Discriminating Self from Non-Self with Finite Mixtures of Multivariate Bernoulli Distributions

Thomas Stibor
Department of Computer Science
Darmstadt University of Technology
64289 Darmstadt, Germany
stibor@sec.informatik.tu-darmstadt.de

ABSTRACT

Affinity functions are the core components in negative selection to discriminate self from non-self. It has been shown that affinity functions such as the r-contiguous distance and the Hamming distance are limited applicable for discrimination problems such as anomaly detection. We propose to model self as a discrete probability distribution specified by finite mixtures of multivariate Bernoulli distributions. As by-product one also obtains information of non-self and hence is able to discriminate with probabilities self from non-self. We underpin our proposal with a comparative study between the two affinity functions and the probabilistic discrimination.

Categories and Subject Descriptors

G.3 [**Probability and Statistics**]: [Multivariate Statistics, Experimental Design]; I.6.4 [**Simulation and Modeling**]: Model Validation and Analysis

General Terms

Algorithms

1. INTRODUCTION

Negative selection is an immune inspired algorithm for detecting anomalies in (binary) data [10]. Inspired by the censoring process of T-Lymphocytes, detector bit strings are censored such that no detectors match with any *self* bit string. After a repertoire of detectors is found, a bit string is classified as self, if no match between the bit string and any detector occurs, and otherwise as *non-self.* In recent years, different affinity functions for negative selection were proposed. Affinity functions define a closeness measure between a detector and the classified bit string [5]. Dilger [8] investigated metric properties of some affinity functions (Hamming and r-contiguous) and showed that not all metric properties are satisfied. González et al. [11] and Stibor et al. [15, 17] showed that the generalization capability of some affinity functions (Hamming, r-contiguous and r-chunk) are limited applicable for anomaly detection problems, because generalization regions occur also in non-self regions. Recently, it has been shown [16, 14] that finding r-contiguous detectors is equivalent to the problem of finding assignments sets for a Boolean formula in k-CNF. This result explained the lack of efficient algorithms for finding detectors.

Summarizing these results, it seems debatable whether these "classical" affinity functions (Hamming, r-contiguous and r-chunk) used in negative selection are appropriate as a closeness measure in self/non-self discrimination problems.

In this paper we discuss whether a probabilistic approach of modeling self can be applied to decide if a bit string belongs to self or non-self. The idea is to model self as a discrete probability distribution. As by-product one also obtains information of non-self and hence is able to discriminate with probabilities self from non-self. We structure this paper as follows: in section 2 the affinity functions (Hamming and r-contiguous) and the corresponding discrimination function are defined. The principle of self/non-self discrimination in negative selection is explained in section 3. The multivariate Bernoulli distribution, the parameter estimation with the EM-algorithm and the link to K-means are explained in sections 4, 4.1 and 4.2. The problem of non-identifiability of multivariate Bernoulli distributions is discussed in section 4.3. The proposed probabilistic self/non-self discrimination method is presented in section 5. Experiments and results are described and discussed in sections 6-6.3.

Throughout this paper sets are denoted in calligraphic letters, e.g. $\mathcal{S}$ and $|\mathcal{S}|$ denotes the cardinality of $\mathcal{S}$. Multivariate variables are denoted in bold letters.

2. AFFINITY AND DISCRIMINATION FUNCTION

Discriminating self from non-self by means of an affinity function in negative selection is originally proposed by Forrest et al. [10] and has formed the foundation for a large amount of work in the field of artificial immune systems.

Let $\mathcal{U}$ be a universe which contains all 2^l distinct bit string of length l. An affinity function $\mathfrak{A}$ takes as input two bit strings $\mathbf{a}, \mathbf{b} \in \mathcal{U}$, where $\mathbf{a} = a_1 a_2 \ldots a_l, \mathbf{b} = b_1 b_2 \ldots b_l$ and outputs some similarity magnitude, that is,

$$\mathfrak{A}(\mathbf{a}, \mathbf{b}) \rightarrow \mathbb{R}.$$

A matching function $\mathfrak{M}_r$ takes as input $\mathfrak{A}(\mathbf{a}, \mathbf{b})$ and thresh-

GECCO'08, July 12–16, 2008, Atlanta, Georgia, USA.
Copyright 2008 ACM 978-1-60558-130-9/08/07 ...$5.00.

old $\tau \in \mathbb{R}$ and outputs either self or non-self, that is,

$$\mathfrak{M}_\tau = \begin{cases} \mathfrak{A}(\mathbf{a},\mathbf{b}) \geq \tau, & \text{self} \\ \text{otherwise}, & \text{non-self.} \end{cases}$$

Frequently used affinity functions for $\mathfrak{A}$ are e.g. the Hamming distance and the r-contiguous distance which are defined below.

Definition 1. Given $\mathbf{a}, \mathbf{b} \in \mathcal{U}$, the Hamming distance between $\mathbf{a}$ and $\mathbf{b}$ is given by

$$H(\mathbf{a},\mathbf{b}) \quad = \quad \sum_{i=1}^{l} a_i \text{ XOR } b_i. \tag{1}$$

The r-contiguous distance can be defined by means of the Hamming distance.

Definition 2. Given $\mathbf{a}, \mathbf{b} \in \mathcal{U}$, the r-contiguous distance between $\mathbf{a}$ and $\mathbf{b}$ is given by

$$\begin{aligned} & R(\mathbf{a},\mathbf{b}) = j - i + 1 \\ & \text{where } (i,j) = \underset{1 \leq i \leq j \leq l}{\text{argmax}} \left(\prod_{t=i}^{j} \overline{a}_t \text{ XOR } b_t \right) \left(\sum_{t=i}^{j} \overline{a}_t \text{ XOR } b_t \right), \\ & \text{and } \overline{a}_t = 1 - a_t. \end{aligned} \tag{2}$$

The Hamming and the r-contiguous distance are not a metric in a strict mathematical sense. To be more precise, Dilger [8] showed that the triangle inequality is not satisfied in both distances. Nevertheless, these two distances are most frequently used in the field of artificial immune systems [5] as a closeness measure for discriminating self from non-self in binary data.

