













$$L' - L = \sum_{\omega \in \Omega} \frac{1}{|\mathcal{S}|} \sum_{\mathbf{x} \in \mathcal{S}} q(\omega|\mathbf{x}, \psi(\mathbf{x})) \cdot \frac{p'(\mathbf{x}|\omega)p'(\omega|\psi(\mathbf{x}))}{p(\mathbf{x}|\omega)p(\omega|\psi(\mathbf{x}))} + \frac{1}{|\mathcal{S}|} \sum_{\mathbf{x} \in \mathcal{S}} \sum_{\omega \in \Omega} q(\omega|\mathbf{x}, \psi(\mathbf{x})) \cdot \frac{q(\omega|\mathbf{x}, \psi(\mathbf{x}))}{q'(\omega|\mathbf{x}, \psi(\mathbf{x}))}. \tag{2}$$

$$I(q, q') = \sum_{\omega \in \Omega} q(\omega|\mathbf{x}, \psi(\mathbf{x})) \cdot \frac{q(\omega|\mathbf{x}, \psi(\mathbf{x}))}{q'(\omega|\mathbf{x}, \psi(\mathbf{x}))} \geq 0, \tag{2}$$

$$L' - L \geq \frac{1}{|\mathcal{S}|} \sum_{\mathbf{x} \in \mathcal{S}} \sum_{\omega \in \Omega} q(\omega|\mathbf{x}, \psi(\mathbf{x})) \cdot \frac{p'(\mathbf{x}|\omega)p'(\omega|\psi(\mathbf{x}))}{p(\mathbf{x}|\omega)p(\omega|\psi(\mathbf{x}))}. \tag{2'}$$

$$L' - L \geq \sum_{\omega \in \Omega} \frac{1}{|\mathcal{S}|} \sum_{\mathbf{x} \in \mathcal{S}} q(\omega|\mathbf{x}, \psi(\mathbf{x})) \cdot \frac{p'(\mathbf{x}|\omega)}{p(\mathbf{x}|\omega)} + \sum_{\omega \in \Omega} \frac{1}{|\mathcal{S}|} \sum_{\mathbf{x} \in \mathcal{S}} q(\omega|\mathbf{x}, \psi(\mathbf{x})) \cdot \frac{p'(\omega|\psi(\mathbf{x}))}{p(\omega|\psi(\mathbf{x}))} \geq 0. \tag{2}$$

$$L' - L = \sum_{\omega \in \Omega} \frac{1}{|\mathcal{S}|} \sum_{\mathbf{x} \in \mathcal{S}} q(\omega|\mathbf{x}, \psi(\mathbf{x})) \cdot \frac{p'(\mathbf{x}|\omega)}{p(\mathbf{x}|\omega)} + \sum_{\tilde{\omega} \in \Omega} \frac{|\mathcal{S}_{\tilde{\omega}}|}{|\mathcal{S}|} \sum_{\omega \in \Omega} \frac{1}{|\mathcal{S}_{\tilde{\omega}}|} \sum_{\mathbf{x} \in \mathcal{S}_{\tilde{\omega}}} q(\omega|\mathbf{x}, \tilde{\omega}) \cdot \frac{p'(\omega|\tilde{\omega})}{p(\omega|\tilde{\omega})}. \tag{2}$$

$$p'(\omega|\tilde{\omega}) = \frac{1}{|\mathcal{S}_{\tilde{\omega}}|} \sum_{\mathbf{x} \in \mathcal{S}_{\tilde{\omega}}} q(\omega|\mathbf{x}, \tilde{\omega}), \omega \in \Omega, \tilde{\omega} \in \Omega \tag{2}$$

$$p'(\cdot|\omega) = \frac{1}{p(\cdot|\omega)} \sum_{\mathbf{x} \in \mathcal{S}} q(\omega|\mathbf{x}, \psi(\mathbf{x})) \cdot p(\mathbf{x}|\omega), \omega \in \Omega, \tag{2}$$

$$\sum_{\omega \in \Omega} p'(\omega|\tilde{\omega}) \cdot \frac{p'(\omega|\tilde{\omega})}{p(\omega|\tilde{\omega})} \geq 0, \tilde{\omega} \in \Omega, \tag{0}$$

$$\frac{1}{|\mathcal{S}|} \sum_{\mathbf{x} \in \mathcal{S}} q(\omega|\mathbf{x}, \psi(\mathbf{x})) \cdot p'(\mathbf{x}|\omega) \geq \frac{1}{|\mathcal{S}|} \sum_{\mathbf{x} \in \mathcal{S}} q(\omega|\mathbf{x}, \psi(\mathbf{x})) \cdot p(\mathbf{x}|\omega), \omega \in \Omega, \tag{1}$$

$$\frac{1}{|\mathcal{S}|} \sum_{\mathbf{x} \in \mathcal{S}} q(\omega|\mathbf{x}, \psi(\mathbf{x})) \cdot \frac{p'(\mathbf{x}|\omega)}{p(\mathbf{x}|\omega)} \geq 0, \omega \in \Omega. \tag{1}$$

