ConvNet and Dempster-Shafer Theory for Object Recognition

Zheng Tong, Philippe Xu, Thierry Denoeux

ConvNet and Dempster-Shafer Theory for Object Recognition*

Zheng Tong[0000−0001−6894−3521], Philippe Xu[0000−0001−7397−4808], and Thierry Denœux[0000−0002−0660−5436]

Université de Technologie de Compiègne,
CNRS, UMR 7253 Heudiasyc, Compiègne, France.
zheng.tong@hds.utc.fr; philippe.xu@hds.utc.fr; thierry.denoeux@utc.fr

Abstract. We propose a novel classifier based on convolutional neural network (ConvNet) and Dempster-Shafer theory for object recognition allowing for ambiguous pattern rejection, called the ConvNet-BF classifier. In this classifier, a ConvNet with nonlinear convolutional layers and a global pooling layer extracts high-dimensional features from input data. The features are then imported into a belief function classifier, in which they are converted into mass functions and aggregated by Dempster's rule. Evidence-theoretic rules are finally used for pattern classification and rejection based on the aggregated mass functions. We propose an end-to-end learning strategy for adjusting the parameters in the ConvNet and the belief function classifier simultaneously and determining the rejection loss for evidence-theoretic rules. Experiments with the CIFAR-10, CIFAR-100, and MNIST datasets show that hybridizing belief function classifiers with ConvNets makes it possible to reduce error rates by rejecting patterns that would otherwise be misclassified.

Keywords: Pattern Recognition · Belief Function · Convolutional Neural Network · Supervised Learning · Evidence Theory.

1 Introduction

Dempster-Shafer (DS) theory of belief functions [3,24] has been widely used for reasoning and making decisions with uncertainty [29]. DS theory is based on representing independent pieces of evidence by completely monotone capacities and aggregating them using Dempster's rule. In the past decades, DS theory has been applied to pattern recognition and supervised classification in three main directions. The first one is classifier fusion, in which classifier outputs are converted into mass functions and fused by Dempster's rule (e.g., [19,2]). Another direction is evidential calibration: the decisions of classifiers are transformed into

* This research was carried out in the framework of the Labex MS2T, which was funded by the French Government, through the program Investments for the future managed by the National Agency for Research (Reference ANR- 11-IDEX-0004-02). It was also supported by a scholarship from the China Scholarship Council.

mass functions (e.g., [28,20]). The last approach is to design evidential classifiers (e.g., [6]), which represent the evidence of each feature as elementary mass functions and combine them by Dempster's rule. The combined mass functions are then used for decision making [5]. Compared with conventional classifiers, evidential classifiers can provide more informative outputs, which can be exploited for uncertainty quantification and novelty detection. Several principles have been proposed to design evidential classifiers, mainly including the evidential k-nearest neighbor rule [4,9], and evidential neural network classifiers [6]. In practice, the performance of evidential classifiers heavily depends on two factors: the training set size and the reliability of object representation. With the development of the "Big Data" age, the number of examples in benchmark datasets for supervised algorithms has increased from 10^2 to 10^5 [14] and even 10^9 [21]. However, little has been done to combine recent techniques for object representation with DS theory.

Thanks to the explosive development of deep learning [15] and its applications [14,25], several approaches for object representation have been developed, such as restricted Boltzmann machines [1], deep autoencoders [26,27], deep belief networks [22,23], and convolutional neural networks (ConvNets) [12,17]. ConvNet, which is maybe the most promising model and the main focus of this paper, mainly consists of convolutional layers, pooling layers, and fully connected layers. It has been proved that ConvNets have the ability to extract local features and compute global features, such as from edges to corners and contours to object parts. In general, robustness and automation are two desirable properties of ConvNets for object representation. Robustness means strong tolerance to translation and distortion in deep representation, while automation implies that object representation is data-driven with no human assistance.

Motivated by recent advances in DS theory and deep learning, we propose to combine ConvNet and DS theory for object recognition allowing for ambiguous pattern rejection. In this approach, a ConvNet with nonlinear convolutional layers and a global pooling layer is used to extract high-order features from input data. Then, the features are imported into a belief function classifier, in which they are converted into Dempster-Shafer mass functions and aggregated by Dempster's rule. Finally, evidence-theoretic rules are used for pattern recognition and rejection based on the aggregated mass functions. The performances of this classifier on the CIFAR-10, CIFAR-100, and MNIST datasets are demonstrated and discussed.

The organization of the rest of this paper is as follows. Background knowledge on DS theory and ConvNet is recalled in Section 2. The new combination between DS theory and ConvNet is then established in Section 3, and numerical experiments are reported in Section 4. Finally, we conclude the paper in Section 5.

2 Background

In this section, we first recall some necessary definitions regarding the DS theory and belief function classifier (Section 2.1). We then provide a description of the architecture of a ConvNet that will be combined with a belief function classifier later in the paper (Section 2.2).

2.1 Dempster-Shafer Theory

Evidence Theory The main concepts regarding DS theory are briefly presented in this section, and some basic notations are introduced. Detailed information can be found in Shafer's original work [24] and some up-to-date studies [8].

