

**Abstract**—Rival penalized competitive learning (RPCL) has been shown to be a useful tool for clustering on a set of sample data in which the number of clusters is unknown. However, the RPCL algorithm was proposed heuristically and is still in lack of a mathematical theory to describe its convergence behavior. In order to solve the convergence problem, we investigate it via a cost-function approach. By theoretical analysis, we prove that a general form of RPCL, called distance-sensitive RPCL (DSRPCL), is associated with the minimization of a cost function on the weight vectors of a competitive learning network. As a DSRPCL process decreases the cost to a local minimum, a number of weight vectors eventually fall into a hypersphere surrounding the sample data, while the other weight vectors diverge to infinity. Moreover, it is shown by the theoretical analysis and simulation experiments that if the cost reduces into the global minimum, a correct number of weight vectors is automatically selected and located around the centers of the actual clusters, respectively. Finally, we apply the DSRPCL algorithms to unsupervised color image segmentation and classification of the wine data.

**Index Terms**—Clustering analysis, competitive learning (CL), convergence, cost function, gradient descent.

As a competitive learning network, a competitive learning (CL) network is a type of artificial neural network that is used for clustering analysis. In a CL network, the input data is presented to a set of weight vectors, and the weight vector that is most similar to the input data is selected as the winner. The winner's weight vector is then updated, and the process is repeated until the network converges to a stable state. CL networks have been used for a variety of applications, including image segmentation, pattern recognition, and data clustering. However, CL networks are often criticized for their lack of a mathematical theory to describe their convergence behavior. In this paper, we investigate the convergence behavior of CL networks using a cost-function approach. We prove that a general form of CL, called distance-sensitive CL (DSC), is associated with the minimization of a cost function on the weight vectors of a competitive learning network. As a DSC process decreases the cost to a local minimum, a number of weight vectors eventually fall into a hypersphere surrounding the sample data, while the other weight vectors diverge to infinity. Moreover, it is shown by the theoretical analysis and simulation experiments that if the cost reduces into the global minimum, a correct number of weight vectors is automatically selected and located around the centers of the actual clusters, respectively. Finally, we apply the DSC algorithms to unsupervised color image segmentation and classification of the wine data.

Competitive learning (CL) is a type of artificial neural network that is used for clustering analysis. In a CL network, the input data is presented to a set of weight vectors, and the weight vector that is most similar to the input data is selected as the winner. The winner's weight vector is then updated, and the process is repeated until the network converges to a stable state. CL networks have been used for a variety of applications, including image segmentation, pattern recognition, and data clustering. However, CL networks are often criticized for their lack of a mathematical theory to describe their convergence behavior. In this paper, we investigate the convergence behavior of CL networks using a cost-function approach. We prove that a general form of CL, called distance-sensitive CL (DSC), is associated with the minimization of a cost function on the weight vectors of a competitive learning network. As a DSC process decreases the cost to a local minimum, a number of weight vectors eventually fall into a hypersphere surrounding the sample data, while the other weight vectors diverge to infinity. Moreover, it is shown by the theoretical analysis and simulation experiments that if the cost reduces into the global minimum, a correct number of weight vectors is automatically selected and located around the centers of the actual clusters, respectively. Finally, we apply the DSC algorithms to unsupervised color image segmentation and classification of the wine data.

Competitive learning (CL) is a type of artificial neural network that is used for clustering analysis. In a CL network, the input data is presented to a set of weight vectors, and the weight vector that is most similar to the input data is selected as the winner. The winner's weight vector is then updated, and the process is repeated until the network converges to a stable state. CL networks have been used for a variety of applications, including image segmentation, pattern recognition, and data clustering. However, CL networks are often criticized for their lack of a mathematical theory to describe their convergence behavior. In this paper, we investigate the convergence behavior of CL networks using a cost-function approach. We prove that a general form of CL, called distance-sensitive CL (DSC), is associated with the minimization of a cost function on the weight vectors of a competitive learning network. As a DSC process decreases the cost to a local minimum, a number of weight vectors eventually fall into a hypersphere surrounding the sample data, while the other weight vectors diverge to infinity. Moreover, it is shown by the theoretical analysis and simulation experiments that if the cost reduces into the global minimum, a correct number of weight vectors is automatically selected and located around the centers of the actual clusters, respectively. Finally, we apply the DSC algorithms to unsupervised color image segmentation and classification of the wine data.