3. SELF/NON-SELF DISCRIMINATION IN NEGATIVE SELECTION

Given self set $\mathcal{S} \subset \mathcal{U}$ and some matching function $\mathfrak{M}_\tau$. To discriminate self from non-self, a detector set $\mathcal{D} \subset \mathcal{U}$ has to be generated, such that for all $\mathbf{d} \in \mathcal{D}$ and all $\mathbf{s} \in \mathcal{S}$, $\mathfrak{M}_\tau(\mathbf{d},\mathbf{s})$ outputs non-self. After the detector set $\mathcal{D}$ is generated, an (unseen) bit string $\mathbf{u} \in \mathcal{U}$ is classified as non-self, if $\mathfrak{M}_\tau(\mathbf{d},\mathbf{u})$ outputs non-self for all $\mathbf{d} \in \mathcal{D}$, otherwise as self. The principle of self/non-self discrimination in negative selection is depicted in Figure 1, where the r-contiguous distance is used and threshold[1] $\tau = r$. Note that $\mathcal{S}$ represents only a subset of the *true* self space, in other words $\mathcal{S}$ contains the *observed* self examples. To generalize beyond the observed self examples, the concept of holes is required. An unobserved bit string that is classified as self and is not a member of $\mathcal{S}$ is called hole. It is clear that holes have to present unobserved self data because the distance of all holes and all detectors is never greater than τ. For a further explanation of this concept see e.g. [7],[16].

4. FINITE MIXTURES OF MULTIVARIATE BERNOULLI DISTRIBUTIONS

The univariate Bernoulli distribution is a discrete probability distribution having two possible outcomes $x = 0$ and $x = 1$. Outcome $x = 1$ occurs with probability Θ and outcome $x = 0$ with probability $1 - \Theta$. It therefore has probability mass function

$$P(x|\Theta) = \Theta^x \, (1 - \Theta)^{1-x}. \tag{3}$$

Extending $P(x|\Theta)$ on the binary space $\{0,1\}^l$, one obtains the multivariate Bernoulli distribution with mass function

$$P(\mathbf{x}|\boldsymbol{\Theta}) = \prod_{i=1}^{l} \Theta_i^{x_i} \, (1 - \Theta_i)^{1-x_i} \tag{4}$$

where $\boldsymbol{\Theta} \in \mathbb{R}^l$ and $0 \leq \Theta_i \leq 1$ for all $1 \leq i \leq l$, and $x_1 x_2 \ldots x_l = \mathbf{x} \in \{0,1\}^l$.

Given an independent and identically distributed sample $\mathcal{X} = \{\mathbf{x}_t\}_{t=1}^N$ from $\{0,1\}^l$, the vector $\widehat{\boldsymbol{\Theta}}$ that maximizes (4) can be derived by means of the maximum likelihood estimation and results in

$$\widehat{\boldsymbol{\Theta}} \quad = \quad \frac{1}{N} \sum_{t=1}^{N} \mathbf{x}_t. \tag{5}$$

If sample $\mathcal{X}$ contains higher order correlations, then (5) gives an unsatisfiable result because the sample covariance matrix is diagonal. However, by combining M mixtures of multivariate Bernoulli distributions:

$$P(\mathbf{x}|\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) = \sum_{m=1}^{M} \alpha_m \, P(\mathbf{x}|\boldsymbol{\Theta}_m), \tag{6}$$

one can capture correlations in the sample. Note that mixture proportion $\boldsymbol{\alpha} \in \mathbb{R}^M$ has to obey the convex combination $\sum_{m=1}^M \alpha_m = 1$ with $\alpha_m \geq 0$ and $\overline{\boldsymbol{\Theta}}$ is composed of $(\boldsymbol{\Theta}_1, \boldsymbol{\Theta}_2, \ldots, \boldsymbol{\Theta}_M)$. For the sake of clearness, components of $\mathbf{x}_t$ are denoted as $(x_{t1} x_{t2} \ldots x_{tl}) = \mathbf{x}_t$ and components of $\boldsymbol{\Theta}_m$ as $(\Theta_{m1} \Theta_{m2} \ldots \Theta_{ml}) = \boldsymbol{\Theta}_m$ for $t = 1, \ldots, N$ and $m = 1, \ldots, M$.

The "fit" of parameters $(\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha})$ with regard to sample $\mathcal{X}$ can be measured in terms of the log-likelihood, that is:

$$\log \left(\prod_{t=1}^{N} P(\mathbf{x}_t|\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) \right) \quad = \quad \sum_{t=1}^{N} \log P(\mathbf{x}_t|\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}). \tag{7}$$

Maximizing term (7) by means of a large number of mixtures increases the model complexity and (usually) results in an overfitted model. To find the best trade-off between an appropriate model complexity and a large value of (7) one needs to penalize the model complexity. The Akaike information criterion (AIC) [1]:

$$\text{AIC} \quad = \quad 2k - 2 \sum_{t=1}^{N} \log P(\mathbf{x}_t|\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) \tag{8}$$

where $k = M - 1 + l \cdot M$, can be used to measure this trade-off. Parameter k denotes the number of parameters independently adjusted for the maximization of (7). Note that (8) is always positive and hence the preferred model is the one with the lowest AIC value.

4.1 Parameter Estimation with Expectation Maximization

Parameters $\overline{\boldsymbol{\Theta}}$ and $\boldsymbol{\alpha}$ that maximizes (7), can not be determined analytically when given sample $\mathcal{X}$. However, by

[1] The threshold is a positive integer value ($1 \leq r \leq l$) when using the Hamming and r-contiguous distance.

