14

A Generic Framework for Soft Subspace Pattern Recognition

Dat Tran, Wanli Ma, Dharmendra Sharma, Len Bui and Trung Le
University of Canberra, Faculty of Information Sciences and Engineering
Australia

1. Introduction

In statistical pattern recognition, hidden Markov model (HMM) is the most important technique for modelling patterns that include temporal information such as speech and handwriting. If the temporal information is not taken into account, Gaussian mixture model (GMM) is used. This GMM technique uses a mixture of Gaussian densities to model the distribution of feature vectors extracted from training data. When little training data are available, vector quantisation (VQ) technique is also effective (Tran & Wagner 2002). In fuzzy set theory-based pattern recognition, fuzzy clustering techniques such as fuzzy c-means and fuzzy entropy are used to design re-estimation techniques for fuzzy HMM, fuzzy GMM, and fuzzy VQ (Tran & Wagner 2000).

The first stage in pattern recognition is data feature selection. A number of features that best characterises the considering pattern is extracted and the selection of features is dependent on the pattern to be recognised and has direct impact on the recognition results. The above-mentioned pattern recognition methods cannot select features automatically because they treat all features equally. We propose that the contribution of a feature to pattern recognition should be measured by a weight that is assigned to the feature in the modelling process. This method is called soft subspace pattern recognition. There have been some algorithms proposed to calculate weights for soft subspace clustering (Huang et al. 2005, Jing et al. 2007). However a generic framework for the above-mentioned modelling methods has not been built.

A generic framework for soft subspace pattern recognition will be proposed in this chapter. A generic objective function will be designed for HMM and maximizing this function will provide an algorithm for calculating weights. Other weight calculation algorithms for GMM and VQ will also be determined from the algorithm for HMM.

The proposed soft subspace pattern recognition methods will be evaluated in network intrusion detection. Some preliminary experiments have been done and experimental results showed that the proposed algorithms could improve the recognition rates.

2. Continuous hidden Markov model

The underlying assumption of the HMM is that the speech signal can be well characterised as a parametric random process, and that the parameters of the stochastic process can be

Source: Theory and Novel Applications of Machine Learning, Book edited by: Meng Joo Er and Yi Zhou,
ISBN 978-3-902613-55-4, pp. 376, February 2009, I-Tech, Vienna, Austria

estimated in a precise, well-defined manner. The HMM method provides a reliable way of recognizing speech for a wide range of applications (Juang 1998, Furui 1997, Rabiner et al. 1996).

There are two assumptions in the first-order HMM. The first is the Markov assumption, i.e. a new state is entered at each time t based on the transition probability, which only depends on the previous state. It is used to characterise the sequence of the time frames of a speech pattern. The second is the output-independence assumption, i.e. the output probability depends only on the state at that time regardless of when and how the state is entered (Huang et al. 1990). A process satisfying the Markov assumption is called a Markov model (Kulkarni 1995). An observable Markov model is a process where the output is a set of states at each instant of time and each state corresponds to an observable event. The hidden Markov model is a doubly stochastic process with an underlying Markov process which is not directly observable (hidden) but which can be observed through another set of stochastic processes that produce observable events in each of the states (Rabiner & Juang 1993).

Let $S = \{s_1, s_2, ..., s_T\}$ and $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T\}$ be a sequence of states and a sequence of continuous feature vectors, respectively. The compact notation $\Lambda = \{\pi, A, B\}$ indicates the complete parameter set of the HMM where

- $\pi = \{\pi_i\}$, $\pi_i = P(s_1 = i \mid \Lambda)$: the initial state distribution
- $A = \{a_{ij}\}$, $a_{ij} = P(s_t = j \mid s_{t-1} = i, \Lambda)$: the state transition probability distribution, and
- $B = \{b_j(\mathbf{x}_t)\}$, $b_j(\mathbf{x}_t) = P(\mathbf{x}_t \mid s_t = j, \Lambda)$: the output probability distribution of feature vector $\mathbf{x}_t$ in state *j*.