Competitive learning (CL) is a type of artificial neural network that is used for clustering analysis. In a CL network, the input data is presented to a set of weight vectors, and the weight vector that is most similar to the input data is selected as the winner. The winner's weight vector is then updated, and the process is repeated until the network converges to a stable state. CL networks have been used for a variety of applications, including image segmentation, pattern recognition, and data clustering. However, CL networks are often criticized for their lack of a mathematical theory to describe their convergence behavior. In this paper, we investigate the convergence behavior of CL networks using a cost-function approach. We prove that a general form of CL, called distance-sensitive CL (DSC), is associated with the minimization of a cost function on the weight vectors of a competitive learning network. As a DSC process decreases the cost to a local minimum, a number of weight vectors eventually fall into a hypersphere surrounding the sample data, while the other weight vectors diverge to infinity. Moreover, it is shown by the theoretical analysis and simulation experiments that if the cost reduces into the global minimum, a correct number of weight vectors is automatically selected and located around the centers of the actual clusters, respectively. Finally, we apply the DSC algorithms to unsupervised color image segmentation and classification of the wine data.

Competitive learning (CL) is a type of artificial neural network that is used for clustering analysis. In a CL network, the input data is presented to a set of weight vectors, and the weight vector that is most similar to the input data is selected as the winner. The winner's weight vector is then updated, and the process is repeated until the network converges to a stable state. CL networks have been used for a variety of applications, including image segmentation, pattern recognition, and data clustering. However, CL networks are often criticized for their lack of a mathematical theory to describe their convergence behavior. In this paper, we investigate the convergence behavior of CL networks using a cost-function approach. We prove that a general form of CL, called distance-sensitive CL (DSC), is associated with the minimization of a cost function on the weight vectors of a competitive learning network. As a DSC process decreases the cost to a local minimum, a number of weight vectors eventually fall into a hypersphere surrounding the sample data, while the other weight vectors diverge to infinity. Moreover, it is shown by the theoretical analysis and simulation experiments that if the cost reduces into the global minimum, a correct number of weight vectors is automatically selected and located around the centers of the actual clusters, respectively. Finally, we apply the DSC algorithms to unsupervised color image segmentation and classification of the wine data.



Let  $\mathcal{G} = \{W_1^{(0)}, \dots, W_n^{(0)}\}$  be a set of  $n$  initial weights.

$$W_1^{(0)}, \dots, W_n^{(0)} \in \mathcal{G}$$

### Separation Nature

$$\mathcal{G} \rightarrow \mathcal{G} \rightarrow \mathcal{G} \rightarrow \mathcal{G} \rightarrow \mathcal{G}$$

### Correct Division

$$\mathcal{G} \rightarrow k \rightarrow \mathcal{G} \rightarrow k \rightarrow \mathcal{G}$$

Correct Location  $k$  is the location of the correct division.

$$\mathcal{G} \rightarrow k \rightarrow \mathcal{G} \rightarrow k \rightarrow \mathcal{G}$$

Correct Location  $k$  is the location of the correct division.

Correct Location  $k$  is the location of the correct division.

Correct Location  $k$  is the location of the correct division.

$$A. \quad D \rightarrow \mathbf{h} \rightarrow C \rightarrow F \rightarrow \mathcal{S} = \{X^\mu\}_{\mu=1}^N \quad X^\mu = [x_1^\mu, x_2^\mu, \dots, x_d^\mu]^\top$$

$$\begin{aligned} E_{\text{MSE}}(\mathbf{W}) &= \frac{1}{2} \sum_{ij\mu} M_i^\mu (x_j^\mu - w_{ij})^2 \\ &= \frac{1}{2} \sum_{i\mu} M_i^\mu \|X^\mu - W_i\|^2 \\ &= \frac{1}{2} \sum_{\mu} \|X^\mu - W_{c(\mu)}\|^2 \end{aligned}$$

$$\mathbf{W} = [W_1, W_2, \dots, W_k] \quad n = k \quad W_i = [w_{i1}, w_{i2}, \dots, w_{id}]^\top$$

$$M_i^\mu = \begin{cases} 1, & i = c(\mu) \\ 0, & \text{otherwise} \end{cases}$$

$$c(\mu) = \arg \min_j \|X^\mu - W_j\|$$

$$E_{\text{MSE}}(\mathbf{W}) = \frac{1}{2} \sum_{\mu} \|X^\mu - W_{c(\mu)}\|^2$$

$$E_1(\mathbf{W}) = E_{\text{MSE}}(\mathbf{W}) = \frac{1}{2} \sum_{\mu} \|X^\mu - W_{c(\mu)}\|^2$$