Given a finite set $\Omega = \{\omega_1, \cdots, \omega_k\}$, called the *frame of discernment*, a *mass function* is a function m from 2^Ω to [0,1] verifying $m(\emptyset) = 0$ and

$$\sum_{A \subseteq \Omega} m(A) = 1. \tag{1}$$

For any $A \subseteq \Omega$, given a certain piece of evidence, $m(A)$ can be regarded as the belief that one is willing to commit to A. Set A is called a *focal element* of m when $m(A) > 0$.

For all $A \subseteq \Omega$, a *credibility* function bel and a *plausibility* function pl, associated with m, are defined as

$$\mathrm{bel}(A) = \sum_{B \subseteq A} m(B) \tag{2}$$

$$\mathrm{pl}(A) = \sum_{A \cap B \neq \emptyset} m(B). \tag{3}$$

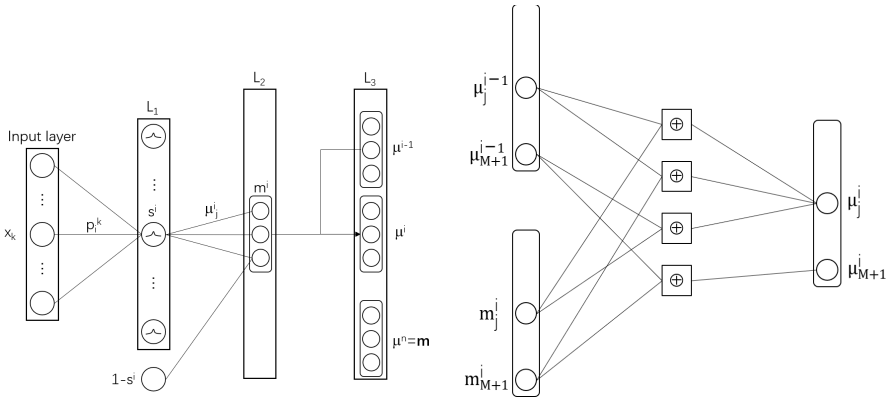
The quantity $\mathrm{bel}(A)$ is interpreted as a global measure of one's belief that hypothesis A is true, while $\mathrm{pl}(A)$ is the amount of belief that could potentially be placed in A.

Two mass functions m_1 and m_2 representing independent items of evidence can be combined by Dempster's rule $\oplus$ [3,24] as

$$(m_1 \oplus m_2)(A) = \frac{\sum_{B \cap C = A} m_1(B)\, m_2(C)}{\sum_{B \cap C \neq \emptyset} m_1(B)\, m_2(C)} \tag{4}$$

for all $A \neq \emptyset$ and $(m_1 \oplus m_2)(\emptyset) = 0$. Mass functions m_1 and m_2 can be combined if and only if the denominator on the right-hand side of (4) is strictly positive. The operator $\oplus$ is commutative and associative.

Belief Function Classifier Based on DS theory, an adaptive pattern classifier, called belief function classifier, was proposed by Denœux [6]. The classifier uses reference patterns as items of evidence regarding the class membership. The evidence is represented by mass functions and combined using Dempster's rule. In this section, we describe the architecture of a belief function classifier. For a more complete introduction, readers are invited to refer to Denœux's original work [6].



(a) Architecture of a belief function classifier (b) Connection between layers L_2 and L_3

Fig. 1: Belief function classifier

We denote by $\boldsymbol{x} \in \mathbb{R}^P$ a pattern to be classified into one of M classes $\omega_1, \cdots, \omega_M$, and by $\mathcal{X}$ a training set of N P-dimensional patterns. A belief function classifier quantifies the uncertainty about the class of $\boldsymbol{x}$ by a belief function on $\Omega = \{\omega_1, \cdots, \omega_M\}$, using a three-step procedure. This procedure can also be implemented in a multi-layer neural network illustrated in Figure 1. It is based on n prototypes $\boldsymbol{p}^1, \cdots, \boldsymbol{p}^n$, which are the weight vectors of the units in the first hidden layer L_1. The three steps are defined as follows.

Step 1: The distance between $\boldsymbol{x}$ and each prototype $\boldsymbol{p}^i$ is computed as

$$d^i = \left\|\boldsymbol{x} - \boldsymbol{p}^i\right\| \quad i = 1, \cdots, n, \tag{5}$$

and the activation of the corresponding neuron is defined by introducing new parameters η^i ($\eta^i \in \mathbb{R}$) as $s^i = \alpha^i \exp(-\left(\eta^i d^i\right)^2)$, where $\alpha^i \in (0, 1)$ is a parameter associated to the prototype $\boldsymbol{p}^i$.

Step 2: The mass function m^i associated to prototype $\boldsymbol{p}^i$ is computed as

$$\boldsymbol{m}^i = (m^i(\{\omega_1\}), \ldots, m^i(\{\omega_M\}), m^i(\Omega))^T \tag{6a}$$

$$= (u_1^i s^i, \ldots, u_M^i s^i, 1 - s^i)^T, \tag{6b}$$

where $\boldsymbol{u}^i = (u_1^i, \ldots, u_M^i)$ is a vector of parameters associated to the prototype $\boldsymbol{p}^i$ verifying $\sum_{j=1}^{M} u_j^i = 1$.

