Gröbner Basis of Non-Negative Matrix Factorization and Feature Extraction of Cross-Site Scripting Attacks

Takeshi Matsuda[1,a)]

Abstract: Non negative matrix factorization (NMF) is the method to decompose a non negative matrix into two non negative matrices. NMF is used in many fields for extracting some features and the effectiveness had been recognized in the application for the analysis of sound signal or text mining. In this paper, we investigate on an affine algebraic variety of NMF, and propose the feature extraction method of cross-site scripting attacks.

1. Introduction

NMF is used as one of the feature extraction methods, and it is used in many fields such as text mining [1], bioinformatics [2] [3], spectral data analysis [4] and image processing [5], and the effectiveness has been widely recognized. The derivation method of NMF had been studied (for example [6]), but the result of NMF depends strongly on the choice of initial numbers [7].

In this paper, we investigate the property of an affine algebraic variety of NMF, and propose the feature extraction method of cross-site scripting attacks by using the property. We had already proposed the classification method of SQL injection attacks by using the algebraic property of NMF [8]. In [8], we considered the following decomposition, and classified SQL injection attack by using the information of $(a_{11}a_{12})$ and $(a_{21}a_{22})$.

$$\begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}$$

Here, we consider the following decomposition $X = AB$,

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \end{pmatrix}$$
$$A = \begin{pmatrix} a_{11} & a_{12} \end{pmatrix}$$
$$B = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \end{pmatrix}$$

and propose the detection method by using the information of A and B.

2. Non Negative Matrix Factorization

Let us consider the following decomposition.

$$X = AB.$$

Here, we define

$X : P \times S$ matrix
$A : P \times R$ matrix
$B : R \times S$ matrix
$x_{ps} : (p, s)$ element of X
$a_{pr} : (p, r)$ element of A
$b_{rs} : (r, s)$ element of B

A lot of algorithms for getting NMF had been studied [6] [9]. In this section, we will introduce the algorithm of Lee and Seung's multiplicative update rule [6]. The multiplicative update rule is given by

$$a_{pr} = a_{pr} \frac{[XB^{\mathrm{T}}]_{pr}}{[ABB^{\mathrm{T}}]_{pr}}$$
$$b_{rq} = b_{rq} \frac{[A^{\mathrm{T}}X]_{rq}}{[A^{\mathrm{T}}AB]_{rq}}.$$

Here, $[\cdot]$ is matrix and $[\cdot]_{ij}$ is (i, j) element of the matrix $[\cdot]$. This multiplicative update rule is derived from the following optimization problem.

$$\text{minimize} \quad \|X - AB\|^2$$
$$\text{subject to} \quad a_{pr} \geq 0, b_{rq} \geq 0.$$

Here, $\|\cdot\|$ is a Frobenius norm. The calculation result of the multiplicative update rule depends on an initial value of a_{pr} and b_{rq} [7]. NMF is determined from the zero set of polynomial from $X = AB$. Therefore, it can be said that the decomposition independs on the zero set of $X = AB$. So in this study, we investigated the property of the zero set of $X = AB$.

3. Affine Algebraic Variety of NMF

Let

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \end{pmatrix},$$
$$A = \begin{pmatrix} a_{11} & a_{12} \end{pmatrix},$$
$$B = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \end{pmatrix}.$$

Then, the decomposition of $X = AB$ is obtained from the following equations.

[1] Shizuoka Institute of Science and Technology 2200-2 Toyosawa, Fukuroi 437–8555, Japan
[a)] tmatsuda@cs.sist.ac.jp

$$f_i = a_{11}b_{1i} + a_{12}b_{2i} - x_{1i} \quad (1)$$

for $i = 1, 2, \cdots, n$. In this study, we will investigate the dimension of affine algebraic variety of the above equations.

Let $\mathbf{Q}_{\geq 0}$ be a set of non negative rational numbers, and

$$S = \mathbf{Q}_{\geq 0}[a_{11}, a_{12}, b_{11}, b_{12}, \cdots, b_{1n}, b_{21}, b_{22}, \cdots, b_{2n}]$$

be a polynomial ring with non negative rational number coefficients. Then, we see that the NMF of Eq. (**??**) corresponds to the affine algebraic variety

$$V = V(f_1, f_2, \cdots, f_n) = \{P \in \mathbf{Q}_{\geq 0}^{2n+2} \mid f_i(P) = 0, 1 \leq i \leq n\}.$$

Let us consider the defining ideal of V

$$I(V) = \{g \in S \mid g(P) = 0, P \in V\}$$

to investigate the property of V.

Let

$$a^{\alpha} = a_{11}^{\alpha_1} a_{12}^{\alpha_2} b_{11}^{\alpha_3} \cdots b_{2n}^{\alpha_{2n+2}}$$

be a monomial in the polynomial ring S. The multi-index $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_{2n+2})$ corresponds to $\mathbf{Z}_{\geq 0}^{2n+2}$, where $\mathbf{Z}_{\geq 0}$ is a set of non negative integers. We define an order $\alpha > \beta$ if and only if $\alpha_1 = \beta_1, \cdots, \alpha_{i-1} = \beta_{i-1}$ and $\alpha_i > \beta_i$ for some i $(1 \leq i \leq 2n+2)$ and $\alpha, \beta \in \mathbf{Z}_{\geq 0}^{2n+2}$. This order is called as a monomial order, and it has the following properties.

