention nor fulfill it yet.

To alleviate these challenges, a novelty partial classification approach named PCBS is introduced, which combines Bell Shaped Function (BSF) and DST. As shown in Fig. 6.7, the model first extracts feature, i.e., model output, from the testing sample. Second, the

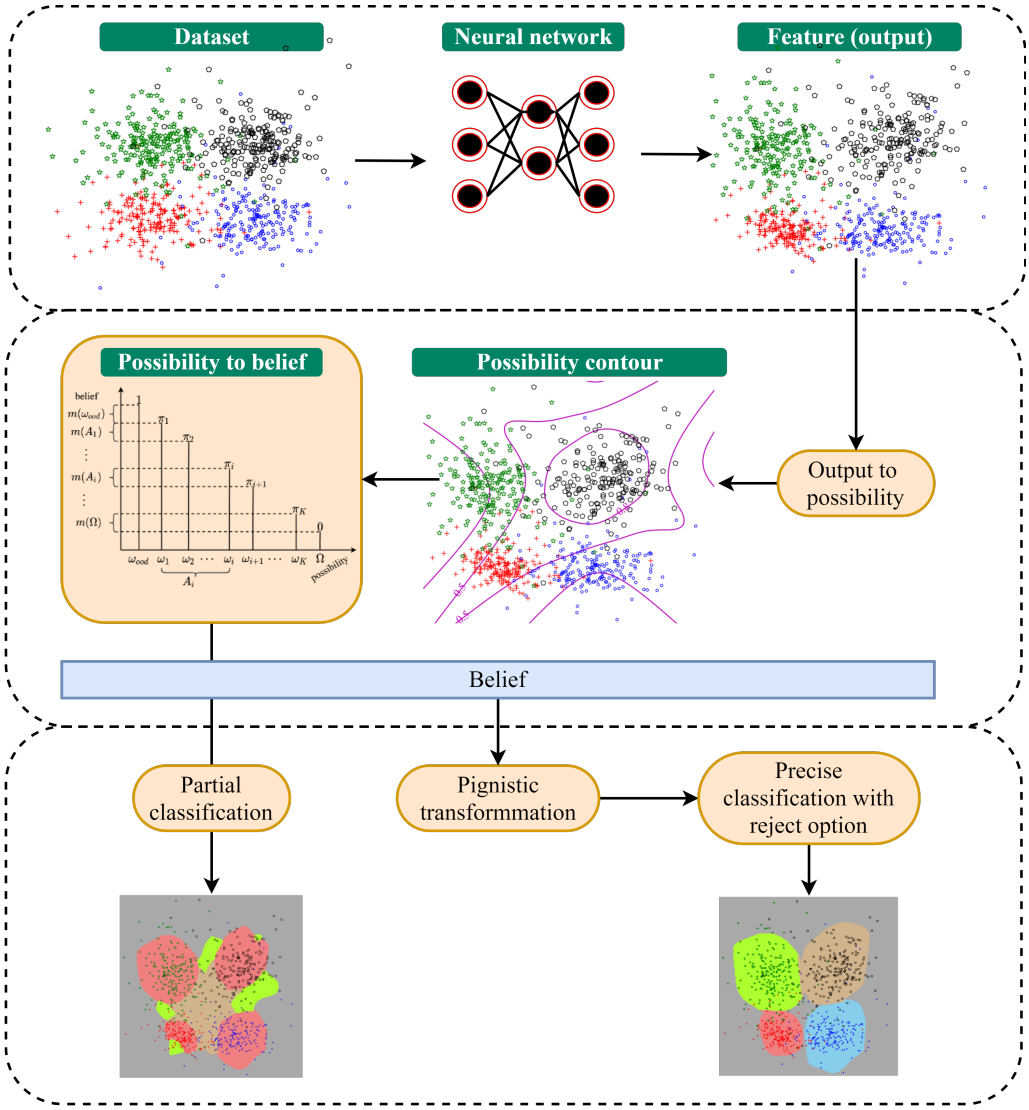
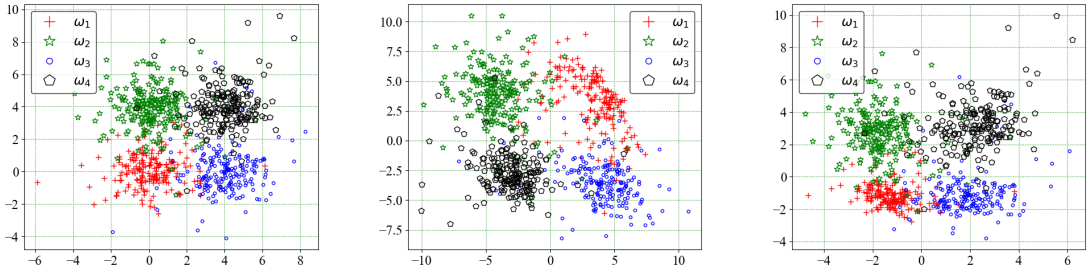


FIGURE 6.7: The framework of the PCBS method. The neural network is showed simply, it can be the classical neural network architecture, e.g., Multilayer Perceptron (MLP), LeNet, or wide residual network. The detailed BSF calculation, output to possibility, possibility to belief, and belief to probability processes are presented in Section 6.2.2. After performing different classification strategies, the samples are classified into different class subsets (or classes) represented by different color partitions, the detail is showed in Fig. 6.11.

feature is transformed into possibility by feeding them through the BSF calculated from the training output, the detailed calculation is showed in Algorithm. 2. Then, convert the possibility into belief as done in literature [Sha76], which can be used for the partial classification and the uncertainty estimation. Finally, performing partial classification by naturally choosing the class subset with the maximum belief as the prediction. Additionally, the belief can be transformed into probability under the pignistic transformation [**?**]. Performing uncertainty estimation by choosing the class with the maximum probability as the prediction.

The contributions can be listed as follows:

- Based on the characteristics and the analysis of the existing methods, a new partial classification combined with the BSF and the DST is proposed. This method



(A) The four-class dataset (the input space).

(B) Features extracted from MLP trained by the cross-entropy loss function (the feature space).

(C) Features extracted from MLP trained by the contrastive-center loss function (the feature space).

FIGURE 6.8: Four classes of 200 points each are generated from multivariate t distributions with five degrees of freedom and centered at $(0,0)$, $(0,4)$, $(4,0)$ and $(4,4)$, respectively. The model adapted is MLP which contains three hidden layers, one dense layer, and takes rectified linear unit (ReLU) as the activation function. In order to directly plot the features on a 2D surface for visualization, we reduce the output number h of the last hidden layer to two.

only requires original training and testing outputs can acquire state-of-the-art performance. The PCBS method saves a lot of effort of the OOD dataset preparation and makes the model more comfortable to apply, which makes challenge (1) less challenging.

- Another impressive achievement is that our method, in the view of partial classification, can calculate belief to class subsets including the empty set and the entire set, which offers challenge (2) a feasible way.
- The PCBS method is a combination of the partial classification and uncertainty estimation, it can be switched simply from the partial classification to the uncertainty estimation under different scenarios, and this can address the challenge (3).
- The PCBS method can be utilized under the pre-trained model directly and receive desirable performances. We also present a fine-tuned strategy by resorting to the contrastive-center loss function, which aims to produce more distinguishable features for different samples leading to better classification performance.

6.2.2 Proposed Method

In order to illustrate the proposed method, consider a synthetic four-class dataset represented in Fig. 6.8a. Let us start with an exploration of the model output, i.e., learned feature. Since the output is four-dimensional and cannot be plotted directly on a two-dimensional plane, we set the output of the last hidden layer to two, using it as an approximation to the learned feature. From Fig. 6.8b We can see that the samples are well separated. Each class has its own output characteristics which graphically means that it occupies a different plane range. This is the basis for transforming the model output into belief.

There is not a direct transformation from the model output to belief, we choose the possibility as the bridge. Since the BSF is able to map high-frequency samples to one and low-frequency samples to zero to get a closed-class-shaped boundary. We use the BSF to transform the model output into possibility as the first step of our method. As shown in Algorithms 2, 3, for each testing sample we can get the corresponding possibility $\boldsymbol{\pi}_n = \{\pi_1, \ldots, \pi_i, \ldots, \pi_K\}, \pi_i \in [0,1]$.

Algorithm 2 The BSF's parameters calculation process

Require:
Class number K
Training output $\boldsymbol{o}^{train}$
Training label $\boldsymbol{y}^{train}$
Ensure:
BSF's parameters $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}$
Initializing BSFs' parameters $\boldsymbol{a} \in \mathbb{R}^{K \times K}$, $\boldsymbol{b} \in \mathbb{R}^{K \times K}$, $\boldsymbol{c} \in \mathbb{R}^{K \times K}$
for $i = 0; i < K; i++$ **do**
  $\boldsymbol{idx} = \boldsymbol{y}^{train} == i$
  $\boldsymbol{out} = \boldsymbol{o}^{train}[\boldsymbol{idx}]$
  for $j = 0; j < K; j++$ **do**
    Calculating the maximum column output $\boldsymbol{out}_{max}$ from $\boldsymbol{out}[:, j]$
    Calculating q_a-quantile Q_{q_a} and q_b-quantile Q_{q_b} from $\boldsymbol{out}[:, j]$
    Calculating $\boldsymbol{a}[i, j] = \left(Q_{q_b} - Q_{q_a}\right) / 2$
    Calculating $\boldsymbol{b}[i, j] = \left(Q_{q_b} - Q_{q_a}\right) / \left(\boldsymbol{out}_{max} - Q_{q_b}\right)$
    Calculating $\boldsymbol{c}[i, j] = \left(Q_{q_b} + Q_{q_a}\right) / 2$
  end for
end for
return $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}$

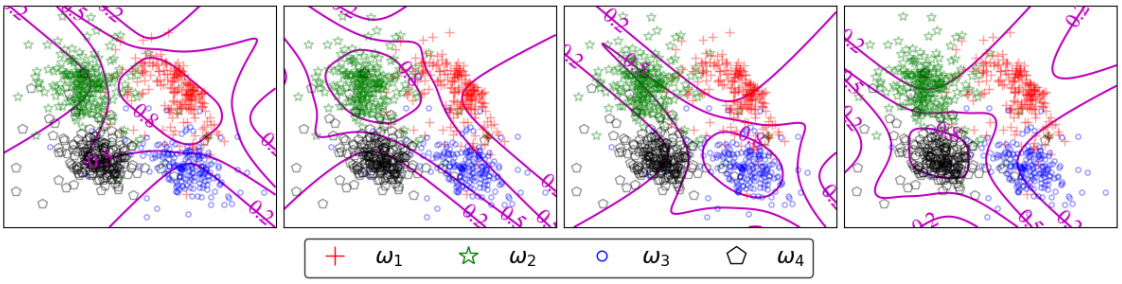
Based on the possibilities collected from all the testing samples, we can draw the possibility contours as shown in Fig. 6.9a. As it can be seen, the obtained BSFs generate closed-class-shaped boundaries for the corresponding classes. There is a distinct separation in that the possibility of corresponding class samples, i.e., In-Domain (ID), is above 0.8 and the remaining, i.e., IM and OOD, is below 0.5. A clear possibility boundary leading to an unambiguous belief boundary can facilitate the classification of ID, IM, and OOD samples.