$$E_2(\mathbf{W}) = \frac{2}{P} \sum_{\mu, i \neq c(\mu)} \|X^\mu - W_i\|^{-P}$$

$$E(\mathbf{W}) = E_1(\mathbf{W}) + E_2(\mathbf{W})$$

$$E_1(\mathbf{W}) = E_{\text{MSE}}(\mathbf{W}) = \frac{1}{2} \sum_{\mu} \|X^\mu - W_{c(\mu)}\|^2$$

$$E_2(\mathbf{W}) = \frac{2}{P} \sum_{\mu, i \neq c(\mu)} \|X^\mu - W_i\|^{-P}$$

$$\mathbf{W} = \text{vec}[W_1, W_2, \dots, W_n] \quad P$$

$$c(\mu) = 1$$

$$c(\mu) = 0$$

$$E(\mathbf{W}) = \frac{1}{2} \sum_{\mu} \|X^\mu - W_{c(\mu)}\|^2$$

$$n = 1$$

$$k$$

$$B. \quad D \rightarrow \mathbf{h} \rightarrow C \rightarrow F \rightarrow \mathcal{S} = \{X^\mu\}_{\mu=1}^N$$

$$\mathbf{W} = [W_1, W_2, \dots, W_n]$$

$$\mathcal{B} \rightarrow \mathcal{B} \rightarrow \mathcal{B} \rightarrow \mathcal{B} \rightarrow \mathcal{B}$$

$$\begin{aligned} & \mathbf{W} \\ & \mathbf{W} \\ & \mathcal{B} \\ & \mathbf{W} \\ & X^\mu \\ & M_i^\mu \\ & \mathbf{W} \quad E(\mathbf{W}) \\ & \mathbf{W} \\ & w_{ij} \end{aligned}$$

$$\begin{aligned} \frac{\partial E(\mathbf{W})}{\partial w_{ij}} &= \frac{\partial E_1(\mathbf{W})}{\partial w_{ij}} + \frac{\partial E_2(\mathbf{W})}{\partial w_{ij}} \\ &= - \sum_{\mu} \delta_{i,c(\mu)} (x_j^\mu - w_{ij}) + \sum_{\mu,i} (1 - \delta_{i,c(\mu)}) \\ &\quad \times \|X^\mu - W_i\|^{-P-2} (x_j^\mu - w_{ij}) \end{aligned}$$

$$\begin{aligned} & \delta_{i,j} \\ & \mathbf{W} \\ & X^\mu \\ & \mathbf{W} \\ & R^{nd} \\ & M_i^\mu \\ & \mathbf{W}' \\ & W'_i \quad W'_j \quad i < j \\ & X^{\mu'} \end{aligned}$$

$$\|W'_i - X^{\mu'}\| = \|W'_j - X^{\mu'}\| = \min_l \|W'_l - X^{\mu'}\| > 0.$$

$$\begin{aligned} & \mathbf{W}' \\ & \|W_i - X^{\mu'}\| = \|W_j - X^{\mu'}\| \\ & \mathcal{A}_{l_i} \quad \mathcal{A}_{l_j} \\ & \|W_i - X^{\mu'}\| = \min_l \|W_l - X^{\mu'}\| \leq \|W_j - X^{\mu'}\| \\ & \mathcal{A}_{l_i} \quad \|W_j - X^{\mu'}\| = \min_l \|W_l - X^{\mu'}\| < \|W_i - X^{\mu'}\| \\ & \mathcal{A}_{l_j} \quad i < j \\ & \mathcal{A}_{l_i} \\ & \mathcal{A}_{l_i} \\ & \mathbf{W}' \\ & M_i^{\mu'} = 1 \\ & M_j^{\mu'} = 0 \\ & \mathcal{A}_{l_j} \\ & \mathbf{W}' \\ & M_i^{\mu'} = 0 \quad M_j^{\mu'} = 1 \quad \|W'_i - X^{\mu'}\| = \|W'_j - X^{\mu'}\| \end{aligned}$$

$$\begin{aligned} C_i &= \mathcal{S} \cap R_i, \quad i = 1, \dots, n. \end{aligned}$$

$$\begin{aligned} \overline{C}_i &= \mathcal{S} - C_i \\ \overline{C}_i &= \mathcal{S} - C_i \end{aligned}$$

$$E_1(\mathbf{W})$$

$$E_2(\mathbf{W})$$

$$E_2(\mathbf{W})$$

$$\|X^\mu - W_{r(\mu)}\|^{-P} \quad r(\mu)$$

$$E_2(\mathbf{W})$$

$$\Delta W_i = \begin{cases} \eta(X^\mu - W_i), & i = c(\mu) \\ -\eta\|X^\mu - W_i\|^{-P-2}(X^\mu - W_i), & i = r(\mu) \\ 0, & \end{cases}$$

