A Generative Model for Self/Non-Self Discrimination in Strings

Matti Pöllä

Adaptive Informatics Research Centre,
Helsinki University of Technology,
FI-02015 Espoo, FINLAND
matti.polla@tkk.fi

Abstract. A statistical generative model is presented as an alternative to negative selection in anomaly detection of string data. We extend the probabilistic approach to binary classification from fixed-length binary strings into variable-length strings from a finite symbol alphabet by fitting a mixture model of multinomial distributions for the frequency of adjacent symbols. Robust and localized change analysis of text documents is viewed as an application area.

1 Introduction

Finding anomalies in a collection of data has been one of the most important research areas in the field of artificial immune system (AIS), i.e., computational methods inspired by the information processing of biological immune systems. The negative selection algorithm (NSA) by Forrest et al. [1] was originally presented as an immunology-inspired method for classifying bit strings into *self* or *non-self* using training examples only from the *self* class.

The NSA is an instance-based learning algorithm which produces a description of non-self for which no samples are available in the training phase. This is achieved by producing an initial detector collection which is then pruned according to negative selection based on the available self samples–that is: any detector that matches a self sample is rejected. The remaining collection of detectors are then used as an instance-based description of non-self data.

Recently, statistical methods have been shown to have good performance in anomaly detection tasks [2]. Among these, the one-class support vector machine [3] and probabilistic generative models [4] have been used as an alternative to instance-based learning. In the following sections, a generative model based on multinomial distributions is presented for anomaly detection in variable-length strings from an arbitrary symbol vocabulary.

In Section 2 we review the principle of negative selection based self/non-self discrimination, and in Section 3 we review a generative model for fixed-length binary strings and a related model for character frequencies. In Section 4 we present a generative model using multinomial distributions which can be seen as a hybrid of the models of [4] and [5]. Section 5 discusses the properties of natural language in terms of applying the developed model. Experimental result from a confined example and a more realistic experiment are presented in Section 6. Topics for further research are outlined discussion in Section 7 with some conclusions in Section 8.

2 Anomaly detection using negative selection

The negative selection approach to anomaly detection [6] is employs an instance-based representation of the unseen data (non-self). The set of all data vectors $\mathcal{U}$ contains the self set $\mathcal{S} \subset \mathcal{U}$ from which a set of self samples $\mathbf{s} \in \mathcal{S}$ is available in the training phase. The self samples are used to prune an initial (often stochastically generated) set $\mathcal{D}_0$ of detector strings such that all detectors $\mathbf{d} \in \mathcal{D}_0$ which have high affinity (similarity) with samples from $\mathcal{S}$ are removed. The affinity function $u(\mathbf{s}, \mathbf{d}) \rightarrow \mathbb{R}$ maps the similarity of two vectors as a real value and is selected to suit the application at hand.

In the pruning phase all self-matching detector candidates are removed from the initial set of detectors according to the discrimination rule

$$m_\tau = \begin{cases} u(\mathbf{s}, \mathbf{d}) \geq \tau, \text{ self} \\ u(\mathbf{s}, \mathbf{d}) < \tau, \text{ non-self} \end{cases} \quad \forall \mathbf{s} \in \mathcal{S}, \quad \forall \mathbf{d} \in \mathcal{D}_0 \tag{1}$$

After the censoring phase, any new vector $\mathbf{x}$ can be classified into non-self if a match between $\mathbf{x}$ and a detector $\mathbf{d} \in \mathcal{D}$ is found according to (1).

Compared to simply classifying according to a thresholded similarity with self samples (positive selection), the NSA has the benefit of being able to make the classification decision to non-self based on a single match between a detector and a data sample, whereas positive selection would require matching with each self sample before assigning $\mathbf{x}$ to non-self.

Originally the NSA was used for fixed-length binary strings and affinity was measured using bitwise-similarity metrics such as the Hamming distance or the related r-contiguous and r-chunk matching rules [7]. Since then, the NSA has been extended with various matching rules and data representation schemes from binary data into multidimensional real-valued vector data [8].

