Fuzzy Learning Machine

Junbiao Cui[1]
945546899@qq.com

Jiye Liang[1*]
ljy@sxu.edu.cn

[1] Key Laboratory of Computational Intelligence and Chinese Information Processing of Ministry of Education, School of Computer and Information Technology, Shanxi University, Taiyuan 030006, Shanxi, China.

Abstract

Classification is one of the most important problems in machine learning and the nature of it is concept cognition. So far, dozens of different classifiers have been designed. Although their working mechanisms vary widely, few of them fully consider concept cognition. In this paper, a new learning machine, fuzzy learning machine (FLM), is proposed from the perspective of concept cognition. Inspired by cognitive science, its working mechanism is of strong interpretability. At the same time, FLM roots in set theory and fuzzy set theory, so FLM has a solid mathematical foundation. The systematic experimental results on a large number of data sets show that FLM can achieve excellent performance, even with the simple implementation.

1 Introduction

Given an input space $\mathcal{X}$, an output space (a finite set) $\mathcal{Y}$, and an unknown function $\varphi : \mathcal{X} \to \mathcal{Y}$, the goal of classification is finding the φ or an approximation to it. Classification is one of the most important problems in machine learning, dating back to the origins of machine learning. In classification, each member of the output space $\mathcal{Y}$ corresponds to a concept. In essence, the process of classification is the process of concept cognition. Concept contains our knowledge about the world, and we use concept to understand and organize the world. Without it, there will be no human intelligence at all. That is why classification plays an important role in machine learning even in artificial intelligence.

Over the past decades, a large number of classifiers have been designed and achieved success on different tasks. The literature [1] systematically summarizes 179 classifiers from 17 different families, which contains almost all existing classifiers. Usually, generalization performance is regarded as an important dimension for evaluating classifiers. With the increasing depth and breadth of machine learning applications, classifiers are expected to possess more and more good properties, such as interpretability [2, 3, 4], robustness [5, 6, 7], and so on.

The existing classifiers follow different design principles (see **Appendix** A.2.1 for detailed analysis), but they are designed without full consideration for the nature of classification and how humans do it. For a specific scenario, the existing classifiers often perform well on one evaluation dimension but poorly on others. For example, deep neural network has exceed the human level in very specific settings [8]. Unfortunately, its working mechanism is difficult to be understood due to the complexity of the network structure [9, 10], and it is also very vulnerable to be attacked [11, 12]. Instead, human can perform well on a variety of tasks and on different evaluation dimensions.

To implement a classifier that is easy to understand and interpretable, one effective approach is to draw on relevant research in cognitive science. As mentioned above, classifying is essentially the

*Corresponding author

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

process of concept cognition. In cognitive science, there is a lot of valuable theories about how human learn, represent and use concept [13]. However, few of the existing classifiers are designed based on these theories, which leads to that the existing classifiers are unable to take into account the performance on multiple evaluation dimensions. Inspires by this, we are encouraged to design a new classifier based on concept cognition, which can simultaneously perform well on different evaluation dimensions. In this paper, a new learning machine is designed based on the following principles.

Similarity is the basis of concept representation. Similarity is a fundamental concept within cognitive science and plays a crucial role in the process of human classification [14]. Concepts are represented based on similarity for children, which is also a basic choice for adults [15]. Based on this principle, we prove the equivalence between classification and equivalence relation (ER) via binary relation in set theory. Considering that ER is a special similarity, we actually demonstrate that all classification problems can be solved based on similarity (see **Section** 2.1).

Concept is fuzzy. An important conclusion in cognitive science is that fuzziness is the intrinsic property of concept [13]. The fuzziness and the randomness are the two most important sources of the uncertainty. In machine learning, probability theory is often used for dealing with the uncertainty, such as Bayesian methods [16, 17], label distribution learning [18], etc. It is sufficient for probability theory to deal with the randomness rather than the fuzziness [19]. However, the intrinsic property of concept is just the fuzziness rather than the randomness. An intuitive example is shown in Figure 1.

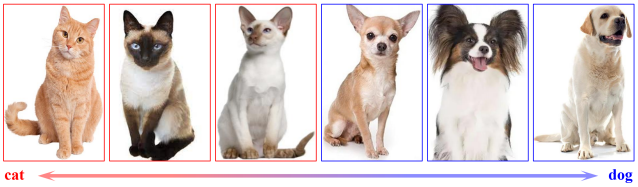

Figure 1: The visual concepts of 'cat' and 'dog' are of fuzziness. In fuzzy set theory, the membership degree of the 3rd image to 'cat' ('dog') can be 0.6 (0.5). In probability theory, the probability of the 3rd image being 'cat' ('dog') can be 0.6 ($1 - 0.6 = 0.4$). Obviously, the latter is unreasonable, because the 3rd image can not be 'cat' in some case and be 'dog' in the other case.

Based on this principle, fuzzy equivalence relation (FER) in fuzzy set theory [20] rather than ER is chosen as the core component of the new learning machine (see **Section** 2.2).

Exemplar theory based concept representation In cognitive science, there are several theories of concept representation [13]. Exemplar theory is chosen to solve concept representation problem, in this paper. Because it not only has a good self-consistency in cognitive science, but also it can be naturally integrated with the data-driven machine learning paradigm (see **Section** 2.3).

To summarize, the main contributions of this paper are four-folds as below:
(1) This paper revisits classification problem from the perspective of concept cognition, and proves the equivalence between classification problem and ER. Furthermore, considering that concept is intrinsically fuzzy, the classification problem is modeled as an FER problem.
(2) A new and general learning machine FLM is proposed based on concept cognition for solving FER problem.
(3) A new fuzziness permissible loss is designed. Given the new loss, this paper prove that FER can be approximated effectively by fuzzy similarity relation (FSR). And then a neural network based FLM (NN-FLM) is designed.
(4) Extensive experimental results demonstrate the superiority of NN-FLM on different evaluation dimensions including interpretability, robustness, and generalization performance.

2 Fuzzy Learning Machine

Basic notations Let $\mathcal{X}$ and $\mathcal{Y} = \{c_1, c_2, \cdots, c_k\}$ be the input and output spaces, respectively. Let $\varphi : \mathcal{X} \to \mathcal{Y}$ be the unknown target function and $\forall x \in \mathcal{X}$, $\varphi(x) \in \mathcal{Y}$ is the true class label of the sample x. The goal of classification is to approximate φ as closely as possible. Let $D_{\text{train}} = \{(x_i, y_i) | x_i \in \mathcal{X}, y_i = \varphi(x_i) \in \mathcal{Y}, i = 1, 2, \cdots, n\}$ and $D_{\text{test}} = \{\ x_j |\ x_j \in \mathcal{X},\ j = n+1, n+2, \cdots, n+m\ \}$ be the training and test data sets, respectively.

2.1 Classification Problem Can Be Solved Based on Similarity

Researches [14, 15] on cognitive science prove that similarity is the basis of concept representation. However, the relationship between similarity and classification (e.g. whether all classification problems can be solved by using similarity) still lacks a mathematical justification. This section will accomplish it. The mathematical tool used in this section is binary relation in set theory. See **Appendix** A.1.2 for a brief introduction.

First, the formal definition of classification problem is given.

Definition 1 *For an input space $\mathcal{X}$, an output space $\mathcal{Y} = \{c_1, c_2, \cdots, c_k\}$, and an unknown target function $\varphi : \mathcal{X} \to \mathcal{Y}$. The $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem is defined as: $\forall x \in \mathcal{X}$, finding $\varphi(x)$.*

It should be pointed out that the **Definition** 1 is so general that almost all classification problems studied in machine learning can be represented by it.

Definition 2 *Given a $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem, (1) let $\mathcal{X} \times \mathcal{X}$ be the new input space, (2) let $\{0, 1\}$ be the new output space, (3) let $\varphi^\dagger : \mathcal{X} \times \mathcal{X} \to \{0, 1\}$ be the new target function, where $\forall (x_i, x_j) \in \mathcal{X} \times \mathcal{X}, \varphi^\dagger ((x_i, x_j)) = \mathbb{I} (\varphi (x_i) = \varphi (x_j))$, then finding the $\varphi^\dagger$ is defined as its adjoint $\left(\mathcal{X} \times \mathcal{X}, \{0, 1\}, \varphi^\dagger\right)$-classification problem.*
*Obviously, the $\varphi^\dagger$ is an equivalence relation (ER) on $\mathcal{X}$ (see **Definition** 7 in **Appendix** A.1.2) and the adjoint $\left(\mathcal{X} \times \mathcal{X}, \{0, 1\}, \varphi^\dagger\right)$-classification problem is a binary-classification problem.*

Based on the above definitions, we have the following proposition.

Proposition 1 *The $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem is equivalent to its adjoint $\left(\mathcal{X} \times \mathcal{X}, \{0, 1\}, \varphi^\dagger\right)$-classification problem, i.e. if one problem is solved, then the other problem will also be solved. (See **Appendix** A.3.1 for proof)*

The **Proposition** 1 establishes the relationship between ER and classification problem. For any $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem, we can always solve it equivalently by solving its adjoint $\left(\mathcal{X} \times \mathcal{X}, \{0, 1\}, \varphi^\dagger\right)$-classification problem. Because ER $\varphi^\dagger : \mathcal{X} \times \mathcal{X} \to \{0, 1\}$ is a special similarity on $\mathcal{X}$. Therefore, we can solve any classification problem by finding a appropriate similarity.

2.2 Capture Fuzziness of Concept

The mathematical tools used in this section are fuzzy set theory and binary fuzzy relation. See **Appendix** A.1.3 for a brief introduction.

A large number of theories and experiments in cognitive science have shown that fuzziness is an intrinsic property of concepts [13]. A short mathematical argument is as follows. For any $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem, we have $|\mathcal{X}| > |\mathcal{Y}|$. So $\exists x_i, x_j \in \mathcal{X}, x_i \neq x_j$, such that $\varphi (x_i) = \varphi (x_j)$. The membership degrees of x_i and x_j to the concept $\varphi (x_i)$ are equal even if x_i and x_j may be very different. When fuzziness of concept is introduced, the dilemma can be solved directly. Because x_i and x_j are different, their membership degrees to the concept $\varphi (x_i)$ are different. As shown in Figure 1, the membership degree of the 1st image to 'cat' is greater than the 3rd image, and the membership degree of 3rd image to 'dog' is less than the 6th image.

The **Section** 2.1 concludes that any classification problem can be solved based on the special similarity, i.e. $\forall x_i, x_j \in \mathcal{X}$, either they belong to the same concept then $\varphi^\dagger ((x_i, x_j)) = 1$, or they don't belong to the same concept then $\varphi^\dagger ((x_i, x_j)) = 0$. Obviously, this assumption is hard to satisfy in real world because it does not capture the intrinsic fuzziness of concept.

Based on the above analysis, a new mathematical tool is needed to effectively deal with the fuzziness of concept. Fortunately, an effective solution is found in the fuzzy set theory that is introduced by Lotfi A. Zadeh [20] in 1965. In it, the crisp definition of the set is extended to a fuzzy one. Specifically, in set theory, the membership degree of an object to a set is crisp, i.e. either it belongs to the set then membership degree is 1, or it does not belong to the set then the membership degree is 0. Instead, in fuzzy set theory, the membership degree of an object to a set is fuzzy, which is represented by a real number in the interval $[0, 1]$ (see **Definition** 11 in **Appendix** A.1.3 and Figure 1).

To capture the fuzziness of concept, ER in **Definition** 2 is replaced by FER (see **Definition** 16 and **Theorem** 3 in **Appendix** A.1.3), which leads to the following definition.

Definition 3 *Given a $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem, (1) let $\mathcal{X} \times \mathcal{X}$ be the new input space, (2) let $[0,1]$ be the new output space, (3) let $\varphi^{\ddagger} : \mathcal{X} \times \mathcal{X} \to [0,1]$ be the new target function, where $\varphi^{\ddagger}$ is an FER on $\mathcal{X}$, and $\forall (x_i, x_j) \in \mathcal{X} \times \mathcal{X}$, $\varphi^{\ddagger}((x_i, x_j))$ be the degree that x_i and x_j belong to the same concept. Then finding the $\varphi^{\ddagger}$ is defined as its adjoint $\left(\mathcal{X} \times \mathcal{X}, [0,1], \varphi^{\ddagger}\right)$-FER problem.*

To solve the $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem, it is more reasonable to solve the $\left(\mathcal{X} \times \mathcal{X}, [0,1], \varphi^{\ddagger}\right)$-FER problem than the $\left(\mathcal{X} \times \mathcal{X}, \{0,1\}, \varphi^{\dagger}\right)$-classification problem. Because it is fuzzy whether two samples belong to the same concept in the former. This assumption fits well with the principle that concept is fuzzy. For a learning task, solving the $\left(\mathcal{X} \times \mathcal{X}, [0,1], \varphi^{\ddagger}\right)$-FER problem is finding an FER on $\mathcal{X}$ that can approximate φ by using D_{train}. The process can be formally described as follows

$$\mathrm{f}^* = \arg\min_{\mathrm{f}} \mathcal{L}\left(D_{\text{train}}, \mathrm{f}\right) + \gamma \mathcal{R}(\mathrm{f}) \ s.t. \ \mathrm{f} \in \{\mathrm{g} | \mathrm{g} \text{ is an FER on } \mathcal{X}\}, \tag{1}$$

where $\mathcal{L}$ is the loss function that measures how well the f fits the training data set D_{train}, $\mathcal{R}$ is the regularization term, and $\gamma > 0$ is the tradeoff parameter.

2.3 Exemplar Theory Based Concept Representation

The main knowledge used in this section is concept representation theory. See **Appendix** A.1.1 for a brief introduction.

Suppose we have obtained an FER f^* (the optimal solution of formula (1)) by using D_{train}. From the perspective of cognitive science, we have constructed the basis used for representing all concepts in $\mathcal{Y}$. Next, we need to complete concept representation based on the basis.

In cognitive science, exemplar theory is about concept representation [13]. Its core idea is that human represents a concept by remembering some objects that belong to it. Therefore, we need to find some exemplars that can represent every concept in $\mathcal{Y}$. In cognitive science, which object is suitable as an exemplar of a concept is still an open question, but it is an unequivocal fact that the similarity is the key of exemplar theory [13]. In a data-driven learning task, the exemplar set of every concept can be determined by the following definition.