As illustrated in Figure 1a, Eq. (6) can be regarded as computing the activations of units in the second hidden layer L_2, composed of n modules of $M+1$ units each. The units of module i are connected to neuron i of the previous layer. The output of module i in the hidden layer corresponds to the belief masses assigned by m^i.

Step 3: The n mass functions m^i, $i = 1, \cdots, n$, are combined in the final layer based on Dempster's rule as shown in Figure 1b. The vectors of activations $\boldsymbol{\mu}^i = (\mu_1^i, \cdots, \mu_{M+1}^i)$, $i = 1, \ldots, n$ of the final layer L_3 is defined by the following equations:

$$\boldsymbol{\mu}^1 = \boldsymbol{m}^1, \tag{7a}$$

$$\mu_j^i = \mu_j^{i-1} m^i(\{\omega_j\}) + \mu_j^{i-1} m^i(\{\Omega\}) + \mu_{M+1}^{i-1} m^i(\{\omega_j\}) \tag{7b}$$

for $i = 2, \cdots, n$ and $j = 1, \cdots, M$, and

$$\mu_{M+1}^i = \mu_{M+1}^{i-1} m^i(\{\Omega\}) \quad i = 2, \cdots, n. \tag{7c}$$

The classifier outputs $\boldsymbol{m} = (m(\{\omega_1\}), \ldots, m(\{\omega_M\}), m(\Omega))^T$ is finally obtained as $\boldsymbol{m} = \boldsymbol{\mu}^n$.

Evidence-Theoretic Rejection Rules Different strategies to make a decision (e.g., assignment to a class or rejection) based on the possible consequences of each action were proposed in [5]. For a complete training set $\mathcal{X}$, we consider actions $\alpha_i,\ i \in \{1, \cdots, M\}$ assigning the pattern to each class and a rejection action α_0. Assuming the cost of correct classification to be 0, the cost of misclassification to be 1 and the cost of rejection to be λ_0, the three conditions for rejection reviewed in [5] can be expressed as

Maximum credibility: $\max_{j=1,\cdots,M} m(\{\omega_j\}) < 1 - \lambda_0$
Maximum plausibility: $\max_{j=1,\cdots,M} m(\{\omega_j\}) + m(\Omega) < 1 - \lambda_0$
Maximum pignistic probability: $\max_{j=1,\cdots,M} m(\{\omega_j\}) + \frac{m(\Omega)}{M} < 1 - \lambda_0$.

Otherwise, the pattern is assigned to class ω_j with $j = \arg\max_{k=1,\cdots,M} m(\{\omega_k\})$. For the maximum plausibility and maximum pignistic probability rules, rejection is possible if and only if $0 \leq \lambda_0 \leq 1 - 1/M$, whereas a rejection action for the maximum credibility rule only requires $0 \leq \lambda_0 \leq 1$.

2.2 Convolutional Neural Network

In this section, we provide a brief description of some state-of-the-art techniques for ConvNets including the nonlinear convolutional operation and global average pooling (GAP), which will be implemented in our new model in Section 3. Detailed information about the two structure layers can be found in [18].

Nonlinear Convolutional Operation The convolutional layer [15] is highly efficient for feature extraction and representation. In order to approximate the representations of the latent concepts related to the class membership, a novel convolutional layer has been proposed [18], in which nonlinear multilayer perceptron (MLP) operations replace classic convolutional operations to convolve over the input. An MLP layer with nonlinear convolutional operations can be summarized as follows:

$$f_{i,j,k}^{1} = ReLU\left(\left(\mathbf{w}_k^1\right)^T \cdot \mathbf{x} + b_k^1\right), k = 1, \cdots, C \tag{8a}$$

$$\vdots$$

$$f_{i,j,k}^{m} = ReLU\left(\left(\mathbf{w}_k^m\right)^T \cdot \mathbf{f}_{i,j}^{m-1} + b_k^m\right), k = 1, \cdots, C. \tag{8b}$$

Here, m is the number of layers in an MLP. Matrix $\mathbf{x}$, called receptive field of size $i \times j \times o$, is a patch of the input data with the size of $(rW - r - p + i) \times (rH - r - p + j) \times o$. An MLP layer with an r stride and a p padding can generate a $W \times H \times C$ tensor, called feature maps. The size of a feature map is $W \times H \times 1$, while the channel number of the feature maps is C. A rectified linear unit (ReLU) is used as an activation function as $ReLU(x) = \max(0, x)$. As shown in Eq. (8), element-by-element multiplications are first performed between $\mathbf{x}$ and the transpositions of the weight matrices $\mathbf{w}_k^1$ $(k = 1, \cdots, C)$ in the 1^{st} layer of the MLP. Each weight matrix $\mathbf{w}_k^1$ has the same size as the receptive field. Then the multiplied values are summed, and the bias b_k^1 $(k = 1, \cdots, C)$ is added to the summed values. The results are transformed by a $ReLU$ function. The output vector is $\mathbf{f}_{i,j}^1 = (f_{i,j,1}^1, f_{i,j,2}^1, \cdots, f_{i,j,C}^1)$. The outputs then flow into the remaining layers in sequence, generating $\mathbf{f}_{i,j}^m$ of size $1 \times 1 \times C$. After processing all patches by the MLP, the input data is transformed into a $W \times H \times C$ tensor. As the channel number C of the last MLP in a ConvNet is the same as the input data dimension P in a belief function classifier, a $W \times H \times P$ tensor is finally generated by a ConvNet.