However, as an instance-based learning scheme the NSA suffers from the curse of dimensionality problem. Stibor et al. [9–11] have shown that in the case of matching bit strings using the r-contiguous bit rule there is no way to generate detectors efficiently as the problem can be reformulated as a k-CNF satisfiability problem. While the unique properties of NSA can be useful in some application domains, the process of searching for non-self matching detectors has limitations in scaling for high-dimensional data. This result motivates the use of statistical affinity measures over negative selection for strings from a non-binary alphabet.

3 Related work

3.1 Finite Bernoulli mixture models

Stibor [4] presented the use of finite multivariate Bernoulli mixtures as a generative model for discriminating self and non-self in l-dimensional bit strings. In this model, a bit string $\mathbf{x} \in \{0,1\}^l$ is considered to be generated by an l-dimensional Bernoulli distribution. In this discrete distribution the outcome of each bit can be either 1 with probability $P(x_i = 1) = \Theta$ or $x_i = 0$ with probability $P(x = 0) = 1 - \Theta$. The one dimensional probability distribution $P(x|\Theta) = \Theta^x(1-\Theta)^{1-x}$ can be extended for l-dimensional bit strings into

$$P(\mathbf{x}|\boldsymbol{\Theta}) = \prod_{i=1}^{l} \Theta_i^{x_i}(1-\Theta_i)^{1-x_i}, \quad x_i \in \{0,1\} \tag{2}$$

where the parameter vector $\boldsymbol{\Theta} = (\Theta_1, \Theta_2, ..., \Theta_l)$ contains the probabilities for each bit position.

To take into consideration the internal correlations in the data set $\mathcal{X} = \{\mathbf{x_1}, ..., \mathbf{x}_{|\mathcal{X}|}\}$, a linear mixture of M distributions can be used such that the mixture proportions of each component is defined by a parameter $\boldsymbol{\alpha} \in \mathbb{R}^M$, $\sum_{m=1}^{M} \alpha_m = 1$ and the probability of the mixture model generating the string $\mathbf{x}$ is thus

$$P(\mathbf{x}|\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) = \sum_{m=1}^{M} \alpha_m P(\mathbf{x}|\boldsymbol{\Theta}_m) \tag{3}$$

where the matrix $\overline{\boldsymbol{\Theta}} = (\boldsymbol{\Theta}_1, \boldsymbol{\Theta}_2, ..., \boldsymbol{\Theta}_M)$ contains the parameter vectors of each mixture component.

To find the maximum likelihood estimates for parameters for $\boldsymbol{\alpha}$ and $\overline{\boldsymbol{\Theta}}$ the EM algorithm [12] can be used to iteratively alternate between computing the posterior probabilities $P(m|\mathbf{x}, \boldsymbol{\alpha}, \overline{\boldsymbol{\Theta}})$ (E step) and re-estimating $\boldsymbol{\alpha}$ and $\overline{\boldsymbol{\Theta}}$ (M step). In the resulting mixture model, discrimination between self and non-self is done according to the thresholded probability (3) such that any string $\mathbf{x}$ for which $P(\mathbf{x}|\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) \geq \tau$ is classified as self and all other for which $P(\mathbf{x}|\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) < \tau$ are classified as non-self.

3.2 Negative representation of character statistics

In [11] Stibor et al. have shown that the use of the r-chunk matching rule becomes infeasible when the binary alphabet $\Sigma = \{0,1\}$ is changed into a larger symbol vocabulary. In specific, to generate a sufficient amount of detectors the r parameter needs to be close to the string length which results in an infeasible space complexity. Applying the r-chunk matching rule directly to language data where the size of the symbol vocabulary is typically above 20 is thus considered of little use.

Recently, Pöllä and Honkela [5] have used a probabilistic model to generate a negative description of a text document by examining the frequencies of individual characters in a sliding window of w characters. Using a character unigram model, the frequency x_i $(0 \leq x_i \leq w)$ of a specific character $i \in \Sigma$ in a multiset of w character has a Binomial distribution

$$P(x_i = k|p) = \binom{w}{k} p^k (1-p)^{w-k} \tag{4}$$

where p is the unigram probability of the character. This property is then used to produce a description of all character frequencies $x_i = k$ in a window of w adjacent characters which are not observed in $\mathcal{S}$.