Definition 4 *Given a $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem and a training data set D_{train}. Let f^* be the FER found by using D_{train}. $\forall c \in \mathcal{Y}$, let $X^c_{\text{train}} = \{x_i | (x_i, y_i) \in D_{\text{train}}, y_i = c\}$. And $\forall x \in X^c_{\text{train}}$, let $\mu(x, c) = \frac{1}{|X^c_{\text{train}}|} \sum_{x_i \in X^c_{\text{train}}} \mathrm{f}^*((x, x_i))$. Then, $\forall c \in \mathcal{Y}$, the exemplar set is defined as*

$$E_c = \{x \,|\, x \in X^c_{\text{train}}, \mu(x, c) \text{ is the top}-n^c_{\text{exe}} \text{ largest value in } \{\mu(x_i, c) \,|\, x_i \in X^c_{\text{train}}\}\},$$

where n^c_{exe} is a manually specified parameter.

The above definition is intuitive. Take Figure 1 as an example. Among the 3 images of 'cat', the 1st image is more suitable as an exemplar of concept 'cat' than the other 2 images, because the 1st image is more similar to the 3 images of 'cat' than the other 2 images.

After obtaining the exemplar set of every concept, the predicting process can be described as

$$\hat{y} = \arg\max_{c \in \mathcal{Y}} \frac{1}{|E_c|} \sum_{x_i \in E_c} \mathrm{f}^*((x, x_i)), \ \forall x \in \mathcal{X}. \tag{2}$$

2.4 The Framework of FLM

Combining the **Sections** 2.1-2.3, the training and test processes of FLM are given as follows.

Algorithm 1: The training process of FLM

Input: The training data set D_{train}, the number of exemplars of every class $n^c_{\text{exe}}, \forall c \in \mathcal{Y}$.
Find f^* by solving optimization problem (1).
for $c \in \mathcal{Y}$ **do**
  Compute the exemplar set E_c of class c by **Definition** 4.
end
Output: f^* and $E_c, \forall c \in \mathcal{Y}$.

Algorithm 2: The test process of FLM

Input: The test data set D_{test}, the learned f^*, and the exemplar set $E_c, \forall c \in \mathcal{Y}$.
for $x_j \in D_{\text{test}}$ **do**
  Compute the predicting label $\hat{y}_j$ of sample x_j by formula (2).
end
Output: $(x_j, \hat{y}_j), \forall x_j \in D_{\text{test}}$.

2.5 Relationship to the Existing Methods

The existing classifiers [1] aim to learn a function $\mathrm{f}: \mathcal{X} \to \mathcal{Y}$, while FLM aims to learn an FER $\mathrm{f}_{\mathrm{FER}}: \mathcal{X} \times \mathcal{X} \to [0,1]$. Therefore, the existing classifiers and FLM solve classification problem from different perspectives. FLM is designed based on concept cognition, while most existing classifiers seldom fully consider concept cognition. There is also a large corpus fuzzy classifiers [21] that use fuzzy set theory for classification task. Compared with most existing fuzzy classifiers, FLM is more suitable for data-driven learning tasks.

The distance metric learning [22, 23, 24], the siamese network [25, 26, 27], and the relation network based methods [28, 29, 30] both aim to learn a similarity $\mathrm{s}: \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. And FLM aims to learn an FER $\mathrm{f}_{\mathrm{FER}}: \mathcal{X} \times \mathcal{X} \to [0,1]$. On the one hand, the $\mathrm{f}_{\mathrm{FER}}$ is a special similarity. So these three methods and FLM are similar in this respect. On the other hand, the $\mathrm{f}_{\mathrm{FER}}$ needs to satisfy four properties (i.e. normalization, reflexivity, symmetry, and transitivity), while the s usually cannot satisfy these properties at the same time. Therefore these methods can not fully capture the nature of classification.

The detailed analysis about the above conclusions can be found in **Appendix** A.2.

3 NN-FLM: A Neural Network Based FLM on $\mathbb{R}^d$

In this section, the most popular input space, $\mathcal{X} = \mathbb{R}^d$, is taken into account. And the goal is to instantiate optimization problem (1) and solve it. The mathematical tools used in this section are binary fuzzy relation and the transitive-closure. See **Appendix** A.1.3 for a brief introduction. And the code implementation is provided in supplementary material.

3.1 The Separability of Adjoint $\left(\mathbb{R}^d \times \mathbb{R}^d, \{0,1\}, \varphi^\dagger\right)$-Classification Problem

To solve the adjoint $\left(\mathbb{R}^d \times \mathbb{R}^d, \{0,1\}, \varphi^\dagger\right)$-classification problem described in the **Definition** 2, classifiers need to deal with pairs of samples on $\mathbb{R}^d$. One of the simplest methods is to concatenate the feature vectors of a pair of samples, i.e.

$$D_{\mathrm{train}}^{\mathrm{concat}} = \left\{\left([\mathbf{x}_i; \mathbf{x}_j], \mathbb{I}\left(y_i = y_j\right)\right) \mid \left(\mathbf{x}_i, y_i\right), \left(\mathbf{x}_j, y_j\right) \in D_{\mathrm{train}}\right\}, \tag{3}$$

where $\forall \mathbf{a}, \mathbf{b} \in \mathbb{R}^d$, $[\mathbf{a}; \mathbf{b}] \in \mathbb{R}^{2d}$ denotes the concatenation of them. Then, a pair of samples can be converted into a composite sample, which can be processed by classifiers designed for $\mathbb{R}^{2d}$. However, the binary-classification problem obtained by the above method is not linearly separable. See the following theorem.

Theorem 1 *For a $\left(\mathbb{R}^d, \mathcal{Y}, \varphi\right)$-classification problem and a training data set D_{train}, if the sample is defined as formula* (3)*, then the corresponding $\left(\mathbb{R}^{2d}, \{0,1\}, \varphi^\dagger\right)$-classification problem is not linearly separable. (see **Appendix** A.3.2 for proof)*

Therefore, FLM on $\mathbb{R}^d$ must be able to handle the linear inseparability of the training set.

3.2 The Design of NN-FLM

To solve the original $\left(\mathbb{R}^d, \mathcal{Y}, \varphi\right)$-classification problem, an FER on $\mathbb{R}^d$ (see **Definition** 3) should be found. It is very difficult to directly obtain a binary fuzzy relation that satisfies transitivity. So an indirect method is adopted. First, construct a binary fuzzy relation that satisfies reflexivity and symmetry, i.e. FSR (see **Definition** 15 in **Appendix** A.1.3). Then convert the FSR into FER.

To fully implement FLM, the following requirements must be considered. (a) The linear inseparability of the problem (see **Theorem** 1). (b) The learned function must be an FER, i.e. the learned function is normalized, reflexive, symmetric, and transitive. (c) The loss function must consider the fuzziness of concept. (d) An effective optimizer must be designed, and it should be efficient especially for large scale training data sets.

Based on the above analysis, NN-FLM includes three components: (1) Fuzzy similarity relation network, which responds to (a) and the first 3 conditions in (b), (2) Fuzziness permissible loss, which responds to (c) and indirectly satisfies the transitivity in (b), and (3) Stochastic gradient descent based optimizer, which responds to (d). An overall design of NN-FLM is shown in Figure 2.

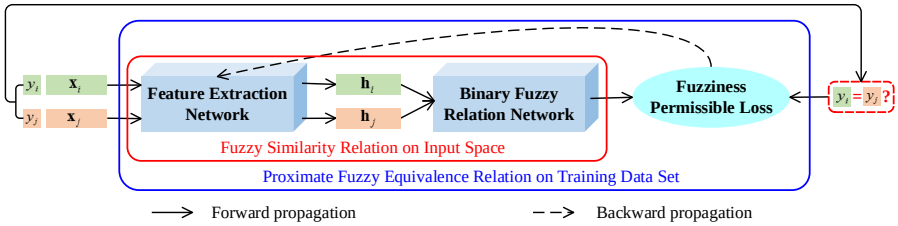

Figure 2: The overall design of NN-FLM.

(1) Fuzzy similarity relation network contains two parts: feature extraction network and binary fuzzy relation network.

First, the feature extraction network can be formally described as follows

$$\forall \mathbf{x} \in \mathbb{R}^d, \mathrm{h}\left(\mathbf{x}; \Theta\right) \in \mathbb{R}_+^{d_h}, \tag{4}$$

where $\mathbb{R}_+$ is the nonnegative real numbers, d_h is the dimension of the latent space, and Θ is the set of learnable parameters. A multi-layer neural network with nonlinear activation function is recommended, to deal with the linear inseparability of the problem.

Second, the cosine similarity is used as the skeleton of the binary fuzzy relation network, i.e. $\forall \mathbf{h}_i, \mathbf{h}_j \in \mathbb{R}_+^{d_h}, \mathrm{g}\left(\mathbf{h}_i, \mathbf{h}_j\right) = \frac{\mathbf{h}_i^T \mathbf{h}_j}{\|\mathbf{h}_i\|_2 \|\mathbf{h}_j\|_2}$. Let $\mathrm{f}\left(\left(\mathbf{x}_i, \mathbf{x}_j\right); \Theta\right) = \mathrm{g}\left(\mathrm{h}\left(\mathbf{x}_i; \Theta\right), \mathrm{h}\left(\mathbf{x}_j; \Theta\right)\right), \forall \mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^d$, be the composite of h and g. Obviously, f is an FSR on $\mathbb{R}^d$, i.e. f is normalized, reflexive, and symmetric.

It is very difficult to make f satisfy transitivity on $\mathbb{R}^d$. However, it is easy to make f satisfy transitivity on the training sample set $X_{\text{train}} = \{\mathbf{x}_i | \left(\mathbf{x}_i, y_i\right) \in D_{\text{train}}\}$. Let $\mathbf{S} \in [0,1]^{n \times n}, s_{ij} = \mathrm{f}\left(\left(\mathbf{x}_i, \mathbf{x}_j\right)\right), \forall i, j = 1, 2, \cdots, n$, be the FSR matrix on X_{train} (see **Definition** 14 in **Appendix** A.1.3). And let

$$\mathbf{T} = \mathrm{t}_{\text{FSR2FER}}\left(\mathbf{S}\right) \tag{5}$$

be the transitive-closure of $\mathbf{S}$ (see **Definition** 20, **Theorem** 4 and 5 in **Appendix** A.1.3). Then, $\mathbf{T} \in [0,1]^{n \times n}$ is an FER matrix on X_{train}.

(2) Fuzziness permissible loss For a pair of training samples $\left(\mathbf{x}_i, y_i\right), \left(\mathbf{x}_j, y_j\right) \in D_{\text{train}}$, let t_{ij} be the predicting FER value of $\left(\mathbf{x}_i, \mathbf{x}_j\right)$ by formula (5). The (α, β)-insensitive loss is used to measure the difference between the predicting value and the true value and can be written as

$$\mathcal{L}_{\alpha,\beta}\left(t_{ij}, y_i, y_j\right) = \begin{cases} \max\left\{t_{ij} - \alpha, 0\right\}, & y_i \neq y_j \\ \max\left\{\beta - t_{ij}, 0\right\}, & y_i = y_j \end{cases}, \tag{6}$$

where $0 \leq \alpha < 0.5 < \beta \leq 1$ are two hyper-parameters. The gap $\beta - \alpha \in (0, 1]$ is used for controlling the fuzziness degree of concept. And the greater the value is, the less the fuzziness will be. Therefore, the loss on the entire training data set is

$$\mathcal{L}_{\text{FER}} = \sum_{i=1}^{n} \sum_{j=1}^{n} \mathcal{L}_{\alpha,\beta}\left(t_{ij}, y_i, y_j\right). \tag{7}$$

However, the time complexity of converting the FSR matrix $\mathbf{S}$ into the FER matrix $\mathbf{T}$ is $O\left(n^3 \log_2 n\right)$ (see **Remark** in **Appendix** A.1.3). In practice, it is not feasible to directly optimize the above loss, especially when the number of training samples n is large. Fortunately, due to the intrinsic property of the transitive-closure, we can indirectly control the loss (7) by optimizing the following loss

$$\mathcal{L}_{\text{FSR}} = \sum_{i=1}^{n} \sum_{j=1}^{n} \mathcal{L}_{\alpha,\beta}\left(s_{ij}, y_i, y_j\right). \tag{8}$$

See the following theorem.

Theorem 2 *Given a training set* $D_{\text{train}} = \left\{\left(\boldsymbol{x}_i, y_i\right) | \boldsymbol{x}_i \in \mathbb{R}^d, y_i = \varphi\left(\boldsymbol{x}_i\right), i = 1, 2, \cdots, n\right\}$ *and an FSR* f *on* $X_{\text{train}} = \{\boldsymbol{x}_i | \left(\boldsymbol{x}_i, y_i\right) \in D_{\text{train}}\}$. *Let* $\boldsymbol{S} \in [0,1]^{n \times n}$, $s_{ij} = \mathrm{f}\left(\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right)\right), \forall i, j = 1, 2, \cdots, n$, *be the FSR matrix on* X_{train}, *and* $\boldsymbol{T} = \mathrm{t}_{\text{FSR2FER}}(\boldsymbol{S})$ *be the transitive-closure of* $\boldsymbol{S}$. $\forall\ 0 \leq \alpha < 0.5 < \beta \leq 1$, *if* $\sum_{i=1}^{n} \sum_{j=1}^{n} \mathcal{L}_{\alpha,\beta}\left(s_{ij}, y_i, y_j\right) = 0$, *then* $\sum_{i=1}^{n} \sum_{j=1}^{n} \mathcal{L}_{\alpha,\beta}\left(t_{ij}, y_i, y_j\right) = 0$. *(see* ***Appendix*** *A.3.3 for proof)*

According to the **Theorem** 2, when the loss (8) reaches its minimum 0, the loss (7) also reaches its minimum 0. The loss (8) can be optimized without calculating the FER matrix on all training samples. Therefore, the loss (8) is used to complete the learning process, in this paper.

(3) Stochastic gradient descent based optimizer To sum up, the learning process of NN-FLM can be written as

$$\min_{\Theta} \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \mathcal{L}_{\alpha,\beta} \left(\mathrm{f} \left(\left(\mathbf{x}_i, \mathbf{x}_j \right); \Theta \right), y_i, y_j \right) + \gamma \mathcal{R}(\Theta). \tag{9}$$

The time complexity of calculating the entire FSR matrix $\mathbf{S}$ is at least $O\left(n^2\right)$, which still has a large computational burden. Fortunately, to optimize the above loss, it is not necessary to calculate the entire matrix $\mathbf{S}$ at the same time. So the above optimization problem can be solved efficiently using the stochastic gradient descent with mini-batch, even when n is large.

4 Experiments

4.1 An Example on MNIST Data Set

Experimental settings In this section, the MNIST data set [31] is chosen to demonstrate the working mechanism of NN-FLM. In NN-FLM, a 5-layer convolutional neural network is used as the feature extraction network, and the fuzzy parameters are fixed as $\alpha = 0.2$, $\beta = 0.8$. See **Appendix** A.4.1 for more experimental details.