2.1.1 Gaussian Classes with Noisy Labels

$$p(\mathbf{x}|\omega) = f(\mathbf{x}|\boldsymbol{\mu}_\omega, \boldsymbol{\Sigma}_\omega), \omega \in \Omega, \tag{2}$$

$$\boldsymbol{\mu}_\omega, \boldsymbol{\Sigma}_\omega = \frac{1}{|\mathcal{S}_{\omega}|} \sum_{\mathbf{x} \in \mathcal{S}_{\omega}} q(\omega|\mathbf{x}, \psi(\mathbf{x})) \cdot f(\mathbf{x}|\boldsymbol{\mu}_\omega, \boldsymbol{\Sigma}_\omega), \omega \in \Omega. \tag{1}$$

$$F(\mathbf{x}|\boldsymbol{\mu}) = \sum_{\mathbf{x} \in \mathcal{S}} q(\mathbf{x}) \cdot f(\mathbf{x}|\boldsymbol{\mu}), \tag{2}$$

$$L_{\boldsymbol{\mu}} = \frac{1}{|\mathcal{S}|} \sum_{\mathbf{x} \in \mathcal{S}} f(\mathbf{x}|\boldsymbol{\mu}) \rightarrow \hat{\boldsymbol{\mu}} = \frac{1}{|\mathcal{S}|} \sum_{\mathbf{x} \in \mathcal{S}} \mathbf{x}. \tag{1}$$

$$q(\mathbf{x}) = \frac{N(\mathbf{x})}{|\mathcal{S}|}, q(\mathbf{x}) = 1, (\mathbf{x} \notin \mathcal{S} \Rightarrow q(\mathbf{x}) = 0), \tag{1}$$

$$L_{\boldsymbol{\mu}} = \sum_{\mathbf{x} \in \mathcal{X}} q(\mathbf{x}) \cdot F(\mathbf{x}|\boldsymbol{\mu}) \rightarrow \hat{\boldsymbol{\mu}} = \sum_{\mathbf{x} \in \mathcal{X}} q(\mathbf{x}) \cdot \mathbf{x}. \tag{5}$$





$$p(\omega|\tilde{\omega}) = \frac{1}{|\mathcal{S}_{\tilde{\omega}}|} \sum_{\mathbf{x} \in \mathcal{S}_{\tilde{\omega}}} h(m, \omega|\mathbf{x}, \tilde{\omega}), \omega \in \Omega, \tilde{\omega} \in \Omega. \tag{5}$$

$$w'_{m\omega} = \frac{1}{|\mathcal{S}|} \sum_{\mathbf{x} \in \mathcal{S}} h(m|\omega, \mathbf{x}, \psi(\mathbf{x})) = \frac{1}{|\mathcal{S}|} \sum_{\mathbf{x} \in \mathcal{S}} \frac{h(m, \omega|\mathbf{x}, \psi(\mathbf{x}))}{\sum_{m \in \mathcal{M}_{\omega}} h(m, \omega|\mathbf{x}, \psi(\mathbf{x}))} \quad m \in \mathcal{M}_{\omega}, \omega \in \Omega, \tag{6}$$