This approach can be considered as a compromise between a classical negative selection algorithm and a probabilistic self-model since the idea of non-self detectors is used but without the need for inefficient negative selection of detectors since the size of the initial detector population is limited to $|\mathcal{D}_0| = |\Sigma|(w+1)$.

4 Multinomial mixture model for non-binary strings

By combining the ideas of modeling self using a parameterized distribution and the sliding window of characters approach, the two can be combined into a generative model using a multinomial distribution and define non-self as any string for which the probability of being generated by the statistical model does not reach a threshold frequency.

Let Σ be an alphabet of symbols and let D be a string from Σ. The size (cardinality) of the alphabet is denoted as $|\Sigma|$ and the length of the document as $|D|$. Further, let $\mathbf{x}$ be a $|\Sigma|$-dimensional categorical random variable counting the frequency of each symbol $i \in \Sigma$ in a window of w adjacent symbols in D. Assuming an independent probability Θ_i for each symbol in Σ, the probability of $\mathbf{x}$ has a multinomial distribution

$$P(\mathbf{x}|\mathbf{\Theta}) = \frac{w!}{\prod_{i=1}^{|\Sigma|} x_i!} \prod_{i=1}^{|\Sigma|} \Theta_i^{x_i} \tag{5}$$

where $\sum_{i=1}^{|\Sigma|} \Theta_i = 1$ and $\sum_{i=1}^{|\Sigma|} x_i = w$. To fit this model for a specific dataset $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{|D|-w+1}\}$ we can find the maximum likelihood estimate for parameters $\mathbf{\Theta}$ by maximizing the likelihood function

$$\mathcal{L}(\mathbf{\Theta}|\mathcal{X}) = \prod_{j=1}^{|\mathcal{X}|} P(\mathbf{x}_j|\mathbf{\Theta}) = \prod_{j=1}^{|\mathcal{X}|} \left(\frac{w!}{\prod_{i=1}^{|\Sigma|} x_{ji}!} \prod_{i=1}^{|\Sigma|} \Theta_i^{x_{ji}} \right) \tag{6}$$

where x_{ji} is the frequency character i in the jth training sample resulting in

$$\mathbf{\Theta}_{\text{ML}} = \frac{1}{|\mathcal{X}|} \sum_{j=1}^{|\mathcal{X}|} \mathbf{x}_j \tag{7}$$

where $|\mathcal{X}|$ is the number of available training samples.

Depending on the data set $\mathcal{X}$ at hand, a single multinomial model can be insufficient to capture the internal correlations in the data and a finite mixture model is justified. For a mixture of M multinomials, the probability of $\mathbf{x}$ is

$$P(\mathbf{x}|\boldsymbol{\alpha}, \overline{\boldsymbol{\Theta}}) = \sum_{m=1}^{M} \alpha_m P(\mathbf{x}|\boldsymbol{\Theta}_m) = \sum_{m=1}^{M} \alpha_m \frac{w!}{\prod_{i=1}^{|\Sigma|} x_i!} \prod_{i=1}^{|\Sigma|} \Theta_{mi}^{x_i} \tag{8}$$

where the coefficients α_m define the mixture proportions of the multinomials defined by $\overline{\boldsymbol{\Theta}} = (\boldsymbol{\Theta}_1, \boldsymbol{\Theta}_2, ..., \boldsymbol{\Theta}_M)$. However, for the mixture model, the optimal values for $\boldsymbol{\alpha}$ and $\overline{\boldsymbol{\Theta}}$ cannot be solved analytically. As in [4] the EM algorithm can be used to alternate between determining the posterior probability and computing new parameter values. The E- and M-steps for a multinomial mixture model (as presented in [13]) are as follows:

- E-step: use the current parameters $\overline{\boldsymbol{\Theta}}$ and $\boldsymbol{\alpha}$ to compute the posterior probability of each sample $\mathbf{x}_j$ being generated by mixture component m

$$\begin{aligned} P(m|\mathbf{x}_j, \overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) &= \frac{P(\mathbf{x}_j|m, \overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha})P(m)}{P(\mathbf{x}_j)} \\ &= \frac{\alpha_m \prod_{i=1}^{|\Sigma|} \Theta_{mi}^{x_i}}{\sum_{m'=1}^{M} \alpha_{m'} \prod_{i=1}^{|\Sigma|} \Theta_{m'i}^{x_i}} \end{aligned} \tag{9}$$

- M-step: compute new parameters $\overline{\boldsymbol{\Theta}}^{(t+1)}$ and $\boldsymbol{\alpha}^{(t+1)}$ according to the new posterior probabilities

$$\boldsymbol{\alpha}_m^{(t+1)} = \frac{1}{|\mathcal{X}|} \sum_{j=1}^{|\mathcal{X}|} P(m|\mathbf{x}_j, \overline{\boldsymbol{\Theta}}^{(t)}, \boldsymbol{\alpha}^{(t)}) \tag{10}$$

$$\Theta_{mi}^{(t+1)} = \frac{\sum_{j=1}^{|\mathcal{X}|} x_{ji} P(m|\mathbf{x}_j, \overline{\boldsymbol{\Theta}}^{(t)}, \boldsymbol{\alpha}^{(t)})}{\sum_{r=1}^{|\Sigma|} \sum_{j=1}^{|\mathcal{X}|} x_{jr} P(m|\mathbf{x}_j, \overline{\boldsymbol{\Theta}}^{(t)}, \boldsymbol{\alpha}^{(t)})} \tag{11}$$

where x_{ji} is the frequency of character i in the jth training vector and Θ_{mi} is the parameter Θ_i of the mth component of the mixture.

After computing the parameters $\overline{\boldsymbol{\Theta}}$ and $\boldsymbol{\alpha}$ for a dataset $\mathcal{X}$, the discrimination between self and non-self can be made by setting a threshold probability τ such that any $\mathbf{x}$ for which $P(\mathbf{x}|\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) \geq \tau$ is classified as self and $P(\mathbf{x}|\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha}) < \tau$ as non-self.

To correctly classify all samples $\mathbf{s} \in \mathcal{S}$ into self, the threshold probability should be set to

$$\tau = \min\{P(\mathbf{s}|\overline{\boldsymbol{\Theta}}, \boldsymbol{\alpha})\}, \quad \forall \mathbf{s} \in \mathcal{S} \tag{12}$$

in order to have the threshold probability as high as possible while still classifying the self samples correctly.

5 Applicability to written language

Difficulties in statistical language modeling often arise from the problem of data sparsity (i.e., insufficient amount of available training data in relation to the dimensionality of the data). Modeling documents using word n-grams statistics is a common approach to gain information about the contents of documents though losing much information in ignoring word order. A bag-of-characters representation of text extends this tradeoff even further as only a fraction of the entropy in the text is preserved. Thus the character-based analysis is limited to simpler tasks such as anomaly detection, language identification [14] and authorship attribution [15].

Anomaly detection in textual data is closely related to the problem of document classification in information retrieval where document membership in a category is often model as a posterior probability using a statistical model. The one-class support vector machine has been applied in document classification [16] tasks with various document representation schemes [17]. A bag-of-words based multinomial model has been used by Novovičová and Malík [13] in document classification with improved results compared to a naïve Bayesian classifier.

6 Experiments

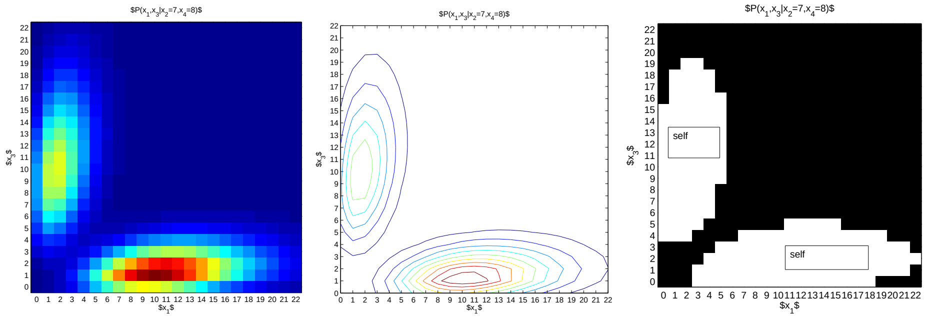
6.1 Mixture model for a 4-symbol vocabulary