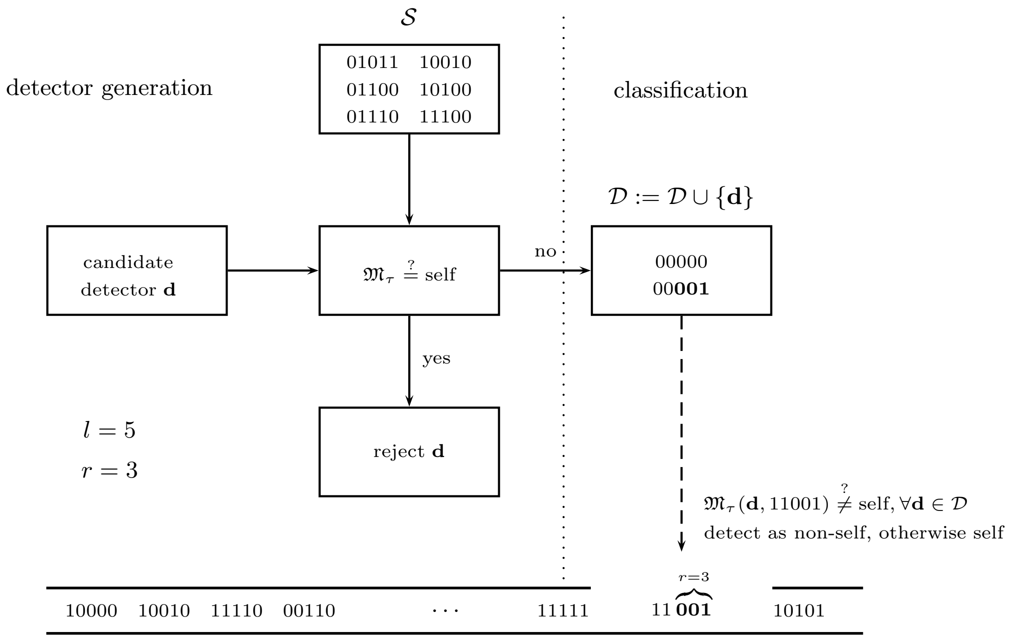

$\mathcal{S}$
01011 10010
01100 10100
01110 11100
detector generation
classification
$\mathcal{D} := \mathcal{D} \cup \{\mathbf{d}\}$
candidate detector $\mathbf{d}$
$\mathfrak{M}_\tau \overset{?}{=}$ self
no
00000
00**001**
yes
reject $\mathbf{d}$
$l = 5$
$r = 3$
$\mathfrak{M}_\tau(\mathbf{d}, 11001) \overset{?}{\neq}$ self, $\forall \mathbf{d} \in \mathcal{D}$
detect as non-self, otherwise self
$r=3$
10000 10010 11110 00110 ··· 11111 11**001** 10101

Figure 1: Principle of negative selection. Detectors are generated in a censoring process called negative selection such that no detector matches with any bit strings of $\mathcal{S}$. Bit strings from $\mathcal{U}$ are then classified with the generated detectors as self if $\mathfrak{M}_\tau$ outputs self, otherwise as non-self bit strings. In this figure the unseen bit string 11001 **is thus classified as non-self.**

differentiating componentwise (7) with regard to $\overline{\boldsymbol{\Theta}}$ and $\boldsymbol{\alpha}$

$$\frac{\partial}{\partial \Theta_{mi}} \sum_{t=1}^{N} \log P(\mathbf{x}_t | \overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) \tag{9}$$

$$\frac{\partial}{\partial \alpha_m} \sum_{t=1}^{N} \log P(\mathbf{x}_t | \overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) \tag{10}$$

one can derive the expectation and the maximization step within the EM-algorithm [6].

Recall, the probability of $\mathbf{x}_t$ being a member of mixture m can be obtained by means of the Bayes theorem, that is:

$$P(m|\mathbf{x}_t, \overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) = \frac{P(\mathbf{x}_t | m, \overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha})\, P(m)}{P(\mathbf{x}_t)} \tag{11}$$

$$= \frac{\alpha_m \prod_{i=1}^{l} \Theta_{mi}^{x_{ti}} \, (1 - \Theta_{mi})^{1 - x_{ti}}}{\sum_{m'=1}^{M} \alpha_{m'} \prod_{i=1}^{l} \Theta_{m'i}^{x_{ti}} \, (1 - \Theta_{m'i})^{1 - x_{ti}}}. \tag{12}$$

The E and M-step hence results in:

- E-step: determine the posterior probability (eq. 11) at iteration step s using current parameters $\boldsymbol{\alpha}^{(s)}$ and $\overline{\boldsymbol{\Theta}}^{(s)}$.
- M-step: determine reestimated parameters $\boldsymbol{\alpha}^{(s+1)}$ and $\overline{\boldsymbol{\Theta}}^{(s+1)}$ as follows:

$$\alpha_m^{(s+1)} = \frac{1}{N} \sum_{t=1}^{N} P(m|\mathbf{x}_t, \overline{\boldsymbol{\Theta}}^{(s)}, \boldsymbol{\alpha}^{(s)}) \tag{13}$$

$$\boldsymbol{\Theta}_m^{(s+1)} = \frac{1}{N\, \alpha_m^{(s+1)}} \sum_{t=1}^{N} P(m|\mathbf{x}_t, \overline{\boldsymbol{\Theta}}^{(s)}, \boldsymbol{\alpha}^{(s)}) \mathbf{x}_t. \tag{14}$$

This result is derived originally by Wolfe [18] and can be found also in the work of Everitt and Hand [9] (pp. 104) and in the recent work of Carreira-Perpiñán and Renals [4].

4.2 The Link between K-Means and the Estimation of $\overline{\Theta}$ and α

It is worthwhile to notice that the E and M-step (eq. (11)-(14)) has a clear and intuitive interpretation within the K-means algorithm[2]. The K-means algorithm is an iterative two-step algorithm which consists of an assignment and an update step. Given data points $\{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_N\}$ and randomly initialize cluster centers $\{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K\}$ both from $\mathbb{R}^d$. In the assignment step, each data point $\mathbf{p}_t$, $t = 1, 2, \ldots, N$ is assigned to the closest cluster center $\mathbf{c}_k, k = 1, 2, \ldots, K$. In the update step, the cluster centers $\mathbf{c}_k$ are adjusted to match the means of the data points that they are responsible for.