Global Average Pooling In a traditional ConvNet, the tensor is vectorized and imported into fully connected layers and a softmax layer for a classification task. However, fully connected layers are prone to overfitting, though dropout [11] and its variation [10] have been proposed. A novel strategy, called global average pooling (GAP), has been proposed to remove traditional fully connected layers [18]. A GAP layer transforms the feature tensor $W \times H \times P$ into a feature vector $1 \times 1 \times P$ by taking the average of each feature map as follows:

$$x_k = \frac{\sum_{i=1}^{W}\sum_{j=1}^{H} f_{i,j,k}^m}{W \cdot H} \quad k = 1, \cdots, P. \tag{9}$$

The generated feature vector is used for classification. From the belief function perspective, the feature vector can be used for object representation and

classified in one of M classes or rejected by a belief function classifier. Thus, a ConvNet can be regarded as a feature generator.

3 ConvNet-BF Classifier

In this section, we present a method to combine a belief function classifier and a ConvNet for objection recognition allowing for ambiguous pattern rejection. The architecture of the proposed method, called ConvNet-BF classifier, is illustrated in Figure 2. A ConvNet-BF classifier can be divided into three parts: a ConvNet as a feature producer, a belief function classifier as a mass-function generator, and a decision rule. In this classifier, input data are first imported into a ConvNet with nonlinear convolutional layers and a global pooling layer to extract latent features related to the class membership. The features are then imported into a belief function classifier, in which they are converted into mass functions and aggregated by Dempster's rule. Finally, an evidence-theoretic rule is used for pattern classification and rejection based on the aggregated mass functions. As the background of the three parts has been introduced in Section 2, we only provide the details of the combination in this section, including the connectionist implementation and the learning strategy.

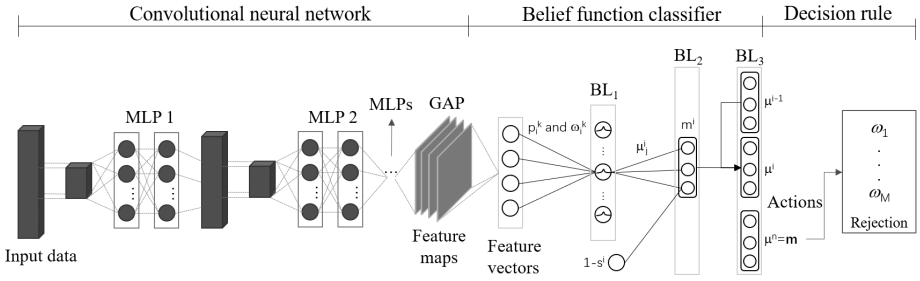


Fig. 2: Architecture of a ConvNet-BF classifier

3.1 Connectionist Implementation

In a ConvNet-BF classifier, the Euclidean distance between a feature vector and each prototype is first computed and then used to generate a mass function. To reduce the classification error when P is large, we assign weights to each feature as

$$d^i = \sqrt{\sum_{k=1}^{P} w_k^i (x_k - p_k^i)^2}, \tag{10}$$

and the weights are normalized by introducing new parameters ζ_k^i ($\zeta_k^i \in \mathbb{R}$) as

$$w_k^i = \frac{(\zeta_k^i)^2}{\sum_{l=1}^{P}(\zeta_l^i)^2}. \tag{11}$$

3.2 Learning

The proposed learning strategy to train a ConvNet-BF classifier consists in two parts: (a) an end-to-end training method to train ConvNet and belief function classifier simultaneously; (b) a data-driven method to select λ_0.

End-to-End Training Compared with the belief function classifier proposed in [6], we have different expressions for the derivatives w.r.t. w_k^i, ζ_k^i, and p_k^i in the new belief function classifier. A normalized error $E_\nu(\boldsymbol{x})$ is computed as:

$$E_\nu(\boldsymbol{x}) = \frac{1}{2N}\sum_{i=1}^{I}\sum_{q=1}^{M}(Pre_{\nu,q,i} - Tar_{q,i})^2, \tag{12a}$$

$$Pre_{\nu,q,i} = m_{q,i}^{'} + \nu m_{M+1,i}^{'}, \tag{12b}$$

$$\boldsymbol{m}_i^{'} = \frac{\boldsymbol{m}_i}{\sum_{k=1}^{M+1} m_i(\{\omega_k\})}. \tag{12c}$$