Revealed by Figs. 6.8b, 6.9a, the first-class features a scattered rectangle shape and has overlaps with the remaining three classes. The overlap among learned features will lead to inaccurate boundaries and misclassified samples from other classes. Inspired by the feature distribution, we used the contrastive-center loss function to improve the discriminative power of the model. The distribution of learned features under the joint supervision of contrastive-center loss is shown in Fig. 6.8c. From the results obtained, there is an improvement that the learned feature is more concentrated and compacted compared to the model trained by cross-entropy loss function. By comparing the learned features and boundaries for pre-trained, i.e., Fig. 6.8b, 6.9a and fine-tuned, i.e., Fig. 6.8c, 6.9b. We can observe: (1) under the supervision of cross-entropy loss function learned features are separable, (2) the learned features are not discriminative enough, since they still show inner-class variations, and contrastive-center loss function outperforms.

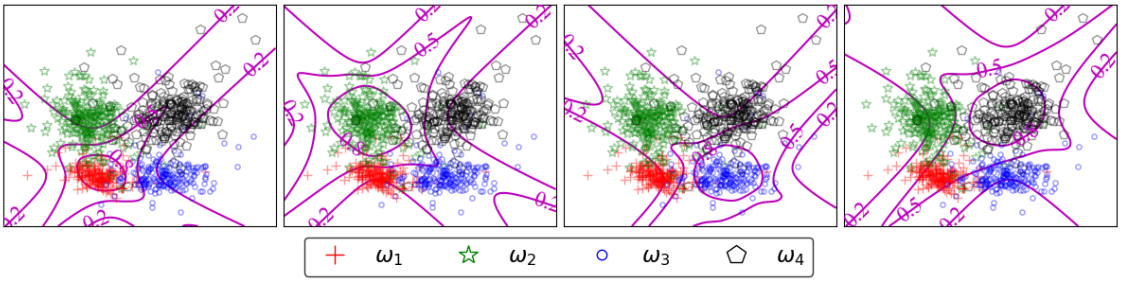
Another challenge is the quantity of predicted subsets increases exponentially with the class number, which could preclude the application of partial classification [TXD21]. We presented a strategy to give beliefs merely to the $K + 1$ main subsets, including the empty set and subsets whose cardinality range from 1 to K. The empty set represents the OOD class, which corresponds to the OOD samples. Then, we define our nested predicted subsets $\{\emptyset, \{\omega_i\}, \{\omega_i, \omega_j\}, \{\omega_i, \omega_j, \omega_k\}, \ldots, \Omega\}$. The subset $\{\omega_i\}$ represents the set of the most probable class, the subset $\{\omega_i, \omega_j\}$ represents the set of the top two probable classes, etc. Following the theory proposed in [DP82, AD08] that any possibility distribution is a plausibility function corresponding to a consonant m. Our possibility distribution π_n can be transformed to belief function m according to Eq. (6.6), the detailed

Algorithm 3 The transformation from the model output to the possibility

Require:
Class number K
Testing sample number N^{test}
Testing output $\boldsymbol{o}^{test}$
Training output $\boldsymbol{o}^{train}$
Training label $\boldsymbol{y}^{train}$
Ensure:
Testing possibility $\boldsymbol{\pi}^{test}$
Initializing $\boldsymbol{\pi}^{test} \in \mathbb{R}^{N^{test} \times K}$ equals 0
Calculating the BSF's parameters $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}$ based on Algorithm. 2 from $\boldsymbol{o}^{train}$ and $\boldsymbol{y}^{train}$.
for $i = 0; i < K; i++$ **do**
  Initializing $\boldsymbol{v} \in \mathbb{R}^{N^{test}}$ equals 0
  for $j = 0; j < K; j++$ **do**
    $\boldsymbol{v} += \frac{1}{1+|\frac{\boldsymbol{o}^{test}[:,j]-\boldsymbol{c}[i,j]}{\boldsymbol{a}[i,j]}|^{2\times \boldsymbol{b}[i,j]}}$
  end for
  $\boldsymbol{\pi}^{test}[:, i] = \boldsymbol{v}$
end for
$\boldsymbol{\pi}^{test} /= K$
return $\boldsymbol{\pi}^{test}$



(A) Model trained by the cross-entropy loss function.



(B) Model trained by the contrastive-center loss function.

FIGURE 6.9: The boundary (possibility contour) obtained by BSF for the four-class dataset.

calculation is presented in Fig. 6.10.

$$m(A_i) = \begin{cases} 1 - \pi_1 & \text{if } A_i = \emptyset, \\ \pi_j - \pi_{j+1} & \text{if } A_i = \{\omega_1, \cdots, \omega_j\}, j \in \{1, \cdots, K-1\}, \\ \pi_K & \text{if } A_i = \Omega, \\ 0 & \text{otherwise.} \end{cases} \tag{6.6}$$

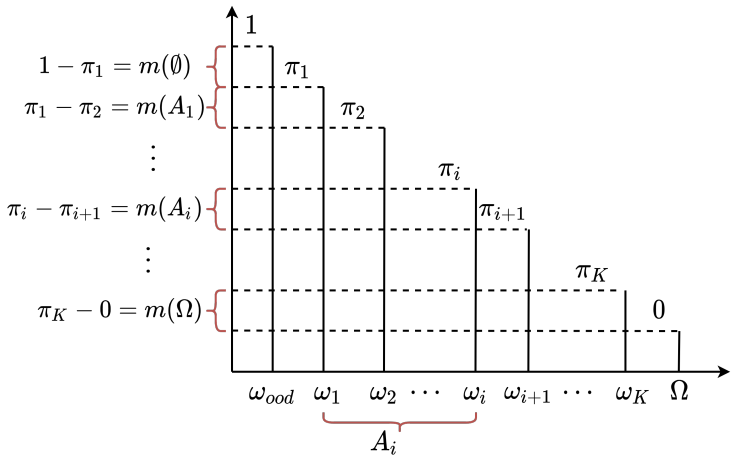


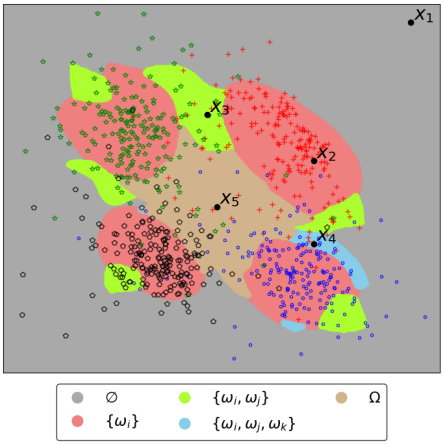
FIGURE 6.10: The belief calculation process in PCBS.

x_n	$m(\emptyset)$	$m(A_1)$	$m(A_2)$	$m(A_3)$	$m(\Omega)$	Prediction
x_1	**0.54**	0.16	0.08	0.06	0.16	$\emptyset$
x_2	0.03	**0.63**	0.02	0.16	0.16	$\{\omega_i\}$
x_3	0.17	0.11	**0.42**	0.12	0.18	$\{\omega_i,\omega_j\}$
x_4	0.13	0.27	0.10	**0.30**	0.19	$\{\omega_i,\omega_j,\omega_k\}$
x_5	0.37	0.03	0.01	0.03	**0.56**	Ω

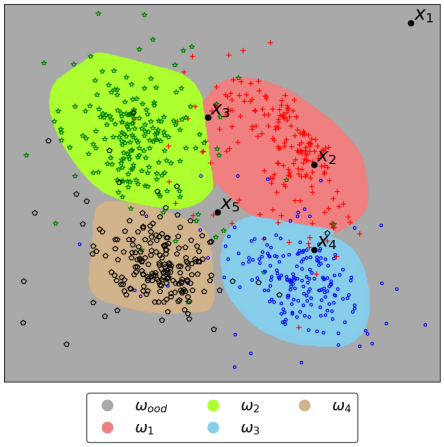
TABLE 6.4: Partial classification of five samples mentioned in Fig. 6.11a based on the MLP trained by the cross-entropy loss function.

Based on the belief calculated through Eq. (6.6), we can execute the partial classification by choosing the subset with the maximum belief as the prediction. The partial classification result for four-class dataset is shown in Figs. 6.11a and 6.11c. As can be seen from the figure, both the pre-trained and fine-tuned models are able to identify ID, IM, and OOD samples. Compared with the fine-tuned model, the pre-trained model has a slight weakness: (1) some of the samples belonging to the second, third, and fourth classes are classified into a set of two classes, and (2) some of the samples belonging to the third and fourth classes are grouped into a set of three classes. In contrast, the clear and concise partition is obtained by the fine-tuned model. To make it more clear, we chose five samples, i.e., $x_1 \sim x_5$ as shown in Figs. 6.11a and 6.11c predicted in different subsets to present the interest of the partial classification. Suppose, the maximum belief is $m(A_i)$, the PCBS method will generate the predicted set $\{\omega_1,\ldots,\omega_i\}$ corresponding to the top i maximum outputs. Tables. 6.4, 6.5 manifest the detailed values as well as prediction according to partial classification based on different training strategies. It is well shown how our method calculates and makes predictions. It also further supports the feasibility of PCBS method.

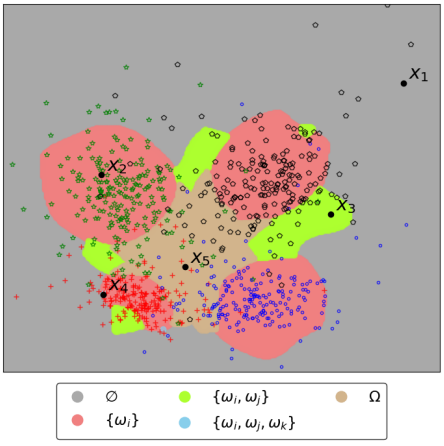
The interchangeability between partial classification and uncertainty estimation can be achieved with the help of pignistic transformation Eq. (6.7). In order to keep the OOD information available provided by the empty set of the evidential space, a new class ω_{ood} is added to the probabilistic universe. Traditional formula $BetP$ Eq. 2.7 is then transformed into Eq. 6.7. Consequently, $BetP$ is taking as entry the universe of probabilistic



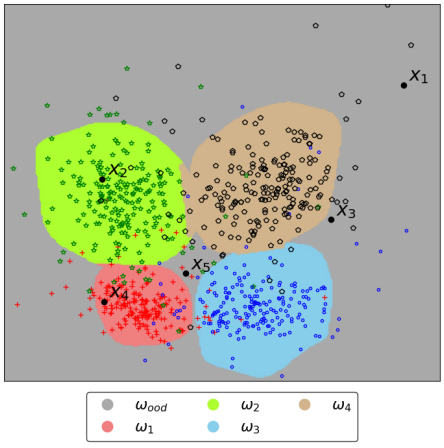
(A) The partial partition based on MLP trained by cross-entropy loss.



(B) The precise partition based on MLP trained by cross-entropy loss.



(C) The partial partition based on MLP trained by contrastive-center loss.



(D) The precise partition based on MLP trained by contrastive-center loss.