To be more precise, let $r_k^{(t)}$ denotes an indicator variable which is set to one if cluster center $\mathbf{c}_k$ is the closest mean to data point $\mathbf{p}_t$ and otherwise to zero. The two steps are performed as follows:

- Assignment step: determine responsibilities

  $$r_k^{(t)} = \begin{cases} 1 & \text{if } k = \operatorname{argmin}_j \|\mathbf{p}_t - \mathbf{c}_j\|^2 \\ 0 & \text{otherwise} \end{cases}$$

  that is, assign the t^{th} data point to the closest cluster center $\mathbf{c}_k$,

[2] This is not a surprising observation because one can derive the K-means algorithm as a particular limit of the EM-algorithm for Gaussian mixtures (see e.g. [2] for more details).

- Update step: recompute the cluster means $\mathbf{c}_k$ to match the means of the data points that they are responsible for, that is,

$$\mathbf{c}_k = \frac{\sum_{t=1}^{N} r_k^{(t)} \mathbf{p}_t}{\sum_{t=1}^{N} r_k^{(t)}}.$$

Repeat both steps until there is no further change in responsibilities or the maximum number of iterations is reached.

Note that indicator variable $r_k^{(t)}$ ensures that data points are assigned exactly to one cluster center and contribute with equal weight to that cluster center.

By transforming the indicator variable $r_k^{(t)}$ in a "soft" responsibility, that is, in a probability of bit string $\mathbf{x}_t$ being a member of mixture[3] k one obtains:

$$r_k^{(t)} \quad = \quad \frac{P(\mathbf{x}_t | k, \overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha})\, P(k)}{P(\mathbf{x}_t)} = P(k | \mathbf{x}_t, \overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}). \qquad (15)$$

Equivalent to the update step in K-means one obtains:

$$\alpha_k \quad = \quad \frac{\sum_{t=1}^{N} r_k^{(t)}}{N} = \frac{1}{N} \sum_{t=1}^{N} P(k | \mathbf{x}_t, \overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) \qquad (16)$$

$$\boldsymbol{\Theta}_k \quad = \quad \frac{\sum_{t=1}^{N} r_k^{(t)} \mathbf{x}_t}{\sum_{t=1}^{N} r_k^{(t)}} = \frac{\sum_{t=1}^{N} P(k | \mathbf{x}_t, \overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) \mathbf{x}_t}{N \frac{\sum_{t=1}^{N} r_k^{(t)}}{N}} \qquad (17)$$

$$\quad = \quad \frac{\sum_{t=1}^{N} P(k | \mathbf{x}_t, \overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) \mathbf{x}_t}{N \boldsymbol{\alpha}_k} \qquad (18)$$

which, in summary, result in the E and M-step within the EM-algorithm. It is important to notice that K-means is a local search algorithm. As a consequence, it converges to local minima and is sensitive to starting points, that is, the final solution highly depends on the initialized starting values. This fact consequently is also valid for the EM-algorithm. It is also important to notice that the EM-algorithm operates in *batch* mode. That is, all bit strings in $\mathcal{X}$ have to be stored and presented for performing the E and M-step. When processing large data sets, this consequently results in high computational complexity and becomes impractical. To overcome this problem Cappé and Moulines [3] proposed a generic version of the EM-algorithm which allows to perform the E and M-step without storing the complete data. Such an *online* version can also be applied here to determine the reestimated parameters $\boldsymbol{\alpha}^{(s+1)}$ and $\overline{\boldsymbol{\Theta}}^{(s+1)}$.

4.3 Non-identifiability Property

Multivariate Bernoulli distributions belong to the class of non-identifiable distributions. That means there exist distinct parameter vectors $(\boldsymbol{\alpha}, \overline{\boldsymbol{\Theta}})$ and $(\boldsymbol{\beta}, \overline{\boldsymbol{\Lambda}})$ (except the trivial permutations) that represent the same distribution.

Example 1. Let $\boldsymbol{\alpha} = \left(\frac{1}{3}, \frac{2}{3}\right)$, $\boldsymbol{\Theta}_1 = \left(\frac{1}{2}, \frac{1}{3}\right)$, $\boldsymbol{\Theta}_2 = \left(\frac{1}{4}, \frac{1}{5}\right)$. Writing down all probabilities results in:

$$\begin{aligned} P(00|\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) &= \tfrac{23}{45}, \quad & P(01|\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) &= \tfrac{7}{45}, \\ P(10|\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) &= \tfrac{11}{45}, \quad & P(11|\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) &= \tfrac{4}{45}. \end{aligned}$$

Finding different parameter vectors that represent the same distribution can be formulated in terms of determining the unknown $\boldsymbol{\beta}$ and $\overline{\boldsymbol{\Lambda}}$ in the following equation system:

$$\begin{aligned} \beta_1(1-\Lambda_{11})(1-\Lambda_{12}) + (1-\beta_1)(1-\Lambda_{21})(1-\Lambda_{22}) &= \tfrac{23}{45} \\ \beta_1(1-\Lambda_{11})(\Lambda_{12}) + \quad (1-\beta_1)(1-\Lambda_{21})(\Lambda_{22}) &= \tfrac{7}{45} \\ \beta_1(\Lambda_{11})(1-\Lambda_{12}) + \quad (1-\beta_1)(\Lambda_{21})(1-\Lambda_{22}) &= \tfrac{11}{45} \\ \beta_1(\Lambda_{11})(\Lambda_{12}) + \quad (1-\beta_1)(\Lambda_{21})(\Lambda_{22}) &= \tfrac{4}{45}. \end{aligned}$$