The following constraints are applied:

$$\sum_{i=1}^{N} \pi_i = 1, \quad \sum_{j=1}^{N} a_{ij} = 1, \quad \text{and} \quad \int b(\mathbf{x}_t) d\mathbf{x}_t = 1 \tag{1}$$

The HMM parameters are estimated such that in some sense, they best match the distribution of the feature vectors in $\mathbf{X}$. The most widely used training method is the maximum likelihood (ML) estimation. For a sequence of feature vectors $\mathbf{X}$, the likelihood of the HMM is

$$P(\mathbf{X} \mid \Lambda) = \prod_{t=1}^{T} P(\mathbf{x}_t \mid \Lambda) \tag{2}$$

The aim of ML estimation is to find a new parameter model $\overline{\Lambda}$ such that $P(\mathbf{X} \mid \overline{\Lambda}) \geq P(\mathbf{X} \mid \Lambda)$. Since the expression in (2) is a nonlinear function of parameters in Λ, its direct maximisation is not possible. However, parameters can be obtained iteratively using the expectation-maximisation (EM) algorithm (Dempster 1977). An auxiliary function *Q* is used

$$Q(\Lambda, \overline{\Lambda}) = \sum_{t=1}^{T-1} \sum_{i=1}^{N} \sum_{j=1}^{N} P(s_t = i, s_{t+1} = j \mid \mathbf{X}, \Lambda) \log[\overline{a}_{ij} \overline{b}_{ij}(\mathbf{x}_{t+1})] \tag{3}$$

where $\overline{\pi}_{s_1=j}$ is denoted by $\overline{a}_{s_0=i\, s_1=j}$ for simplicity. Setting derivatives of the *Q* function with respect to $\overline{\Lambda}$ to zero, the following reestimation formulas are found

$$\overline{\pi}_i = \gamma_1(i) \qquad \overline{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{4}$$

where

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)\,, \quad \xi_t(i,j) = P(s_t = i, s_{t+1} = j \mid \mathbf{X}, \Lambda) \tag{5}$$

The most general representation of the output probability distribution is a mixture of Gaussians

$$b_j(\mathbf{x}_t) = P(\mathbf{x}_t \mid s_t = j, \Lambda) = \sum_{k=1}^{K} P(k \mid s_t = j, \Lambda) P(\mathbf{x}_t \mid k, s_t = j, \Lambda)$$

$$b_j(\mathbf{x}_t) = \sum_{k=1}^{K} c_{jk} N(\mathbf{x}_t, \boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk}) \tag{6}$$

where $c_{jk} = P(k \mid s_t = j, \Lambda)$, *j* = 1,…, *N*, *k* = 1,…, *K* are mixture coefficients, and $N(\mathbf{x}_t, \boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk})$ is a Gaussian with mean vector $\boldsymbol{\mu}_{jk}$ and covariance matrix $\boldsymbol{\Sigma}_{jk}$ for the *k*th mixture component in state *j*. The following constraints are satisfied

$$c_{jk} > 0 \quad \text{and} \quad \sum_{k=1}^{K} c_{jk} = 1 \tag{7}$$

The mixture coefficients, mean vectors and covariance matrices are calculated as follows

$$\overline{c}_{jk} = \frac{1}{T} \sum_{t=1}^{T} P(k \mid \mathbf{x}_t, s_t = j, \Lambda) \qquad \overline{\boldsymbol{\mu}}_{jk} = \frac{\sum_{t=1}^{T} P(k \mid \mathbf{x}_t, s_t = j, \Lambda) \mathbf{x}_t}{\sum_{t=1}^{T} P(k \mid \mathbf{x}_t, s_t = j, \Lambda)}$$

$$\overline{\boldsymbol{\Sigma}}_{jk} = \frac{\sum_{t=1}^{T} P(k \mid \mathbf{x}_t, s_t = j, \Lambda)(\mathbf{x}_t - \boldsymbol{\mu}_{jk})(\mathbf{x}_t - \boldsymbol{\mu}_{jk})'}{\sum_{t=1}^{T} P(k \mid \mathbf{x}_t, s_t = j, \Lambda)} \tag{8}$$

where the prime denotes vector transposition, and

$$P(k \mid \mathbf{x}_t, s_t = j, \Lambda) = \frac{c_{jk} N(\mathbf{x}_t, \boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk})}{\sum_{n=1}^{K} c_{jn} N(\mathbf{x}_t, \boldsymbol{\mu}_{jn}, \boldsymbol{\Sigma}_{jn})} \tag{9}$$