FIGURE 6.11: The partition is obtained from different strategies (partial and precise) based on the calculated beliefs. Five special cases, i.e., $x_1 \sim x_5$ are selected from different predicted subsets to show the detailed calculation process in Tables. 6.4, 6.5, 6.6, and 6.7.

x_n	$m(\emptyset)$	$m(A_1)$	$m(A_2)$	$m(A_3)$	$m(\Omega)$	Prediction
x_1	**0.56**	0.05	0.07	0.27	0.05	$\emptyset$
x_2	0.02	**0.62**	0.14	0.07	0.15	$\{\omega_i\}$
x_3	0.39	0.04	**0.46**	0.01	0.10	$\{\omega_i, \omega_j\}$
x_4	0.27	0.27	0.02	**0.28**	0.16	$\{\omega_i, \omega_j, \omega_k\}$
x_5	0.36	0.05	0.03	0.01	**0.55**	Ω

TABLE 6.5: Partial classification of five samples mentioned in Fig. 6.11c based on the MLP trained by the contrastive-center loss function.

theory, i.e., $\Omega^{BetP} = \{\omega_{ood}, \omega_1, \ldots, \omega_i, \ldots, \omega_K\}$. Once the belief is transformed into probability, we can fulfill uncertainty estimation easily by choosing the class with the maximum probability.

$$BetP(\omega) = \begin{cases} m(\emptyset) & \text{if } \omega = \omega_{ood}, \\ \sum_{\omega \in A} \frac{m(A)}{|A|} & \text{if } \omega \neq \omega_{ood}, \omega \in \Omega. \end{cases} \tag{6.7}$$

As mentioned, $BetP(\omega_{ood}) = m(\emptyset)$, the sum of the probability equals one, the mathematical proof follows.

$$\begin{aligned}\sum_{\omega\in\Omega^{BetP}} BetP(\omega) &= BetP(\omega_{ood}) + \sum_{i=1}^{K} BetP(\omega_i) \\ &= m(\emptyset) + \sum_{i=1}^{K} \frac{m(A_i)}{|A_i|} + \cdots + \sum_{i=K}^{K} \frac{m(A_i)}{|A_i|} \\ &= m(\emptyset) + 1 \times \frac{m(A_1)}{|A_1|} + \cdots + K \times \frac{m(A_K)}{|A_K|} \\ &= m(\emptyset) + m(A_1) + \cdots + m(A_K) = \mathbf{1}.\end{aligned} \tag{6.8}$$

In a similar way, the risk relative to uncertainty estimation is minimized by choosing the class with maximum probability. As can be seen from Fig. 6.11, both the pre-trained model and the fine-tuned model are able to identify the testing sample accurately. In addition, PCBS method can classify both the OOD and IM samples into OOD class $\emptyset$. Intuitively, the fine-tuned model outperforms the pre-trained model. The detailed calculation based on the same five samples as showed in Figs. 6.11b and 6.11d are shown in Tables. 6.6 and 6.7. From the tables, we can see how the belief converts into probability. Consider the second sample x_2, for example, fed it into the fine-tuned model obtains with belief $\{0.02, 0.62, 0.14, 0.07, 0.15\}$ whose probability for the second class after transformation is $0.75 = \frac{0.62}{1} + \frac{0.14}{2} + \frac{0.07}{3} + \frac{0.15}{4}$.

x_n	$BetP(\omega_{ood})$	$BetP(\omega_1)$	$BetP(\omega_2)$	$BetP(\omega_3)$	$BetP(\omega_4)$	Prediction
x_1	**0.54**	0.26	0.10	0.06	0.04	ω_{ood}
x_2	0.03	**0.73**	0.09	0.10	0.05	ω_1
x_3	0.17	**0.41**	0.30	0.09	0.03	ω_1
x_4	0.13	0.20	0.05	**0.47**	0.15	ω_3
x_5	**0.37**	0.18	0.15	0.16	0.14	ω_{ood}

TABLE 6.6: Uncertainty estimation of five samples mentioned in Fig. 6.11b based on the MLP trained by the cross-entropy loss.

x_n	$BetP(\omega_{ood})$	$BetP(\omega_1)$	$BetP(\omega_2)$	$BetP(\omega_3)$	$BetP(\omega_4)$	Prediction
x_1	**0.56**	0.01	0.10	0.14	0.19	ω_{ood}
x_2	0.02	0.06	**0.75**	0.04	0.13	ω_2
x_3	**0.39**	0.03	0.03	0.26	0.29	ω_{ood}
x_4	0.27	**0.42**	0.14	0.13	0.04	ω_1
x_5	**0.36**	0.20	0.14	0.16	0.14	ω_{ood}

TABLE 6.7: Uncertainty estimation of five samples mentioned in Fig. 6.11d based on the MLP trained by the contrastive-center loss.

6.2.3 Experiments

This section reports some experimental results that show various aspects of the proposed method. It is subdivided into three parts:

1. For choosing the optimum BSFs parameters for the pre-trained model as well as the fine-tuned model, the main parameters q_a and q_b are monitored, the observation of the variability of the results over the 12 UCI datasets is discussed in Section 6.2.4.

2. Take different reject rates, present and discuss the partial classification performances of different methods, i.e., partial classifier in the belief function framework (hereafter called PCBF) [MD21], pre-trained PCBS, and fine-tuned PCBS) on 12 UCI datasets in Section 6.2.5. We also conduct an abundant analysis that leads to an improved understanding of our approach.

3. In Section 6.2.6, we describe experimental steps of how PCBS method can be used as the uncertainty estimation. And show the effectiveness of PCBS method on a wide range of BSF evaluation benchmarks, i.e., softmax score, pre-trained energy score [LWOL20], fine-tuned energy score [LWOL20], pre-trained PCBS, and fine-tuned PCBS and different datasets, i.e., Cifar10 [Kri12], Cifar100 [Kri12], Isun [YZS+15], Places365 [ZKL+17], Textures [CMK+14], Svhn [NWC+11], Lsun Crop [YZS+15], and Lsun Resize [YZS+15].

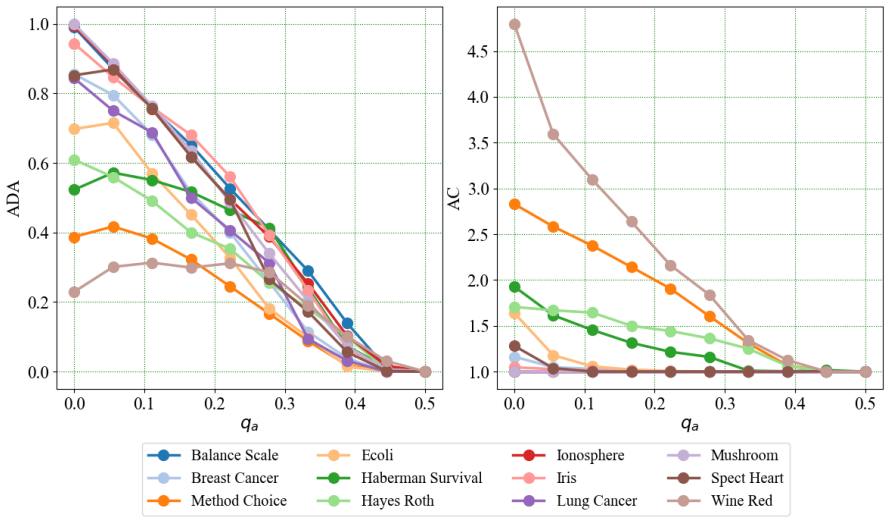
6.2.4 Parameter choosing

As we can see from Algorithm. 2 line 6, the BSF is calculated based on two principal parameters q_a and q_b. In reality, $q_a + q_b \leq 1$, but in this thesis, the q_a and q_b constraint to $q_a + q_b = 1$, and $q_a \leq q_b$. Consequently, we can calculate $q_b = 1 - q_a$ from q_a. From Algorithm. 2 lines 7, 8, and 9, we can see that a, b and c all greater than zero. Consequently, for both pre-trained and fine-tuned model, set different q_a from 0 to 0.5 (not included), and test with various datasets to get the optimum settings for these two hyperparameters. As displayed in Fig. 6.12, the left parts of each subfigure show the ADA, while the right parts show the AC. The smaller the q_a (as displayed in the Algorithm 2 line 7, $a = (q_b - q_a)/2 = (1 - 2q_a)/2$) the wider the BSF function is, resulting in containing more samples. Mapping more certain samples' possibility to one leads to larger ADA and smaller AC values. But there are some datasets, e.g. Wine Red, where the small initial ADA, i.e., 0.2 accompanied by a large initial AC, i.e., 5.0. This is because (1) the dataset contains too many confusing samples, which is due to the characteristics inherently, and (2) the BSF function maps the confusing sample's possibility to one leading to the predicted set having a large cardinality. In contrast, the Mushroom dataset occupies a large initial ADA, i.e., 1.0, and small initial AC, i.e., 1.0. This indicates that (1) the dataset is clear, inherently, and (2) the learned feature is more concentrated, the samples are well classified. As q_a increases, BSF maps fewer sample possibilities to one, which in turn increases the confidence of precise classification. The reflection in the figure is the AC and ADA values gradually decrease as q_a increases. The user can choose different parameter settings depending on the needs of the different scenarios. Here, to compare with other methods, we set q_a to 0.05 to achieve the best ADA and avoid overconfidence.

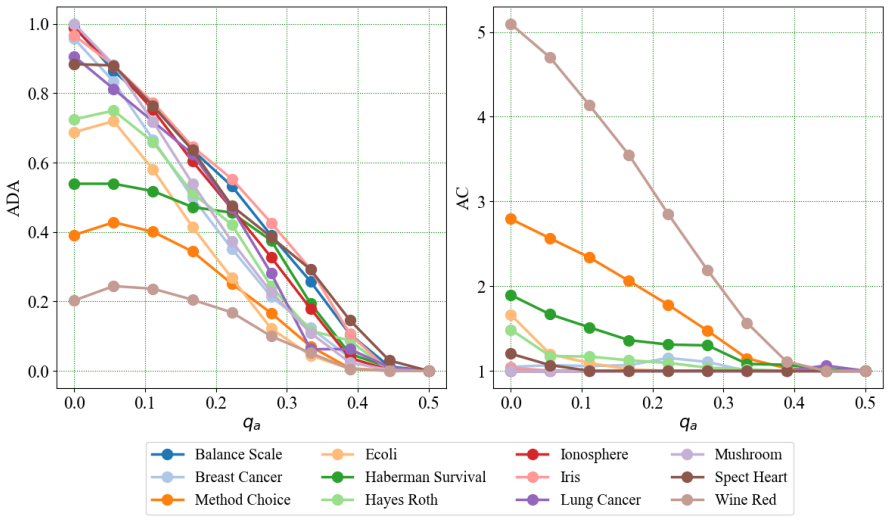
6.2.5 Partial classification

6.2.5.1 Experiment protocol

For partial classification, we selected 12 UCI classification datasets for our experiments, as summarized in Table. 6.8. The UCI dataset is not the real image dataset, but the attribute of each UCI dataset can be regarded as the output extracted by the Neural Network (NN)s. Consequently, the usage of UCI dataset is hemogerous with the thesis subject. The classical MLP contains three hidden layers and two fully connected layers is



(A) MLP trained by the cross-entropy loss function.



(B) MLP trained by the contrastive-center loss function.

FIGURE 6.12: The ADA and AC curves were plotted as a function of q_a based on 12 UCI datasets. In order to achieve the best ADA and avoid overconfidence, we set q_a to 0.05.