$$\alpha_c = \eta, \quad \alpha_r = \eta\|X^\mu - W_{r(\mu)}\|^{-P-2}.$$

$$\frac{\alpha_c}{\alpha_r} = \|X^\mu - W_{r(\mu)}\|^{2+P}.$$

$$\begin{aligned} \alpha_c/\alpha_r &= \|X^\mu - W_{r(\mu)}\|^{2+P} \\ \alpha_c &= \alpha_r \end{aligned}$$

$$\eta$$

$$\eta$$

$$\begin{aligned}
 & \mathbf{W}^{(t)} = [W_1^{(t)}, \dots, W_n^{(t)}] \\
 & \mathbf{W} = [W_1^{(0)}, \dots, W_n^{(0)}] \in R^{nd} - \mathcal{B} \\
 & I: \quad \quad \quad W_i^{(t)} \\
 & \quad \quad \quad t \\
 & \quad \quad \quad W_i^{(t)} \\
 & \quad \quad \quad t \\
 & \Delta W_i = \eta \sum_{\mu} \|X^{\mu} - W_i\|^{-P-2} (W_i - X^{\mu}). \\
 & W_i^{(t+1)}, \\
 & E(\mathbf{W}^{(t)}) \\
 & E(\mathbf{W}^{(t)}) \\
 & W_j \\
 & W_i^{(t+1)} \\
 & W_j \\
 & W_i^{(t+1)} \\
 & W_i^{(t+1)} \\
 & W_i^{(t+1)} \\
 & W_i \\
 & \blacksquare \\
 & \boldsymbol{h} \quad I: \\
 & \eta \quad \quad \quad \mathcal{G} \\
 & W_i^{(t)} \quad \mathcal{G} \quad \mathcal{G} \\
 & \vdots \\
 & E(\mathbf{W}^{(t)}) \quad \quad \quad t \\
 & \eta \\
 & E(\mathbf{W}^{(t)}) \\
 & E^* \\
 & t \\
 & \quad \quad \quad \{W_i^{(t)}\} \\
 & \quad \quad \quad \{W_i^{(t)}\} \\
 & W_i^{(t)} \\
 & E^* \quad \quad \quad \{W_i^{(t)}\} \\
 & \quad \quad \quad W_i^{(t)} \\
 & \{W_i^{(t)}\} \quad \mathcal{G} \\
 & T \quad \quad \quad \{W_i^{(t)}\} \\
 & \quad \quad \quad t > T \quad W_i^{(t)} \quad \mathcal{G} \quad \blacksquare \\
 & \eta \\
 & E(\mathbf{W}) \\
 & \hat{\mathbf{W}}^* \quad E(\hat{\mathbf{W}}) \\
 & \hat{\mathbf{W}} \\
 & \hat{\mathbf{W}}^* \\
 & \eta \\
 & E(\hat{\mathbf{W}}) \\
 & E(\hat{\mathbf{W}}) \\
 & \|W_i - W_j\| \geq \delta \quad \quad \quad i \neq j \quad \quad \delta \\
 & E(\mathbf{W}) \\
 & \{ \mathbf{W} : E(\mathbf{W}) \leq C \} \\
 & C > 0 \quad E(\mathbf{W}) \\
 & \eta
 \end{aligned}$$

$$E(\mathbf{W}^{(t)})$$
$$k, \dots, k, k, \dots, k, m_1, \dots, m_k, \dots, m_i$$

$$\begin{array}{ccccccc} & & \mathbf{W} & & k & & C_1, \dots, C_k \\ E_1(\mathbf{W}) & & & & & & \\ E_1(\mathbf{W}) & & & & & & \\ & & \mathbf{W}^0 & & E_1(\mathbf{W}^0) & & \mathbf{W} \end{array} \quad C_i$$