Interpretability analysis First, Figure 3a shows the FSR matrix predicted by NN-FLM on 10,000 test samples. The following observations can be seen from it. (1) The FSR value between samples from the same classes is significantly larger than that of samples from different classes. (2) The FSR matrix exhibits an obvious block-diagonal structure, where each block corresponds to a class. Combining (1) and (2), we can conclude that NN-FLM can learn a high-quality FSR from the training data. Based on the learned FSR, the concepts of 10 visual digits can be accurately represented. Second, Figure 3b shows the exemplars selected from the training set according to **Definition** 4. The following observations can be seen from it. (1) The 5 exemplars of every concept are regular, with no interrupted strokes, no distortions, well-proportioned parts of the digits, and the digits are located in the center of the image. It shows that the exemplars of the every concept selected by NN-FLM can accurately capture the characteristics of every concept. (2) There are certain changes among the 5 exemplars of every concept, which ensures that the exemplars can cover the intra-concept variations.

In summary, we can conclude that NN-FLM can learn high-quality FSR from training data. And then based on learned FSR, the high-quality exemplars can be selected to accurately represent the corresponding concept.

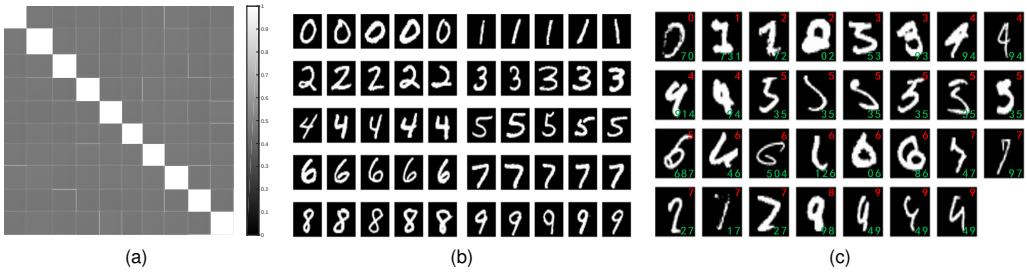

Figure 3: (a) The predicting FSR matrix on the test set (the samples are sorted by class labels). (b) The exemplars of 10 classes selected from the training set. (c) The 31 errors of NN-FLM, with annotated label (up right) and top-2 or top-3 predicting labels (down right).

Robustness analysis A total of 31 test samples are misclassified and they are shown in Figure 3c. The following conclusions can be drawn from it. (1) All the misclassified samples are irregular digital images. (2) Among them, 26 images (3 images) are hit by the 2nd (3rd) class label predicted by NN-FLM. (3) For the 2 images (1st and 3rd in row 3 of Figure 3c), none of the top-3 class labels predicted by NN-FLM hits the annotated label. These 31 images are so irregular that it is difficult

to give crisp and uncontroversial labels even for humans. In this case, the class labels predicted by NN-FLM are in line with human cognition to a certain degree.

In summary, we can conclude that NN-FLM can adequately capture fuzziness of concept. Therefore, the samples with a high degree of fuzziness can be identified and assigned reasonable candidate concepts by NN-FLM. This verifies that NN-FLM is robust to the controversial samples.

4.2 Comparison With 179 Classifiers on 121 Data Sets

Experimental settings In the literature [1], 179 classifiers from 17 families (see Table 3 in **Appendix** A.4.2) are systematically compared on 121 benchmark data sets (see Table 2 in **Appendix** A.4.2). In order to verify the generalization performance of FLM, NN-FLM is chosen and compared with the above 179 classifiers. In all experiments, NN-FLM adopts a 3-layer fully connected network as the feature extraction network. The fuzzy parameters are fixed as $\alpha = 0.2, \beta = 0.8$. More details about the experimental settings can been found in **Appendix** A.4.2.

Table 1: The rank of NN-FLM among 180 classifiers on 121 data sets

ID	Rank	ID	Rank	ID	Rank	ID	Rank	ID	Rank	ID	Rank
1	1	21	1	41	1	61	2	81	10	101	21
2*	1	22	1	42	1	62*	2	82	10.5	102	23
3	1	23	1	43	1	63	2.5	83	11	103*	23.5
4	1	24	1	44	1.5	64*	2.5	84	12	104	24.5
5	1	25	1	45	1.5	65	3	85	13	105	26
6	1	26	1	46	1.5	66	3	86	13.5	106	26
7	1	27	1	47	1.5	67	3	87	14	107	29
8	1	28	1	48*	1.5	68	3	88*	14.5	108	31.5
9	1	29	1	49*	1.5	69	3.5	89	15	109	32
10	1	30	1	50	1.5	70*	4	90	15	110	35
11	1	31	1	51	1.5	71	4.5	91*	15	111	36
12	1	32	1	52	1.5	72	5	92	15.5	112	38
13	1	33	1	53	1.5	73	5.5	93	15.5	113	39
14	1	34	1	54*	2	74	6	94	16	114	39
15	1	35	1	55	2	75	6.5	95	16	115	39
16	1	36*	1	56	2	76	7.5	96	16.5	116	42
17	1	37	1	57	2	77	8	97	17.5	117	47
18	1	38	1	58	2	78	9	98	18.5	118	60
19	1	39	1	59*	2	79	9	99	19	119	62
20	1	40	1	60	2	80	10	100	20	120	63.5
										121	65

(a) ID: The unique ID of every data set (see Table 2 in **Appendix** A.4.2). (b) $*$: The 12 small sample data sets (see Table 5 in **Appendix** A.4.2). (c) The rank considers that two or more classifiers have the same test accuracy. For example, on data set 121 there are 129 classifiers with 100% test accuracy, so the ranks of these 129 classifiers are $\left(\sum_{r=1}^{129} r\right)/129 = 65$.

The top-10 classifiers The 10 classifiers with the smallest mean rank among 180 classifiers on 121 data sets are selected for comparison analysis. They are NN-FLM (1st), 3 classifiers from the random forest family (2nd parRF-t, 3rd rf-t, 5th rforest-R), 3 classifiers from the support vector machine family (4th svm-C, 6th svmPoly-t, 8th svmRadCost-t), 1 classifier from the decision tree family (7th C5.0-t), and 2 classifiers from the neural network family (9th elm-kernel-m, 10th avNNet-t).

Accuracy analysis The test accuracy of the 179 comparison methods are from the literature [1]. We run NN-FLM on 121 data sets and record the test accuracy. Then the accuracy of 180 classifiers are used for evaluating their performance. The following conclusions can be drawn from it. (1) Due to space constraint, the accuracy of NN-FLM and other top-9 comparison methods are shown in Table 4 in **Appendix** A.4.2. And it can be seen that the mean accuracy on 121 data sets of NN-FLM is 5.49% higher than the 2nd parRF-t and 6.00% higher than the 3rd rf-t. Therefore, NN-FLM achieves a significant improvement compared with the 179 classifiers. (2) The rank of NN-FLM among 180

methods on 121 data sets is recorded in Table 1. And it can be seen that NN-FLM ranks first on 43 data sets (data sets 1-43) and ties for first place on 10 data sets (data sets 44-53), and ranks in the top-3 on more than half of the data sets (data sets 1-68). Therefore, NN-FLM achieves outstanding rank among the the 180 classifiers.

Comparative advantage analysis Figure 4a shows the proportion of the top-10 classifiers ranked in top-K ($K = 1, 3, 5, 10, 15, 20$) among the 121 data sets. And Figure 4b shows the mean and standard deviation of the rank of the top-10 classifiers on 121 data sets. The following conclusions can be drawn from it. (1) NN-FLM ranks the top-1 on $43/121 \approx 35.5\%$ of the data sets and the top-3 on $68/121 \approx 56.2\%$ of the data sets. And NN-FLM ranks in top-20 on more that 80% of the data sets. (2) None of the other 9 comparison classifiers ranks in top-20 on more than 50% of the data sets. (3) NN-FLM has the smallest mean rank 10.51. And the mean rank is significantly superior to the second-ranked classifier, parFR-t, 33.50. (4) NN-FLM has the smallest standard deviation 14.86. And the standard deviation is significantly lower than to the other comparison classifiers. In summary, the performance of NN-FLM is significantly superior to the comparison classifiers. At the same time, it can perform well on various tasks. In contrast, the comparison classifiers can only perform well on limited tasks. Therefore, the performance of NN-FLM is more stable than the others. The possible reason is that NN-FLM can adequately capture the nature of classification.

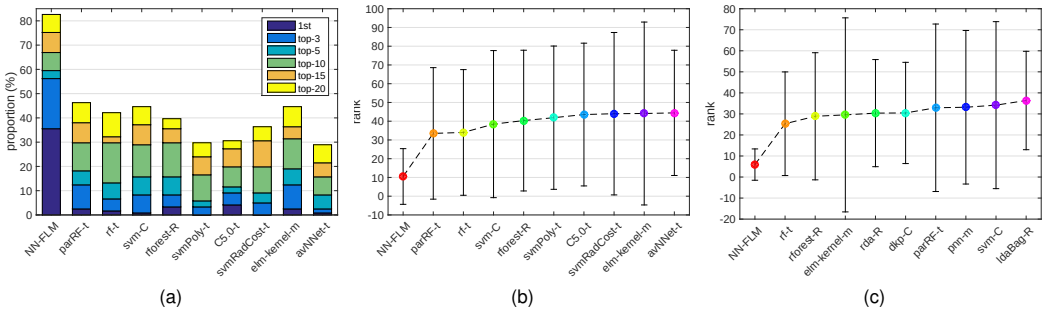

Figure 4: The statistics on top-10 methods among 180 classifiers. (a) The proportion of top-K rank on 121 data sets. (b) The mean and standard deviation of the rank on 121 data sets. (c) The mean and standard deviation of the rank on 12 small data sets.

Experiments on small sample data sets To verify the performance of NN-FLM on small sample classification problem, we selected 12 data sets with small number of samples (see Table 5 in **Appendix** A.4.2) from the 121 data sets. In these data sets, the mean number of training samples for each class is less than 15.

We selected the top-10 methods with the smallest average rank on the 12 small sample data sets for comparison, see Figure 4c. The following conclusions can be drawn from it. (1) Compared Figures 4b with 4c, it can be seen that the top-10 classifiers on the 12 small sample data sets differ from that on the all 121 data sets. Some classifiers (svmPoly-t, C5.0-t, svmRadCost-t, and avNNet-t) rank in the top-10 on the all data sets but not on the small sample data sets. (2) The mean rank of NN-FLM on the 12 small sample data sets is still the best. (3) The rank of NN-FLM is significantly superior to the second-ranked method, fr-t (5.91 ± 7.44 vs 25.33 ± 24.62). In summary, it can be concluded that when there are fewer training samples, the superiority of NN-FLM is more significant. The reason is that NN-FLM can capture the characteristic and intra-concept variations of concept through a small number of exemplars. Therefore, the exemplar theory based concept representation is reasonable.

5 Conclusion

A new learning machine based on concept cognition, fuzzy learning machine, is proposed. The new learning machine has the advantages of strong interpretability and solid mathematical foundation. And it is of certain degree of robustness because it can effectively deal with the intrinsic fuzziness in classification. At the same time, a large number of comparison experiments show that it can reach an advanced level even with a simple implementation.

Acknowledgments and Disclosure of Funding

This work is supported by the National Key Research and Development Program of China (under grant 2020AAA0106100).

References

[1] Manuel Fernández Delgado, Eva Cernadas, Senén Barro, and Dinani Gomes Amorim. Do we need hundreds of classifiers to solve real world classification problems? *Journal of Machine Learning Research*, 15(1):3133–3181, 2014.

[2] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

[3] Zhifeng Kong and Kamalika Chaudhuri. Understanding instance-based interpretability of variational auto-encoders. In *Advances in Neural Information Processing Systems, virtual*, 2021.

[4] Aya Abdelsalam Ismail, Héctor Corrada Bravo, and Soheil Feizi. Improving deep learning interpretability by saliency guided training. In *Advances in Neural Information Processing Systems, virtual*, 2021.

[5] Cheng Zhang, Kun Zhang, and Yingzhen Li. A causal view on robustness of neural networks. In *Advances in Neural Information Processing Systems, virtual*, pages 289–301, 2020.

[6] Shahd Safarani, Arne Nix, Konstantin Willeke, Santiago A. Cadena, Kelli Restivo, George H. Denfield, Andreas S. Tolias, and Fabian H. Sinz. Towards robust vision by multi-task learning on monkey visual cortex. In *Advances in Neural Information Processing Systems, virtual*, 2021.

[7] Songwei Ge, Vasu Singla, Ronen Basri, and David W. Jacobs. Shift invariance can reduce adversarial robustness. In *Advances in Neural Information Processing Systems, virtual*, 2021.

[8] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[9] Alexey Dosovitskiy and Thomas Brox. Inverting visual representations with convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA*, pages 4829–4837, 2016.

[10] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA*, pages 5188–5196, 2015.

[11] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations, Banff, AB, Canada*, 2014.

[12] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations, Vancouver, BC, Canada*, 2018.

[13] Gregory Murphy. *The Big Book of Concepts*. MIT press, 2004.

[14] Ulrike Hahn. Similarity. *Wiley Interdisciplinary Reviews: Cognitive Science*, 5(3):271–280, 2014.

[15] Heidi Kloos and Vladimir M Sloutsky. What's behind different kinds of kinds: effects of statistical density on learning and representation of categories. *Journal of Experimental Psychology: General*, 137(1):52–72, 2008.

[16] Pedro M. Domingos and Michael J. Pazzani. On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning*, 29(2-3):103–130, 1997.

[17] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Series in Representation and Reasoning. Morgan Kaufmann, 1989.

[18] Xin Geng. Label distribution learning. *IEEE Transactions on Knowledge and Data Engineering*, 28(7):1734–1748, 2016.

[19] Lotfi A. Zadeh. Is probability theory sufficient for dealing with uncertainty in AI: A negative view. In *Annual Conference on Uncertainty in Artificial Intelligence, Los Angeles, CA, USA*, pages 103–116, 1985.

[20] Lotfi Asker Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.

[21] Ludmila I. Kuncheva. *Fuzzy Classifier Design*, volume 49 of *Studies in Fuzziness and Soft Computing*. Springer, 2000.

[22] Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, and Stuart J. Russell. Distance metric learning with application to clustering with side-information. In *Advances in Neural Information Processing Systems, Vancouver, British Columbia, Canada*, pages 505–512, 2002.