Dataset	# Sample	# Attribute	# Used class
Balance Scale	625	4	3
Breast Cancer	105	9	4
Method Choice	1473	9	3
Ecoli	336	7	8
Haberman Survival	306	3	2
Hayes Roth	160	4	3
Ionosphere	351	33	2
Iris	150	4	3
Lung Cancer	32	56	3
Mushroom	8124	21	2
Spect Heart	267	22	2
Wine Red	1599	11	6

TABLE 6.8: An overview of datasets involved in PCBS for partial classification.

used as the target model. For training, we use the holdout strategy, that is to say, 70% of samples are used for training, and 30% of samples used for testing. The cross-entropy loss function is used for pre-trained model and contrastive-center loss function is adopted for fine-tuned model. For contrastive-center loss function, we adopt the optimum parameter configurations, i.e., $\delta = 1$, and $\lambda = 1$ as the original paper [QS17]. For method comparison, since the PCBS can be used under both pre-trained and fine-tuned contexts, we choose PCBF [MD21] and the same experiment settings for comparison. For the NN, we abandon the previous used NNs, e.g., LeNet, GoogLeNet. The motivation behind this is the superior NN can bring better performance.

6.2.5.2 Performance

By choosing the predicted set with the maximum belief, the PCBS method can perform partial classification. Naturally, the bigger the predicted set cardinality indicates the more confusing sample is. Consequently, to show the efficiency and capacity of the PCBS method of performing partial classification and reducing the classification risk, the most confusing samples are rejected according to different reject rates [NZH09].

On the one hand, to manifest the efficiency of different methods against a small reject rate, the reject rate is set to equal 0.1 and receive Fig. 6.13. For PCBS method, the fine-tuned outperforms the pre-trained. In addition, except for the Haberman Survival and Wine Red datasets, both pre-trained and fine-tuned PCBS method is better than the existing PCBF method. This is due to the inherent characteristics of the dataset. As we can see from Fig. 6.14, the overlap among samples, both pre-trained and fine-tuned, is severe that PCBS method is unable to obtain clear closed-class-shaped boundaries. Thus, to be on the cautious side, PCBS method assigns the samples into predicted sets containing more than one class, leading to a decrease in ADA and an increase in AC.

On the other hand, the average values for different reject rates based on 12 UCI datasets are demonstrated in Fig. 6.15. When the confusing samples are rejected, a more confident prediction can be made, so that the AC decreases from large to small until it reaches one. At the same time, the ADA increases. Ideally, the ADA should reach one when all the confusing samples are rejected. But, due to the overlap of the learned features and the models' bias, some certain samples are misclassified as confusing samples and are rejected. As a result, ADA can not reach one. This is particularly obvious for the PCBF method. For our method, the fine-tuned is slightly better than the pre-trained.

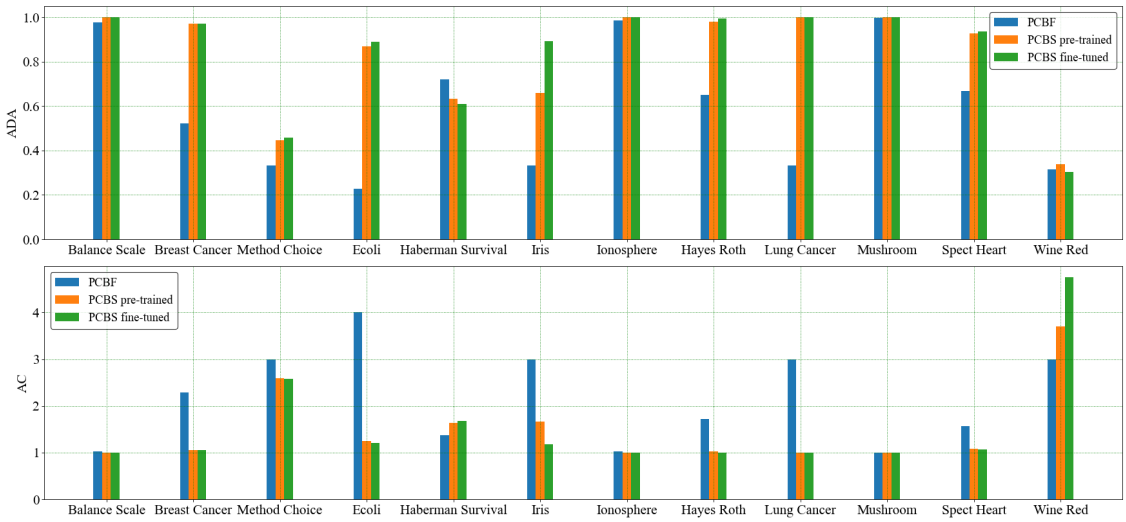
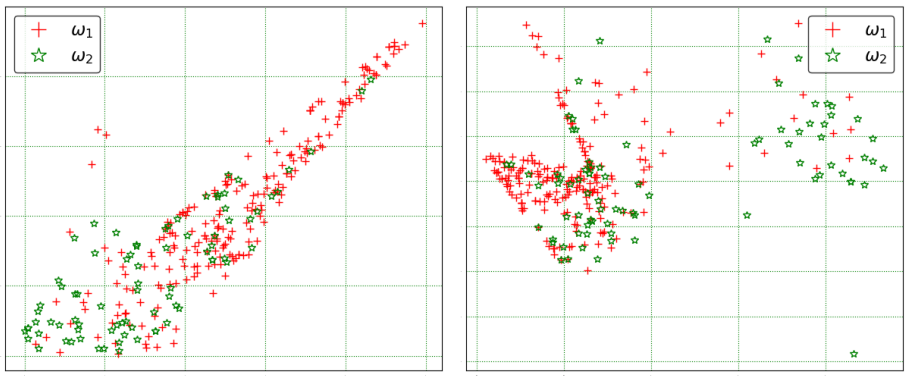
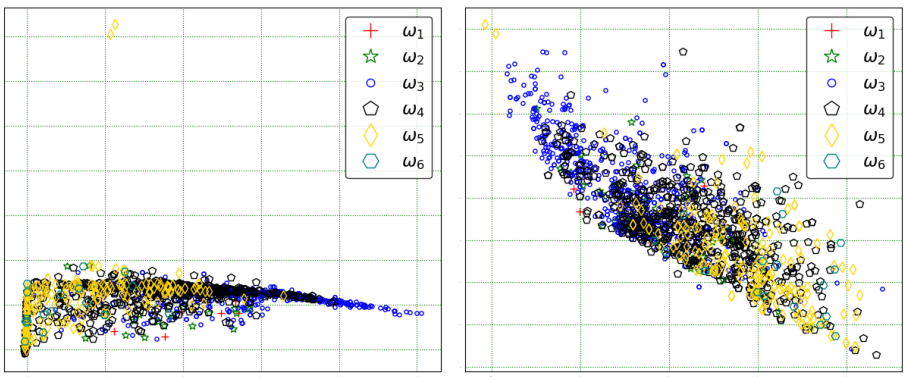


FIGURE 6.13: The comparison among different methods when reject rate equals 0.1.



(A) Haberman Survival dataset.



(B) Wine Red dataset.

FIGURE 6.14: Learned feature distributions of two UCI datasets based on pre-trained MLP (left) and fine-tuned MLP (right).

Compared to the insensitivity of the PC method, PCBS method performs better and is more feasible.

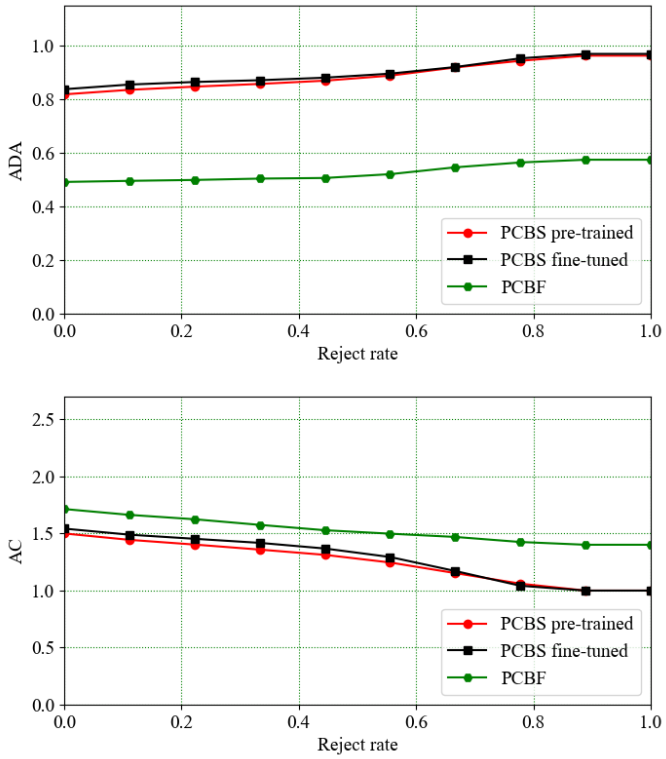


FIGURE 6.15: The ADA and AC values are based on different reject rates for 12 UCI datasets. First, calculate ADA and AC for 12 UCI datasets based on different reject rates. Then, for each single reject rate calculate the average values based on 12 UCI datasets.

Type	Name	# Used class	# Training sample	# Testing sample
ID	Cifar10	10	50000	10000
	Cifar100	100	50000	10000
OOD	Isun	-	-	2000
	Places365	-	-	2000
	Textures	-	-	2000
	Svhn	-	-	2000
	Lsun Crop	-	-	2000
	Lsun Resize	-	-	2000

TABLE 6.9: An overview of datasets involved in PCBS for uncertainty estimation.

6.2.6 Uncertainty estimation

6.2.6.1 Experiment protocol

For uncertainty estimation, we use Cifar10 [Kri12], and Cifar100 [Kri12] datasets as ID datasets. For the OOD datasets, we use six prevalent benchmarks, Isun [YZS+15], Places365 [ZKL+17], Textures [CMK+14], Svhn [NWC+11], Lsun Crop [YZS+15], and Lsun Resize [YZS+15]. Table 6.9 reports the two types of datasets selected for the experiments. A classical WideResNet which contains 40 layers is chosen as the target model. For method comparison, we choose energy score [LWOL20] and the same experiment configurations.

$\mathcal{D}_{in}^{test}$	fine-tuned?	$\mathcal{D}_{ood}^{test}$	FPR95↓	AUROC ↑	AUPR↑
			PCBS score / Energy score		
WideResNet Cifar10	No	iSUN	**19.85** / 32.10	**97.13** / 92.92	**99.48** / 98.38
		Places365	**35.35** / 39.30	**96.08** / 90.53	**99.29** / 97.47
		Texture	**0.00** / 52.95	**98.41** / 85.43	**99.71** / 95.68
		SVHN	**0.00** / 33.85	**98.98** / 91.30	**99.82** / 97.74
		LSUN-Crop	80.65 / **8.40**	89.77 / **98.29**	98.09 / **99.63**
		LSUN-Resize	**18.25** / 26.25	**97.16** / 94.54	**99.48** / 98.73
		average	**25.68** / 32.14	**96.26** / 92.17	**99.31** / 97.94
			PCBS score / Energy score		
WideResNet Cifar10	Yes	iSUN	18.25 / **1.40**	97.24 / **99.32**	99.50 / **99.87**
		Places365	36.75 / **8.20**	96.03 / **97.77**	99.28 / **99.40**
		Texture	**0.00** / 5.80	98.49 / **98.57**	**99.72** / 99.69
		SVHN	**0.00** / 1.55	98.69 / **99.25**	99.76 / **99.85**
		LSUN-Crop	78.05 / **1.55**	90.21 / **99.25**	98.16 / **99.85**
		LSUN-Resize	19.50 / **1.40**	97.15 / **99.32**	99.48 / **99.86**
		average	25.43 / **3.32**	96.30 / **98.92**	99.32 / **99.75**

TABLE 6.10: The uncertainty estimation performance comparison between PCBS and energy score based on Cifar10 dataset. All values are percentages, ↑ indicates larger values are better, ↓ indicates smaller values are better, and the best value is bold. The PCBS method outperforms, except the fine-tuned model based on the Cifar10 dataset that is due to our method does not need the BSF dataset for fine-tuning. The PCBS method is superior when the BSF dataset is unavailable.