Here, $\boldsymbol{Tar}_i = (Tar_{1,i}, \cdots, Tar_{M,i})$ and $\boldsymbol{m}_i = (m_i(\{\omega_1\}), \ldots, m_i(\{\omega_M\}), m_i(\Omega))^T$ are the target output vector and the unnormalized network output vector for pattern $\boldsymbol{x}_i$, respectively. We transform $\boldsymbol{m}_i$ to a vector $(Pre_{\nu,1,i}, \ldots, Pre_{\nu,M,i})$ by distributing a fraction ν of $m_i(\Omega)$ to each class under the constraint $0 \leq \nu \leq 1$. The numbers $Pre_{1,q,i}$, $Pre_{0,q,i}$ and $Pre_{1/M,q,i}$ represent, respectively, the credibility, the plausibility, and the pignistic probability of class ω_q. The derivatives of $E_\nu(\boldsymbol{x})$ w.r.t p_k^i, w_k^i, and ζ_k^i in a belief function classifier can be expressed as

$$\frac{\partial E_\nu(\boldsymbol{x})}{\partial p_k^i} = \frac{\partial E_\nu(\boldsymbol{x})}{\partial s^i}\frac{\partial s^i}{\partial p_k^i} = \frac{\partial E_\nu(\boldsymbol{x})}{\partial s^i} \cdot 2(\eta^i)^2 s^i \cdot \sum_{k=1}^{P} w_k^i(x_k - p_k^i), \tag{13}$$

$$\frac{\partial E_\nu(\boldsymbol{x})}{\partial w_k^i} = \frac{\partial E_\nu(\boldsymbol{x})}{\partial s^i}\frac{\partial s^i}{\partial w_k^i} = \frac{\partial E_\nu(\boldsymbol{x})}{\partial s^i} \cdot \left(\eta^i\right)^2 s^i \cdot \left(x_k - p_k^i\right)^2, \tag{14}$$

and

$$\frac{\partial E_\nu(\boldsymbol{x})}{\partial \zeta_k^i} = \frac{\partial E_\nu(\boldsymbol{x})}{\partial w_k^i}\frac{\partial w_k^i}{\partial \zeta_k^i} \tag{15a}$$

$$= \frac{2\zeta_k^i}{\left(\sum_{k=1}^{P}\left(\zeta_k^i\right)^2\right)^2}\left[\frac{\partial E_\nu(\boldsymbol{x})}{\partial w_k^i}\sum_{k=1}^{P}(\zeta_k^i)^2 - \sum_{k=1}^{P}(\zeta_k^i)^2\frac{\partial E_\nu(\boldsymbol{x})}{\partial w_k^i}\right]. \tag{15b}$$

Finally, the derivatives of the error w.r.t. x_k, $w^m_{i,j,k}$ and b^m_k in the last MLP are given as

$$\frac{\partial E_\nu(\boldsymbol{x})}{\partial x_k} = \frac{\partial E_\nu(\boldsymbol{x})}{\partial s^i}\frac{\partial s^i}{\partial x_k} = -\frac{\partial E_\nu(\boldsymbol{x})}{\partial s^i} \cdot 2(\eta^i)^2 s^i \cdot \sum_{k=1}^{P} \omega^i_k (x_k - p^i_k), \tag{16}$$

$$\frac{\partial E_\nu(\boldsymbol{x})}{\partial w^m_{i,j,k}} = \frac{\partial E_\nu(\boldsymbol{x})}{\partial f^m_{i,j,k}} \cdot \frac{\partial f^m_{i,j,k}}{\partial w^m_{i,j,k}} = w^m_{i,j,k} \cdot \frac{\partial E_\nu(\boldsymbol{x})}{\partial f^m_{i,j,k}} \quad k = 1, \cdots, P, \tag{17}$$

and

$$\frac{\partial E_\nu(\boldsymbol{x})}{\partial b^m_k} = \frac{\partial E_\nu(\boldsymbol{x})}{\partial f^m_{i,j,k}} \cdot \frac{\partial f^m_{i,j,k}}{\partial b^m_k} = \frac{\partial E_\nu(\boldsymbol{x})}{\partial f^m_{i,j,k}} \quad k = 1, \cdots, P \tag{18}$$

with

$$\frac{\partial E_\nu(\boldsymbol{x})}{\partial f^m_{i,j,k}} = \frac{\partial E_\nu(\boldsymbol{x})}{\partial x_k} \cdot \frac{\partial x_k}{\partial f^m_{i,j,k}} = \frac{1}{W \cdot H}\frac{\partial E_\nu(\boldsymbol{x})}{\partial x_k} \quad k = 1, \cdots, P. \tag{19}$$

Here, $w^m_{i,j,k}$ is the component of the weight matrix $\mathbf{w}^m_k$, while $f^m_{i,j,k}$ is the component of vector $\mathbf{f}^m_{i,j}$ in Eq. (8).

Determination of $\boldsymbol{\lambda_0}$ A data-driven method for determining λ_0 to guarantee a ConvNet-BF classifier with a certain rejection rate is shown in Figure 3. We randomly select three-fifths of a training set χ to train a ConvNet-BF classifier, while random one-fifth of the set is used as a validation set. The remaining one-fifth of the set is used to draw a $\lambda_0^{(1)}$-rejection curve. We can determine the value of $\lambda_0^{(1)}$ for a certain rejection rate from the curve. We repeat the process and take the average of $\lambda_0^{(i)}$ as the final λ_0 for the desired rejection rate.