[23] Dewei Li and Yingjie Tian. Survey and experimental study on metric learning methods. *Neural Networks*, 105:447–462, 2018.

[24] Mang Ye, Jianbing Shen, Gaojie Lin, Tao Xiang, Ling Shao, and Steven C.H. Hoi. Deep learning for person re-identification: A survey and outlook. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021, DOI: 10.1109/TPAMI.2021.3054775.

[25] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a siamese time delay neural network. In *Advances in Neural Information Processing Systems, Denver, Colorado, USA*, pages 737–744, 1993.

[26] Gregory Koch. Siamese neural networks for one-shot image recognition. In *International Conference on Machine Learning, Deep Learning Workshop*, 2015.

[27] Davide Chicco. Siamese neural networks: An overview. *Artificial Neural Networks*, pages 73–94, 2021.

[28] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H. S. Torr, and Timothy M. Hospedales. Learning to compare: Relation network for few-shot learning. In *IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA*, pages 1199–1208, 2018.

[29] Binyuan Hui, Pengfei Zhu, Qinghua Hu, and Qilong Wang. Self-attention relation network for few-shot learning. In *IEEE International Conference on Multimedia and Expo Workshops, Shanghai, China*, pages 198–203, 2019.

[30] Xiaoxu Li, Jintao Yan, Jijie Wu, Yuxin Liu, Xiaochen Yang, and Zhanyu Ma. Anti-noise relation network for few-shot learning. In *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, Auckland, New Zealand*, pages 1719–1724, 2020.

[31] 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.

[32] Kenneth H. Rosen. *Discrete Mathematics and Its Applications*. McGraw-Hill Education, 2018.

[33] Bernard De Baets and Etienne Kerre. Fuzzy relations and applications. In *Advances in Electronics and Electron Physics*, volume 89, pages 255–324. Elsevier, 1994.

[34] Ling Zhang and Bo Zhang. *Quotient Space Based Problem Solving: A Theoretical Foundation of Granular Computing*. Tsinghua University Press, 2014.

[35] W. Bandler and L. J. Kohout. Special properties, closures and interiors of crisp and fuzzy relations. *Fuzzy Sets and Systems*, 26(3):317–331, 1988.

[36] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[37] J. Ross Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[38] Yann LeCun, Yoshua Bengio, and Geoffrey E. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[39] Harry Zhang. The optimality of naive bayes. In *International Florida Artificial Intelligence Research Society Conference, Miami Beach, Florida, USA*, pages 562–567, 2004.

[40] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[41] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[42] Ryan M. Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *Journal of Machine Learning Research*, 5:101–141, 2004.

[43] Thomas M. Cover and Peter E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

[44] Siyuan Shan, Yang Li, and Junier B. Oliva. Meta-neighborhoods. In *Advances in Neural Information Processing Systems, virtual*, pages 5047–5057, 2020.

[45] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations, San Diego, CA, USA*, 2015.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [No]
   (c) Did you discuss any potential negative societal impacts of your work? [No]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See **Appendix** A.3.
   (b) Did you include complete proofs of all theoretical results? [Yes] See **Appendix** A.3.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The code of the proposed method is provided in supplemental material. And the data sets can be download from the links in **Appendix** A.4.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See **Appendix** A.4.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [N/A]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] The code of the proposed method is provided in the supplemental material.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Appendix

A.1 Preliminaries

A.1.1 Preliminaries of Cognitive Science

Theories in cognitive science about how concepts are represented includes classical theory, prototype theory, exemplar theory and knowledge theory [13].

(1) Classical theory The classical theory can traced back to Aristotle. In it, any concept can be precisely defined.

However, modern cognitive science has shown that concepts are almost impossible to define precisely in the real world.

(2) Prototype theory In prototype theory, a concept can be represented by a prototype, but how to give a prototype corresponding to a concept is still a very difficult problem.

(3) Exemplar theory Unlike classical theory and prototype theory, exemplar theory argues that no definition or prototype can fully encompass the entire concept. Conversely, the representation of a concept in the human mind is the set of objects in human memory that belong to the concept.

(4) Knowledge theory Knowledge theory holds that concepts are part of our common sense knowledge about the world. Human are unable to learn a concept in isolation from anything else. On the contrary, the process by which human learn concepts is part of all our understanding of the world around us.

From the perspective of cognitive science, knowledge theory seems to be the most complete, and a large number of cognition experimental results can support it. However, knowledge theory is difficult to implement for designing a purely data-driven learning machine. Therefore, exemplar theory is chosen to solve the problem of concept representation in this paper. The reason is that it not only has good self-consistency in cognitive science, but also it can be naturally integrated with the data-driven machine learning paradigm.

(5) Concept is fuzzy. The number of objects in the real world is far greater than the number of concepts, so a small number of concepts can never map perfectly onto all objects. At the same time, objects belonging to the same concept may have certain differences, and objects belonging to different concepts may also have certain similarities. Therefore, concept must be of fuzziness. See page 21 of literature [13] for a more detailed discussion.

A.1.2 Preliminaries of Binary Relation

Definition 5 ([32]) *Given two sets A, B, a map $\mathrm{R} : A \times B \to \{0, 1\}$ is called a binary relation from A to B.*

Definition 6 ([32]) *Given a set A, a map $\mathrm{R} : A \times A \to \{0, 1\}$ is called a binary relation on A.*

Definition 7 ([32]) *Given a set A and a binary relation R on A, if R satisfies the following three properties:*
(1) Reflexivity: $\forall a \in A, \mathrm{R}((a, a)) = 1$,
(2) Symmetry: $\forall a, b \in A$, if $\mathrm{R}((a, b)) = 1$, then $\mathrm{R}((b, a)) = 1$,
(3) Transitivity: $\forall a, b, c \in A$, if $\mathrm{R}((a, b)) = 1$ and $\mathrm{R}((b, c)) = 1$, then $\mathrm{R}((a, c)) = 1$,
then R is called a equivalence relation (ER) on A.

Remark: To unify the description of the full paper, the above definitions are given in a different form from the literature [32], but they are equivalent mathematically.

Definition 8 ([32]) *Given two sets A and $P = \{P_1, P_2, \cdots, P_m\}, m \geq 1$, if P satisfies the following three properties:*
(1) $\forall P_i \in P$, $P_i \neq \phi$,
(2) $\cup_{P_i \in P} = A$,
(3) $\forall P_i, P_j \in P$, if $i \neq j$, then $P_i \cap P_j = \phi$,
then P is called a partition on A.

Definition 9 ([32]) *Given a set A and a equivalence relation R on A. $\forall a \in A$, $[a]_{\mathrm{R}} = \{b | b \in A, \mathrm{R}((a,b)) = 1\}$ is called the equivalence class of a derived by R.*

Definition 10 *Given a set A and a equivalence relation R on A. Obviously, the set $A/\mathrm{R} = \{[a]_{\mathrm{R}} | a \in A\}$ is a partition on A, and it is called the partition on A derived by R.*

A.1.3 Preliminaries of Fuzzy Set

Definition 11 ([33]) *Given a set A, a map $\mu : A \to [0,1]$ is called a fuzzy set on A. And $\forall a \in A$, the $\mu(a)$ is called the membership degree of a to the fuzzy set μ.*

Definition 12 ([33]) *Given two sets A, B, a map $\mathrm{F} : A \times B \to [0,1]$ is called a binary fuzzy relation from A to B.*

Definition 13 ([33]) *Given a set A, a map $\mathrm{F} : A \times A \to [0,1]$ is called a binary fuzzy relation on A.*

Definition 14 *Given a finite set A and a binary fuzzy relation F on A. Without loss of generality, let $A = \{a_1, a_2, \cdots, a_{|A|}\}$, then the F can be described equivalently as a binary fuzzy relation matrix: $\boldsymbol{F} \in [0,1]^{|A| \times |A|}$, $f_{ij} = \mathrm{F}((a_i, a_j)), \forall i, j = 1, 2, \cdots, |A|$.*

Obviously, the binary fuzzy relation F on A is a fuzzy set on A's Cartesian product $A \times A$. As can be seen from the definition, the binary fuzzy relation is a extension of the binary relation described in **Definition** 6.

Definition 15 ([33]) *Given a set A and a binary fuzzy relation F on A. If the F satisfies*
(1) Reflexivity: $\forall a \in A, \mathrm{F}((a,a)) = 1$,
(2) Symmetry: $\forall a, b \in A, \mathrm{F}((a,b)) = \mathrm{F}((b,a))$,
then it's called a fuzzy similarity relation (FSR) on A.

Definition 16 ([33]) *Given a set A and a fuzzy similarity relation F on A. If the F satisfies*
Transitivity: $\forall a, b \in A, \mathrm{F}((a,b)) \geq \max_{c \in A} \min[\mathrm{F}((a,c)), \mathrm{F}((c,b))]$,
then it's called an fuzzy equivalence relation (FER) on A.

As can be seen from the **Definition** 16, the fuzzy equivalence relation (FER) and the equivalence relation (ER) described in the **Definition** 7 are highly correlated. The following theorem clearly shows the relationship between them.

Definition 17 ([34]) *Given a set A and a binary fuzzy relation F on A. $\forall \lambda \in [0,1]$, the binary relation*

$$\forall a, b \in A,\ \mathrm{F}^{[\lambda]}((a,b)) = \mathbb{I}[\mathrm{F}((a,b)) \geq \lambda]$$

is called the λ-cut relation of F.

Theorem 3 *Given a set A and a fuzzy equivalence relation F on A. $\forall \lambda \in [0,1]$, the λ-cut relation $\mathrm{F}^{[\lambda]}$ is an equivalence relation on A. (The proof process is straightforward and omitted.)*

Definition 18 ([20]) *Given a set A and two binary fuzzy relations F and G on A. The product (composition) binary fuzzy relation of them $\mathrm{H} = \mathrm{F} \otimes \mathrm{G}$ is defined as*

$$\forall a, b \in A,\ \mathrm{H}((a,b)) = \max_{c \in A} \min[\mathrm{F}((a,c)), \mathrm{G}((c,b))].$$

Definition 19 *Based on the **Definition** 18, given a finite set A and a binary fuzzy relation F on A, the power of F can be written as $\mathrm{F}^1 = \mathrm{F}$, $\mathrm{F}^2 = \mathrm{F} \otimes \mathrm{F}$, $\mathrm{F}^3 = \mathrm{F}^2 \otimes \mathrm{F}$, $\cdots$.*
*Let $\boldsymbol{F} \in [0,1]^{|A| \times |A|}$ be the binary fuzzy relation matrix of F (see **Definition** 14), then the power of fuzzy relation matrix $\boldsymbol{F}$ can be written as $\boldsymbol{F}^1 = \boldsymbol{F}$, $\boldsymbol{F}^2 = \boldsymbol{F} \otimes \boldsymbol{F}$, $\boldsymbol{F}^3 = \boldsymbol{F}^2 \otimes \boldsymbol{F}$, $\cdots$.*

Definition 20 *Given a finite set A, and a binary fuzzy relation F on A. G is the transitive-closure of F if and only if G satisfies*
(1) G is transitive,
(2) $\mathrm{F} \subseteq \mathrm{G}$, i.e. $\forall a, b \in A$, $\mathrm{F}((a,b)) \leq \mathrm{G}((a,b))$,
(3) if $\mathrm{F} \subseteq \mathrm{H}$ and H is transitive, then $\mathrm{G} \subseteq \mathrm{H}$.

In **Definition** 5.1 of reference [35], a more general definition, P-closure, was given, where P can be one of many properties.

Theorem 4 *Given a set A and a binary fuzzy relation F on A. The transitive-closure of F exists and is unique. (It is a direct corollary to the **Theorem** 5.3 of literature [35].)*

Theorem 5 *Given a finite set A, and a fuzzy similarity relation S on A. The transitive-closure of S can be obtained by*
*(1) Compute S^{2^0}, S^{2^1}, S^{2^2}, $\cdots$, by **Definition** 19, until $\mathrm{S}^{2^{k-1}} = \mathrm{S}^{2^k}$;*
(2) Then $\mathrm{S}^{2^{k-1}}$ is the transitive-closure of S and $2^{k-1} \leq |A|$.
The above process converts a Fuzzy Similarity Relation into a Fuzzy Equivalence Relation and is abbreviated as $\mathrm{T} = \mathrm{t}_{\mathrm{FSR2FER}}(\mathrm{S})$.
*(It is a direct corollary to the **Theorem** 8.2 of literature [35].)*

Remark: Given a finite set A, and a binary fuzzy relation F on it. According to **Definition** 18, the time complexity on calculating $\mathrm{F} \otimes \mathrm{F}$ is $O\left(|A|^3\right)$. So the time complexity of the $\mathrm{t}_{\mathrm{FSR2FER}}(\mathrm{F})$ is $O\left(|A|^3 \log_2 |A|\right)$ according to **Theorem** 5.

A.2 Related Work

A.2.1 Relationship to Existing Classifiers

So far, dozens of different classifiers have been designed. The literature [1] systematically summarizes existing classifiers and divides them into 17 families. The 17 families are: (1) Discriminant analysis, (2) Bayesian, (3) Neural networks, (4) Support vector machines, (5) Decision trees, (6) Rule-based classifiers, (7) Boosting, (8) Bagging, (9) Stacking, (10) Random forests, (11) Other ensembles, (12) Generalized linear models, (13) Nearest neighbors, (14) Partial least squares and principal component regression, (15) Logistic and multinomial regression, (16) Multiple adaptive regression splines, and (17) Other methods, which contains almost all existing classifiers.

The goal of the existing classifiers is to learn a function $\mathrm{f} : \mathcal{X} \to \mathcal{Y}$, which is an estimation of the posterior probability $\mathrm{p}(y|x), \forall x \in \mathcal{X}, y \in \mathcal{Y}$ from the point of view of probability theory. According to how to estimate the posterior probability, the existing classifiers can be divided as: discriminative methods, generative methods, and ensemble learning methods.

The discriminative methods model the $\mathrm{p}(y|x)$ directly and the typical examples include Support Vector Machine [36], Decision Tree [37], Deep Neural Network [38] and so on.

The generative methods estimate the $\mathrm{p}(y|x)$ indirectly by modeling the $\mathrm{p}(y, x)$. And the typical examples include Naive Bayes [16, 39], Bayes-Network [17] and so on.

The ensemble learning methods obtain the final $\mathrm{p}(y|x)$ by integrating multiple base classifiers, where each base classifier can be discriminative method or generative method. And the typical examples include AdaBoosting [40], Random forest [41] and so on.

The relationship between existing classifiers and FLM can be discussed from the following aspects.