6.2.6.2 Performance

We begin by assessing the improvement of PCBS method over the energy score [LWOL20]. Tables. 6.10 and 6.11 contain a detailed comparison based on the Cifar10 and Cifar100 datasets. We show that using PCBS method based on pre-trained model reduces the average False Positive Rate 95 (FPR95) compared to the pre-trained energy score based on the Cifar10 dataset. However, for the fine-tuned model, PCBS method is not as good as the energy score on the Cifar10 dataset.

The reason is, as we can see from Table. 6.12, the information required varies from method to method. For softmax, it only requires the testing output, so it performs the worst. For the pre-trained model, the PCBS method requires the additional training output than the energy score, and it can be seen that PCBS method outperforms both on Cifar10 and Cifar100. For fine-tuned model, in contrast to the energy score, which requires information from the BSF dataset, PCBS method only requires information from training output. In some cases where BSF datasets are unavailable, our method is more advantageous. In Table. 6.13, we presented the average values for different methods based on six OOD datasets. The PCBS method achieves comparable and better performance than the energy score.

6.2.7 Other decision making strategies for partial classification

In addition to the mentioned heuristic decision-making strategy, i.e., choosing the subset with maximum belief as the prediction. We also proposed several decision-making strategies based on minimum classification loss. We will start with the precise classification and gradually transition to partial classification.

$\mathcal{D}_{in}^{test}$	fine-tuned?	$\mathcal{D}_{ood}^{test}$	FPR95↓	AUROC ↑	AUPR↑
			PCBS score / Energy score		
WideResNet Cifar100	No	iSUN	**0.00** / 80.60	**98.39** / 79.05	**99.71** / 94.86
		Places365	**55.90** / 80.25	**93.64** / 75.94	**98.81** / 93.63
		Texture	**22.05** / 79.85	**97.07** / 75.48	**99.46** / 93.31
		SVHN	**0.00** / 85.45	**99.02** / 74.51	**99.82** / 93.79
		LSUN-Crop	92.70 / **35.00**	82.67 / **93.57**	96.65 / **98.63**
		LSUN-Resize	**0.00** / 80.00	**98.60** / 78.97	**99.75** / 94.94
		average	**28.44** / 73.53	**94.90** / 79.59	**99.03** / 94.86
			PCBS score / Energy score		
WideResNet Cifar100	Yes	iSUN	**0.00** / 69.10	**98.30** / 78.19	**99.69** / 94.11
		Places365	54.10 / **50.20**	**93.75** / 89.78	**98.83** / 97.69
		Texture	**17.90** / 52.95	**97.21** / 88.08	**99.49** / 97.14
		SVHN	**0.00** / 20.50	**98.31** / 96.52	**99.69** / 99.29
		LSUN-Crop	92.20 / **15.00**	83.85 / **97.26**	96.90 / **99.42**
		LSUN-Resize	**0.00** / 67.85	**98.57** / 80.21	**99.74** / 94.79
		average	**27.37** / 45.93	**95.00** / 88.34	**99.06** / 97.07

TABLE 6.11: The uncertainty estimation performance comparison between PCBS and energy score based on Cifar100 dataset. All values are percentages, ↑ indicates larger values are better, ↓ indicates smaller values are better, and the best value is bold. The PCBS method outperforms in regards to both pre-trained and fine-tuned model.

Method	Information		
	Training output	Testing output	OOD dataset
Softmax	-	✓	-
Energy pre-trained	-	✓	-
PCBS pre-trained	✓	✓	-
Energy fine-tuned	-	✓	✓
PCBS fine-tuned	✓	✓	-
PCMO	-	✓	-

TABLE 6.12: Information needed of different methods. ✓ indicates needed, and - indicates no needed.

6.2.7.1 Precise classification

For precise classification, its goal is to choose the label that can minimize the classification risk, i.e., $\hat{y} = \underset{y_i}{argmin}\, R(y_i \mid \boldsymbol{x})$. The $R(y_i \mid \boldsymbol{x})$ represents the risk of classify sample $\boldsymbol{x}$ as y_i.

$$R(y_i \mid \boldsymbol{x}) = \sum_{j=1}^{K} \ell(y_i \mid y_j) p(y_j \mid \boldsymbol{x}) = 1 - p(y_i \mid \boldsymbol{x}), \tag{6.9}$$

where

$$\ell(y_i \mid y_j) = \begin{cases} 0, & y_i = y_j, \\ 1, & y_i \neq y_j. \end{cases} \tag{6.10}$$

To minimize the classification risk, the class with maximum probability is chosen as the prediction.

$$\hat{y} = \underset{y_i}{argmin}\, R(y_i \mid \boldsymbol{x}) = \underset{y_i}{argmin}\, 1 - p(y_i \mid \boldsymbol{x}) = \underset{y_i}{argmax}\, p(y_i \mid \boldsymbol{x}). \tag{6.11}$$

$\mathcal{D}_{in}^{test}$	method	FPR95↓	AUROC ↑	AUPR↑
Cifar10	PCBS pre-trained	25.68	96.26	99.31
	PCBS fine-tuned	25.43	96.30	99.32
	Energy pre-trained	32.14	92.17	97.94
	Energy fine-tuned	3.32	98.92	99.75
	Softmax	50.16	91.26	98.04
Cifar100	PCBS pre-trained	28.44	94.90	99.03
	PCBS fine-tuned	27.37	95.00	99.06
	Energy pre-trained	73.53	79.59	94.86
	Energy fine-tuned	45.93	88.34	97.07
	Softmax	80.25	75.62	93.95

TABLE 6.13: The comparison among different uncertainty estimation methods. ↑ indicates larger values are better, and ↓ indicates smaller values are better. All values are percentages and are averaged over the six BSF test datasets described in Table. 6.9.

6.2.7.2 Partial classification

Same to the precise classification, the goal of partial classification is to make a decision by choosing the subset that can minimize the classification risk. As can be seen from the Eq. (6.9), the problem to be solved is the determination of the loss function $\ell(\cdot)$ and the belief function $m(\cdot)$.

On the one hand, we can build a new loss function $\ell(A_i \mid A_j)$ inspired by the IoU loss. Since we have got the belief function, the classification risk can be derived easily.

$$\hat{A} = \underset{A_i}{argmin}\, R(A_i \mid \boldsymbol{x}), \tag{6.12}$$

$$R(A_i \mid \boldsymbol{x}) = \sum_{A_j \subseteq \Omega'} \ell(A_i \mid A_j) m(A_j \mid \boldsymbol{x}), \tag{6.13}$$

$$\ell(A_i \mid A_j) = 1 - \frac{\mid A_j \cap A_i \mid}{\mid A_j \cup A_i \mid}. \tag{6.14}$$

On the other hand, we can use $\ell(A_i \mid y_j)$ approximate $\ell(A_i \mid A_j)$. Now, the new problem is to construct the $\ell(A_i \mid y_j)$ loss. Fortunately, several similar loss functions have been proposed, e.g., discount accuracy, utility function or F_β score.

$$\ell(A_i \mid A_j) = \min_{y_k \in A_j} \ell(A_i \mid y_k), \tag{6.15}$$

$$\ell(A_i \mid A_j) = \max_{y_k \in A_j} \ell(A_i \mid y_k). \tag{6.16}$$

We can use discount accuracy, utility function or F_β score ($K \times K$) to extend the original "0/1" loss to extended loss ($2^K \times K$).

$$\ell(A_i \mid y_k)^{discount} = \begin{cases} 1 - \frac{1}{A_i}, & y_k \in A_i, \\ 0, & otherwise. \end{cases} \tag{6.17}$$

$$\ell(A_i \mid y_k)^{utility} = \begin{cases} 1 - g(\frac{1}{A_i}),\ g(x) = -0.6x^2 + 1.6x, & y_k \in A_i, \\ 0, & otherwise. \end{cases} \tag{6.18}$$

$$\ell(A_i \mid y_k)^{F_\beta} = \frac{(1+\beta^2)PR}{\beta^2 P + R},\ R = \mathbb{I}(y_k \in A_i),\ P = \frac{R}{\mid A_i \mid}. \tag{6.19}$$

Based on Eq.6.16, it is easy to get the $\ell(A_i \mid A_j)$. Furthermore, partial classification can be made use Eq. (6.9).

6.2.8 Uncertainty estimation comparison between SLUE and PCBS

Since both SLUE and PCBS can fulfill uncertainty estimation methods, we compared these performances dealing with confusing samples. First, the WideResNet is trained using Cifar10 and Cifar100, respectively. Then six datasets are used as OOD dataset and FPR95, Area Under the Receiver Operating Characteristic (AUROC), and Area Under the Precision Recall Curve (AUPRC) are computed as shown in the Table 6.14. From the results, we can see that the SLUE method is inferior to the PCBS method for the Cifar10 dataset. However, for Cifar100, SLUE achieved better results in terms of AUROC and AUPRC. The only difference between the two datasets is the different class numbers, thus it can be inferred that for a small class number, the PCBS can achieve a more accurate transformation from model output to belief by the BSF. However, when the classes number is large, it is better to compute uncertainty directly through a simple formula. This is because the direct calculation reduces the information lost in the transformation from model output to belief.

$\mathcal{D}_{in}^{test}$	fine-tuned?	$\mathcal{D}_{ood}^{test}$	FPR95↓	AUROC ↑	AUPR↑
			PCBS score / SLUE score		
WideResNet Cifar10	Yes	LSUN-Crop	78.05 / **22.86**	**90.21** / 99.04	**98.16** / 95.52
		Svhn	**0.00** / 34.65	**98.69** / 97.24	**99.76** / 87.06
		LSUN-Resize	**19.50** / 47.46	**97.15** / 96.14	**99.48** / 81.23
		Isun	**18.25** / 37.52	**97.24** / 95.58	**99.50** / 78.87
		Texture	**0.00** / 76.64	**98.49** / 94.07	**99.72** / 74.92
		Places365	**36.75** / 51.64	**96.03** / 94.71	**99.28** / 76.20
		average	**25.43** / 45.13	**96.30** / 96.13	**99.32** / 82.30
			PCBS score / SLUE score		
WideResNet Cifar100	Yes	LSUN-Crop	**92.20** / 99.74	83.85 / **99.95**	96.90 / **99.96**
		Svhn	**0.00** / 96.31	98.31 / **99.91**	99.69 / **99.92**
		LSUN-Resize	**0.00 / 0.00**	98.57 / **100.00**	99.74 / **100.00**
		Isun	**0.00** / 99.74	98.30 / **99.85**	99.69 / **99.88**
		Texture	**17.90** / 99.90	97.21 / **97.29**	**99.49** / 97.64
		Places365	54.10 / **0.00**	93.75 / **100.00**	98.83 / **100.00**
		average	**27.37** / 65.95	95.00 / **99.50**	99.06 / **99.57**

TABLE 6.14: The uncertainty estimation performance comparison between PCBS and SLUE. All values are percentages, ↑ indicates larger values are better, ↓ indicates smaller values are better, and the best value is bold.