A simple training set consisting of four-character strings from vocabulary $\Sigma = \{\text{a,b,c,d}\}$ is used to fit a multinomial mixture of two components for self/non-self discrimination. The training data set consists of strings where the characters have a strong correlation such that each string in the training data set consists of an equal amount of 'a' and 'b' or alternatively 'c' and 'd' (e.g. 'baab', 'bbaa', 'dccd', or 'cdcd'). The multinomial parameters $\overline{\boldsymbol{\Theta}}$ are initialized randomly in $[0,1]^4$ and the mixture coefficients are initially set to $\boldsymbol{\alpha} = (0.5,\ 0.5)$. After 170 iteration rounds using EM, the mixture model has learned the parameters

$$\begin{bmatrix}\alpha_1\\ \alpha_2\end{bmatrix} = \begin{bmatrix}0.475\\ 0.525\end{bmatrix} \qquad \begin{bmatrix}\boldsymbol{\Theta}_1\\ \boldsymbol{\Theta}_2\end{bmatrix} = \begin{bmatrix}0.07 & 0.07 & 0.43 & 0.43\\ 0.41 & 0.41 & 0.09 & 0.09\end{bmatrix}$$

A conditional probability distribution for this mixture model is shown in Figure 1a with various probability contours for selecting the threshold probability τ in Figure 1b. Classification regions are shown for a threshold frequency of $\tau = 0.1129$ in Figure 1c.

Table 1 presents a listing of all $\binom{4+4-1}{4} = 35$ possible 4 character multisets in a descending order of probability according to the mixture model. For example, by setting $\tau = 0.09$ the model classifies each permutation of strings “aabb” and “ccdd” as self and everything else as non-self.



a) b) c)

Fig. 1. Conditional probability distributions $P(x_1, x_3|x_2 = 7, x_4 = 8)$ for strings generated by a mixture model $m = 2$, $|\Sigma| = 4$ (a), contour lines for possible threshold probabilities τ (b) and decision regions for self and non-self for a given τ (c).

Table 1. List of all 35 possible 4-character multisets in a descending order of probability.

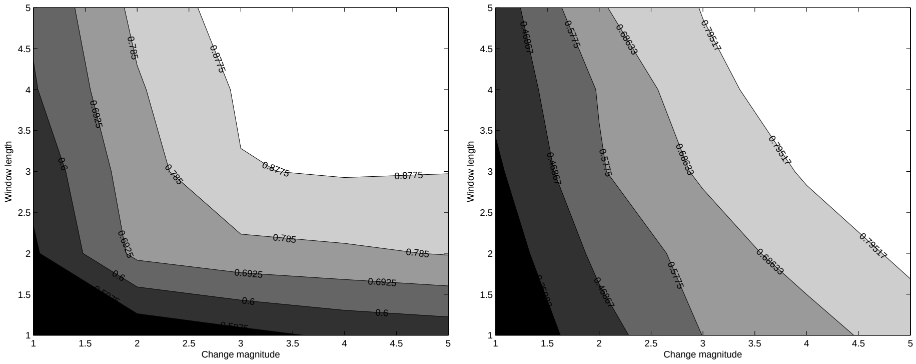
$\mathbf{x}$	D example	$P(\mathbf{x}\|\boldsymbol{\alpha}, \overline{\boldsymbol{\Theta}})$	$\mathbf{x}$	D example	$P(\mathbf{x}\|\boldsymbol{\alpha}, \overline{\boldsymbol{\Theta}})$
(0 0 2 2)	“cdcd”	**0.098831**	(0 4 0 0)	“bbbb”	0.015445
(2 2 0 0)	“abab”	**0.092671**	(3 0 1 0)	“aaac”	0.013075
(0 0 3 1)	“cccd”	0.065887	(3 0 0 1)	“aaad”	0.013075
(0 0 1 3)	“dddc”	0.065887	(0 3 1 0)	“bbbc”	0.013075
(3 1 0 0)	“aaab”	0.061781	(0 3 0 1)	“bbbd”	0.013075
(1 3 0 0)	“bbba”	0.061781	(2 0 1 1)	“aacd”	0.012973
(2 1 1 0)	“aabc”	0.039224	(1 1 2 0)	“abcc”	0.012973
(2 1 0 1)	“aabd”	0.039224	(1 1 0 2)	“abdd”	0.012973
(1 2 1 0)	“abbc”	0.039224	(0 2 1 1)	“bbcd”	0.012973
(1 2 0 1)	“abbd”	0.039224	(1 0 3 0)	“accc”	0.011022
(1 0 2 1)	“accd”	0.033065	(1 0 0 3)	“addd”	0.011022
(1 0 1 2)	“acdd”	0.033065	(0 1 3 0)	“bccc”	0.011022
(0 1 2 1)	“bccd”	0.033065	(0 1 0 3)	“bddd”	0.011022
(0 1 1 2)	“bcdd”	0.033065	(2 0 2 0)	“aacc”	0.006487
(1 1 1 1)	“abcd”	0.025946	(2 0 0 2)	“aadd”	0.006487
(0 0 4 0)	“cccc”	0.016472	(0 2 2 0)	“bbcc”	0.006487
(0 0 0 4)	“dddd”	0.016472	(0 2 0 2)	“bbdd”	0.006487
(4 0 0 0)	“aaaa”	0.015445			