1. **Existing classifiers and FLM solve classification problems from different perspectives.** Given an $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem (see **Definition** 1), the goal of existing classifiers is to learn the mapping $\mathrm{f} : \mathcal{X} \to \mathcal{Y}$ to approximate the unknown target function φ. Unlike it, the goal of FLM is to learn a mapping $\mathrm{f}_{\mathrm{FER}} : \mathcal{X} \times \mathcal{X} \to \{0, 1\}$ to approximate the equivalence relation derived by the target function R_φ (see **Definition** 21). It is motivated by that concept is represented on the bias of similarity, a conclusion of cognitive science. And, it is proved that any classification problem can be solved by the this approach (see **Proposition** 1).
2. **Most of the existing classifiers and FLM deal with the fuzziness in opposite ways.** The fuzziness is an intrinsic property of concept. However, most of existing classifiers ignore or even unintentionally eliminate it. Given a $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem, the 0-1 loss, $\mathbb{I}\left(\mathrm{f}(x) = \varphi(x)\right)$, has always been regarded as the ideal infallible loss function. However, it does not tolerate the fuzziness. In order to simplify the solving process, some surrogate loss functions with elegant mathematical properties are designed, such as the exponential loss
$$\exp(-\varphi(x)\mathrm{f}(x)),$$

where $\mathcal{Y} = \{-1, +1\}, \forall x \in \mathcal{X}, \mathrm{f}(x) \in \mathbb{R}$ and the cross entropy loss

$$-\sum_{i \in \mathcal{Y}} \mathbb{I}\left(\varphi(x) = i\right) \log(\mathrm{f}(x)_i),$$

where $\mathcal{Y} = \{1, 2, \cdots\}, \forall x \in \mathcal{X}, \mathrm{f}(x) \in [0,1]^{|\mathcal{Y}|}, \sum_{i \in \mathcal{Y}} \mathrm{f}(x)_i = 1$, etc. Based on these loss functions, some classical classifiers are built, such as AdaBoosting [40], Deep Neural Network [38], etc. However, minimizing these losses indirectly removes the fuzziness. In contrast, in NN-FLM, a fuzziness permissible loss function (see formula (6)) is designed, and the fuzziness is deliberately preserved during the learning process.

3. **Existing classifiers can be embedded into FLM.** FLM is a general framework and almost all existing classifiers can be embedded into it. It enables existing classifiers designed for binary-classification problem (e.g. Support Vector Machine [36]) to be directly used for multi-class classification problems without any modification (see **Proposition** 1). Different from the conventional strategies for converting multi-class classification problems into a series of binary-classification problems such as 'one-versus-one' and 'one-versus-rest' [42], the new strategy obtains only one binary-classification problem. That is to say, only one binary classifier is needed to solve the original multi-class classification problem.

Relationship to Nearest Neighbor Classifier

Nearest Neighbor Classifier (NNC) [43] is a classical family of classifiers that predicts the class label for a test sample through finding it's nearest neighbors. To date, there are still studies on improving it [44]. The differences between the NNC and FLM mainly include:

1. **They are different in the choice of similarity (distance).** In NNC, a predefined similarity is usually used. Therefore, performance of NNC depends heavily on the selection of the similarity, and it is difficult to adaptively handle different tasks. Unlike NNC, in FLM, a special similarity (FER) is learned from the training data (see formula (1)). This allows FLM to automatically adapt to different tasks.
2. **They represent concept in different ways.** In NNC, all samples are stored to represent every concept in $\mathcal{Y}$. Unlike NNC, in FLM, every concept in $\mathcal{Y}$ is represented by the selected exemplar set (see **Definition** 4). Although in exemplar theory, how many exemplars are stored for a concept in human memory is a leftover open question. Obviously, in order to form representation of concept 'dog', human do not remember all the dogs they have ever seen. What's more, storing all the training data for prediction is not a sensible approach from the perspective of time and space complexity, especially when the number of training samples is large.

Relationship to Existing Fuzzy Classifiers

In literature [21], the fuzzy classifier (FC) is defined as a classifier that uses fuzzy sets or fuzzy logic in the course of its training or operation. FCs can deal with ambiguity effectively, has strong interpretability and can easily be fused with the knowledge of experts. According to this definition, FLM is a kind of FC. The differences between FLM and the existing FCs are discussed as follows. (In order to be clearer, we take NN-FLM as an example.)

In literature [21], the existing FCs are divided into fuzzy if-then and non if-then fuzzy classifiers.

For fuzzy if-then classifiers, fuzzy sets need to be defined on each feature to construct classification rules. The number of fuzzy sets and the membership function of each fuzzy set need to be set manually. When the semantic information of features are unknown, this step is difficult to complete effectively. In some case, although the semantic information of features are known, it is almost meaningless to define fuzzy sets on them. For example, in image classification tasks, the feature of the sample is pixel. Because a single pixel often can not express high-level semantic information, the fuzzy set defined on a single pixel is often useless for classification. At the same time, defining fuzzy sets on each feature also faces efficiency problem. Assuming that the sample is image with 256×256 pixels and only 3 fuzzy sets are defined on each pixels. Then there are $3 \times 256 \times 256$ fuzzy sets to be processed and $(256 \times 256)^3$ rules will be generated, which brings a huge computational burden to training and test of classifiers.

Different from fuzzy if-then classifiers, firstly, the design of NN-FLM does not rely semantic information of features. Secondly, thanks to the excellent feature extraction ability of deep neural

network, NN-FLM can effectively extract useful features for classification from low-level semantic features. Thirdly, by skillfully designing the optimization model, NN-FLM can effectively complete training and prediction. Therefore, NN-FLM is suitable for learning from large-scale data.

Non if-then fuzzy classifiers mainly include fuzzy k-nearest neighbor (FKNN) and fuzzy-prototype (FPC) classifiers. FKNN is an extension of KNN and uses the distances to the neighbors as well as their soft labels for predicting. Unlike FKNN, FPC uses the nearest class prototype instead of the nearest training sample to predict. Among these methods, the distance between samples, the construction of class prototype, and the soft labels all directly affect the performance. All these depend on manual construction, so it is difficult to give a reasonable design when there is a lack of domain experts.

Unlike FKNN and FPC, NN-FLM takes similarity learning as the core goal. With the excellent feature extraction ability and well-designed loss, the adaptive similarity learning is realized. Meanwhile, the exemplar selection method can automatically select samples from training set without the participation of human experts.

In conclusion, the NN-FLM is more suitable for data-driven learning tasks.

A.2.2 Relationship to Distance Metric Learning

The goal of the distance metric learning [22, 23, 24] is to learn a task specific distance function,

$$\mathrm{d} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_+, \tag{10}$$

to improve the performance of the model on the corresponding task. Unlike it, the goal of FLM is to learn an FER (see formula (1)), $\mathrm{f}_{\mathrm{FER}} : \mathbb{R}^d \times \mathbb{R}^d \to [0,1]$ to solve original $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem. In fact, let

$$\mathrm{d}_{\mathrm{FER}} : \mathbb{R}^d \times \mathbb{R}^d \to [0,1]\,,\ \mathrm{d}_{\mathrm{FER}}\left(\mathbf{x}_i, \mathbf{x}_j\right) = 1 - \mathrm{f}_{\mathrm{FER}}\left(\mathbf{x}_i, \mathbf{x}_j\right),\ \forall \mathbf{x}_i, \mathbf{x}_j \in \mathcal{X} \subseteq \mathbb{R}^d. \tag{11}$$

Then $\mathrm{d}_{\mathrm{FER}}$ can also be viewed as a distance function on $\mathbb{R}^d$.

The differences between formula (10) and (11) include:

1. The $\mathrm{d}_{\mathrm{FER}}$ is a distance metric on $\mathbb{R}^d$ (i.e. $\mathrm{d}_{\mathrm{FER}}$ satisfies the basic properties of the distance metric: nonnegativity, identity of indiscernibles, symmetry, and triangle inequality, see **Theorem** 2.3 of literature [34]) while d doesn't have to be a distance metric on $\mathbb{R}^d$. For example, the Mahalanobis-like distance, $\mathrm{d}\left(\mathbf{x}_i, \mathbf{x}_j\right) = \sqrt{\left(\mathbf{x}_i - \mathbf{x}_j\right)^T \mathbf{M}\left(\mathbf{x}_i - \mathbf{x}_j\right)}$, is commonly used, where $\mathbf{M} \in \mathbb{R}^{d \times d}$ be the learnable parameter. Usually, $\mathbf{M}$ is a positive semi-definite matrix, which will not guarantee that the d satisfies the identity of indiscernibles.
2. Further, $\mathrm{d}_{\mathrm{FER}}$ is a normalized isosceles distance (see **Definition** 2.18 and **Proposition** 2.13 of literature [34]). In addition to the basic properties of the distance metric, it also satisfies two other properties:
   (a) $\mathrm{d}_{\mathrm{FER}}$ is normalized, i.e. $\forall x_i, x_j \in \mathcal{X}, 0 \leq \mathrm{d}_{\mathrm{FER}}\left(x_i, x_j\right) \leq 1$,
   (b) $\mathrm{d}_{\mathrm{FER}}$ is isosceles, i.e. $\forall x_i, x_j, x_k \in \mathcal{X}$, the triangle composed by $\mathrm{d}_{\mathrm{FER}}\left(x_i, x_j\right)$, $\mathrm{d}_{\mathrm{FER}}\left(x_i, x_k\right)$ and $\mathrm{d}_{\mathrm{FER}}\left(x_j, x_k\right)$ is an isosceles triangle, and its congruent legs are the longest side.

   The first property is derived from the definition of the binary fuzzy relation (see **Definition** 13 in **Appendix** A.1.3), and the second one is derived from the transitivity of FER (see **Definition** 16 in **Appendix** A.1.3). These two properties are crucial for classification problem, because the target function of the $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem is transitive (see **Definition** 21). Unfortunately, distance metric learning ignores these two crucial properties, so it can not capture the nature of classification.

A.2.3 Relationship to Siamese Network

Siamese network [25, 26, 27] is a widely used neural network that computes the similarity between two objects. Siamese network consists of two identical neural networks to learning the hidden representation of an input vector. These two neural networks work parallelly in tandem and compare their outputs at the end.

Essentially, siamese network can be viewed as a special kind of distance metric learning method. Therefore, the relationship between siamese network and FLM is the same as the relationship between distance metric learning and FLM, which will not be repeated here.

A.2.4 Relationship to Relation Network Based Methods

Recently, methods based on 'relation' network [28, 29, 30] have been proposed to solve the few-shot learning problem. The differences between these methods and FLM include:

1. The methods in the above literatures are designed for few-shot learning problems, while FLM is designed for general classification problems (see **Definition** 1). FLM can directly deal with few-shot learning problems almost without adjustment.
2. In the above literatures, the term 'relation' is only an intuitive expression, and there is no clear mathematical definition. In contrast, this paper adopts binary relation in classical set theory (see **Definition** 6 in **Appendix** A.1.2) and binary fuzzy relation (see **Definition** 13 in **Appendix** A.1.3) in fuzzy set theory. The structure of classification problems is studied with these tools and the equivalence between ER and classification problems is demonstrated (see **Proposition** 1).
3. In the above literatures, the relation network can be formally written as: $\mathrm{r} : \mathcal{X} \times \mathcal{X} \to [0, 1]$. According to the theoretical system adopted in this paper, its mathematical meaning should be that r is a binary fuzzy relation on $\mathcal{X}$ (see **Definition** 13 in **Appendix** A.1.3). And there is no special requirement for r, such as reflexivity, symmetry, transitivity, etc. In contrast, the core component of FLM is $\mathrm{f}_{\mathrm{FER}} : \mathcal{X} \times \mathcal{X} \to [0, 1]$, where $\mathrm{f}_{\mathrm{FER}}$ is an FER on $\mathcal{X}$ (see **Definition** 16 in **Appendix** A.1.3).
4. In the above literatures, the relation network is only used to calculate the relation score between the support samples (or the mean of support samples) and the query samples during the training process. It ignores the relation score between the support samples and the relation score between the query samples. The supervised information contained in these sample pairs is important for training classifiers. Especially in learning tasks such as few-shot learning, there are only a very small number of labeled samples. In contrast, in FLM, the sample pair composed of any two training samples must participate in the training process (see formula (8)), so as to ensure that a high-quality FER is learned.

A.3 Proofs

A.3.1 The Proof of Proposition 1

First, two useful definitions and a lemma are given as follows.

Definition 21 *Given a $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem, the binary relation (see **Definition** 6 in Appendix A.1.2) derived by the unknown target function φ is defined as*

$$\forall x_i, x_j \in \mathcal{X}, \mathrm{R}_{\varphi}\left(\left(x_i, x_j\right)\right) = \mathbb{I}\left(\varphi\left(x_i\right) = \varphi\left(x_j\right)\right). \tag{12}$$

*Obviously, R_{φ} is a equivalence relation (ER) on $\mathcal{X}$ (see **Definition** 7 in **Appendix** A.1.2).*

Definition 22 *Given a $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem, finding the R_{φ} is defined as the $(\mathcal{X}, \mathcal{Y}, \varphi)$-ER problem.*

Lemma 1 *The $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem is equivalent to the $(\mathcal{X}, \mathcal{Y}, \varphi)$-ER problem, i.e. if one problem is solved, then the other problem will also be solved.*

Proof

The proof consists two parts.

(1) If the $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem is solved, then the $(\mathcal{X}, \mathcal{Y}, \varphi)$-ER problem will also be solved.

If the $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem is solved, then $\forall x \in \mathcal{X}$, the true class label $\varphi(x)$ is known. Then according to the formula (12), the R_{φ}, the binary relation derived by the target function φ, can be obtained directly. That is to say, the $(\mathcal{X}, \mathcal{Y}, \varphi)$-ER problem is solved.

(2) If the $(\mathcal{X}, \mathcal{Y}, \varphi)$-ER problem is solved, then the $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem will also be solved.

If the $(\mathcal{X}, \mathcal{Y}, \varphi)$-ER problem is solved, i.e. R_φ is known, then the $\mathcal{X}/\mathrm{R}_\varphi = \left\{[x]_{\mathrm{R}_\varphi} | x \in \mathcal{X}\right\} = \{\mathcal{X}_i | i = 1, 2, \cdots, |\mathcal{Y}|\}$, the partition on $\mathcal{X}$ derived by R_φ, can be obtained by **Definition** 9, 10. Then, one can obtained the target function φ: $\forall x \in \mathcal{X}$, $\varphi(x) = c_i$, if $x \in \mathcal{X}_i \in \mathcal{X}/\mathrm{R}_\varphi$. That is to say, $\forall x \in \mathcal{X}$, the $\varphi(x)$ is found, i.e. the $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem is solved.