In the *M*-dimensional feature space, the Guassian function can be written as follows

$$N(\mathbf{x}_t, \boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk}) = P(\mathbf{x}_t \mid k, s_t = j, \Lambda) = \prod_{m=1}^{M} P(x_{tm} \mid k, s_t = j, \Lambda) \tag{10}$$

where

$$P(x_{tm} \mid k, s_t = j, \Lambda) = \frac{1}{\sqrt{2\pi\sigma_{jkm}^2}} e^{-\frac{(x_{tm} - \mu_{jkm})^2}{2\sigma_{jkm}^2}} \tag{11}$$

2. Fuzzy subspace continuous hidden Markov model

It can be observed in (10) that features are treated equally in the HMM. In order to differentiate the contribution of features, we propose to assign a weight to each feature as follows

$$N(\mathbf{x}_t, \boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk}) = \prod_{m=1}^{M} \left[P(x_{tm} \mid k, s_t = j, \Lambda)\right]^{w_{jkm}^{\alpha}} \tag{12}$$

where w_{jkm}^{α} , m = 1, 2, …, M are components of an M-dimensional weight vector $\mathbf{w}_{jm}^{\alpha}$, and α is a parameter weight for w_{jkm}^{α} . Weight values satisfy the following conditions:

$$0 \le w_{jkm} \le 1 \quad \forall m, \; \sum_{m=1}^{M} w_{jkm} = 1 \tag{13}$$

The weight values can be determined by considering the following function which is part of the *Q* function in (3):

$$Q_j(\Lambda, \overline{\Lambda}) = \sum_{k=1}^{K} \sum_{t=1}^{T} P(k \mid \mathbf{x}_t, s_t = j, \Lambda) \log[\overline{c}_{jk} N(\mathbf{x}_t, \overline{\boldsymbol{\mu}}_{jk}, \overline{\boldsymbol{\Sigma}}_{jk})] \tag{14}$$

where

$$\log N(\mathbf{x}_t, \overline{\boldsymbol{\mu}}_{jk}, \overline{\boldsymbol{\Sigma}}_{jk}) = \sum_{m=1}^{M} w_{jkm}^{\alpha} \log P(x_{tm} \mid k, s_t = j, \overline{\Lambda}) \tag{15}$$

The basic idea of this approach is to maximize the function $Q_j(\Lambda, \overline{\Lambda})$ over the variable w_{jkm}^{α} on the assumption that the weight vector $\mathbf{w}_{jm}^{\alpha}$ identifies a good contribution of the features. Maximizing the function $Q_j(\Lambda, \overline{\Lambda})$ in (14) using (13) and (15) gives

$$w_{jkm} = \frac{1}{\sum_{n=1}^{M} (D_{jkm} / D_{jkn})^{1/(\alpha - 1)}} \tag{16}$$

where $\alpha \neq 1$ and $D_{jkm} = -\sum_{t=1}^{T} P(k \mid \mathbf{x}_t, s_t = j, \Lambda) \log P(x_{tm} \mid k, s_t = j, \overline{\Lambda})$

The advantage of this approach is that it does not change the structure of the HMM listed in (4) through (11). This means that these equations are still applied in fuzzy subspace HMM. Therefore, this approach can be considered as a generic framework and can extend to other models that relate to the HMM such as Gaussian mixture model (GMM) and Vector Quantization (VQ). Fuzzy subspace GMM can be obtained by setting the number of states in fuzzy subspace continuous HMM to one. The VQ will be considered in the next section.