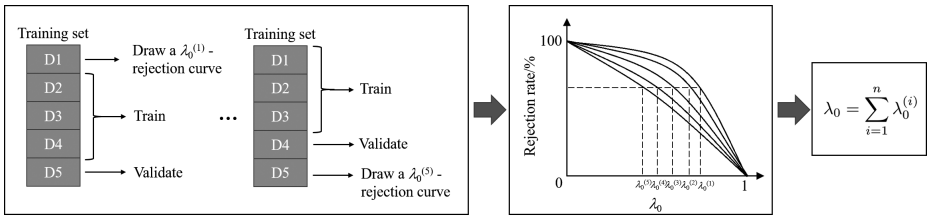


Fig. 3: Illustration of the procedure for determining λ_0

4 Numerical Experiments

In this section, we evaluate ConvNet-BF classifiers on three benchmark datasets: CIFAR-10 [13], CIFAR-100 [13], and MNIST [16]. To compare with traditional ConvNets, the architectures and training strategies of the ConvNet parts in

ConvNet-BF classifiers are the same as those used in the study of Lin et al, called NIN [18]. Feature vectors from the ConvNet parts are imported into a belief function classifier in our method, while they are directly injected into softmax layers in NINs.

In order to make a fair comparison, a probability-based rejection rule is adopted for NINs as $\max_{j=1,\cdots,M} p_j < 1 - \lambda_0$, where p_j is the output probability of NINs.

4.1 CIFAR-10

The CIFAR-10 dataset [13] is made up of 60,000 RGB images of size 32×32 partitioned in 10 classes. There are 50,000 training images, and we randomly selected 10,000 images as validation data for the ConvNet-BF classifier. We then randomly used 10,000 images of the training set to determine λ_0.

The test set error rates without rejection of the ConvNet-BF and NIN classifiers are 9.46% and 9.21%, respectively. The difference is small but statistically significant according to McNemar's test (p-value: 0.012). Error rates without rejection mean that we only consider $\max_{j=1,\cdots,M} p_j$ and $\max_{j=1,\cdots,M} m(\{\omega_j\})$. If the selected class is not the correct one, we regard it as an error. It turns out in our experiment that using a belief function classifier instead of a softmax layer only slightly impacts the classifier performance.

The test set error rates with rejection of the two models are presented in Figure 4a. A rejection decision is not regarded as an incorrect classification. When the rejection rate increases, the test set error decreases, which shows that the belief function classifier rejects a part of incorrect classification. However, the error decreases slightly when the rejection rate is higher than 7.5%. This demonstrates that the belief function classifier rejects more and more correctly classified patterns with the increase of rejection rates. Thus, a satisfactory $\lambda_0^{(i)}$ should be determined to guarantee that the ConvNet-BF classifier has a desirable accuracy rate and a low correct-rejection rate. Additionally, compared with the NIN, the ConvNet-BF classifier rejects significantly more incorrectly classified patterns. For example, the p-value of McNemar's test for the difference of error rates between the two classifiers with a 5.0% rejection rate is close to 0. We can conclude that a belief function classifier with an evidence-theoretic rejection rule is more suitable for making a decision allowing for pattern rejection than a softmax layer and the probability-based rejection rule.

Table 1 presents the confusion matrix of the ConvNet-BF classifier with the maximum credibility rule, whose rejection rate is 5.0%. The ConvNet-BF classifier tends to select rejection when there are two or more similar patterns, such as dog and cat, which can lead to incorrect classification. In the view of evidence theory, the ConvNet part provides conflicting evidence when two or more similar patterns exist. The maximally conflicting evidence corresponds to $m(\{\omega_i\}) = m(\{\omega_j\}) = 0.5$ [7]. Additionally, the additional mass function $m(\Omega)$ provides the possibility to verify whether the model is well trained because we have $m(\Omega) = 1$ when the ConvNet part cannot provide any useful evidence.