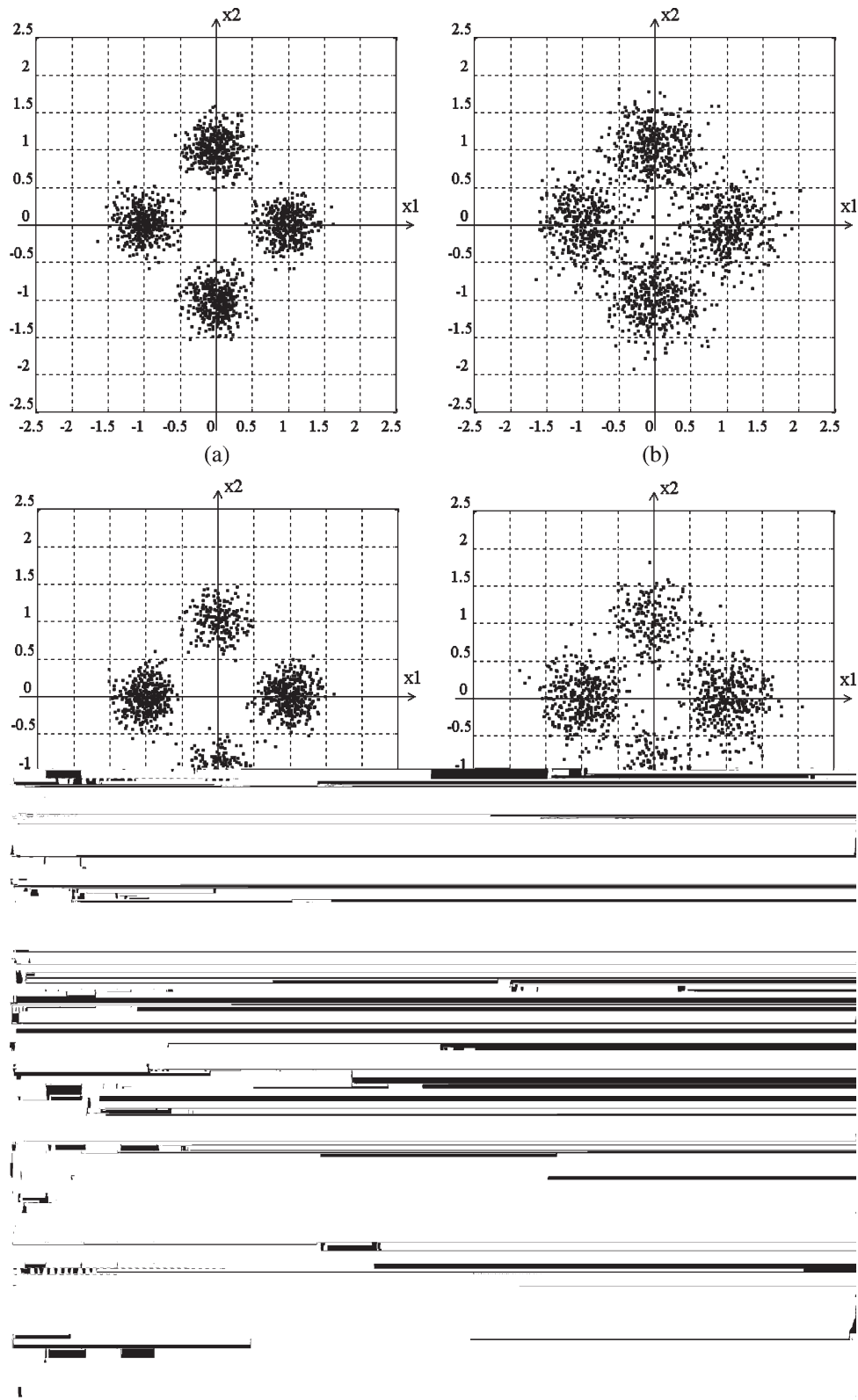


Figure 1: Scatter plots of data points for different stages:  $S_1$  (top-left),  $S_2$  (top-right),  $S_3$  (bottom-left),  $S_4$  (bottom-right), and  $S_5$  (bottom-center).

2)  $\mathbf{D} = \mathbf{C} \mathbf{A}^T$  :  $\mathbf{D}$  is the distance matrix,  $\mathbf{C}$  is the cluster matrix, and  $\mathbf{A}$  is the affinity matrix. The distance matrix  $\mathbf{D}$  is defined as  $D_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|$ , where  $\mathbf{x}_i$  and  $\mathbf{x}_j$  are data points. The cluster matrix  $\mathbf{C}$  is defined as  $C_{ij} = 1$  if  $\mathbf{x}_i$  and  $\mathbf{x}_j$  belong to the same cluster, and 0 otherwise. The affinity matrix  $\mathbf{A}$  is defined as  $A_{ij} = \exp(-D_{ij}^2 / \sigma^2)$ , where  $\sigma^2$  is a parameter. The expected value of the distance matrix is  $E(\mathbf{D})$ , the expected value of the cluster matrix is  $E(\mathbf{C})$ , and the expected value of the affinity matrix is  $E(\mathbf{A})$ . The expected value of the distance matrix is  $E(\mathbf{D}) = \mathbf{C} \mathbf{A}^T$ , the expected value of the cluster matrix is  $E(\mathbf{C}) = \mathbf{C}$ , and the expected value of the affinity matrix is  $E(\mathbf{A}) = \mathbf{A}$ . The expected value of the distance matrix is  $E(\mathbf{D}) = \mathbf{C} \mathbf{A}^T$ , the expected value of the cluster matrix is  $E(\mathbf{C}) = \mathbf{C}$ , and the expected value of the affinity matrix is  $E(\mathbf{A}) = \mathbf{A}$ .