3. Fuzzy subspace vector quantization

3.1 Vector quantization

The VQ modelling is an efficient data reduction method, which is used to convert a feature vector set into a small set of distinct vectors using a clustering technique. Advantages of this reduction are reduced storage and computation. The distinct vectors are called code vectors and the set of code vectors that best represents the training set is called the codebook. Since there is only a finite number of code vectors, the process of choosing the best representation of a given feature vector is equivalent to quantizing the vector and leads to a certain level of quantization error. This error decreases as the size of the codebook increases, however the storage required for a large codebook is non-trivial. The VQ codebook can be used as a model in pattern recognition. The key point of VQ modelling is to derive an optimal codebook which is commonly achieved by using a clustering technique.

In VQ modeling, the model Λ is a set of cluster centers $\Lambda = \{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_K\}$ where $\mathbf{c}_k = (c_{k1}, c_{k2}, ..., c_{kM})$, *k* = 1, 2, …, *K* are code vectors. Each code vector $\mathbf{c}_k$ is assigned to an encoding region R_k in the partition $\Omega = \{R_1, R_2, ..., R_K\}$. Then the source vector $\mathbf{x}_t$ can be represented by the encoding region R_k and expressed by

$$V(\mathbf{x}_t) = \mathbf{c}_k \quad \text{if} \quad \mathbf{x}_t \in R_k \tag{17}$$

Let $U = [u_{kt}]$ be a matrix whose elements are memberships of x_t in the nth cluster, *k* = 1, 2, …, *K*, *t* = 1, 2, …, *T*. A *K*-partition space for **X** is the set of matrices *U* such that

$$u_{kt} \in \{0,1\} \ \forall k, t, \quad \sum_{k=1}^{K} u_{kt} = 1 \ \forall t, \quad 0 < \sum_{t=1}^{T} u_{kt} < T \ \forall k \tag{18}$$

where $u_{kt} = u_k(\mathbf{x}_t)$ is 1 or 0, according to whether $\mathbf{x}_t$ is or is not in the *k*th cluster, $\sum_{k=1}^{K} u_{kt} = 1 \; \forall t$ means each $\mathbf{x}_t$ is in exactly one of the *K* clusters, and $0 < \sum_{t=1}^{T} u_{kt} < T \; \forall k$ means that no cluster is empty and no cluster is all of **X** because of 1 < *K* < *T*.

3.2 Fuzzy subspace VQ

The fuzzy subspace VQ method is based on minimization of the $Q_j(\Lambda, \overline{\Lambda})$ function in (14) considered as the following sum-of-squared-errors function (the index *j* for state is omitted)

$$J(U,W,\Lambda) = \sum_{k=1}^{K}\sum_{t=1}^{T} u_{kt} \sum_{m=1}^{M} w_{km}^{\alpha} d_{ktm} \tag{19}$$

where $\overline{\Lambda}$ is included in d_{ktm}, which is the Euclidean norm of $(\mathbf{x}_t - \mathbf{c}_k)$. Applying the equations (8) through (16), we obtain the following equations for fuzzy subspace VQ

$$\mathbf{c}_k = \sum_{t=1}^{T} u_{kt}\mathbf{x}_t \Big/ \sum_{t=1}^{T} u_{kt}, \quad 1 \le k \le K \tag{20}$$

$$u_{kt} = \begin{cases} 1: & d_{kt} < d_{jt}, \quad j = 1,...,K, j \ne k \\ 0: & otherwise \end{cases} \tag{21}$$

$$w_{km} = \frac{1}{\sum_{n=1}^{M} (D_{km} / D_{kn})^{1/(\alpha-1)}}, \; D_{km} = \sum_{t=1}^{T} u_{kt} d_{ktm} \tag{22}$$

where

$$d_{ktm} = (c_{km} - x_{tm})^2, \; d_{kt} = \sum_{m=1}^{M} w_{km}^{\alpha} d_{ktm}^2 \tag{23}$$

The fuzzy subspace VQ modeling algorithm is summarized as follows:

1. Given a training data set $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T\}$, where $\mathbf{x}_t = (x_{t1}, x_{t2}, ..., x_{tM})$, *t* = 1, 2,…, *T*.
2. Initialize memberships u_{kt}, $1 \le t \le T$, $1 \le k \le K$, at random satisfying (18)
3. Initialize weight values w_{km}, $1 \le k \le K$, $1 \le m \le M$ at random satisfying (13)
4. Given $\alpha \ne 1$ and $\varepsilon > 0$ (small real number)
5. Set *i* = 0 and $J^{(i)}(U,W,\Lambda) = 0$. Iteration:
    a. Compute cluster centers using (20)
    b. Compute distance components d_{ktm} and distances d_{kt} using (23)
    c. Update weight values using (22)

d. Update membership values using (21)

e. Compute $J^{(i+1)}(U,W,\Lambda)$ using (19)

f. If

$$\frac{J^{(i+1)}(U,W,\Lambda) - J^{(i)}(U,W,\Lambda)}{J^{(i+1)}(U,W,\Lambda)} > \varepsilon \tag{24}$$

set $J^{(i)}(U,W,\Lambda) = J^{(i+1)}(U,W,\Lambda)$, $i = i + 1$ and go to step (a).

4. Network anomaly detection

Assuming Λ is the *normal* model. Given an unknown network feature vector **x**, the task is to determine **x** is normal or intrusive. The following algorithm is proposed

1. Given an unknown network feature vector **x** and the *normal* model Λ
2. Set a threshold value θ
3. Calculate the minimum distance between **x** and Λ

$$d_{\min} = \min_k d(\mathbf{x}, \mathbf{c}_k) \tag{25}$$

where $d(.)$ is defined in (23) and $\mathbf{c}_k$ is the *k*th code vector in Λ.

4. If $d_{\min} < \theta$ then **x** is normal else **x** is intrusive

It can be seen that when the threshold value increases, the anomaly detection rate and the false alarm rate also increase. If the false alarm rate is fixed, we can determine the corresponding values for the threshold value and the anomaly detection rate.

5. Experimental results

5.1 Network data and attack types

We consider a sample dataset which is the KDD CUP 1999 dataset. This dataset was based on MIT Lincoln Lab intrusion detection dataset, also known as DARPA dataset (DARPA, KDD CUP 1999). The data was produced for "The Third International Knowledge Discovery and Data Mining Tools Competition", which was held in conjunction with the Fifth International Conference on Knowledge Discovery and Data Mining. The raw network traffic records have already been converted into vector format. Each feature vector consists of 41 features. The meanings of these features can be found in (Tran et al. 2007). In this paper, we ignore features with symbolic values.

The attacks listed in feature vectors of KDD CUP 1999 dataset come from MIT Lincoln intrusion detection dataset web site (KDD CUP 1999). The labels are mostly the same except a few discrepancies. The MIT Lincoln lab web site lists 2 types of buffer overflow attack: *eject* and *ffb*. The former explores the buffer overflow problem of *eject* program of Solaris, and the later explores the buffer overflow problem of *ffb* config program. Guessing user logon names and passwords through remote logon via telnet session is labeled as *guess_passwd* in the KDD CUP 1999 dataset, but listed as *dict* on the MIT Lincoln lab web site. Finally, we cannot find the counterparts of *syslog* and *warez* in the KDD CUP 1999 dataset. In addition to the attack labels, the KDD CUP 1999 dataset has also the label *normal*, which means that the traffic is normal and free from any attack.

5.2 Anomaly detection and false alarm results

The proposed method for network intrusion detection was evaluated using the KDD CUP 1999 data set for training and the *Corrected* data set for testing. For training, the number of feature vectors for training the *normal* model was set to 5000. For testing, there were not sufficient data for all attack types, so we selected the *normal* network pattern and the 5 attacks which were *ipsweep*, *neptune*, *portsweep*, *satan*, and *smurf*. The testing data set contains 60593 feature vectors for the *normal* network pattern, and 306, 58001, 354, 1633 and 164091 feature vectors for the five attacks, respectively.