Solving this system gives free choice of $\Lambda_{21}, \Lambda_{22}$ and determined the rest:

$$\Lambda_{12} = \frac{11\Lambda_{21} - 4}{15(3\Lambda_{21} - 1)}, \quad \Lambda_{11} = \frac{15\Lambda_{22} - 4}{45\Lambda_{22} - 11} \quad \text{and}$$

$$\beta_1 = \frac{11 - 45\,\Lambda_{22} + 135\,\Lambda_{21}\Lambda_{22} - 33\,\Lambda_{21}}{3\,(45\,\Lambda_{21}\Lambda_{22} + 4 - 15\Lambda_{22} - 11\,\Lambda_{21})}.$$

By choosing for instance $\boldsymbol{\Lambda}_2 = \left(\frac{1}{6}, \frac{1}{7}\right)$, one obtains $\boldsymbol{\Lambda}_1 = \left(\frac{13}{32}, \frac{13}{45}\right)$, $\boldsymbol{\beta} = \left(\frac{16}{23}, \frac{7}{23}\right)$ and therefore preserves the same distribution, that is

$$P(\mathbf{x}|\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) = P(\mathbf{x}|\overline{\boldsymbol{\Lambda}}, \boldsymbol{\beta}) \quad \text{for all } \mathbf{x} \in \{0,1\}^2.$$

The general fact that *no* finite mixtures of multivariate Bernoulli distributions are identifiable is due to Gyllenberg et al. [12]. With respect to the problem of self/non-self discrimination (or pattern classification in general) however this fact seems to play a negligible role. Recall: the problem is to maximize term (7) and mutually to find a trade-off between the model complexity (i.e. the number of mixtures) and a small prediction error for unseen bit strings (i.e. to minimize the AIC value). If distinct parameter vectors satisfy this property, there exists no argument to prefer certain parameter vectors and discriminate the equivalent remaining ones.

5. PROBABILISTIC SELF/NON-SELF DISCRIMINATION

Given self set $\mathcal{S} \subset \mathcal{U}$ and parameter vectors $\boldsymbol{\alpha}$ and $\overline{\boldsymbol{\Theta}}$ that minimizes term (8) with regard to $\mathcal{S}$. A bit string $\mathbf{u} \in \mathcal{U}$ is classified as self, if

$$P(\mathbf{u}|\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) \geq \tau \qquad (19)$$

and otherwise as non-self. Note that, expression (19) corresponds to matching function $\mathfrak{M}_\tau$ presented in section 2.

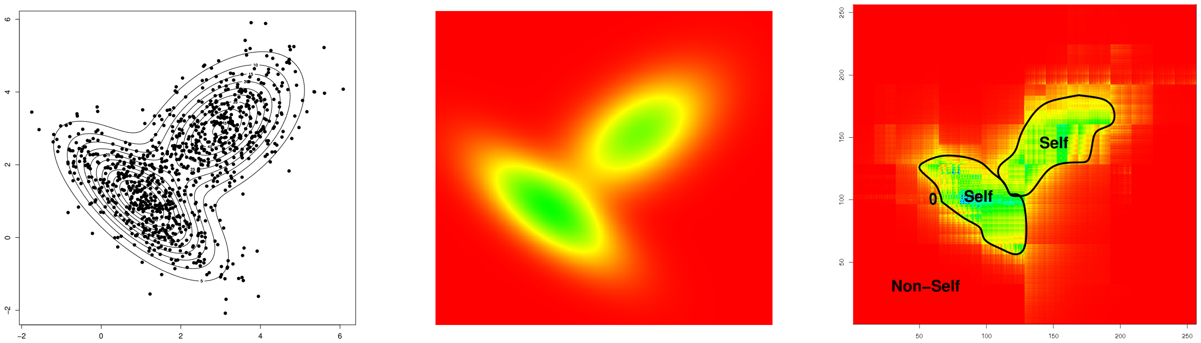
To minimize term (8) one can use different methods. Here we use the EM-algorithm to maximize the posterior probability (see section 4.1). Once the parameter vectors are determined, an appropriate value for τ can be determined by using techniques such as the leave-one-out method.

In this experiment however we focus on the problem whether it is feasible to estimate the distribution which approximates the true distribution closely when given self data only. In other words, is it feasible to determine decision boundaries that enclose *most* of the self data (see Fig. 2(c)). To demonstrate the applicability of this probabilistic self/non-self discrimination method, an artificially generated data set is created. The discrimination results are visualized and compared to results obtained with negative selection.

6. EXPERIMENTS

Self data is generated by a mixture of 2-dim. Gaussian distributions with different mean vectors and covariance matrices and consists of 5000 data points. The generated self data is visualized in Figure 2(a), the corresponding density image

[3] Note that in view of K-means this can be interpreted as the responsibility of $\mathbf{p}_t$ belonging to cluster center $\mathbf{c}_k$.



(a) Self data is generated by a mixture of two multivariate Gaussian distributions with different mean vectors and covariance matrices.

(b) Density image of the underlying distributions. Self data is concentrated in regions of high probability (green regions).

(c) Enclosed decision regions parametrized by finite mixtures of multivariate Bernoulli distributions. Data enclosed within the decision region is classified as self, and outside of the decision region as non-self. The decision boundary is parametrized as $P(\mathbf{u}|\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) = \tau$.

Figure 2: Self data is sampled from a mixture of multivariate Gaussian distributions.

is depicted in Figure 2(b). One can see in Figure 2(a) that self data is concentrated in regions of high density. This is a common assumption in the field of novelty detection and leads to the problem of finding regions where most of the normal data (in our terminology self data) is concentrated [13].