### 3 Related work

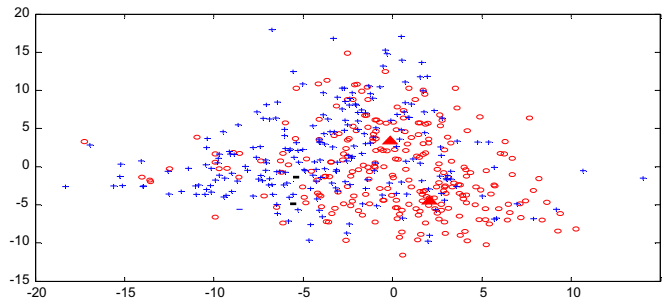
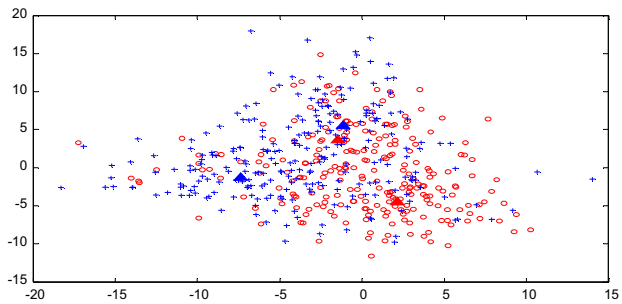
Deep learning has become a dominant paradigm in many areas of artificial intelligence, including computer vision, natural language processing, and speech recognition. The success of deep learning is largely due to its ability to learn hierarchical representations of data, which allows it to capture complex patterns and relationships. However, deep learning models are often computationally expensive and require large amounts of data to train. This has led to the development of various techniques for reducing the computational cost and data requirements of deep learning models. One such technique is knowledge distillation, which involves training a smaller model to learn from a larger, pre-trained model. Another technique is model pruning, which involves removing unnecessary components of a model to reduce its size and computational cost. In this paper, we propose a novel method for reducing the computational cost and data requirements of deep learning models. Our method is based on the idea of knowledge distillation, but it uses a different approach to transfer knowledge from the teacher model to the student model. Specifically, we use a set of weights to control the amount of knowledge that is transferred, allowing us to fine-tune the student model to achieve a desired level of performance. We evaluate our method on a variety of tasks, including image classification and natural language processing, and show that it achieves state-of-the-art results while requiring significantly less computation and data than other methods. Our method is simple and easy to implement, making it a practical solution for reducing the computational cost and data requirements of deep learning models.

The proposed method is based on the idea of knowledge distillation, which involves training a smaller model to learn from a larger, pre-trained model. In our case, the larger model is the teacher model, and the smaller model is the student model. The teacher model is trained on a large dataset, and its weights are used to initialize the student model. The student model is then trained on a smaller dataset, and its weights are updated based on the loss between its output and the target. The key to our method is the use of a set of weights to control the amount of knowledge that is transferred from the teacher model to the student model. These weights are learned during the training process, and they allow us to fine-tune the student model to achieve a desired level of performance. For example, we can use a weight of 1 to transfer all of the knowledge from the teacher model to the student model, or we can use a weight of 0 to transfer no knowledge at all. By using a range of weights, we can explore different levels of knowledge transfer and find the one that works best for a given task. We evaluate our method on a variety of tasks, including image classification and natural language processing, and show that it achieves state-of-the-art results while requiring significantly less computation and data than other methods. Our method is simple and easy to implement, making it a practical solution for reducing the computational cost and data requirements of deep learning models.

**Table 1** Comparison of computational cost and data requirements for different methods

Method	Computation Cost	Data Requirements	Performance
1	2000	0	5
2	1000	2	2
3	10	50	1
4	1	1	22
5	2	1	2
6	50	1	2
7	5000	21	





## 4 Experiments and discussion

### 4.1 Datasets and preprocessing

The dataset consists of two classes of handwritten digits: '1' and 'A'. The '1' class is represented by blue crosses and the 'A' class by red circles. The data is split into training and testing sets. The training set contains 10,000 samples, and the testing set contains 1,000 samples. The data is preprocessed by normalizing the pixel values to the range [0, 1].

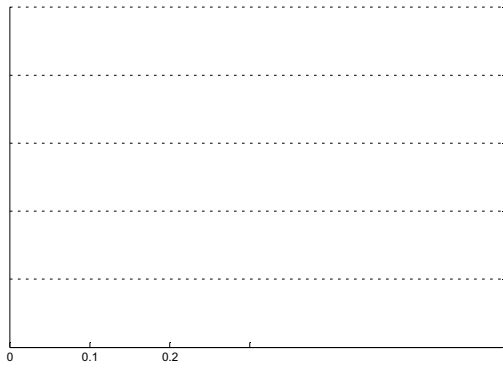
The training set contains 10,000 samples, and the testing set contains 1,000 samples. The data is preprocessed by normalizing the pixel values to the range [0, 1].