Combining (1) and (2), the lemma is proved. □

Based on the above conclusion, the proof of the **Proposition** 1 is given as follows.

Proof

The target function $\varphi^\dagger$ of the adjoint $\left(\mathcal{X} \times \mathcal{X}, \{0, 1\}, \varphi^\dagger\right)$-classification problem is equivalent to R_φ, the binary relation derived by the target function of $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem, because of

$$\forall x_1, x_2 \in \mathcal{X},\ \mathrm{R}_\varphi\left((x_1, x_2)\right) = \mathbb{I}\left[\varphi(x_1) = \varphi(x_2)\right] = \varphi^\dagger\left((x_1, x_2)\right).$$

That is to say, the adjoint $\left(\mathcal{X} \times \mathcal{X}, \{0, 1\}, \varphi^\dagger\right)$-classification problem is equivalent to the $(\mathcal{X}, \mathcal{Y}, \varphi)$-ER problem.

According to **Lemma** 1, one can obtained that the adjoint $\left(\mathcal{X} \times \mathcal{X}, \{0, 1\}, \varphi^\dagger\right)$-classification problem is equivalent to the original $(\mathcal{X}, \mathcal{Y}, \varphi)$-classification problem. □

A.3.2 The Proof of Theorem 1

Proof

Reduction to absurdity.

For $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2) \in D_{\text{train}}$ and $y_1 \neq y_2$, let $([\mathbf{x}_1; \mathbf{x}_1], 1)$, $([\mathbf{x}_2; \mathbf{x}_2], 1)$, $([\mathbf{x}_1; \mathbf{x}_2], 0)$, and $([\mathbf{x}_2; \mathbf{x}_1], 0)$ be 4 composite labeled samples obtained by formula (3).

Assume that these four new sample are linearly separable, i.e. $\exists \mathbf{w} \in \mathbb{R}^{2d}, b \in \mathbb{R}$, such that

$$[\mathbf{w}_1; \mathbf{w}_2]^T [\mathbf{x}_1; \mathbf{x}_1] + b > 0, \tag{13a}$$

$$[\mathbf{w}_1; \mathbf{w}_2]^T [\mathbf{x}_2; \mathbf{x}_2] + b > 0, \tag{13b}$$

$$[\mathbf{w}_1; \mathbf{w}_2]^T [\mathbf{x}_1; \mathbf{x}_2] + b < 0, \tag{13c}$$

$$[\mathbf{w}_1; \mathbf{w}_2]^T [\mathbf{x}_2; \mathbf{x}_1] + b < 0, \tag{13d}$$

where $[\mathbf{w}_1; \mathbf{w}_2] = \mathbf{w}$, and $\mathbf{w}_1, \mathbf{w}_2 \in \mathbb{R}^d$.

From formula (13a) and (13b), we have

$$(\mathbf{w}_1 + \mathbf{w}_2)^T \mathbf{x}_1 + b > 0, \tag{14a}$$

$$(\mathbf{w}_1 + \mathbf{w}_2)^T \mathbf{x}_2 + b > 0. \tag{14b}$$

Further, we have

$$(\mathbf{w}_1 + \mathbf{w}_2)^T (\mathbf{x}_1 + \mathbf{x}_2) + 2b > 0. \tag{15}$$

From formula (13c) and (13d), we have

$$\mathbf{w}_1^T \mathbf{x}_1 + \mathbf{w}_2^T \mathbf{x}_2 + b < 0, \tag{16a}$$

$$\mathbf{w}_1^T \mathbf{x}_2 + \mathbf{w}_2^T \mathbf{x}_1 + b < 0. \tag{16b}$$

Further, we have

$$(\mathbf{w}_1 + \mathbf{w}_2)^T (\mathbf{x}_1 + \mathbf{x}_2) + 2b < 0. \tag{17}$$

There is a contradiction between formula (15) and (17), so the assumption is false, i.e. these 4 composite labeled samples obtained by formula (3) are not linearly separable.

So the $\left(\mathbb{R}^{2d}, \{0, 1\}, \varphi^\dagger\right)$-classification problem with the samples defined as formula (3) is not linearly separable. □

A.3.3 The Proof of Theorem 2

First, a useful definition and a lemma are given as follows.

Definition 23 *Given a finite set A and a binary fuzzy relation matrix $\boldsymbol{R}$ on it, $\forall\lambda \in [0,1]$ the tuple $\left(A, \boldsymbol{R}^{(\lambda)}\right)$ is called as the λ-strictly-cut graph, where, A is the set of the nodes of the graph and λ-strictly-cut relation matrix, $\boldsymbol{R}^{(\lambda)} \in \{0,1\}^{|A|\times|A|}$, $\forall i,j = 1,2,\cdots,|A|,\ r_{ij}^{(\lambda)} = \mathbb{I}\left(r_{ij} > \lambda\right)$, is the adjacency matrix of the graph.*
Obviously, if $\boldsymbol{R}$ is symmetric, then $\forall\lambda \in [0,1]$, the strictly-cut graph $\left(A, \boldsymbol{R}^{(\lambda)}\right)$ is undirected graph.

Lemma 2 *Given a finite set A and a fuzzy similarity relation matrix $\boldsymbol{S}$ on A, then $\forall l = 1,2,3,\cdots$, $\forall i,j = 1,2,\cdots,|A|$, and $\forall\lambda \in [0,1]$, $s_{ij}^{l} > \lambda$ if and only if $\exists$ path $\langle a_i,\cdots,a_j\rangle$ in graph $\left(A, \boldsymbol{S}^{(\lambda)}\right)$ and the length of it is l. Where s_{ij}^{l} is entry in the ith row and the jth column of matrix $\boldsymbol{S}^{l}$, and $\boldsymbol{S}^{l}$ is the the l-th power of $\boldsymbol{S}$ (see **Definition** 19).*

Proof

By mathematical induction.

(1) When $l = 1$, $\forall i,j = 1,2,\cdots,|A|$, and $\forall\lambda \in [0,1]$,

$$s_{ij}^{1} > \lambda \Leftrightarrow s_{ij} > \lambda \Leftrightarrow s_{ij}^{(\lambda)} = 1 \Leftrightarrow \exists \text{ path } \langle a_i, a_j\rangle \text{ in graph } \left(A, \mathbf{S}^{(\lambda)}\right) \text{ and the length of it is } 1.$$

(2) When $l = 2$, $\forall i,j = 1,2,\cdots,|A|$, and $\forall\lambda \in [0,1]$,

$$\begin{aligned} s_{ij}^{2} > \lambda \quad &\Leftrightarrow \max_{k=1,2,\cdots,|A|} \min\left[s_{ik}, s_{kj}\right] > \lambda \\ &\Leftrightarrow \exists\, k \in \{1,2,\cdots,|A|\}, \min\left[s_{ik}, s_{kj}\right] > \lambda \\ &\Leftrightarrow \exists\, k \in \{1,2,\cdots,|A|\}, s_{ik} > \lambda \wedge s_{kj} > \lambda \\ &\Leftrightarrow \exists \text{ path } \langle a_i, a_k, a_j\rangle \text{ in graph } \left(A, \mathbf{S}^{(\lambda)}\right) \text{ and the length of it is } 2. \end{aligned}$$

(3) When $l = L-1, L > 3$, assuming that $\forall i,j = 1,2,\cdots,|A|$, and $\forall\lambda \in [0,1]$,

$$s_{ij}^{L-1} > \lambda \Leftrightarrow \exists \text{ path } \left\langle a_i, a_{p_2}, a_{p_3}, \cdots, a_{p_{L-1}}, a_j\right\rangle \text{ in graph } \left(A, \mathbf{S}^{(\lambda)}\right) \text{ and the length of it is } L-1.$$

(4) When $l = L, L > 3$,$\forall i,j = 1,2,\cdots,|A|$, and $\forall\lambda \in [0,1]$,

$$\begin{aligned} s_{ij}^{L} > \lambda \quad &\Leftrightarrow \max_{k=1,2,\cdots,|A|} \min\left[s_{ik}^{L-1}, s_{kj}\right] > \lambda \\ &\Leftrightarrow \exists\, k \in \{k = 1,2,\cdots,|A|\}, \min\left[s_{ik}^{L-1}, s_{kj}\right] > \lambda \\ &\Leftrightarrow \exists\, k \in \{k = 1,2,\cdots,|A|\}, s_{ik}^{L-1} > \lambda \wedge s_{kj} > \lambda \\ &\Leftrightarrow \exists\, k \in \{k = 1,2,\cdots,|A|\}, \\ &\qquad \exists \text{ path } \left\langle a_i, a_{p_2}, a_{p_3}, \cdots, a_{p_{L-1}}, a_k\right\rangle \text{ in graph } \left(A, \mathbf{S}^{(\lambda)}\right) \wedge \\ &\qquad \exists \text{ edge } \langle a_k, a_j\rangle \text{ in graph } \left(A, \mathbf{S}^{(\lambda)}\right) \\ &\Leftrightarrow \exists \text{ path } \left\langle a_i, a_{p_2}, a_{p_3}, \cdots, a_{p_{L-1}}, a_k, a_j\right\rangle \text{ in graph } \left(A, \mathbf{S}^{(\lambda)}\right) \text{ and the length of it is } L. \end{aligned}$$

Combining (1)-(4), the lemma is proved. □

Based on the above conclusion, the proof of **Theorem** 2 is given as follows.

Proof

The proof consists two parts.

(1) $\forall p,q = 1,2,\cdots,n,\ y_p = y_q$,
if $\sum_{i=1}^{n}\sum_{j=1}^{n}\mathcal{L}_{\alpha,\beta}\left(s_{ij}, y_i, y_j\right) = 0$, then $\mathcal{L}_{\alpha,\beta}\left(t_{pq}, y_p, y_q\right) = 0$.

$\because \sum_{i=1}^{n}\sum_{j=1}^{n}\mathcal{L}_{\alpha,\beta}\left(s_{ij}, y_i, y_j\right) = 0$
$\therefore \mathcal{L}_{\alpha,\beta}\left(s_{pq}, y_i, y_j\right) = \max\{\beta - s_{ij}, 0\} = 0$
$\therefore s_{pq} \geq \beta$.

And
$\because$ **T** is the transitive-closure of **S**
$\therefore t_{pq} \geq s_{pq}$ (see **Definition** 20)

$\therefore t_{pq} \geq s_{pq} \geq \beta$
$\therefore \mathcal{L}_{\alpha,\beta}\left(t_{pq}, y_p, y_q\right) = \max\{\beta - t_{ij}, 0\} = 0.$

(2) $\forall p, q = 1, 2, \cdots, n,\ y_p \neq y_q,$
if $\sum_{i=1}^{n}\sum_{j=1}^{n}\mathcal{L}_{\alpha,\beta}\left(s_{ij}, y_i, y_j\right) = 0,$ then $\mathcal{L}_{\alpha,\beta}\left(t_{pq}, y_p, y_q\right) = 0.$

Reduction to absurdity.

Let $X_{\text{train}} = \{\mathbf{x}_i | \left(\mathbf{x}_i, y_i\right) \in D_{\text{train}}\}$ be the set of training samples and let $\left(X_{\text{train}}, \mathbf{S}^{(\alpha)}\right)$ be the corresponding α-strictly-cut graph.

Assume that $\exists p, q = 1, 2, \cdots, n,\ y_p \neq y_q,$ such that $\mathcal{L}_{\alpha,\beta}\left(t_{pq}, y_p, y_q\right) > 0.$

$\because \mathcal{L}_{\alpha,\beta}\left(t_{pq}, y_p, y_q\right) = \max\{t_{pq} - \alpha, 0\} > 0$
$\therefore t_{pq} > \alpha.$

According to **Theorem** 5, **T** can be written as the power of **S**. Without loss of generality, let $\mathbf{T} = \mathbf{S}^K$. Then we have $t_{pq} = s_{pq}^K > \alpha$.

According to **Lemma** 2, we have

$$\begin{aligned} s_{pq}^K > \alpha \quad &\Leftrightarrow \quad \exists \text{ path } \langle \mathbf{x}_p, \mathbf{x}_{r_2}, \mathbf{x}_{r_3}, \cdots, \mathbf{x}_{r_K}, \mathbf{x}_q \rangle \text{ with length } K \text{ in graph } \left(X_{\text{train}}, \mathbf{S}^{(\alpha)}\right) \\ &\Leftrightarrow \quad \exists\, \mathbf{x}_p, \mathbf{x}_{r_2}, \mathbf{x}_{r_3}, \cdots, \mathbf{x}_{r_K}, \mathbf{x}_q \in X_{\text{train}}, \\ &\qquad\quad \text{such that } s_{pr_2} > \alpha, s_{r_2 r_3} > \alpha, \cdots, s_{r_{K-1} r_K} > \alpha, s_{r_K q} > \alpha. \end{aligned}$$

$\because y_p \neq y_q$
$\therefore \exists \left(p^*, q^*\right) \in \{\left(p, r_2\right), \left(r_2, r_3\right), \cdots, \left(r_{K-1}, r_K\right), \left(r_K, q\right)\}$, such that $y_{p^*} \neq y_{q^*}$
$\therefore \mathcal{L}_{\alpha,\beta}\left(s_{p^*q^*}, y_{p^*}, y_{q^*}\right) = \max\{s_{p^*q^*} - \alpha, 0\} > 0$
$\therefore \sum_{i=1}^{n}\sum_{j=1}^{n}\mathcal{L}_{\alpha,\beta}\left(s_{ij}, y_i, y_j\right) \geq \mathcal{L}_{\alpha,\beta}\left(s_{p^*q^*}, y_{p^*}, y_{q^*}\right) = \max\{s_{p^*q^*} - \alpha, 0\} > 0$
It contradicts to the known condition.

Combining (1) and (2), we have

$$\begin{aligned} \sum_{i=1}^{n}\sum_{j=1}^{n}\mathcal{L}_{\alpha,\beta}\left(t_{ij}, y_i, y_j\right) &= \sum_{i=1}^{n}\sum_{j=1, y_i = y_j}^{n}\mathcal{L}_{\alpha,\beta}\left(t_{ij}, y_i, y_j\right) \\ &+ \sum_{i=1}^{n}\sum_{j=1, y_i \neq y_j}^{n}\mathcal{L}_{\alpha,\beta}\left(t_{ij}, y_i, y_j\right) \\ &= 0 + 0 \\ &= 0 \end{aligned},$$

i.e. the theorem is proved. □

A.4 Experiment Details

A.4.1 Experiment Details of Section 4.1

(1) Data set The MNIST handwritten digit data set [31], one of the most basic data sets used to test performance learning algorithm, is chosen to demonstrate how FLM works. It has a training set of 60,000 samples, and a test set of 10,000 samples, and each sample is a gray image of a handwritten digits. There are a total of 10 concepts, i.e. $0, 1, \cdots, 9$.