6.2.9 Partial classification comparison between PCMO and PCBS

Same as uncertainty estimation, we compared the partial classification performance of PCBS and PCMO. The MLP is trained based on the UCI datasets, and ADA and AC are calculated based on the model output, as shown in Fig. 6.16, we calculated the average ADA and AC based on the 12 UCI datasets. As can be seen, the PCMO is inferior to the PCBS. It is reasonable because that PCMO merely uses information from the testing output as shown in Table 6.12. In addition, the ADA (AC) of both methods gradually increases (decreases) as the reject rate increases.

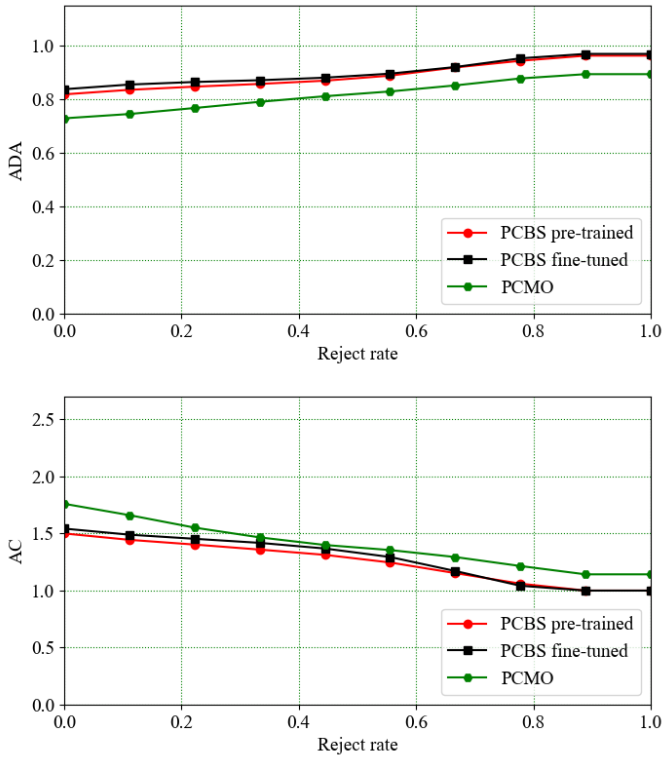


FIGURE 6.16: The comparison between PCBS and PCMO of ADA and AC values based on different reject rates for 12 UCI datasets. First, calculate ADA and AC for 12 UCI datasets based on different reject rates. Then, for each single reject rate calculate the average values based on 12 UCI datasets.

6.2.10 Conclusions

In order to deal with a variety of confusing samples, a new partial classification named PCBS is proposed. The time-consuming associates to the BSF calculation, thus, the time complexity related to the class number is $O(K^2)$. It can not only pay attention to the partial classification but also can be transformed into uncertainty estimation by the pignistic transformation. At first, we theoretically and empirically demonstrate the PCBS method in the view of the four-class dataset which comprises mainly four modules: BSFs generation, output to possibility transformation, possibility to belief transformation, and belief to probability transformation. Meanwhile, we fine-tuned PCBS method, by employing the contrastive-center loss function, and the fine-tuned model outperforms the pre-trained model. Then, we verified PCBS method with different state-of-art methods based on several datasets. There are two branches, for partial classification, we compare with the PCBF method based on the ADA and AC for different reject rates. For precise classification with reject option, we compared with the energy score, on behave of the FPR95, AUPRC, and AUROC. Moreover, we demonstrate how to choose the optimum BSF parameters, in order to get the best ADA, we set q_a equals 0.05. From the production, we can tell that PCBS method performs better than state-of-the-art methods, as it can provide high confidence to a certain sample, as well as a high uncertainty to a confusing sample. The PCBS method proved effective in increasing the reliability of classifiers and ultimately reducing the classification risk.

6.3 Conclusions

The two methods proposed in this chapter, i.e., PCMO and PCBS, are both fulfilled based on the model output. Both two methods use possibility as a bridge to achieve the transformation from model output to belief. Then, the partial classification is fulfilled by selecting the subset with the maximum belief. Both methods are able to achieve partial classification without changing the model architecture or constructing a new loss function. Meanwhile, compared to PCMO, PCBS uses BSF to achieve the transformation from output to possibility. It enables the PCBS to provide belief for both the empty set and the entire set to make more cautious predictions. For experiments, different models are trained based on different datasets. The results for different measurements showed the superiority of the proposed two partial classification methods.

Chapter 7

Conclusions

7.1 Summary of works

In this thesis, we have studied two categories of methods for dealing with confusing samples in the classification based model, i.e., uncertainty estimation and partial classification. We have proposed three methods, i.e., SLUE [XCA21], PCMO [XAC21], and PCBS, to improve both the accuracy and robustness when dealing with confusing samples. A summary of works is listed below:

Uncertainty estimation: The plain strategy to deal with the confusing sample is to detect them based on a scalar value, representing the uncertainty. Then the sample can be classified into the Out-of-Domain (OOD) subset by comparing to a predefined threshold. The Evidential Deep Learning (EDL) [SKK18] is a method based on the Subjective Logic (SL), which uses the Dirichlet distribution to connect the model output and the uncertainty derivation. One of the advantages of SL is that it can use base rate to fulfill better evidence allocation. However, the EDL merely use the default base rate settings, overlooking the import role played by the base rate. Consequently, we propose to extend the existing EDL method by taking base rates into account. At first, we provide four potential base rate initial strategies and evaluate their impact during the training process. At the same time, we also study the hyperparameter $\mathcal{C}$. Since classical accuracy only focuses on the right classified sample we propose new criteria named extended accuracy, which takes the right classified sample and rejected wrong classified sample into account. We also verified the SLUE method uncertainty performance through OOD and adversarial datasets. The most obvious finding is that guided by base rates, the SLUE method works better not only in terms of extended accuracy but also on the uncertainty estimation performance. As it can reject confusing samples, this approach will prove useful in improving model robustness.

Partial classification: In order to reduce the risk by classifying confusing samples into OOD class only based on a scalar value. We propose to use partial classification to make cautious predictions by classifying confusing samples into class subsets. Meanwhile, making the partial classification feasible and packageable as an auxiliary module is another major motivation. We present a new partial classification method named PCMO, which is fulfilled based on pre-trained Convolutional Neural Network (CNN) based model outputs. Based on different datasets and Neural Network (NN)s, we showed that a sample far from the training dataset can provide high outputs and lead to high probabilities for several classes. We adopted possibility as the bridge fulfilling the transformation from model outputs to beliefs for the predicted sets. This process includes two sub-processes, the output to possibility transformation and the possibility to belief transformation. We adopt the log function and the max-min normalization fulfilled the first sub-process, and we also tested different normalization strategies. The obtained possibility is converted into belief for nested subsets whose cardinality range from 1 to $K-1$ under the Dempster-Shafer Theory (DST). The PCMO method has been verified on four

prevalent datasets, four classical CNN-based models, and compared with three existing methods. The performance showed that the PCMO method makes it possible to improve classification accuracy and to make cautious decisions by assigning a sample to a class subset.

There are two weaknesses in the PCMO method. The belief calculation process in PCMO involves many steps, such as computing a temporary variable and performing normalization. Those steps might lead to a loss of information. At the same time, the PCMO method is not able to provide belief for the empty set or the entire set. Consequently, a new partial classification named PCBS is proposed. First, the PCBS method can generate the Bell Shaped Function (BSF)s based on the training output. Second, the testing output is converted into the possibility under the obtained BSFs. The advantage of using BSF is that it can generate a closed-class-shaped boundary for each class and make it possible to assign beliefs to both the empty set and the entire set. Finally, the possibility is transformed into the belief to perform partial classification. Additionally, with the help of pignistic transformation, the calculated belief can be converted into probability to achieve the uncertainty estimation. This feature enhances the interchangeability between the partial classification and the uncertainty estimation. Moreover, the contrastive-center loss function is employed to efficiently ensure inner-class compactness and inter-class separability to have a better classification. Experiments with 12 UCI datasets and eight prevalent datasets based on five different criteria show the excellent performance of PCBS method as compared to the state-of-the-art methods. It can be concluded that PCBS method makes it possible to improve models' robustness by either rejecting confusing samples or assigning confusing samples to class subsets.

Publications: Based on the works during this thesis, we have published two papers at the International conferences and submitted one to an international journal.

1. Jiarui Xie, Thierry Chateau, and Violaine Antoine. A subjective-logic-based model uncertainty estimation mechanism for out-of-domain detection. In International Joint Conference on Neural Networks, pages 1–6, 2021. *Accepted.*

2. Jiarui Xie, Violaine Antoine, and Thierry Chateau. PCMO: Partial classification from CNN-based model outputs. In Neural Information Processing, pages 150–163, 2021. *Accepted.*

3. Jiarui Xie, Violaine Antoine, and Thierry Chateau. Partial Classification Based on Bell-Shaped Function and Dempster-Shafer Theory. Pattern Recognition. *Under Review.*

7.2 Future works

In this section, we present several future directions of research that could come out from this thesis.

7.2.1 Short-term future works

For the SLUE method:

1. In SLUE, we adopt the mean square loss function proposed in EDL. The EDL method just integrates the Dirichlet distribution into the classical mean square loss function by multiplying. Consequently, we want to explore more potential options.

2. Apart from the base rate update strategy mentioned in the SLUE method, another strategy can also be an option that is to use the frequency of each class to update the base rate.

3. Although extensive research has been carried out, the strategy that rejecting samples only based on the uncertainty value is sometimes improper. Since SL can generate opinions over the hyper domain (reduced power set $2^{\Omega}/\{\emptyset, \Omega\}$), we are going to give belief to states in the hyper domain. One critical problem of giving belief for subsets is that the cardinality of the reduced power set $2^{|\Omega|} - 2$ increased exponentially along with the cardinality of Ω. In addition, generating the evidence for the element of reduced power set from K outputs is challenging. Consequently, we proposed to use the Hierarchical Agglomerative Clustering (HAC) algorithm [Sib73, Def77] to give evidence to the primary subset. The main steps are summarized as follows.

   (a) Using the HAC algorithm to calculate the Euclidean distance between the current cluster and the rest clusters.

   (b) Since evidence cannot be provided for the entire set, the cluster number is calculated one by one from 2 to K. The cluster number that maximizes the Calinski-Harabasz Index (CHI) [CH74] value is selected as the target cluster number.

   (c) The average evidence of the elements in each subset is regarded as the corresponding evidence.

   (d) Calculating beliefs for the obtained subsets (clusters), and choosing the subset with the maximum belief as the prediction.

For the PCMO and PCBS methods:

1. The PCMO cannot give beliefs to the entire set or the empty set due to the max-min normalization, it is interesting to invent a normalization strategy to alleviate the drawbacks brought by the max-min normalization.

2. For the PCMO method, apart from focusing on creating a new normalization strategy, we can also try to give beliefs based on the obtained beliefs. For example, we can gradually remove the belief less than the threshold and assign the removed belief to the rest subsets, until the current belief is greater than the threshold.