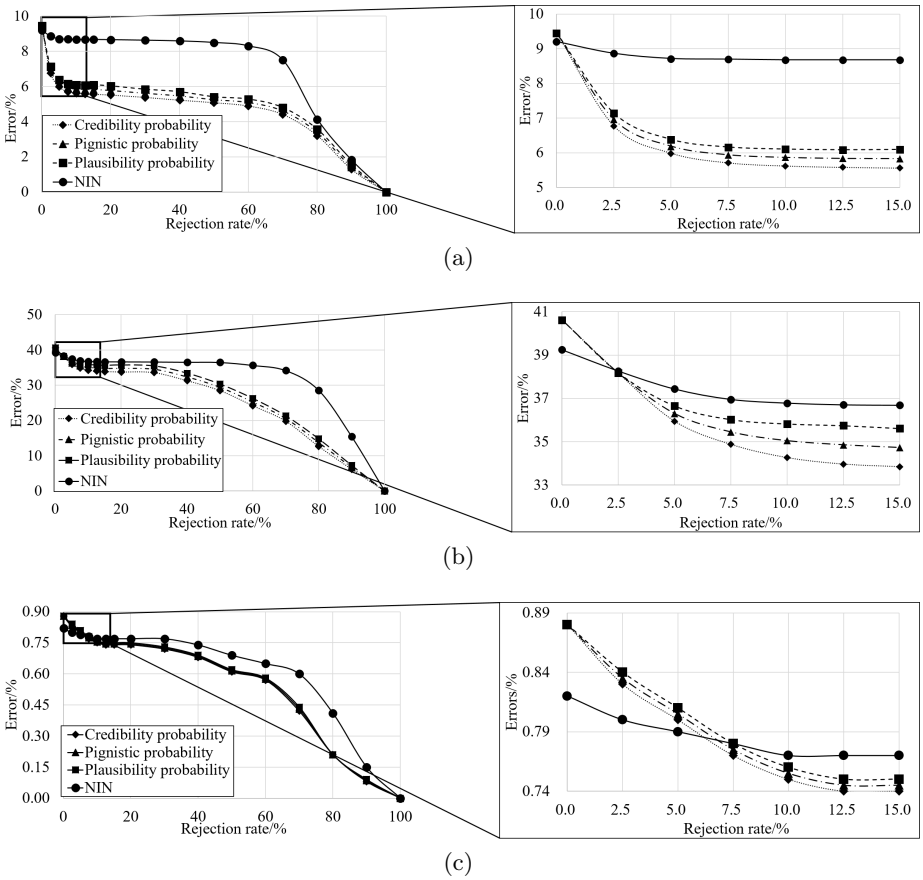


Fig. 4: Rejection-error curves: CIFAR-10 (a), CIFAR-100 (b), and MNIST (c)

Table 1: Confusion matrix for Cifar10.

	Airplane	Automobile	Bird	Cat	Deer	Dog	Frog	Horse	Ship	Truck
Airplane	-	0.03	0.03	0.01	0.02	0.05	0.04	0.01	0.04	0.05
Automobile	0	-	0.04	0.04	0.08	0.08	0.04	0.06	0.03	0.07
Bird	0.02	0.04	-	0.05	0.04	0.07	0.03	0.08	0	0.04
Cat	0.02	0.03	0.13	-	0.06	**0.44**	0.11	0.04	0.05	0.06
Deer	0.01	0.04	0.07	0.12	-	0.03	0.12	**0.34**	0.04	0.08
Dog	0.02	0.03	0.05	**0.49**	0.11	-	0.06	0.09	0.01	0.04
Frog	0.02	0.04	0.08	0.06	0.12	0.06	-	0.06	0.06	0.05
Horse	0.01	0.02	0.04	0.06	**0.31**	0.10	0.04	-	0.04	0.04
Ship	0.04	0.05	0.02	0.04	0.12	0.05	0.04	0.18	-	0.02
Truck	0.02	0	0.06	0.09	0.03	0.06	0.07	0.06	0.04	-
Rejection	0.20	0.13	0.14	**1.05**	**0.84**	**1.07**	0.14	**1.14**	0.18	0.11

4.2 CIFAR-100

The CIFAR-100 dataset [13] has the same size and format at the CIFAR-10 dataset, but it contains 100 classes. Thus the number of images in each class is only 100. For CIFAR-100, we also randomly selected 10,000 images of the training set to determine λ_0. The ConvNet-BF and NIN classifiers achieved, respectively, 40.62% and 39.24% test set error rates without rejection, a small but statistically significant difference (p-value: 0.014). Similarly to CIFAR-10, it turns out that the belief function classifier has a similar error rate as a network with a softmax layer. Figure 4b shows the test set error rates with rejection for the two models. Compared with the rejection performance in CIFAR-10, the ConvNet-BF classifier rejects more incorrect classification results. We can conclude that the evidence-theoretic classifier still performs well when the classification task is difficult and the training set is not adequate. Similarly, Table 2 shows that the ConvNet-BF classifier tends to select the rejection action when two classes are similar, in which case we have $m(\{\omega_i\}) \approx m(\{\omega_j\})$. In contrast, the classifier tends to produce $m(\Omega) \approx 1$ when the model is not trained well because of an inadequate training set.

4.3 MNIST

The MNIST database of handwritten digits consists of a training set of 60,000 examples and a test set of 10,000 examples. The training strategy for the ConvNet-BF classifier was the same as the strategy in CIFAR-10 and CIFAR-100. The test set error rates without rejection of the two models are close (0.88% and 0.82%) and weakly signifiant (p-value: 0.077). Again, using a belief function classifier instead of a softmax layer introduce no negative effect on the network in MNIST. The test set error rates with rejection of the two models are shown in Figure 4c. The ConvNet-BF classifier rejects a small number of classification results because the feature vectors provided by the ConvNet part include little confusing information.

5 Conclusion

In this work, we proposed a novel classifier based on ConvNet and DS theory for object recognition allowing for ambiguous pattern rejection, called "ConvNet-BF

Table 2: Confusion matrix for the superclass *flowers*.

	Orchids	Poppies	Roses	Sunflowers	Tulips
Orchids	-	0.24	0.23	0.28	0.15
Poppies	0.14	-	**0.43**	0.10	**0.90**
Roses	0.27	0.12	-	0.16	0.13
Sunflowers	0.18	0.15	0.12	-	0.22
Tulips	0.08	**1.07**	**0.76**	0.17	-
Rejection	0.09	**0.37**	**0.63**	0.12	**0.34**

classifier". This new structure consists of a ConvNet with nonlinear convolutional layers and a global pooling layer to extract high-dimensional features and a belief function classifier to convert the features into Dempster-Shafer mass functions. The mass functions can be used for classification or rejection based on evidence-theoretic rules. Additionally, the novel classifier can be trained in an end-to-end way.