(2) Settings of NN-FLM

1. The feature extraction network (see formula (4)) is a 5-layer convolutional network [2]. It can be formally described as follows

$$\begin{aligned}
\mathbf{h}^{(1)} &= \mathrm{ReLU}\left(\mathrm{BN}\left(\mathrm{Conv}\left(\mathbf{x};\mathbf{K}_1\right);\mathbf{w}_1,\mathbf{b}_1\right)\right)\\
\mathbf{h}^{(2)} &= \mathrm{ReLU}\left(\mathrm{BN}\left(\mathrm{Conv}\left(\mathbf{h}^{(1)};\mathbf{K}_2\right);\mathbf{w}_2,\mathbf{b}_2\right)\right)\\
\mathbf{h}^{(3)} &= \mathrm{ReLU}\left(\mathrm{BN}\left(\mathrm{Conv}\left(\mathbf{h}^{(2)};\mathbf{K}_3\right);\mathbf{w}_3,\mathbf{b}_3\right)\right)\\
\mathbf{h}^{(4)} &= \mathrm{ReLU}\left(\mathrm{BN}\left(\mathrm{Conv}\left(\mathbf{h}^{(3)};\mathbf{K}_4\right);\mathbf{w}_4,\mathbf{b}_4\right)\right)\\
\mathbf{h}^{(5)} &= \mathrm{Softmax}\left(\mathrm{BN}\left(\mathrm{Linear}\left(\mathbf{h}^{(4)};\mathbf{K}_5\right);\mathbf{w}_5,\mathbf{b}_5\right)\right)
\end{aligned},$$

   where $\Theta = \{\ \mathbf{K}_1 \in \mathbb{R}^{48\times1\times7\times7}$, $\mathbf{w}_1 \in \mathbb{R}^{48}$, $\mathbf{b}_1 \in \mathbb{R}^{48}$, $\mathbf{K}_2 \in \mathbb{R}^{96\times48\times7\times7}$, $\mathbf{w}_2 \in \mathbb{R}^{96}$, $\mathbf{b}_2 \in \mathbb{R}^{96}$, $\mathbf{K}_3 \in \mathbb{R}^{144\times96\times7\times7}$, $\mathbf{w}_3 \in \mathbb{R}^{144}$, $\mathbf{b}_3 \in \mathbb{R}^{144}$, $\mathbf{K}_4 \in \mathbb{R}^{192\times144\times7\times7}$, $\mathbf{w}_4 \in \mathbb{R}^{192}$, $\mathbf{b}_4 \in \mathbb{R}^{192}$, $\mathbf{K}_5 \in \mathbb{R}^{10\times3072}$, $\mathbf{w}_5 \in \mathbb{R}^{10}$, $\mathbf{b}_5 \in \mathbb{R}^{10}\ \}$ is the set of the learnable parameters.

2. For the loss (8), $\alpha = 0.2$, $\beta = 0.8$ is set to control the degree of the fuzziness.

3. The number of exemplars of every class (see **Definition** 4), n_{exe}, is set to 5.

4. A regularization term $\mathcal{R}(\Theta)$ is added into the loss (8) to control the complexity of the model,

$$\mathcal{R}(\Theta) = \frac{\gamma}{n_{\mathrm{para}}}\sum_{i=1}^{5}\|\mathbf{K}_i\|_F^2 + \|\mathbf{w}_i\|_2^2 + \|\mathbf{b}_i\|_2^2,$$

   where $n_{\mathrm{para}} = (48\times1\times7\times7+48+48)+(96\times48\times7\times7+96+96)+(144\times96\times7\times7+144+144)+(192\times144\times7\times7+192+192)+(10\times3072+10+10)$ is the number of learnable parameters and $\gamma = 0.1$ is the trade-off parameter.

5. The stochastic gradient descent method Adam [45] optimizer are used to train the model. The size of the batch is set to 2048. We stopped iterating until the loss values of 10 consecutive epochs do not change significantly.

(3) Supplementary analysis of experimental results

The top-1, top-2, and top-3 accuracy of NN-FLM are 99.69%, 99.95%, and 99.98%, respectively.

It should be pointed out that the accuracy obtained in the experiment is slightly lower than the current state-of-the-ar on MNIST data[3]. On the one hand, the goal of this experiment is to illustrate the working mechanism of NN-FLM, rather than pursuing higher accuracy. In most of the state-of-the-art methods, strategies (such as designing specific neural network structure, data augmentation, and ensemble learning, etc) are used to achieve higher accuracy. These strategies can be embedded in NN-FLM for higher accuracy. On the other hand, when the accuracy reaches a certain level, pursuing higher accuracy is not only meaningless, but will lead to other problems. Suppose there is a classifier that predicts the 'correct' labels for all the images in Figure 3c. When it is applied to a field where the cost of misclassification is extremely high (e.g. the medical field), a serious consequences may be caused. In this case, it would be wiser to leave the fuzzy images to human as NN-FLM does.

A.4.2 Experiment Details of Section 4.2

(1) Data sets A total of 121 data sets are used in the experiments. In these data sets, the number of samples varies from 10 (trains) to 130,064 (miniboone), the number of features varies from 3 (haberman-survival) to 262 (arrhythmia), and the number of classes varies from 2 (a total of 54 data sets) to 100 (plant-texture, plant-margin, and plant-shape). More information about these data sets can be found in Table 1, 2 of the literature [1]. For the convenience of discussion, each data set is given a unique ID, see Table 2.

(2) Training-test split In order to ensure the reproducibility of the experiments and the fairness of the comparisons, in literature [1], all methods are subjected to the same 4-fold cross validation on every data set. This experiment also used the same training-test split on all the 121 data sets. The

[2]The network structure design comes from `https://github.com/ansh941/MnistSimpleCNN`

[3]See the leaderboard at `https://paperswithcode.com/sota/image-classification-on-mnist`

whole 121 data sets and partitions are available from: `http://persoal.citius.usc.es/manuel.fernandez.delgado/papers/jmlr/data.tar.gz`.

(3) Comparison methods and settings A total of 179 classifiers from these 17 families are used as comparison methods (see Table 3). The names and implementation details of these 179 classifiers can be found in **Section** 2.2 of literature [1].

(4) Settings of NN-FLM In this experiment, the implementation details of NN-FLM are as follows.

1. The feature extraction network (see formula (4)) is a 3-layer full connected network. It can be formally described as follows

$$\mathrm{h}(\mathbf{x};\Theta) = \mathrm{softmax}\left\{\mathrm{LeakyReLU}\left[\mathrm{LeakyReLU}\left(\mathbf{x}\mathbf{W}_1+\mathbf{b}_1\right)\mathbf{W}_2+\mathbf{b}_2\right]\mathbf{W}_3+\mathbf{b}_3\right\},$$

   where $\Theta = \left\{\mathbf{W}_1 \in \mathbb{R}^{d\times 128},\ \mathbf{b}_1 \in \mathbb{R}^{128},\ \mathbf{W}_2 \in \mathbb{R}^{128\times 64},\ \mathbf{b}_2 \in \mathbb{R}^{64},\ \mathbf{W}_3 \in \mathbb{R}^{64\times 32},\ \mathbf{b}_3 \in \mathbb{R}^{32}\ \right\}$ is the set of the learnable parameters and d is the number of the features of the data set.
2. For the loss (8), $\alpha = 0.2$, $\beta = 0.8$ is set to control the degree of the fuzziness for all the experiments.
3. The number of exemplars of every concept (see **Definition** 4), n_{exe}, is set to 5 for all the experiments. If the number of training samples in the concept is less than 5, then all the training samples are selected as exemplars.
4. For all the experiments, a regularization term $\mathcal{R}(\Theta)$ is added into the loss (8) to control the complexity of the model,

$$\mathcal{R}(\Theta) = \frac{\gamma}{n_{\mathrm{para}}}\sum_{i=1}^{3}\|\mathbf{W}_i\|_F^2 + \|\mathbf{b}_i\|_2^2,$$

   where $n_{\mathrm{para}} = d\times 128 + 128 + 128\times 64 + 64 + 64\times 32 + 32$ is the number of learnable parameters and $\gamma = 0.1$ is the trade-off parameter.
5. For all the experiments, the stochastic gradient descent method Adam [45] optimizer are used to train the model. The size of the batch is set to $\min(4096, n)$, where n is the number of the training samples. We stopped iterating until the loss values of 10 consecutive epochs do not change significantly.

(5) Experimental results The results of the 179 comparison methods on the 121 data set are available from: `http://persoal.citius.usc.es/manuel.fernandez.delgado/papers/jmlr/results.txt`.

The average accuracy of NN-FLM on 4-fold cross validations are recorded in Table 4. Limited by space, it is impractical to list the accuracy of the 179 comparison methods on 121 data sets. Therefore, the top-10 methods with the smallest average rank on the 121 data sets are picked for comparison, and their accuracies are recorded in Table 4.

Table 4: The prediction accuracy (%) of the top-10 methods among 180 methods on 121 data sets, with the rank of the method among the 10 methods (up right).

ID	**NN-FLM**	parRF-t	rf-t	svm-C	rforest-R	svm Poly-t	C5.0-t	svm Radial Cost-t	elm-kernel-m	avNNet-t
1	80.5[1]	74.9	75.3	77.3	74.2	77.4	77.9	80.1	77.5	79.5
2	78.2[1]	73.1	75.5	75.0	71.2	73.1	74.8	74.8	74.6	73.4
3	98.0[1]	96.6	97.0	96.1	97.0	96.9	96.7	95.8	96.1	96.9
4	83.2[1]	80.9	80.8	81.1	80.5	78.3	79.4	79.3	82.7	77.8
5	99.6[1]	98.3	98.6	99.0	97.5	96.1	98.7	99.2	98.1	95.3
6	57.5[1]	55.1	54.0	55.6	52.4	55.9	54.7	55.1	55.9	56.4
7	88.5[1]	87.5	86.8	86.2	88.4	86.7	86.5	85.5	86.6	87.0
8	100.0[1]	80.3	78.9	78.3	81.2	77.5	79.9	78.7	73.6	77.5
9	88.3[1]	84.7	84.0	86.4	83.3	85.5	87.1	82.5	75.0	81.0

Continuation of Table 4

ID	**NN-FLM**	parRF-t	rf-t	svm-C	rforest-R	svm Poly-t	C5.0-t	svm Radial Cost-t	elm-kernel-m	avNNet-t
10	91.0[1]	88.0	88.0	88.0	85.0	88.0	88.0	90.0	84.0	88.0
11	78.3[1]	67.7	69.6	74.0	71.4	74.2	72.9	75.2	72.7	76.1
12	65.7[1]	59.4	59.7	58.6	57.6	59.1	56.1	58.4	58.6	59.7
13	54.2[1]	41.4	35.8	35.5	41.1	44.8	40.0	45.5	42.7	43.1
14	45.0[1]	37.4	32.5	31.5	35.0	36.5	38.0	32.0	31.5	38.1
15	90.2[1]	82.6	83.2	85.9	81.4	82.5	85.7	85.2	84.0	83.9
16	82.8[1]	55.3	54.1	53.6	55.1	55.3	49.3	52.5	70.3	66.7
17	88.5[1]	28.6	28.7	28.9	28.8	28.8	30.8	28.9	21.4	82.2
18	96.0[1]	93.2	92.9	94.9	93.5	92.3	94.6	95.5	94.0	92.6
19	75.3[1]	71.7	72.8	72.4	72.2	73.6	71.3	71.8	74.5	71.7
20	94.5[1]	90.4	89.8	93.4	91.9	90.8	87.0	88.3	90.2	90.6
21	91.9[1]	61.1	61.1	51.9	61.1	55.6	72.2	64.1	66.0	78.2
22	68.5[1]	65.7	65.7	65.3	66.2	66.9	62.5	65.5	62.3	66.4
23	96.7[1]	53.7	53.7	53.2	53.7	67.6	53.9	48.6	62.5	60.4
24	94.2[1]	92.8	92.1	92.3	92.4	92.4	92.0	92.2	94.0	93.0
25	95.8[1]	92.5	92.0	95.1	92.1	93.5	93.0	94.1	94.4	94.2
26	99.0[1]	97.0	97.4	97.2	97.4	98.0	81.6	97.4	98.7	93.7
27	95.3[1]	89.2	90.8	93.4	88.8	91.3	90.8	93.3	94.4	90.3
28	99.7[1]	95.5	95.6	97.6	95.5	97.6	96.1	97.7	97.8	89.9
29	93.3[1]	84.0	84.9	79.8	84.6	86.8	86.8	85.9	88.5	86.8
30	86.0[1]	74.8	71.0	72.1	71.2	69.9	69.8	70.0	65.4	67.0
31	80.5[1]	65.1	70.9	80.4	65.2	75.0	71.8	69.6	71.7	71.7
32	94.0[1]	88.2	86.3	80.0	86.5	87.3	86.3	89.2	88.0	90.3
33	73.9[1]	71.4	72.5	72.8	66.3	71.4	71.4	71.4	72.8	71.4
34	75.0[1]	61.2	68.9	70.5	70.5	71.1	71.1	71.1	72.7	68.9
35	54.3[1]	49.4	48.5	47.3	50.3	46.1	45.8	47.3	52.7	46.1
36	93.1[1]	89.6	90.2	88.6	90.7	89.4	58.8	88.0	90.4	85.6
37	81.6[1]	55.1	55.1	55.1	55.1	55.1	57.8	55.1	58.3	59.9
38	69.5[1]	67.7	67.8	68.2	67.4	67.8	67.5	67.8	67.3	67.8
39	89.9[1]	83.0	83.0	88.1	85.7	85.6	81.1	84.1	86.9	85.6
40	85.9[1]	75.8	76.4	84.7	75.4	85.1	76.2	84.2	84.8	82.6
41	66.9[1]	62.1	62.2	48.0	59.2	56.9	57.7	55.0	47.4	60.3
42	88.3[1]	85.2	84.5	85.4	82.4	86.1	85.2	86.5	86.7	86.1
43	88.0[1]	86.4	85.1	83.8	84.6	85.5	84.2	83.9	82.5	86.8
44	79.8[1]	71.7	68.8	68.3	77.9	69.1	70.0	74.5	70.2	69.8
45	63.2[1]	62.5	62.1	60.8	61.5	63.0	61.4	62.1	61.2	62.3
46	90.4[1]	82.7	84.2	85.6	84.1	87.5	80.8	87.0	90.4[1]	81.7
47	86.6[1]	82.3	83.0	86.0	83.1	83.7	81.3	84.7	82.5	83.3
48	95.8[1]	83.3	87.3	83.3	83.3	88.1	87.5	83.3	91.7	87.5
49	75.0[1]	60.0	62.5	62.5	53.1	56.2	59.0	44.6	37.5	52.4
50	89.2[1]	87.9	85.2	87.2	87.2	85.1	85.1	85.9	87.2	81.8
51	93.7[1]	89.9	88.7	87.8	87.6	92.0	92.4	93.7[1]	87.8	92.2
52	86.8[1]	81.4	81.9	86.0	80.9	85.1	81.0	85.5	86.8[1]	85.3
53	99.7[1]	98.3	98.7	99.5	98.8	99.0	97.8	99.5	99.2	98.7
54	100.0[1]	66.7	87.5	81.2	87.5	81.2	48.3	100.0[1]	93.8	75.0
55	93.4[1]	92.1	92.3	90.2	89.5	90.5	91.7	92.2	90.8	91.4
56	83.9[1]	80.9	82.6	82.5	81.9	83.4	83.6	82.1	81.0	83.7
57	93.3[1]	88.7	90.6	86.5	92.3	85.1	86.9	81.1	86.5	82.9
58	85.7[2]	81.6	81.1	84.1	77.9	82.9	80.2	83.3	86.0[1]	84.4
59	75.0[2]	64.9	66.8	76.0[1]	72.1	58.1	62.8	60.8	64.4	62.1
60	78.5[1]	77.1	77.1	76.8	77.9	76.9	75.9	77.6	78.1	74.6