Figure 1: The estimated trajectories.

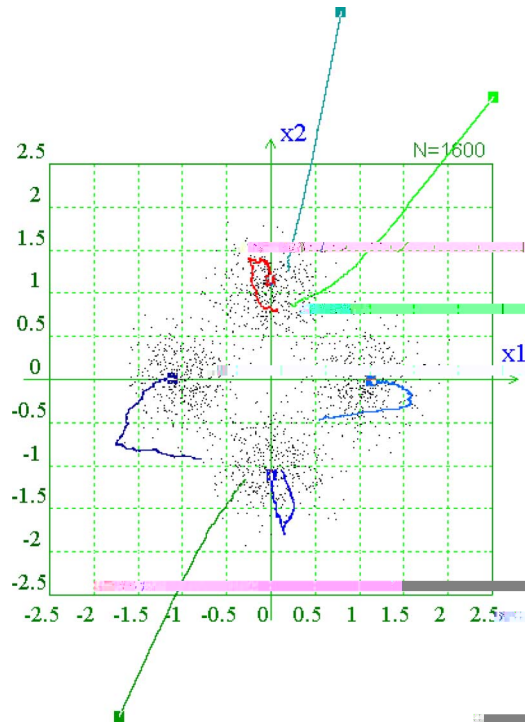
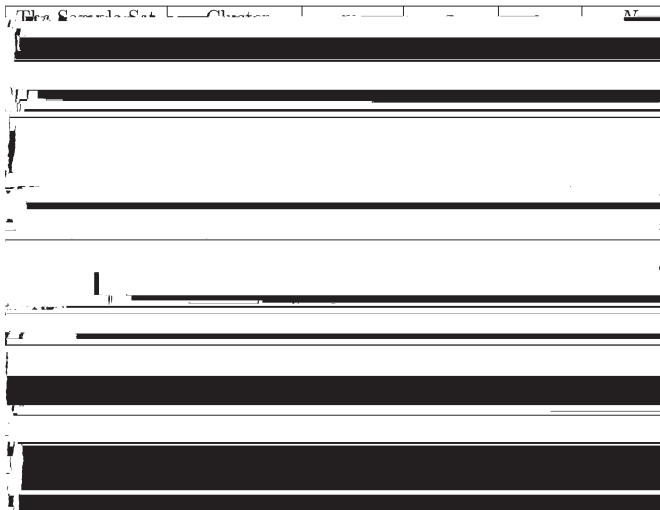


Figure 2: The estimated trajectories of the stochastic process  $X_t$  for  $n = 1000$ ,  $k = 4$  and  $\eta = 0.001$ . The trajectories are shown for  $t = 0, \dots, 1000$ .

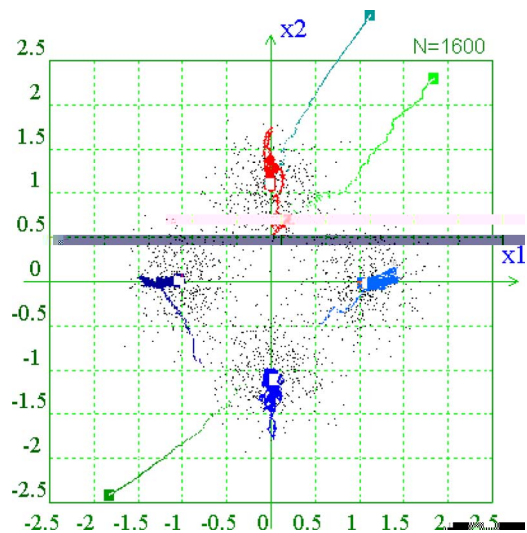
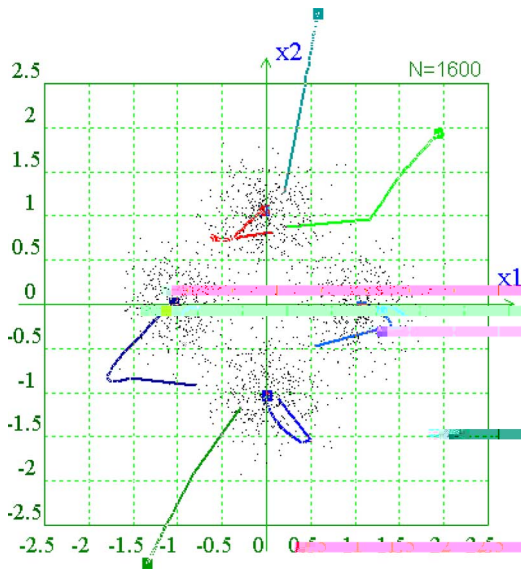