Note that the domain of (1),(2) and (19) is $\mathcal{U}$. We therefore use the mapping from $\mathbb{R}^2 \rightarrow \mathcal{U}$ proposed in [11]. That is, the data is min-max normalization to $[0,1]^2$ and discretized to bit strings of length $l = 16$

$$\underbrace{b_1, b_2, \ldots, b_8}_{b_x}, \underbrace{b_9, b_{10}, \ldots, b_{16}}_{b_y},$$

where the first 8 bits encode the integer x-value

$$i_x := \lceil 255 \cdot x + 0.5 \rceil$$

and the last 8 bits the integer y-value

$$i_y := \lceil 255 \cdot y + 0.5 \rceil,$$

that is,

$$\begin{aligned} [0,1]^2 &\rightarrow (i_x, i_y) \in (1, \ldots, 256) \times (1, \ldots, 256) \\ &\rightarrow (b_x, b_y) \in \{0,1\}^8 \times \{0,1\}^8. \end{aligned}$$

For the sake of clarity the binary self data (i.e. the sample) is denoted as $\mathcal{S}$.

6.1 Experimental Setup

The starting parameters for the EM-algorithm are randomly initialized as follows:

$$\Theta_{mi} \in_R [1/4, 3/4] \text{ for } i = 1 \ldots l \text{ and } m = 1 \ldots M$$

and the mixture proportions are deterministically initialized with:

$$\alpha_m = \frac{1}{M} \text{ for } m = 1 \ldots M.$$

In our experiment, we setup the EM-algorithm to terminate if between two succeeding iterations the log-likelihood of sample $\mathcal{S}$ is smaller than 10^{-2} or 1000 iterations are reached.

The negative selection experiments are setup as follows: for both affinity functions all detectors for each threshold $r = 1, \ldots, 16$ are generated.

6.2 Probabilistic Discrimination Results

The visualized results presented in Figure 6 are obtained as follows: the number of mixtures M is chosen and the parameter vectors $(\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha})$ are determined by means of the EM-algorithm. For all $\mathbf{u} \in \mathcal{U}$ the probability value of $P(\mathbf{u}|\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha})$ is calculated. This corresponds to the evaluation of all pixels in a 256×256 grid, where the color of each pixel is determined by the corresponding probability value. One can see (last page Figure 6(a)) that it is not feasible to reconstruct the higher order correlations of the true distribution by using only one mixture ($M = 1$). This is not a big surprise due to the fact that the EM-algorithm gives the same result as the maximum likelihood term (5) when using only one mixture. By increasing stepwise the number of the mixtures one also increases the model complexity. For some M the best trade-off between model complexity and generalization error is given, that is, the estimated distribution approximates the true distribution closely. In our experiment, values $M = 12, 13, \ldots, 20$ give satisfiable results. This can be observed in Figure 3 and Table 1. Informally this can also be seen by comparing results in Figures 6(a)-6(t) with result in Figure 2(b).

6.3 Negative Selection Discrimination Results

The negative selection discrimination results are depicted in Figure 4 and 5. One can see that for $r = 13$ no detectors can be generated, whereas for $r = 16$ each bit string (except the self bit strings) is covered by detectors. For $r = 14$ one can observe that holes (bit strings not covered by detectors) are *not* distributed within the region where most self data is concentrated. As a consequence, this implies a poor gen-

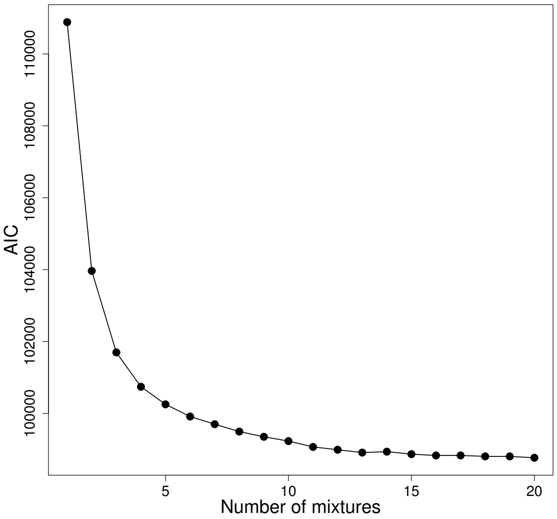


Figure 3: Coherence between number of mixtures and corresponding mean AIC values.

M	AIC	loglik	iter
1	110884.46 (± 0)	-55426.23 (± 0)	1 (± 0)
2	103963.92 (± 77.28)	-51948.96 (± 38.64)	22.40 (± 10.73)
3	101697.27 (± 195.27)	-50798.64 (± 97.63)	28.85 (± 12.24)
4	100740.92 (± 462.52)	-50303.46 (± 231.26)	40.70 (± 14.77)
5	100252.64 (± 264.54)	-50042.32 (± 132.27)	58.30 (± 28.11)
6	99914.02 (± 72.94)	-49856.01 (± 36.47)	95.45 (± 41.19)
7	99700.13 (± 158.75)	-49732.06 (± 79.37)	99.50 (± 41.20)
8	99494.73 (± 115.51)	-49612.36 (± 57.75)	105.00 (± 22.06)
9	99350.07 (± 135.60)	-49523.04 (± 67.80)	139.50 (± 37.38)
10	99230.21 (± 138.44)	-49446.11 (± 69.22)	146.55 (± 51.25)
11	99067.84 (± 83.42)	-49347.92 (± 41.71)	168.65 (± 42.41)
12	98989.64 (± 107.73)	-49291.82 (± 53.86)	183.20 (± 74.94)
13	98910.76 (± 71.90)	-49235.38 (± 35.95)	215.20 (± 74.61)
14	98936.35 (± 79.44)	-49231.18 (± 39.72)	185.35 (± 51.02)
15	98867.20 (± 50.26)	-49179.60 (± 25.13)	221.40 (± 74.88)
16	98829.95 (± 64.51)	-49143.98 (± 32.25)	265.90 (± 93.70)
17	98831.32 (± 97.91)	-49127.66 (± 48.95)	227.25 (± 55.32)
18	98805.13 (± 68.29)	-49097.57 (± 34.14)	269.20 (± 86.00)
19	98804.85 (± 47.29)	-49080.43 (± 23.64)	259.95 (± 70.02)
20	98766.77 (± 27.05)	-49044.38 (± 13.52)	302.45 (± 91.47)

Table 1: Mean AIC and log-likelihood results for stepwise increased number of mixtures. Values depicted in the brackets denote the standard deviation. The last column denotes the average number of iterations until convergence. The experiment was repeated 20 times.