Continuation of Table 4

ID	**NN-FLM**	parRF-t	rf-t	svm-C	rforest-R	svm Poly-t	C5.0-t	svm Radial Cost-t	elm-kernel-m	avNNet-t
61	100.0[1]	98.9	98.9	98.3	98.9	98.3	98.3	98.9	100.0[1]	98.9
62	99.0[1]	98.1	96.1	98.0	95.0	94.2	95.1	92.3	96.0	93.9
63	98.1[1]	96.8	97.0	97.0	95.8	97.9	96.0	98.1[1]	96.1	97.5
64	100.0[1]	91.7	87.5	50.0	87.5	75.0	79.2	62.5	87.5	87.5
65	98.9[2]	85.3	84.8	99.0[1]	86.5	91.8	84.0	96.6	91.8	93.8
66	89.0[2]	87.2	86.6	86.3	88.1	87.5	83.6	86.3	89.6[1]	86.6
67	88.1[2]	82.5	84.7	88.6[1]	79.7	82.5	80.6	86.1	84.7	73.6
68	92.5[1]	92.0	92.0	92.0	92.0	92.0	92.0	92.0	91.4	71.7
69	97.1[1]	94.3	93.4	95.7	94.7	93.8	92.9	93.4	95.2	96.2
70	88.1[1]	12.0	52.0	64.0	52.0	52.0	8.0	48.0	76.0	60.0
71	76.4[1]	68.4	70.2	71.1	69.7	71.4	70.7	71.4	72.8	72.2
72	93.3[2]	93.8[1]	−−	−−	−−	−−	93.8[1]	−−	−−	92.2
73	99.6[2]	97.7	97.8	99.4	97.8	99.6[2]	98.7	99.5	99.8[1]	98.6
74	90.2[2]	89.9	90.0	90.0	90.5[1]	89.5	90.0	89.8	89.6	89.6
75	97.3[3]	97.8[1]	97.5[2]	95.6	93.9	87.8	96.4	96.6	91.7	90.2
76	97.9[3]	97.8	98.2[1]	96.7	97.9	96.3	98.1[2]	96.5	97.3	95.0
77	98.7[1]	96.7	96.0	96.6	96.7	98.0	95.3	94.0	98.6	97.3
78	66.3[2]	65.1	64.7	66.0	64.9	65.9	64.5	66.5[1]	66.1	67.4
79	99.8[3]	99.1	99.2	100.0[1]	99.0	99.0	98.6	99.6	99.9[2]	98.3
80	90.3[5]	91.0[3]	90.8[4]	91.9[1]	91.0[3]	89.8	89.8	90.8	91.3[2]	86.9
81	66.0[4]	69.0[1]	69.0[1]	64.0	68.9[2]	61.1	68.1[3]	63.4	61.6	61.4
82	86.2[4]	87.4[1]	87.5[2]	−−	85.2	−−	90.4[1]	−−	−−	77.8
83	94.8[5]	95.1[3]	95.0[4]	93.8	95.2[2]	93.5	95.4[1]	93.4	93.3	94.3
84	94.4[5]	94.1	93.7	95.8[4]	94.5	95.9[2]	90.5	95.5[3]	96.4[1]	83.7
85	97.2[1]	97.1	97.1	97.2[1]	97.1	97.1	97.2[1]	97.1	96.8	97.1
86	99.4[3]	99.4[3]	99.5[2]	98.9	98.6	99.5[2]	99.6[1]	99.2	99.5[2]	99.4[3]
87	77.2[5]	79.0[2]	78.6[3]	76.1	78.5[4]	76.2	80.4[1]	75.8	75.9	73.2
88	82.3[6]	86.2[2]	87.2[1]	83.1	85.3[3]	−−	74.9	−−	85.0	32.9
89	73.0[4]	77.4[1]	76.1[2]	72.3	75.2[3]	70.6	71.9	65.5	71.5	63.5
90	74.0[5]	78.0[1]	75.7[3]	68.9	77.8[2]	70.2	74.2[4]	70.6	69.3	66.3
91	81.0[5]	83.4[3]	84.5[2]	84.5[2]	83.1[4]	−−	77.0	−−	85.7[1]	31.9
92	97.8[2]	97.8[2]	97.8[2]	96.7	98.1[1]	97.5	96.7	98.1[1]	96.7	98.1[1]
93	95.3[8]	96.4	96.5	97.4[1]	96.7[3]	96.4	95.7	96.8[2]	−−	64.6
94	85.9[4]	88.1[2]	88.1[2]	84.4	86.6[3]	83.1	88.5[1]	84.3	83.6	79.7
95	78.3[1]	76.6	76.3	78.3[1]	76.4	78.0	76.4	77.2	78.1	77.1
96	97.9[2]	97.2	97.3	97.8	97.5	98.0[1]	96.8	97.8	97.9[2]	97.8
97	87.2[6]	88.0[3]	88.1[2]	87.7	88.3[1]	86.7	87.7	87.3	−−	86.9
98	93.4[4]	94.3[2]	94.1[3]	91.9	94.7[1]	92.6	94.7[1]	93.1	92.7	91.9
99	95.1[4]	94.0	94.0	97.0[2]	96.0[3]	97.0[2]	89.0	96.0[3]	97.0[2]	98.0[1]
100	86.7[3]	85.7	85.6	86.5	84.9	87.1[1]	84.2	86.8[2]	86.6	87.1[1]
101	99.5[4]	99.7[3]	99.8[2]	99.7[3]	98.1	−−	100.0[1]	−−	99.4	93.8
102	98.1[3]	96.6	96.7	98.6[1]	95.8	98.1	97.0	98.6[1]	98.5[2]	90.2
103	62.2[6]	63.9	64.1	72.2[1]	64.9[3]	−−	57.8	−−	71.7[2]	31.1
104	85.1[4]	86.1[2]	86.2[1]	85.0	86.1[2]	84.9	85.6[3]	85.0	−−	85.1
105	86.3[4]	92.9[1]	85.7[3]	85.7[3]	85.7[3]	64.3	89.3[2]	85.7[3]	85.7[3]	42.9
106	86.3[5]	85.7	85.8	86.7[3]	86.9[2]	86.6	84.9	86.6	87.0[1]	86.2
107	98.3[5]	99.4[3]	99.4[3]	100.0[1]	98.7	99.8[2]	93.1	99.8[2]	100.0[1]	82.0
108	97.0[3]	97.5[1]	97.4[2]	96.7	97.4[2]	96.6	97.4[2]	96.4	96.3	96.7
109	98.4[2]	98.9[1]	98.9[1]	96.5	98.9[1]	96.6	93.4	97.1	94.9	98.4[2]
110	87.3[2]	88.2[1]	88.2[1]	86.8	88.2[1]	88.2[1]	76.5	88.2[1]	88.2[1]	85.3
111	60.1[5]	63.0[2]	63.0[2]	60.8	63.7[1]	59.8	61.3	60.5	61.6[3]	59.3

Continuation of Table 4

ID	**NN-FLM**	parRF-t	rf-t	svm-C	rforest-R	svm Poly-t	C5.0-t	svm Radial Cost-t	elm-kernel-m	avNNet-t
112	78.7[3]	79.1[1]	78.9[2]	78.5	77.5	78.3	77.4	78.9[2]	78.0	78.6
113	100.0[1]	100.0[1]	100.0[1]	100.0[1]	100.0[1]	100.0[1]	100.0[1]	100.0[1]	100.0[1]	100.0[1]
114	99.8[3]	100.0[1]	100.0[1]	99.9[2]	100.0[1]	99.8[3]	95.5	99.8[3]	−−	99.6
115	59.1[6]	68.6[2]	68.0[3]	64.4	69.1[1]	56.3	66.2	59.0	65.6	55.9
116	60.4[5]	70.1[1]	69.1[2]	46.4	67.7[3]	53.1	65.0	51.0	51.0	52.6
117	63.3[7]	86.4[3]	86.5[2]	82.6	70.5	51.7	88.8[1]	67.9	−−	33.7
118	92.7[5]	99.6[3]	99.7[2]	92.2	99.4	91.6	99.9[1]	92.5	92.3	85.7
119	100.0[1]	100.0[1]	100.0[1]	100.0[1]	100.0[1]	100.0[1]	100.0[1]	100.0[1]	100.0[1]	100.0[1]
120	88.6[5]	95.2[3]	95.7[1]	86.8	95.4[2]	85.9	94.9	87.0	85.3	86.4
121	100.0[1]	100.0[1]	100.0[1]	100.0[1]	100.0[1]	100.0[1]	100.0[1]	100.0[1]	100.0[1]	100.0[1]
Mean	**86.5**[1]	82.0[2]	81.6[3]	80.4	81.2	77.1	80.6	77.3	77.2	79.4

−−: The classifier made an error on the corresponding data set (see **Section** 3.1 of literature [1]).

(6) The 12 small sample data sets To verify the performance of NN-FLM on small sample classification problem, we selected 12 data sets with small number of samples. And the basic information of these data sets is shown in Table 5.

Table 2: The ID of 121 data sets

ID	Data set	ID	Data set	ID	Data set
1	blood	41	teaching	81	wine-quality-red
2	breast-cancer	42	vertebral-column-2clases	82	connect-4
3	breast-cancer-wisc	43	vertebral-column-3clases	83	spambase
4	breast-cancer-wisc-prog	44	breast-tissue	84	semeion
5	car	45	congressional-voting	85	ozone
6	contrac	46	conn-bench-sonar-mines-rocks	86	chess-krvkp
7	credit-approval	47	heart-hungarian	87	steel-plates
8	cylinder-bands	48	lenses	88	plant-margin
9	echocardiogram	49	lung-cancer	89	arrhythmia
10	fertility	50	lymphography	90	glass
11	haberman-survival	51	musk-1	91	plant-texture
12	heart-cleveland	52	oocytes-trisopterus-states-2f	92	dermatology
13	heart-switzerland	53	synthetic-control	93	letter
14	heart-va	54	balloons	94	cardiotocography-10clases
15	hepatitis	55	energy-y2	95	pima
16	hill-valley	56	mammographic	96	twonorm
17	image-segmentation	57	molec-biol-promoter	97	magic
18	ionosphere	58	oocytes-merluccius-nucleus-4d	98	cardiotocography-3clases
19	led-display	59	pittsburg-bridges-TYPE	99	annealing
20	low-res-spect	60	statlog-german-credit	100	waveform
21	monks-1	61	wine	101	nursery
22	monks-2	62	zoo	102	ringnorm
23	monks-3	63	breast-cancer-wisc-diag	103	plant-shape
24	oocytes-merluccius-states-2f	64	trains	104	adult
25	oocytes-trisopterus-states-5b	65	balance-scale	105	hayes-roth
26	optical	66	ecoli	106	waveform-noise
27	parkinsons	67	libras	107	conn-bench-vowel-deterding
28	pendigits	68	spectf	108	page-blocks
29	pittsburg-bridges-MATERIAL	69	seeds	109	thyroid
30	pittsburg-bridges-REL-L	70	audiology-std	110	horse-colic
31	pittsburg-bridges-SPAN	71	ilpd-indian-liver	111	yeast
32	pittsburg-bridges-T-OR-D	72	miniboone	112	titanic
33	planning	73	musk-2	113	mushroom
34	post-operative	74	bank	114	statlog-shuttle
35	primary-tumor	75	energy-y1	115	wine-quality-white
36	soybean	76	statlog-image	116	flags
37	spect	77	iris	117	chess-krvk
38	statlog-australian-credit	78	abalone	118	wall-following
39	statlog-heart	79	tic-tac-toe	119	acute-nephritis
40	statlog-vehicle	80	statlog-landsat	120	molec-biol-splice
				121	acute-inflammation

Table 3: The 179 comparison classifiers from 17 families

ID	Family	# Classifiers
F1	Discriminant analysis	20
F2	Bayesian approaches	6
F3	Neural networks	21
F4	Support vector machines	10
F5	Decision trees	14
F6	Rule-based methods	12
F7	Boosting	20
F8	Bagging	24
F9	Stacking	2
F10	Random forests	8
F11	Other ensembles	11
F12	Generalized Linear Models	5
F13	Nearest neighbor methods	5
F14	Partial least squares and principal component regression	6
F15	Logistic and multinomial regression	3
F16	Multivariate adaptive regression splines	2
F17	Other Methods	10
	Sum	**179**

Table 5: The 12 small sample data sets

ID	Data set	# samples	# classes	# training samples per class
S1	trains	10	2	3.75
S2	lenses	24	3	6.00
S3	balloons	16	2	6.00
S4	audiology-std	171	18	7.13
S5	lung-cancer	32	3	8.00
S6	zoo	101	7	10.82
S7	plant-margin	1600	100	12.00
S8	plant-texture	1600	100	12.00
S9	plant-shape	1600	100	12.00
S10	soybean	307	18	12.79
S11	pittsburg-bridges-TYPE	105	6	13.13
S12	breast-tissue	106	6	13.25

training samples per class = $\frac{3}{4} \times$ (# samples $\div$ # classes), because the 4 fold cross validation is used in this experiments.