We also conducted a set of experiments for the network data using the normalization technique as follows

$$x'_{tm} = \frac{x_{tm} - \mu_m}{\sigma_m}, \ \sigma_m = \frac{1}{T}\sum_{t=1}^{T} | x_{tm} - \mu_m | \tag{26}$$

where x_{tm} is the *m*th feature of the *t*th feature vector, μ_m the mean value of all *T* feature vectors for feature *m*, and σ_m the mean absolute deviation.

Anomaly detection rates versus false alarm rates are presented in Tables 1, 2, 3, and 4, where the codebook size is set to 4, 8, 16, and 32, respectively. The value of α was set to 4. All network data were normalised. We chose 5 false alarm rates (in %) which were 0.0, 0.1, 1.0, 10.0, and 100.0 to compare the corresponding anomaly detection rates for the standard VQ modelling and the proposed fuzzy subspace VQ modeling method. The ideal value for false alarm rate is 0.0, and from the 4 tables, we can see that the fuzzy subspace VQ performed outperformed the standard VQ modeling even with the smallest codebook size.

All the considered methods could not achieved the highest anomaly detection rate of 100% even though we changed the threshold value to accept all attack patterns (i.e., the false alarm rate is 100%). With codebook size of 32, the fuzzy subspace VQ modeling achieved very good results even with the lowest false alarm rate. The training data set contained 5000 feature vectors. If all training data for the *normal* pattern were used to train the model, the result would be better.

Modelling	False Alarm Rate (in %)				
	0.0	0.1	1.0	10.0	100.0
VQ	45.6	46.1	46.7	48.4	77.4
Fuzzy Subspace VQ	98.1	98.1	98.3	98.4	98.8

Table 1. Anomaly detection results (in %). Codebook size = 4

Modelling	False Alarm Rate (in %)				
	0.0	0.1	1.0	10.0	100.0
VQ	45.9	50.8	54.2	60.3	79.6
Fuzzy Subspace VQ	98.2	98.3	98.3	98.5	98.9

Table 2. Anomaly detection results (in %). Codebook size = 8

Modelling	False Alarm Rate (in %)				
	0.0	0.1	1.0	10.0	100.0
VQ	64.9	81.2	82.1	83.3	94.8
Fuzzy Subspace VQ	98.8	98.9	98.9	98.9	99.2

Table 3. Anomaly detection results (in %). Codebook size = 16

Modelling	False Alarm Rate (in %)				
	0.0	0.1	1.0	10.0	100.0
VQ	83.5	84.7	86.5	87.0	95.0
Fuzzy Subspace VQ	98.9	99.0	99.0	99.0	99.3

Table 4. Anomaly detection results (in %). Codebook size = 32

5. Conclusion

We have proposed a generic framework for soft subspace pattern recognition. The framework has been designed for continuous hidden Markov model. The framework for fuzzy subspace Gaussian mixture model has been extracted by setting the number of states in continuous hidden Markov model to one. With an assumption on covariance matrix and density, a fuzzy subspace model for vector quantization has been determined. The proposed methods are based on fuzzy *c*-means modeling to assign fuzzy weight values to features depending on which subspace they belong to. We have also applied the vector quantization model to anomaly network detection problem. We have used the KDD CUP 1999 dataset as the sample data to evaluate the proposed methods. The fuzzy subspace vector quantization method outperformed the standard vector quantization model.

6. References

Anderson R. and Khattak A. (1998). The use of Information Retrieval Techniques for Intrusion Detection, in First International Workshop on Recent Advances in Intrusion Detection (RAID'98), Louvain-la-Neuve, Belgium

Balasubramaniyan, J. S., Garcia-Fernandez, J. O. et al. (1998). An Architecture for Intrusion Detection using Autonomous Agents, in Proceedings of the 14th IEEE ACSAC, Scottsdale, AZ, USA, pp. 13-24

Caruso C. and Malerba D. (2004). Clustering as an add-on for firewalls, Data Mining, WIT Press