3. The loss function used in the PCBS is only based on the learned feature. It is promising to extend the current loss function by combining the BSF with the contrastive-center loss function to perform a more effective fine-tune strategy. In addition, the contrastive-center loss function uses the Euclidean distance, but for some nonlinear datasets, the Mahalanobis distance is more proper.

4. For both PCMO [XAC21] and PCBS, we only use the simplest way to make the decision, i.e., choosing the subset with the maximum belief as the prediction. It is better to explore the performance by using other decision approaches, such as presented in the [MD21].

7.2.2 Long-term future works

1. In many real-life scenarios, it is difficult to obtain labeled samples, which requires experts in the field to do manual labeling, and the time cost and economic cost

are both large. Active learning uses a certain algorithm to query the most useful unlabeled samples, hands them to experts for labeling, and then uses the queried samples to train a classification model to improve the accuracy of the model. The most commonly used strategies for the design of query functions in active learning methods are uncertainty criteria and diversity criteria. The obtained uncertainty determines which unlabeled samples should be labeled next. Uncertainty based active learning strategies for deep learning applications have been successfully used in several works [GIG17, NDH19]. Consequently, we are going to integrate the proposed uncertainty estimation method into active learning to improve the precision and robustness of sample labeling.

2. Image segmentation is the technique and process of dividing an image into a number of specific regions with unique properties and interests. The existing image segmentation methods are divided into the following categories: threshold based segmentation methods, region based segmentation methods, edge based segmentation methods, and specific theories based segmentation methods. Images may contain noise, and each pixel also needs to be partially classified or rejected. Intuitively, it is promising to apply the uncertainty estimation and partial classification method to image segmentation tasks.

Bibliography

[AB18] Murat Seçkin Ayhan and Philipp Berens. Test-time data augmentation for estimation of heteroscedastic aleatoric uncertainty in deep neural networks. 2018.

[AD08] Astride Aregui and Thierry Denœux. Constructing consonant belief functions from sample data using confidence sets of pignistic probabilities. *International Journal of Approximate Reasoning*, 49(3):575–594, 2008.

[Anz12] Yuichiro Anzai. *Pattern recognition and machine learning*. Elsevier, 2012.

[ASKR18] Alexander Amini, Ava Soleimany, Sertac Karaman, and Daniela Rus. Spatial uncertainty sampling for end-to-end control. *CoRR*, 2018.

[BB98] David Barber and Christopher M Bishop. Ensemble learning in bayesian neural networks. *Nato ASI Series F Computer and Systems Sciences*, 168:215–238, 1998.

[BB15] A. Bendale and T. Boult. Towards open world recognition. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1893–1902, jun 2015.

[BCM01] Malcolm Beynon, Darren Cosker, and David Marshall. An expert system for multi-criteria decision making using dempster shafer theory. *Expert Systems with Applications*, 20(4):357–367, 2001.

[BW08] Peter L. Bartlett and Marten H. Wegkamp. Classification with a reject option using a hinge loss. *Journal of Machine Learning Research*, 9(59):1823–1840, 2008.

[BZK18] Leonard Berrada, Andrew Zisserman, and M. Pawan Kumar. Smooth loss functions for deep top-k classification. In *6th International Conference on Learning Representations*, 2018.

[CFG14] Tianqi Chen, Emily Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In *International conference on machine learning*, pages 1683–1691. PMLR, 2014.

[CGYY20] Yue Cao, Thomas Andrew Geddes, Jean Yee Hwa Yang, and Pengyi Yang. Ensemble deep learning in bioinformatics. *Nature Machine Intelligence*, 2(9):500–508, 2020.

[CH74] T. Caliński and J Harabasz. A dendrite method for cluster analysis. *Communications in Statistics*, 3(1):1–27, 1974.

[CMK+14] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Computer Vision and Pattern Recognition*, pages 3606–3613, 2014.

[DCDB09] Juan José Del Coz, Jorge Díez, and Antonio Bahamonde. Learning nondeterministic classifiers. *Journal of Machine Learning Research*, 10(10), 2009.

[Def77] D. Defays. An efficient algorithm for a complete link method. *The Computer Journal*, 20(4):364–366, 01 1977.

[Dem67] Arthur P Dempster. Upper and lower probabilities induced by a multivalued mapping. In *Classic works of the Dempster-Shafer theory of belief functions*, page 325–339. 1967.

[Den97] Thierry Denœux. Analysis of evidence-theoretic decision rules for pattern classification. *Pattern Recognit.*, 30:1095–1107, 1997.

[Den00] Thierry Denœux. A neural network classifier based on dempster-shafer theory. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 30(2):131–150, 2000.

[Den20] Yong Deng. Uncertainty measure in evidence theory. *Science China Information Sciences*, 63(11):1–19, 2020.

[DFB+14] Nan Ding, Youhan Fang, Ryan Babbush, Changyou Chen, Robert D Skeel, and Hartmut Neven. Bayesian sampling using stochastic gradient thermostats. *Advances in neural information processing systems*, 27, 2014.

[DKPR87] Simon Duane, Anthony D Kennedy, Brian J Pendleton, and Duncan Roweth. Hybrid monte carlo. *Physics letters B*, 195(2):216–222, 1987.

[DKS19] Thierry Denœux, Orakanya Kanjanatarakul, and Songsak Sriboonchitta. A new evidential k-nearest neighbor rule based on contextual discounting with partially supervised learning. *International Journal of Approximate Reasoning*, 113:287–302, 2019.

[DM93] Bernard Dubuisson and Mylène Masson. A statistical decision rule with incomplete knowledge about classes. *Pattern Recognition*, 26(1):155–165, 1993.

[DP82] D. Dubois and H. Prade. On several representations of an uncertainty body of evidence. *Fuzzy Information and Decision Processes*, page 167–181, 1982.

[DT72] A. De Luca and S. Termini. A definition of a nonprobabilistic entropy in the setting of fuzzy sets theory. *Information and Control*, 20(4):301–312, 1972.

[DWCH12] Krzysztof Dembczyński, Willem Waegeman, Weiwei Cheng, and Eyke Hüllermeier. On label dependence and loss minimization in multi-label classification. *Machine Learning*, 88(1-2):5–45, 2012.

[DWWZ19] Xueyuan Dai, Xiaofeng Wu, Bin Wang, and Liming Zhang. Semisupervised scene classification for remote sensing images: A method based on convolutional neural networks and ensemble learning. *IEEE Geoscience and Remote Sensing Letters*, 16(6):869–873, 2019.

[DY14] Sébastien Destercke and Gen Yang. Cautious ordinal classification by binary decomposition. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 323–337, 2014.

[eco18] The economist. Why uber's self-driving car killed a pedestrian, 2018. Available online: https://www.economist.com/the-economist-explains/2018/05/29/why-ubers-self-driving-car-killed-a-pedestrian.

[Ele16] Electrek. Understanding the fatal tesla accident on autopilot and the nhtsa probe, 2016. Available online: https://electrek.co/2016/07/01/understanding-fatal-tesla-accident-autopilot-nhtsa-probe/.

[FXFL19] Liguo Fei, Jun Xia, Yuqiang Feng, and Luning Liu. A novel method to determine basic probability assignment in dempster–shafer theory and its application in multi-sensor information fusion. *International Journal of Distributed Sensor Networks*, 15(7), 2019.

[GG16] 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.

[GIG17] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. 03 2017.

[GPSW17] 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, 2017.

[GTA+21] Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, et al. A survey of uncertainty in deep neural networks. *arXiv preprint arXiv:2107.03342*, 2021.

[Ha97] T.M. Ha. The optimum class-selective rejection rule. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6):608–615, 1997.

[HAAZ21] Rula A Hamid, AS Albahri, OS Albahri, and AA Zaidan. Dempster–shafer theory for classification and hybridised models of multi-criteria decision analysis for prioritisation: a telemedicine framework for patients with heart diseases. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–35, 2021.

[HG17] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *Proceedings of International Conference on Learning Representations*, 2017.

[HK82] Masahiko Higashi and George J. Klir. Measures of uncertainty and information based on possibility distributions. *International Journal of General Systems*, 9(1):43–58, 1982.

[Hop84] Fred M. Hoppe. Pólya-like urns and the ewens' sampling formula. *Journal of Mathematical Biology*, 20(1):91–94, Aug 1984.

[HS90] Lars Kai Hansen and Peter Salamon. Neural network ensembles. *IEEE transactions on pattern analysis and machine intelligence*, 12(10):993–1001, 1990.

[HVC93] Geoffrey E Hinton and Drew Van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the sixth annual conference on Computational learning theory*, pages 5–13, 1993.

[HZC+17] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017.

[HZRS16] 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.

[Jøs18] Audun Jøsang. *Subjective Logic: A formalism for reasoning under uncertainty*. Springer Publishing Company, Incorporated, 2018.

[KCD+18] Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and Pieter Abbeel. Model-ensemble trust-region policy optimization, 2018.

[KG17] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, page 5580–5590, 2017.

[Kri12] Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2012.

[KSW15] Durk P Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. *Advances in neural information processing systems*, 28:2575–2583, 2015.

[KV18] Michael Kläs and Anna Maria Vollmer. Uncertainty in machine learning applications: A practice-driven classification of uncertainty. In *International conference on computer safety, reliability, and security*, pages 431–438, 2018.

[Kyb84] Henry E. Kyburg. The enterprise of knowledge: An essay on knowledge, credal probability, and chance. *Noûs*, 18(2):347–354, 1984.

[LBBH98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[LCB10] Yann LeCun, Corinna Cortes, and CJ Burges. MNIST handwritten digit database. *ATT Labs*, 2, 2010.

[LHS15] Maksim Lapin, Matthias Hein, and Bernt Schiele. Top-k multiclass svm. In *NIPS*, 2015.

[LHS16] M. Lapin, M. Hein, and B. Schiele. Loss functions for top-k error: Analysis and insights. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1468–1477, 2016.

[LHZD20] Zhun-Ga Liu, Lin-Qing Huang, Kuang Zhou, and Thierry Denœux. Combination of transferable classification with multisource domain adaptation based on evidential reasoning. *IEEE Transactions on Neural Networks and Learning Systems*, 32(5):2015–2029, 2020.

[LJS21] Titouan Lorieul, Alexis Joly, and Dennis Shasha. Classification under ambiguity: When is average-k better than top-k?, 2021.

[LLS17] Shiyu Liang, Yixuan Li, and Rayadurgam Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. 2017.

[LLY+16] Biao Leng, Yu Liu, Kai Yu, Xiangyang Zhang, and Zhang Xiong. 3D object understanding with 3D convolutional neural networks. *Information sciences*, 366:188–201, 2016.

[LP07] Martin Leutbecher and T.N. Palmer. Ensemble forecasting. (514):31, 02 2007.

[LPB17] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in neural information processing systems*, pages 6402–6413, 2017.

[LW17] Christos Louizos and Max Welling. Multiplicative normalizing flows for variational bayesian neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 2218–2227, 2017.

[LW21] Stanley E. Lazic and Dominic P. Williams. Quantifying sources of uncertainty in drug discovery predictions with probabilistic models. *Artificial Intelligence in the Life Sciences*, 1:100004, 2021.

[LWOL20] Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. Energy-based out-of-distribution detection. *Advances in Neural Information Processing Systems*, 2020.

[Man20] Iswariya Manivannan. A comparative study of uncertainty estimation methods in deep learning based classification models. Technical report, 2020.