6.2 Anomaly detection in written English

Anomaly detection in written natural language was simulated by using short segments from the Reuters corpus[1] and modifying a part of the string to test the sensitivity of detection. As a preprocessing stage, a lowercase conversion was made and all punctuation was removed from the data to limit the symbol vocabulary into 26 characters ('a' to 'z').

In this experiment, a 20-character string form the corpus was selected at random and a single multinomial model was computed from the string by setting $w = 20$. A random segment of 1 to 20 characters was then replaced to simulate an edit in the original string and the probability of the model generating the modified string was used to detect the anomaly.

Figure 2a shows the detection rate (proportion of successful detection) for various window lengths and sizes of the modified segment when a substring of 1 to 5 characters was replaced with a random character. Figure 2b shows the same result when a substring was swapped with another substring of the Reuters corpus.



a)

b)

Fig. 2. Detection rate of replacing a substring of 1 to 5 characters with a random string (a) and another segment of the Reuters corpus (b) into the original string D ($|D| = 20$). Window length on the vertical axis. Mean result of 1000 trials.

In Figure 3 the probability (5) of the multiset of characters is shown for each $|D| - w + 1$ window positions for $|D| = 200$ and $w = 100$. An edit in the original string has resulted in probability values which are lower than the threshold $\tau = 4 \cdot 10^{-17}$ and the change is detected.

[1] http://about.reuters.com/researchandstandards/corpus/

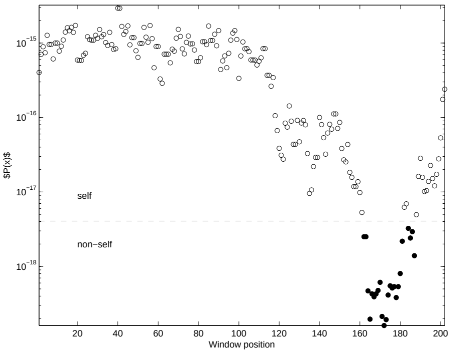


Fig. 3. Probability of each character multiset in a document. The changed part in the document is detected in the region where the probability is below the threshold τ (black dots).

7 Discussion and future work

Negative representations and statistical generative models are recent developments in text change detection where traditional tools based on hash functions and line-by-line comparisons have been the standard approach.

Many practical questions related to applying the multinomial mixture model for natural language are yet to be answered. Using mixture models involves making a compromise between the model complexity and the approximation accuracy by selecting an appropriate number of mixtures. In the confined example of Section 6.1 the number of components could be easily defined *a priori* using information on the correlation structure of the data. However, the problem of selecting an appropriate model complexity (i.e., using the Akaike information criterion) is a relevant topic for further research.