### 4.2 Results and discussion

The results show that the model achieves a classification accuracy of 12.0% on the testing set. This is significantly lower than the baseline accuracy of 20%. The model's performance is evaluated using the confusion matrix, which shows that the model is unable to distinguish between the two classes. The confusion matrix is as follows:

	Actual 1	Actual A
Predicted 1	100	100
Predicted A	100	100









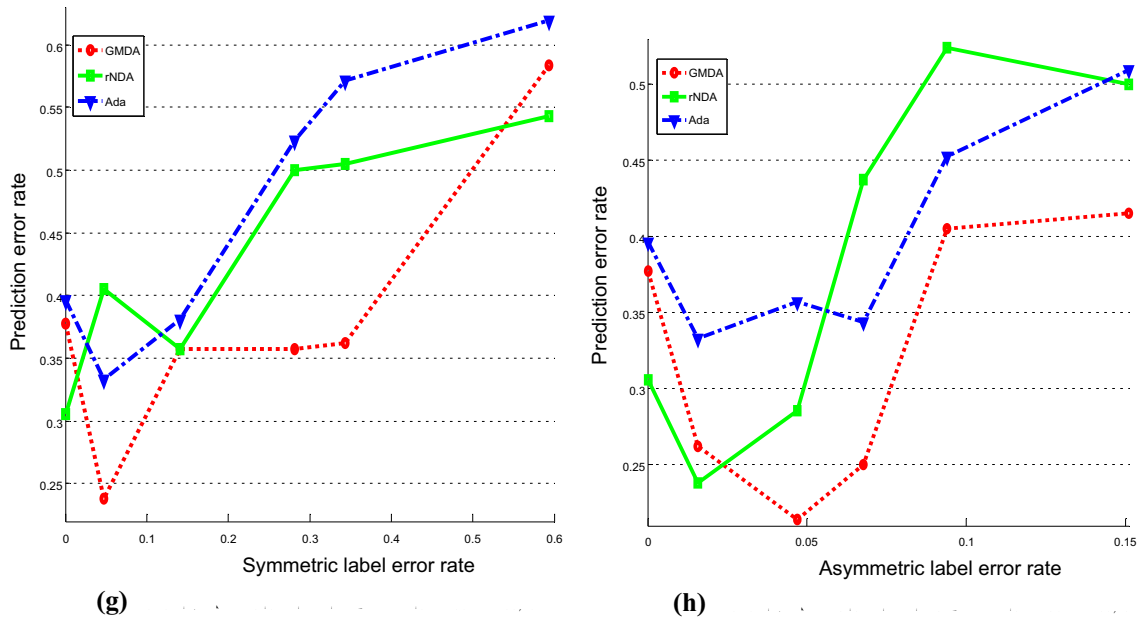


Fig. 6

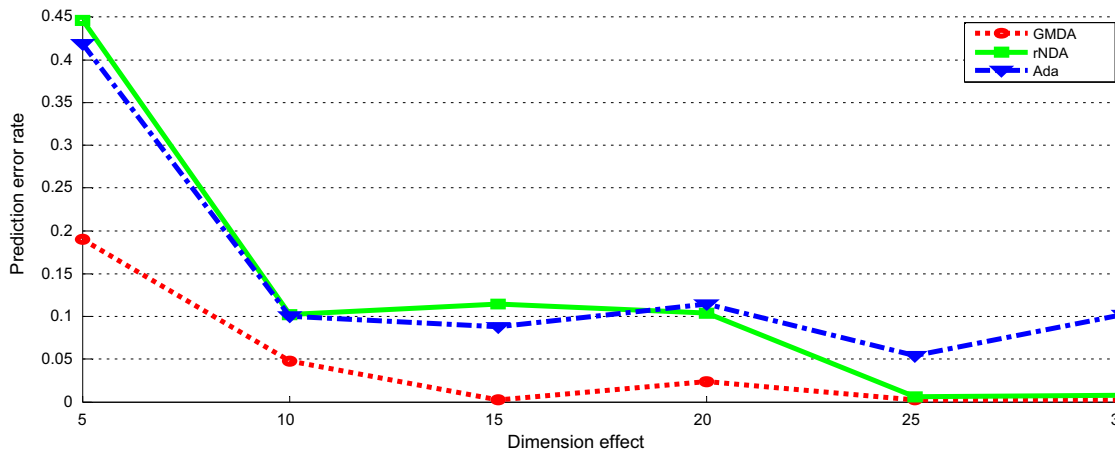


Fig. 7

Fig. 8

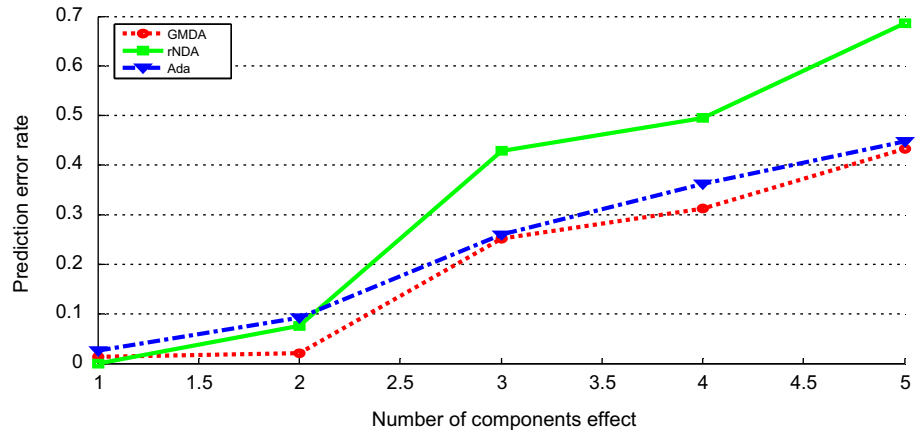




Fig. 9

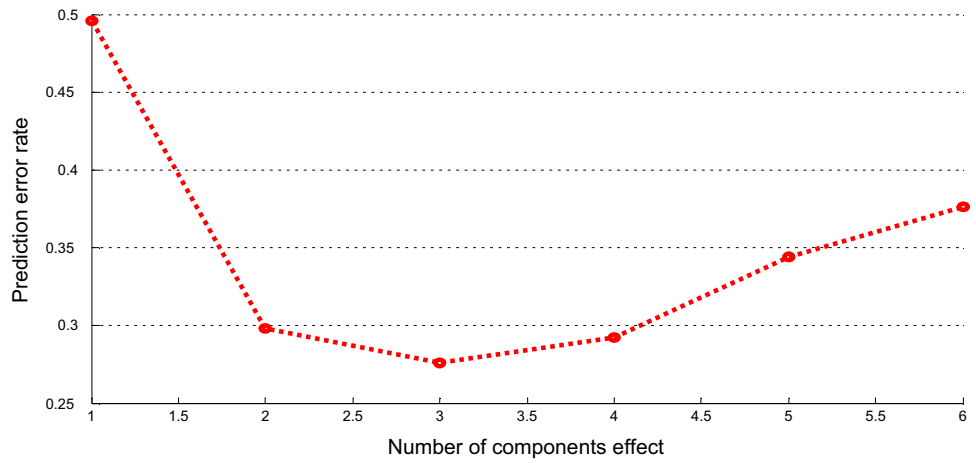


Fig. 10

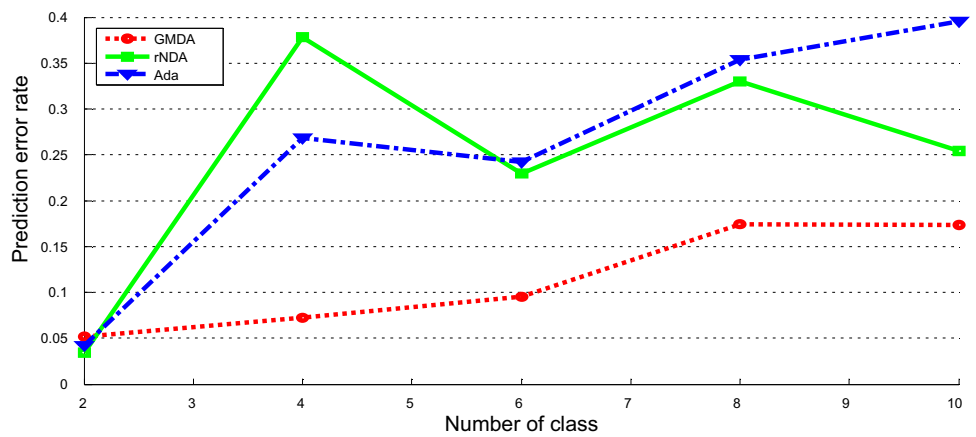


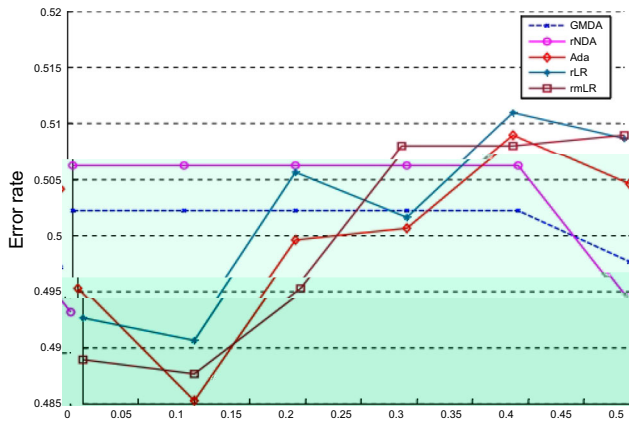
Table 6

	r		r		b	1	b	2