[MD08] Marie-Hélène Masson and T. Denœux. ECM: An evidential version of the fuzzy c-means algorithm. *Pattern Recognition*, 41(4):1384–1397, 2008.

[MD20] EE Marushko and AA Doudkin. Methods of using ensembles of heterogeneous models to identify remote sensing objects. *Pattern Recognition and Image Analysis*, 30(2):211–216, 2020.

[MD21] Liyao Ma and Thierry Denœux. Partial classification in the belief function framework. *Knowledge-Based Systems*, page 106742, 2021.

[MG18] Andrey Malinin and M.J.F. Gales. Predictive uncertainty estimation via prior networks. 02 2018.

[MG19] Andrey Malinin and Mark Gales. Reverse kl-divergence training of prior networks: Improved uncertainty and adversarial robustness. In *Advances in Neural Information Processing Systems*, pages 14547–14558, 2019.

[MH12] Andino Maseleno and Md Mahmud Hasan. Avian influenza (h5n1) expert system using dempster-shafer theory. *International Journal of Information and Communication Technology*, 4(2-4):227–241, 2012.

[MMKF+20] Nikita Moshkov, Botond Mathe, Attila Kertesz-Farkas, Reka Hollandi, and Peter Horvath. Test-time augmentation for deep learning-based cell segmentation on microscopy images. *Scientific reports*, 10(1):1–7, 2020.

[MWD+21] Thomas Mortier, Marek Wydmuch, Krzysztof Dembczyński, Eyke Hüllermeier, and Willem Waegeman. Efficient set-valued prediction in multi-class classification. *Data Mining and Knowledge Discovery*, 35(4):1435–1469, 2021.

[MWT+20] Alireza Mehrtash, William M Wells, Clare M Tempany, Purang Abolmaesumi, and Tina Kapur. Confidence calibration and predictive uncertainty estimation for deep medical image segmentation. *IEEE transactions on medical imaging*, 39(12):3868–3878, 2020.

[MWZY20] Weijie Mai, Mingzhu Wei, Junqing Zhang, and Feng Yuan. Research on chinese text and application based on the latent dirichlet allocation. In *2020 3rd International Conference on Advanced Electronic Materials, Computers and Software Engineering (AEMCSE)*, pages 545–553, 2020.

[NCH15] Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[NDH19] Vu-Linh Nguyen, Sébastien Destercke, and Eyke Hüllermeier. Epistemic uncertainty sampling. In *International Conference on Discovery Science*, pages 72–86. Springer, 2019.

[Nea92] Radford M Neal. Bayesian training of backpropagation networks by the hybrid monte carlo method. Technical report, Citeseer, 1992.

[Nea12] Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

[NGB20] Loris Nannia, Stefano Ghidoni, and Sheryl Brahnam. Ensemble of convolutional neural networks for bioimage classification. *Applied Computing and Informatics*, 2020.

[NHL20] Jay Nandy, Wynne Hsu, and Mong-Li Lee. Towards maximizing the representation gap between in-domain out-of-distribution examples. In *NeurIPS*, 2020.

[NWC+11] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[NZH09] Malik Sajjad Ahmed Nadeem, Jean-Daniel Zucker, and Blaise Hanczar. Accuracy-rejection curves (ARCs) for comparing classification methods with a reject option. volume 8, pages 65–81, 2009.

[Par13] Wendy S Parker. Ensemble modeling, uncertainty and robust predictions. *Wiley Interdisciplinary Reviews: Climate Change*, 4(3):213–223, 2013.

[PB94] Nikhil R Pal and James C Bezdek. Measuring fuzzy uncertainty. *IEEE Transactions on Fuzzy Systems*, 2(2):107–118, 1994.

[QS17] Ce Qi and Fei Su. Contrastive-center loss for deep neural networks. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 2851–2855, 2017.

[RBS+19] Maithra Raghu, Katy Blumer, Rory Sayres, Ziad Obermeyer, Robert Kleinberg, Sendhil Mullainathan, and Jon Kleinberg. Direct uncertainty prediction for medical second opinions, 2019.

[RDS+15] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[RGP18] AA Rassafi, Seyedreza Seyedalizadeh Ganji, and H Pourkhani. Road safety assessment under uncertainty using a multi attribute decision analysis based on dempster–shafer theory. *KSCE Journal of Civil Engineering*, 22(8):3137–3152, 2018.

[RM20] Tiago Ramalho and Miguel Miranda. Density estimation in representation space to predict model uncertainty. In *International Workshop on Engineering Dependable and Secure Machine Learning Systems*, pages 84–96, 2020.

[RV09] Phillippe Rigollet and Régis Vert. Optimal rates for plug-in estimators of density level sets. *Bernoulli*, 15(4):1154–1178, 2009.

[SCC17] Shengyang Sun, Changyou Chen, and Lawrence Carin. Learning structured weight uncertainty in bayesian neural networks. In *Artificial Intelligence and Statistics*, pages 1283–1292, 2017.

[SCL19] Heidy Shi, John Caddell, and Julia Lensing. Analyzing us army officer evaluation reports with natural language processing: A log-odds and latent dirichlet allocation exploration. *Industrial and Systems Engineering Review*, 7(1):44–55, 2019.

[Sha76] Glenn Shafer. *A mathematical theory of evidence*, volume 42. Princeton university press, 1976.

[Sha01] Claude Elwood Shannon. A mathematical theory of communication. *ACM SIGMOBILE mobile computing and communications review*, 5(1):3–55, 2001.

[SHK+14] 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.

[Sib73] R. Sibson. Slink: An optimally efficient algorithm for the single-link cluster method. *The Computer Journal*, 16(1):30–34, 01 1973.

[SK94] P. Smets and R. Kennes. The transferable belief model. *Artificial Intelligence*, 66:191–234, 1994.

[SKK18] Murat Sensoy, Lance Kaplan, and Melih Kandemir. Evidential deep learning to quantify classification uncertainty. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 3183–3193, 2018.

[SLJ+15] 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.

[SLW19] Mauricio Sadinle, Jing Lei, and Larry Wasserman. Least ambiguous set-valued classifiers with bounded error levels. *Journal of the American Statistical Association*, 114(525):223–234, 2019.

[SR18] Omer Sagi and Lior Rokach. Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1249, 2018.

[SZ15] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[TGJ+15] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 648–656, 2015.

[TLS89] Naftali Tishby, Esther Levin, and Sara A Solla. Consistent inference of probabilities in layered networks: Predictions and generalization. In *International Joint Conference on Neural Networks*, volume 2, pages 403–409, 1989.

[TSH17] Noureen Talpur, Mohd Najib Mohd Salleh, and Kashif Hussain. An investigation of membership functions on performance of anfis for solving classification problems. In *IOP Conference Series: Materials Science and Engineering*, volume 226, page 012103, 2017.

[TXD21] Zheng Tong, Philippe Xu, and Thierry Denœux. An evidential classifier based on dempster-shafer theory and deep learning. *Neurocomputing*, 450:275–293, 2021.

[VGS05] Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic learning in a random world*. Springer Science & Business Media, 2005.

[VNF+17] Vladimir Vovk, Ilia Nouretdinov, Valentina Fedorova, Ivan Petej, and Alex Gammerman. Criteria of efficiency for set-valued classification. *Annals of Mathematics and Artificial Intelligence*, 81(1–2):21–46, oct 2017.

[Vov12] Vladimir Vovk. Conditional validity of inductive conformal predictors. In *Asian conference on machine learning*, pages 475–490, 2012.

[Wal91] Peter Walley. Statistical reasoning with imprecise probabilities. *Chapman and Hal*, 1991.

[WJC+20] Jingling Wang, Rong Ju, Yuanyuan Chen, Guijun Liu, and Zhang Yi. Automated diagnosis of neonatal encephalopathy on aEEG using deep neural networks. *Neurocomputing*, 398:95–107, 2020.

[WLA+19] Guotai Wang, Wenqi Li, Michael Aertsen, Jan Deprest, Sébastien Ourselin, and Tom Vercauteren. Aleatoric uncertainty estimation with test-time augmentation for medical image segmentation with convolutional neural networks. *Neurocomputing*, 338:34–45, 2019.

[WLOV18] Guotai Wang, Wenqi Li, Sébastien Ourselin, and Tom Vercauteren. Automatic brain tumor segmentation using convolutional neural networks with test-time augmentation. In *International MICCAI Brainlesion Workshop*, pages 61–72, 2018.

[XAC21] Jiarui Xie, Violaine Antoine, and Thierry Chateau. PCMO: Partial classification from CNN-based model outputs. In *Neural Information Processing*, pages 150–163, 2021.

[XCA21] Jiarui Xie, Thierry Chateau, and Violaine Antoine. A subjective-logic-based model uncertainty estimation mechanism for out-of-domain detection. In *2021 International Joint Conference on Neural Networks*, pages 1–6, 2021.

[Yag92] Ronald R Yager. Decision making under dempster-shafer uncertainties. *International Journal of General System*, 20(3):233–245, 1992.

[Yag96] Ronald R Yager. On the normalization of fuzzy belief structures. *International Journal of Approximate Reasoning*, 14(2-3):127–153, 1996.

[YCVPK21] Vahid Yaghoubi, Liangliang Cheng, Wim Van Paepegem, and Mathias Kersemans. A novel multi-classifier information fusion based on dempster–shafer theory: Application to vibration-based fault detection. *Structural Health Monitoring*, 2021.

[YDM14] Gen Yang, Sébastien Destercke, and Marie-Hélène Masson. Nested dichotomies with probability sets for multi-class classification. In *ECAI*, pages 363–368, 2014.

[YDM17] Gen Yang, Sébastien Destercke, and Marie-Hélène Masson. Cautious classification with nested dichotomies and imprecise probabilities. *Soft Computing*, 21:7447–7462, 2017.

[YHD16] Yi Yang, Deqiang Han, and Jean Dezert. A new non-specificity measure in evidence theory based on belief intervals. *Chinese Journal of Aeronautics*, 29(3):704–713, 2016.

[YZS+15] Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. LSUN: construction of a large-scale image dataset using deep learning with humans in the loop. *CoRR*, abs/1506.03365, 2015.

[Zad65] L.A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.

[Zad99] L.A. Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 100:9–34, 1999.

[Zaf02] Marco Zaffalon. The naive credal classifier. *Journal of Statistical Planning and Inference*, 105(1):5–21, 2002.

[ZCM12] Marco Zaffalon, Giorgio Corani, and Denis Mauá. Evaluating credal classifiers by utility-discounted predictive accuracy. *International Journal of Approximate Reasoning*, 53(8):1282–1301, 2012.

[ZKL+17] Bolei Zhou, Aditya Khosla, Àgata Lapedriza, Antonio Torralba, and Aude Oliva. Places: An image database for deep scene understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1452—1464, 2017.

[ZL20] Yifan Zhang and Shizhong Liao. A kernel perspective for the decision boundary of deep neural networks. In *2020 IEEE 32nd International Conference on Tools with Artificial Intelligence*, pages 653–660, 2020.

[ZSDG18] Guodong Zhang, Shengyang Sun, David Duvenaud, and Roger Grosse. Noisy natural gradient as variational inference. In *International Conference on Machine Learning*, pages 5852–5861, 2018.

[ZYZZ16] Lei Zhang, Fan Yang, Yimin Daniel Zhang, and Ying Julie Zhu. Road crack detection using deep convolutional neural network. In *IEEE international conference on image processing*, pages 3708–3712, 2016.