Figure 4: The estimated trajectories of the stochastic process  $X_t$  for  $n = 1000$ ,  $k = 4$  and  $\eta = 0.001$ . The trajectories are shown for  $t = 0, \dots, 1000$ .

Figure 5: The estimated trajectories of the stochastic process  $X_t$  for  $n = 1000$ ,  $k = 4$  and  $\eta = 0.001$ . The trajectories are shown for  $t = 0, \dots, 1000$ .

$S_1, \dots, S_2$

$E(W)$

$S_2$

$E_2(W)$

$E_2(W)$

$\eta/m$

$m = \lfloor t/5 \rfloor$

$n = 7, k = 4$

$t = 1000$

$m = \lfloor t/5N \rfloor$

$x_t$

$N$

$n = 7, k = 4$

$t = 1000$

$$\begin{array}{l} E(\mathbf{W}) \\ E(\mathbf{W}) \\ n \leq 2k \end{array} \quad \begin{array}{l} t < M \\ \lambda < \varepsilon \\ n \end{array} \quad \begin{array}{l} t = t + 1 \\ T = T + 1 \\ n \end{array}$$

$M$   $n$   $\eta$   $\varepsilon$   $10^{-6}$   $T$   $k_0$   $k_1$   $c_0$   $c_1$   $[a, b]$

$$w_{ij} = \frac{[a, b]}{T} \exp(-k_1 T - k_0) \quad T = 0$$

$$\eta = \eta_0 / (c_1 T + c_0) \quad t = 0$$





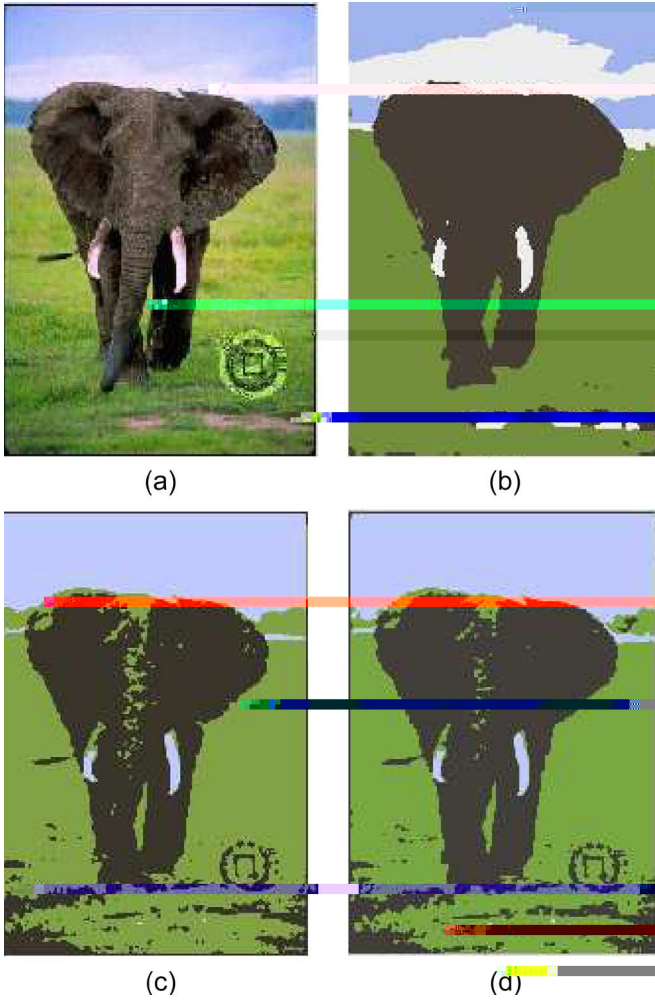


Figure 1: Four panels (a, b, c, d) showing an elephant in a savanna landscape. Panel (a) is the original image. Panel (b) shows the image with a green horizontal line and a blue horizontal line. Panel (c) shows the image with a red horizontal line and a blue horizontal line. Panel (d) shows the image with a red horizontal line and a blue horizontal line. The images are arranged in a 2x2 grid.