	N	D	r	r				
1	5,000	200	50	5	5, 2		0	
2	5,000	200	0	5	1		1	
	5,000	200	50	10			5	
	5,000	200	0	10	1		1	
5	5,000	200	50		0		2	
	5,000	200	0				20	

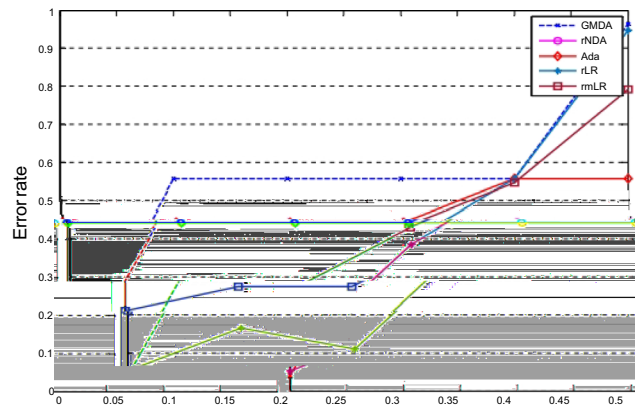
(b) ... 50% ... 1%.

(c) ... 0% ...

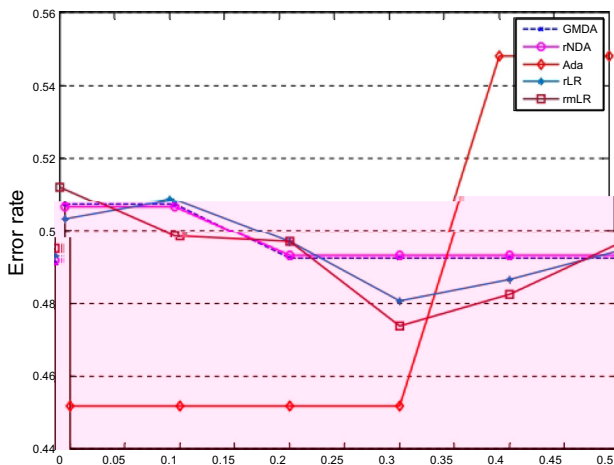




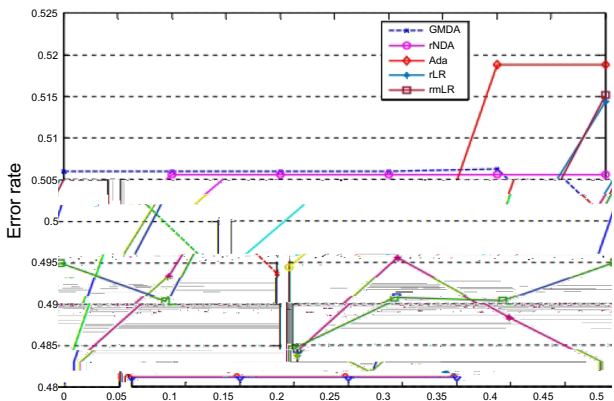
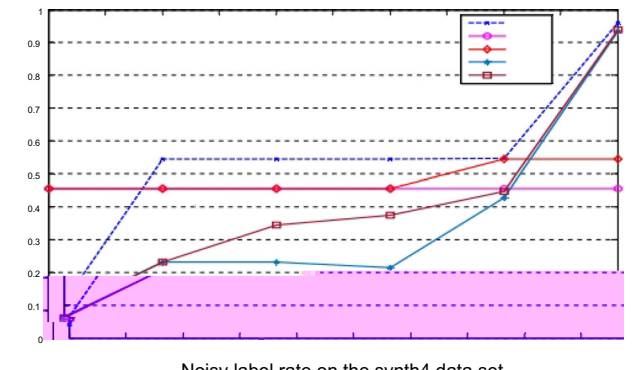
(a1)



(a2)



(b1)



(c1)