The use of belief function classifiers in ConvNets had no negative effect on the classification performances on the CIFAR-10, CIFAR-100, and MNIST datasets. The combination of belief function classifiers and ConvNet can reduce the errors by rejecting a part of the incorrect classification. This provides a new direction to improve the performance of deep learning for object recognition. The classifier is prone to assign a rejection action when there are conflicting features, which easily yield incorrect classification in the traditional ConvNets. In addition, the proposed method opens a way to explain the relationship between the extracted features in convolutional layers and class membership of each pattern. The mass $m(\Omega)$ assigned to the set of classes provides the possibility to verify whether a ConvNet is well trained or not.

References

1. Bengio, Y.: Learning deep architectures for AI. Foundations and Trends in Machine Learning **2**(1), 1–127 (2009)
2. Bi, Y.: The impact of diversity on the accuracy of evidential classifier ensembles. International Journal of Approximate Reasoning **53**(4), 584–607 (2012)
3. Dempster, A.P.: Upper and Lower Probabilities Induced by a Multivalued Mapping, pp. 57–72. Springer Berlin Heidelberg, Berlin, Heidelberg (2008)
4. Denœux, T.: A k-nearest neighbor classification rule based on Dempster-Shafer theory. IEEE Transactions on Systems, Man, and Cybernetics **25**(5), 804–813 (1995)
5. Denœux, T.: Analysis of evidence-theoretic decision rules for pattern classification. Pattern Recognition **30**(7), 1095 – 1107 (1997)
6. Denœux, T.: 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)
7. Denœux, T.: Logistic regression, neural networks and Dempster-Shafer theory: A new perspective. Knowledge-Based Systems **176**, 54–67 (2019)
8. Denœux, T., Dubois, D., Prade, H.: Representations of uncertainty in artificial intelligence: Beyond probability and possibility. In: Marquis, P., Papini, O., Prade, H. (eds.) A Guided Tour of Artificial Intelligence Research, chap. 4. Springer Verlag (2019)
9. Denœux, T., Kanjanatarakul, O., Sriboonchitta, S.: A new evidential k-nearest neighbor rule based on contextual discounting with partially supervised learning. International Journal of Approximate Reasoning **113**, 287–302 (2019)
10. Gomez, A.N., Zhang, I., Swersky, K., Gal, Y., Hinton, G.E.: Targeted dropout. In: CDNNRIA Workshop at the 32nd Conference on Neural Information Processing Systems (NeurIPS 2018). Montréal, Canada (2018)
11. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R.: Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580 (2012)

12. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing. pp. 1746–1751. Doha, Qatar (2014)
13. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images. Tech. rep., University of Toronto (2009)
14. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Communications of the ACM **60**(6), 84–90 (2017)
15. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436 (2015)
16. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., et al.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (1998)
17. Leng, B., Liu, Y., Yu, K., Zhang, X., Xiong, Z.: 3D object understanding with 3D convolutional neural networks. Information Sciences **366**, 188–201 (2016)
18. Lin, M., Chen, Q., Yan, S.: Network in network. In: International Conference on Learning Representations (ICLR 2014). pp. 1–10. Banff, Canada (2014)
19. Liu, Z., Pan, Q., Dezert, J., Han, J.W., He, Y.: Classifier fusion with contextual reliability evaluation. IEEE Transactions on Cybernetics **48**(5), 1605–1618 (2018)
20. Minary, P., Pichon, F., Mercier, D., Lefevre, E., Droit, B.: Face pixel detection using evidential calibration and fusion. International Journal of Approximate Reasoning **91**, 202–215 (2017)
21. Sakaguchi, K., Post, M., Van Durme, B.: Efficient elicitation of annotations for human evaluation of machine translation. In: Proceedings of the Ninth Workshop on Statistical Machine Translation. pp. 1–11. Baltimore, Maryland, USA (2014)
22. Salakhutdinov, R., Hinton, G.: Deep boltzmann machines. In: Artificial intelligence and statistics. pp. 448–455. Florida, USA (2009)
23. Salakhutdinov, R., Tenenbaum, J.B., Torralba, A.: Learning with hierarchical-deep models. IEEE transactions on Pattern Analysis and Machine Intelligence **35**(8), 1958–1971 (2012)
24. Shafer, G.: A mathematical theory of evidence. Princeton University Press, Princeton, N.J. (1976)
25. Tong, Z., Gao, J., Zhang, H.: Recognition, location, measurement, and 3d reconstruction of concealed cracks using convolutional neural networks. Construction and Building Materials **146**, 775–787 (2017)
26. Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.A.: Extracting and composing robust features with denoising autoencoders. In: Proceedings of the 25th International Conference on Machine learning. pp. 1096–1103. New York, USA (2008)
27. Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.A.: Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. Journal of Machine Learning Research **11**(Dec), 3371–3408 (2010)
28. Xu, P., Davoine, F., Zha, H., Denœux, T.: Evidential calibration of binary svm classifiers. International Journal of Approximate Reasoning **72**, 55–70 (2016)
29. Yager, R.R., Liu, L.: Classic works of the Dempster-Shafer theory of belief functions, vol. 219. Springer, Heidelberg (2008)