Figure 1: Four panels (a, b, c, d) showing an elephant in a savanna landscape. Panel (a) is the original image. Panel (b) shows the image with a green horizontal line and a blue horizontal line. Panel (c) shows the image with a red horizontal line and a blue horizontal line. Panel (d) shows the image with a red horizontal line and a blue horizontal line. The images are arranged in a 2x2 grid.

### C. $C \rightarrow \mathcal{H} \rightarrow D$

Figure 1: Four panels (a, b, c, d) showing an elephant in a savanna landscape. Panel (a) is the original image. Panel (b) shows the image with a green horizontal line and a blue horizontal line. Panel (c) shows the image with a red horizontal line and a blue horizontal line. Panel (d) shows the image with a red horizontal line and a blue horizontal line. The images are arranged in a 2x2 grid.

Figure 1: Four panels (a, b, c, d) showing an elephant in a savanna landscape. Panel (a) is the original image. Panel (b) shows the image with a green horizontal line and a blue horizontal line. Panel (c) shows the image with a red horizontal line and a blue horizontal line. Panel (d) shows the image with a red horizontal line and a blue horizontal line. The images are arranged in a 2x2 grid.

### D. $D \rightarrow \mathcal{H} \rightarrow B \rightarrow \mathcal{H} \rightarrow D \rightarrow C \rightarrow A \rightarrow \mathcal{H}$

Figure 1: Four panels (a, b, c, d) showing an elephant in a savanna landscape. Panel (a) is the original image. Panel (b) shows the image with a green horizontal line and a blue horizontal line. Panel (c) shows the image with a red horizontal line and a blue horizontal line. Panel (d) shows the image with a red horizontal line and a blue horizontal line. The images are arranged in a 2x2 grid.

Figure 1: Four panels (a, b, c, d) showing an elephant in a savanna landscape. Panel (a) is the original image. Panel (b) shows the image with a green horizontal line and a blue horizontal line. Panel (c) shows the image with a red horizontal line and a blue horizontal line. Panel (d) shows the image with a red horizontal line and a blue horizontal line. The images are arranged in a 2x2 grid.

Figure 1: Four panels (a, b, c, d) showing an elephant in a savanna landscape. Panel (a) is the original image. Panel (b) shows the image with a green horizontal line and a blue horizontal line. Panel (c) shows the image with a red horizontal line and a blue horizontal line. Panel (d) shows the image with a red horizontal line and a blue horizontal line. The images are arranged in a 2x2 grid.

Figure 1: Four panels (a, b, c, d) showing an elephant in a savanna landscape. Panel (a) is the original image. Panel (b) shows the image with a green horizontal line and a blue horizontal line. Panel (c) shows the image with a red horizontal line and a blue horizontal line. Panel (d) shows the image with a red horizontal line and a blue horizontal line. The images are arranged in a 2x2 grid.

Figure 1: Four panels (a, b, c, d) showing an elephant in a savanna landscape. Panel (a) is the original image. Panel (b) shows the image with a green horizontal line and a blue horizontal line. Panel (c) shows the image with a red horizontal line and a blue horizontal line. Panel (d) shows the image with a red horizontal line and a blue horizontal line. The images are arranged in a 2x2 grid.





**Jinwen Ma**

Dr. Jinwen Ma received his B.S. degree in Computer Science from Tsinghua University, Beijing, China, in 2008, and his M.S. degree in Computer Science from Tsinghua University, Beijing, China, in 2010. He is currently a Ph.D. student in the Department of Computer Science, Tsinghua University, Beijing, China.

His research interests include machine learning, computer vision, and natural language processing. He has published several papers in the field of machine learning and computer vision. He is currently working on the project of deep learning for natural language processing.



**Taijun Wang**

Dr. Taijun Wang received his B.S. degree in Computer Science from Tsinghua University, Beijing, China, in 2008, and his M.S. degree in Computer Science from Tsinghua University, Beijing, China, in 2010. He is currently a Ph.D. student in the Department of Computer Science, Tsinghua University, Beijing, China.

His research interests include machine learning, computer vision, and natural language processing. He has published several papers in the field of machine learning and computer vision. He is currently working on the project of deep learning for natural language processing.

Dr. Jinwen Ma received his B.S. degree in Computer Science from Tsinghua University, Beijing, China, in 2008, and his M.S. degree in Computer Science from Tsinghua University, Beijing, China, in 2010. He is currently a Ph.D. student in the Department of Computer Science, Tsinghua University, Beijing, China.

His research interests include machine learning, computer vision, and natural language processing. He has published several papers in the field of machine learning and computer vision. He is currently working on the project of deep learning for natural language processing.