Chan, P. K., Mahoney, M. V., and Arshad, M. H. (2003). A Machine Learning Approach to Anomaly Detection, Technical Report CS-2003-06

DARPA Intrusion Detection Evaluation Data Sets 1999, available at http://www.ll.mit.edu/IST/ideval/data/data_ index.html

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1997). Maximum Likelihood from Incomplete Data via the EM algorithm, *Journal of the Royal Statistical Society*, Ser. B, 39: pp. 1-38

Eskin, E. (2000). Anomaly Detection over Noisy Data Using Learned Probability Distributions, in the 17th International Conference on Machine Learning, Morgan Kaufmann, San Francisco, USA, pp. 255-262

Furui, S. (1997). Recent advances in speaker recognition, *Patter Recognition Lett.*, vol. 18, pp. 859-872

Huang, J.Z., Ng, M.K., Rong, H., and Li, Z. (2005). Automated Variable Weighting in k-means Type Clustering, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, no. 5, pp. 657-668

Huang, X., Acero, A., Alleva, F., Huang, M., Jiang, L., and Mahajan, M. (1996). From SPHINX-II to WHISPER: Making speech recognition usable, chapter 20 in *Automatic Speech and Speaker Recognition, Advanced Topics*, edited by Chin-Hui Lee, Frank K. Soong, and Kuldip K. Paliwal, Kluwer Academic Publishers, USA, pp. 481-508

Jing, L., Ng., M. K., Huang, J. Z., (2007). An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data, *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 6, pp. 1026-1041

Juang, B.-H. (1998). The Past, Present, and Future of Speech Processing, *IEEE Signal Processing Magazine*, vol. 15, no. 3, pp. 24-48

KDD CUP 1999 Data Set, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

Stanifor, Hoagland and McAlerney (2002). Practical Automated Detection of Stealthy PortScans, *Journal of Computer Security*, vol. 10, no. 1, pp. 105-136

Kulkarni. V. G. (1995). *Modeling and analysis of stochastic systems*, Chapman & Hall, UK

Li X. and Ye N. (2004). Mining Normal and Intrusive Activity Patterns for Computer Intrusion Detection, in Intelligence and Security Informatics: Second Symposium on Intelligence and Security Informatics, Tucson, USA, Springer-Verlag, vol. 3073, pp. 1611-3349

Lee W. and Xiang D. (2001). Information theoretic measures for anomaly detection, in IEEE Synposium on Security and Privacy, pp. 130-143

Mahoney, M. V. and Chan, P.K. (2001). PHAD: Packet Header Anomaly Detection for Identifying Hostile Network Traffic, Technical report, Florida Tech., CS-2001-4

Mahoney, M. (2003). Network Traffic Anomaly Detection Based on Packet Bytes, Proc. ACM. Symposium on Applied Computing, pp. 346-350

Ourston, D., Matzner, S., et al. (2004). Coordinated Internet attacks: responding to attack complexity, *Journal of Computer Security*, vol. 12, pp. 165-190

Paxson, V. (1998). Bro: A system for detecting network intruders in real-time, in Proceedings of the 7th USENIX Security Symposium, Texas, USA, pp. 3-7

Portnoy, L., Eskin, E., and Stolfo, S. (2001). Intrusion detection with unlabeled data using clustering, in Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001), Philadelphia, USA, pp. 333-342

Rabiner, L. R., Juang B. H., and Lee, C. H., (1996). An Overview of Automatic Speech Recognition, chapter 1 in *Automatic Speech and Speaker Recognition, Advanced Topics*, edited by Chin-Hui Lee, Frank K. Soong, and Kuldip K. Paliwal, Kluwer Academic Publishers, USA, pp. 1-30

Rabiner, L. R. and Juang, B. H. (1993). *Fundamentals of speech recognition*, Prentice Hall PTR, USA

Sherif, J.S., Ayers, R. and Dearmond, T. G. (2003). Intrusion Detection: the art and the practice, Part 1. *Information Management and Computer Security*, vol. 11, no. 4, pp. 175-186