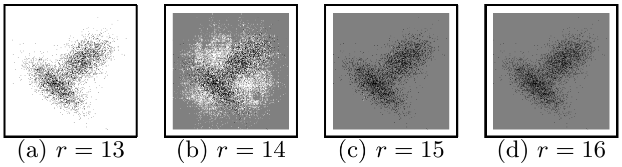
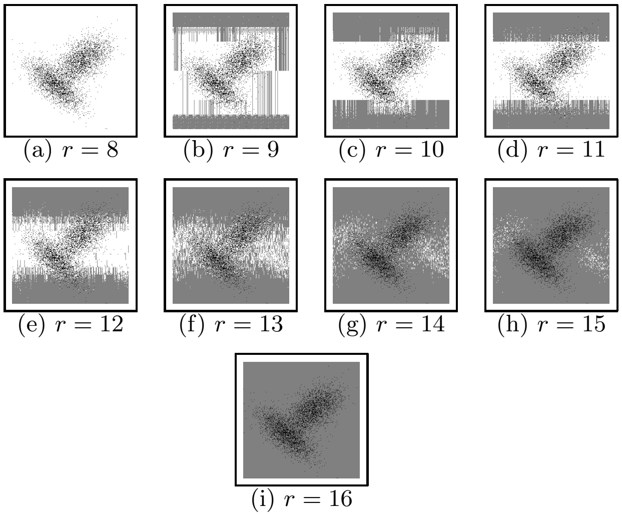

(a) $r = 13$ (b) $r = 14$ (c) $r = 15$ (d) $r = 16$

Figure 4: Detector coverage (Hamming distance) for threshold values $r = 13, \dots, 16$. The gray shaded area is covered by the generated detectors, the white area represents holes. The black points represent self examples ($|\mathcal{S}| = 5000$) which are generated by the underlying distribution (see Fig. 2(a)). Note that for $r = 1, 2, \dots, 13$ no detectors can be generated.


(a) $r = 8$ (b) $r = 9$ (c) $r = 10$ (d) $r = 11$

(e) $r = 12$ (f) $r = 13$ (g) $r = 14$ (h) $r = 15$

(i) $r = 16$

Figure 5: Detector coverage (r-contiguous distance) for threshold values $r = 8, \dots, 16$. Note that for $r = 1, \dots, 8$ no detectors can be generated.

eralization performance in terms of correctly discriminating unseen bit strings. Note that for $r = 1, 2, \dots, 13$ no detectors can be generated. The r-contiguous distance suffers under the same generalization problems as the Hamming distance, that is, unseen self bit strings (holes) are not concentrated in regions of high probability (see Fig. 5).

7. CONCLUSION

Discriminating self from non-self with negative selection is a popular method in the field of artificial immune systems. Latest research results, however, revealed several problems regarding the complexity of finding detectors and the generalization capabilities of the used affinity functions. To overcome these problems, we proposed to model self as a discrete probability distribution specified by finite mixtures of multivariate Bernoulli distributions. The EM-algorithm was used to find the parameters that maximize the likelihood, minimize the AIC value, respectively, of a given sample. As by-product we showed that the E and M-step within the EM-algorithm are linked to the iterated optimization steps performed in K-means clustering. Furthermore, the non-identifiability property of finite mixtures of multivariate Bernoulli distributions was discussed. A comparative study on the self/non-self discrimination capabilities of negative selection and the proposed probabilistic discrimination approach was performed. Results revealed that finite mixtures of multivariate Bernoulli distributions are a promising approach to tackle the self/non-self discrimination problem. To underpin this statement however, experiments on real-world data sets and with regard to classification rates are required in future work.

8. ACKNOWLEDGMENT

The author thanks Erin Gardner for her valuable suggestions and comments.

9. REFERENCES

[1] H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

[2] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag, 2006.

[3] O. Cappé and E. Moulines. Online EM algorithm for latent data models. Preprint available at http://arxiv.org/abs/0712.4273v1, 2007.

[4] M. A. Carreira-Perpiñán and S. Renals. Practical identifiability of finite mixtures of multivariate bernoulli distributions. *Neural Computation*, 12(1):141–152, 2000.

[5] L. N. de Castro and J. Timmis. *Artificial Immune Systems: A New Computational Intelligence Approach*. Springer Verlag, 2002.

[6] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

[7] P. D'haeseleer, S. Forrest, and P. Helman. An immunological approach to change detection: algorithms, analysis, and implications. In *Proceedings of the Symposium on Research in Security and Privacy*, pages 110–119. IEEE Computer Society Press, May 1996.

[8] W. Dilger. Structural properties of shape-spaces. In *Proceedings of the 5th International Conference on Artificial Immune Systems (ICARIS)*, volume 4163 of *Lecture Notes in Computer Science*, pages 178–192. Springer-Verlag, 2006.

[9] B. S. Everitt and D. J. Hand. *Finite Mixture Distributions*. Chapman and Hall, 1981.

[10] S. Forrest, A. S. Perelson, L. Allen, and R. Cherukuri. Self-nonself discrimination in a computer. In *Proceedings of the Symposium on Research in Security and Privacy*, pages 202–212. IEEE Computer Society Press, 1994.

[11] F. González, D. Dasgupta, and J. Gómez. The effect of binary matching rules in negative selection. In *Genetic and Evolutionary Computation (GECCO)*, volume 2723 of *Lecture Notes in Computer Science*, pages 195–206, Chicago, 12-16 July 2003. Springer-Verlag.

[12] M. Gyllenberg, T. Koski, E. Reilink, and M. Verlann. Non-uniqueness in probabilistic numerical identification of bacteria. *Journal of Applied Probability*, 31(2):542–548, 1994.