8 Conclusions

Biologically inspired anomaly detection based on negative selection suffers from the curse of dimensionality when extending standard NSA algorithms to non-binary strings. Recent work on statistical models for self/non-self discrimination are thus expected to be more successful for textual data. A generative model for variable-length strings from a general finite symbol alphabet was presented for the application of change detection in textual data. The use of multinomial models on the character and word level was discussed.

Our experiments on artificial data showed that the use of a probability based similarity measure in binary classification is justified especially if there are strong correlations in the data and if information on the symbol order in a set of w adjacent symbols can be omitted for anomaly detection. A large scale experiment on natural language was considered necessary to evaluate the performance of the proposed model in practical settings.

References

1. Forrest, S., Perelson, A.S., Allen, L., Cherukuri, R.: Self-nonself discrimination in a computer. In: Proceedings of the 1994 IEEE Symposium on Research in Security and Privacy, Oakland, CA, IEEE Computer Society Press (1994) 202–212
2. Stibor, T.: An empirical study of self/non-self discrimination in binary data with a kernel estimator. In: Proceedings of 7th International Conference on Artificial Immune Systems (ICARIS). Volume 5132 of Lecture Notes in Computer Science., Springer-Verlag (2008) 352–363
3. Schölkopf, B., Smola, A.J.: Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press, Cambridge, MA, USA (2001)
4. Stibor, T.: Discriminating self from non-self with finite mixtures of multivariate Bernoulli distributions. In: Proceedings of Genetic and Evolutionary Computation Conference – GECCO, ACM Press (2008) 127–134
5. Pöllä, M., Honkela, T.: Change detection of text documents using negative first-order statistics. In: Poceedings of AKRR'08, The Second International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning, Porvoo, Finland (September 2008) 48–55
6. D'haeseleer, P.: An immunological approach to change detection: theoretical results. In: Proceedings of the 9th Computer Security Foundations Workshop, IEEE Computer Society Press (1996) 18–26
7. de Castro, L.N., Timmis, J., eds.: Artificial Immune Systems: A New Computational Intelligence Approach. Springer-Verlag (2002)
8. González, F.A.: Anomaly detection using real-valued negative selection. Genetic programming and evolvable machines. In: Journal of Genetic Programming and Evolvable Machines. (2003) 4–383
9. Stibor, T., Timmis, J., Eckert, C.: The link between r-contiguous detectors and k-CNF satisfiability. In: Congress On Evolutionary Computation – CEC, IEEE Press (2006) 491–498 Revised and extended version.
10. Stibor, T., Mohr, P., Timmis, J., Eckert, C.: Is negative selection appropriate for anomaly detection? In: GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation, New York, NY, USA, ACM (2005) 321–328
11. Stibor, T., Bayarou, K.M., Eckert, C.: An investigation of R-chunk detector generation on higher alphabets. In: Proceedings of Genetic and Evolutionary Computation Conference – GECCO-2004. Volume 3102 of Lecture Notes in Computer Science., Springer-Verlag (2004) 299–307
12. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum-likelihood from incomplete data via the EM algorithm. Journal of Royal Statistical Society B **39** (1977) 1–38
13. Novovičová, J., Malík, A.: Application of multinomial mixture model to text classification. In: Pattern Recognition and Image Analysis. Volume 2652 of Lecture Notes in Computer Science., Springer Berlin / Heidelberg (2003) 646–653
14. Cavnar, W.B., Trenkle, J.M.: N-gram-based text categorization. (1994) 161–175
15. Keselj, V., Peng, F., Cercone, N., Thomas, C.: N-gram-based author profiles for authorship attribution (2003)
16. Manevitz, L.M., Yousef, M.: One-class SVMs for document classification. Journal of Machine Learning Research **2** (2001) 139–154
17. Srihari, X.W.R., Zheng, Z.: Document representation for one-class SVM. In: Proceedings of ECML 2004: European conference on machine learning. Volume 3201 of Lecture Notes in Computer Science., Springer Berlin / Heidelberg (2004) 489–500