Sherif J.S. and Ayers R. (2003). Intrusion detection: methods and systems, Part II. *Information Management and Computer Security*, vol. 11, no. 5, pp. 222-229

Stolfo, S.J. , Fan, W., Lee, W., Prodromidis, A. and Chan, P.K. (2000). Cost-based Modeling and Evaluation for Data Mining With Application to Fraud and Intrusion Detection: Results from the JAM Project, in Proceedings of DARPA Information Survivability Conference and Exposition, 2000, pp. 1130-1144

Taylor C. and Alves-Foss, J. (2002). An Empirical Analysis of NATE: Network Analysis of Anomalous Traffic Events, in 10th New Security Paradigms Workshop, Virginia Beach, Virginia, USA, pp. 18-26

Taylor C. and Alves-Foss J. (2001). NATE: Network Analysis of Anomalous Traffic Events, a low-cost approach, in Proceedings of New Security Paradigms Workshop, Cloudcroft, New Mexico, USA, pp. 89-96

Tran D., Ma W., Sharma D. and Nguyen T. (2007). Fuzzy Vector Quantization for Network Intrusion Detection, IEEE International Conference on Granular Computing, Silicon Valley, USA

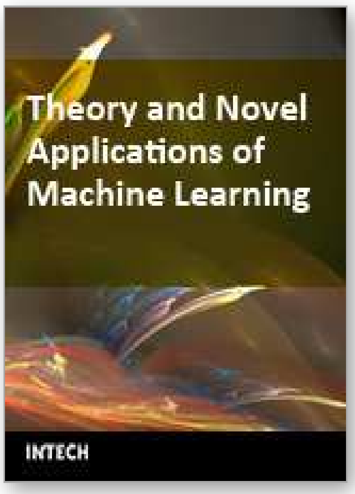
Tran D., Ma W., and Sharma D. (2008). Automated Feature Weighting for Network Anomaly Detection, *IJCSNS International Journal of Computer Science and Network Security*, Vol. 8 No. 2 pp. 173-178

Tran D. and Wagner M. (2002). Generalised Fuzzy Hidden Markov Models for Speech Recognition, *Lecture Notes in Computer Science: Advances in Soft Computing* - AFSS 2002, N.R. Pal, M. Sugeno (Eds.), pp. 345-351, Springer-Verlag.

Tran D. and Wagner M. (2000). A General Approach to Hard, Fuzzy, and Probabilistic Models for Pattern Recognition, *Advances in Intelligent Systems: Theory and Applications*, M. Mohammadian (ed.), pp. 244-251, IOS Press, Netherlands

Yasami, Y., Farahmand, M., Zargari, V. (2007). An ARP-based Anomaly Detection Algorithm Using Hidden Markov Model in Enterprise Networks, Second International Conference on Systems and Networks Communications, pp. 69 - 75

Yang, H., Xie, F. and Lu, Y. (2006). Clustering and Classification Based Anomaly Detection, Lecture Notes in Computer Science, vol. 4223, pp. 1611-3349



Theory and Novel Applications of Machine Learning
Edited by Meng Joo Er and Yi Zhou

ISBN 978-953-7619-55-4
Hard cover, 376 pages
Publisher InTech
Published online 01, January, 2009
Published in print edition January, 2009

Even since computers were invented, many researchers have been trying to understand how human beings learn and many interesting paradigms and approaches towards emulating human learning abilities have been proposed. The ability of learning is one of the central features of human intelligence, which makes it an important ingredient in both traditional Artificial Intelligence (AI) and emerging Cognitive Science. Machine Learning (ML) draws upon ideas from a diverse set of disciplines, including AI, Probability and Statistics, Computational Complexity, Information Theory, Psychology and Neurobiology, Control Theory and Philosophy. ML involves broad topics including Fuzzy Logic, Neural Networks (NNs), Evolutionary Algorithms (EAs), Probability and Statistics, Decision Trees, etc. Real-world applications of ML are widespread such as Pattern Recognition, Data Mining, Gaming, Bio-science, Telecommunications, Control and Robotics applications. This books reports the latest developments and futuristic trends in ML.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821