[13] I. Steinwart, D. Hush, and C. Scovel. A classification framework for anomaly detection. *Journal of Machine Learning Research*, 6:211–232, 2005.

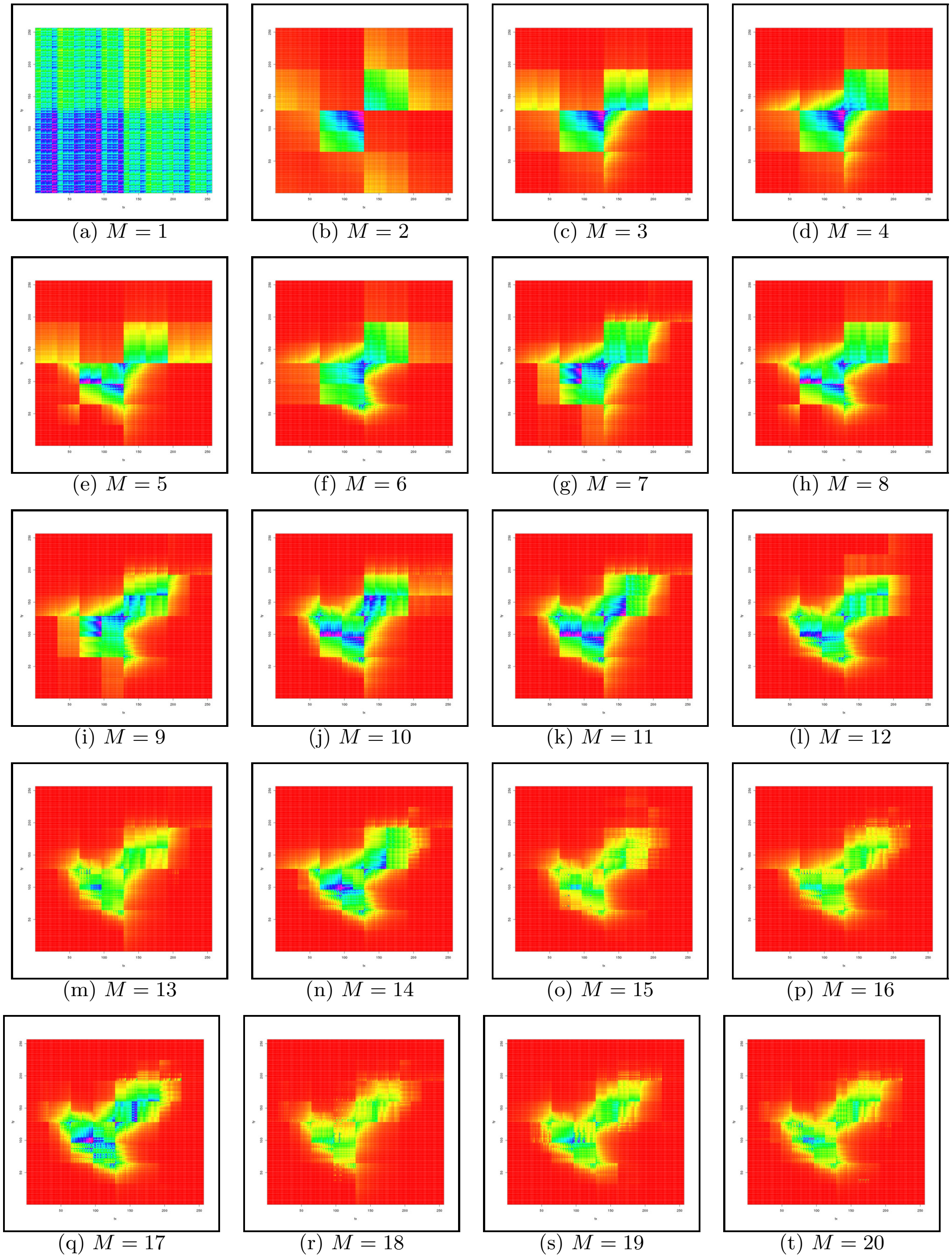
[14] T. Stibor. Phase transition and the computational complexity of generating r-contiguous detectors. In *Proceedings of 6th International Conference on Artificial Immune Systems (ICARIS)*, Lecture Notes in Computer Science, pages 142–155. Springer-Verlag, 2007.

[15] T. Stibor, J. Timmis, and C. Eckert. Generalization regions in hamming negative selection. In *Intelligent Information Processing and Web Mining*, Advances in Soft Computing, pages 447–456. Springer-Verlag, 2006.

[16] T. Stibor, J. Timmis, and C. Eckert. The link between r-contiguous detectors and k-CNF satisfiability. In *Proceedings of Congress On Evolutionary Computation (CEC)*, pages 491–498. IEEE Press, 2006.

[17] T. Stibor, J. Timmis, and C. Eckert. On permutation masks in hamming negative selection. In *Proceedings of 5th International Conference on Artificial Immune Systems (ICARIS)*, Lecture Notes in Computer Science, pages 122–135. Springer-Verlag, 2006.

[18] J. H. Wolfe. Pattern clustering by multivariate mixture analysis. *Multivariate Behavior Research*, 5:329–359, 1970.



(a) $M = 1$ (b) $M = 2$ (c) $M = 3$ (d) $M = 4$

(e) $M = 5$ (f) $M = 6$ (g) $M = 7$ (h) $M = 8$

(i) $M = 9$ (j) $M = 10$ (k) $M = 11$ (l) $M = 12$

(m) $M = 13$ (n) $M = 14$ (o) $M = 15$ (p) $M = 16$

(q) $M = 17$ (r) $M = 18$ (s) $M = 19$ (t) $M = 20$

Figure 6: Results of estimated probability distribution specified by finite mixtures of multivariate Bernoulli distributions for stepwise increased number of mixtures. Each pixel in the 256×256 grid represents a bit string $\mathbf{u}$ of length 16 bits. The color corresponds to the probability $P(\mathbf{u}|\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha})$. One can see that the *true* underlying distribution can be closely approximated with a certain number of mixtures.