- There exists a minimum element in an arbitrary subset of $\mathbf{Z}_{\geq 0}^{2n+2}$.
- It holds $\alpha > \beta, \alpha = \beta$ or $\alpha < \beta$.
- If $\alpha > \beta$, then $\alpha + \gamma > \beta + \gamma$ for any $\gamma \in \mathbf{Z}_{\geq 0}^{2n+2}$.

Here, we assumed

$$\begin{aligned} \alpha + \gamma &= (\alpha_1, \alpha_2, \cdots, \alpha_{2n+2}) + (\gamma_1, \gamma_2, \cdots, \gamma_{2n+2}) \\ &= (\alpha_1 + \gamma_1, \alpha_2 + \gamma_2, \cdots, \alpha_{2n+2} + \gamma_{2n+2}), \end{aligned}$$

and $a^{\alpha} = 0$ if $\alpha = (0, 0, \cdots, 0)$. In this study, we assume the following lexicographic order:

$$a_{11} > a_{12} > b_{11} > \cdots > b_{1n} > b_{21} > \cdots > b_{2n}.$$

Let $\mathrm{LT}(f)$ be the maximum term of the polynomial f in the lexicographic order. $\mathrm{LT}(f)$ is called as a leading term of f. Moreover, we define

$$\mathrm{LT}(I) = < \mathrm{LT}(g) \mid g \in I >$$

for any ideal $I \subset S$.

Definition 1 Let I be an ideal of S and $\{h_1, h_2, \cdots, h_k\}$ be generator of I. If

$$\mathrm{LT}(I) = < \mathrm{LT}(h_1), \mathrm{LT}(h_2), \cdots, \mathrm{LT}(h_k) >$$

holds, then $\{h_1, h_2, \cdots, h_k\}$ is called a Gröbner basis of I.

Definition 2 Let d be a non negative integer, and $S_{\leq d}$ be the set of total degree is less than or equal to d in S. We define

$$I_{\leq d} = I \cap S_{\leq d}.$$

Then, the affine Hilbert function of I is defined by

$$\mathrm{HF}_I(d) = \dim(S_{\leq d}/I_{\leq d}).$$

The affine Hilbert function is the function on d.

Definition 3 The polynomial which equals to $\mathrm{HF}_I(d)$ for sufficiently large d is called the affine Hilbert polynomial of I. We denote the affine Hilbert polynomial of I by $\mathrm{HP}_I(d)$.

Theorem 1 Let I be an ideal on S. The dimension of I is defined by the degree of $\mathrm{HP}_I(d)$.

We can see the proof of Theorem 1 in [10].

In general, there is no guarantee that the set $\{h_1, h_2, \cdots, h_k\}$ of the generator of I is the Gröbner basis of I. However, it is well known that Gröbner basis is obtained by using the Buchberger's Algorithm. The following $S - polynomial$ is fundamental to get Gröbner basis. The $S - polynomial$ of f and g is defined by

$$S(f, g) = \frac{\mathrm{LCM}(f, g)}{\mathrm{LT}(f)} f - \frac{\mathrm{LCM}(f, g)}{\mathrm{LT}(g)} g,$$

where $\mathrm{LC}(f)$ is the coefficient of $\mathrm{LT}(f)$ and $\mathrm{LCM}(f, g)$ is the least common multiple of $\mathrm{LC}(f)\mathrm{LT}(f)$ and $\mathrm{LC}(g)\mathrm{LT}(g)$.

For the decomposition of $X = AB$, we obtained the following main theorem of this study.

Theorem 2 The dimension of V is $2n - 1$.

(Outline of Proof of Theorem 2)

Firstly, we will compute the Gröbner basis of the ideal

$$I = < f_1, f_2, \cdots, f_n >,$$

where $f_i = a_{11}b_{1i} + a_{12}b_{2i} - x_{1i}$ and $i = 1, 2, \cdots, n$. For $1 \leq i < j \leq n$, let

$$f_{ij} = \frac{\mathrm{LCM}(f_i, f_j)}{\mathrm{LT}(f_i)} f_i - \frac{\mathrm{LCM}(f_i, f_j)}{\mathrm{LT}(f_j)} f_j,$$

and

$$J = < f_1, \cdots, f_n, f_{12}, \cdots, f_{n-1,n} > .$$

Then, we can see that

$$S(f_i, f_{ij}) = S(f_j, f_{ij}) = S(f_i, f_{kl}) = S(f_{ij}, f_{kl}) = 0$$

in S/J, for $1 \leq i < j < k < l \leq n$. Therefore,

$$\{f_i, f_{ij} | 1 \leq i < j \leq n\}$$

is the Gröbner basis of I. From this, we can see that the degree of the affine Hilbert polynomial of $I(V)$ is $2n + 1$. (Q. E. D).

Theorem 2 shows the freedom degree of NMF in Eq. (**??**). In order to decompose matrices uniquely, we define some equations on $a_{11}, a_{12}, b_{11}, \cdots, b_{2n}$ depending on $x_{11}, \cdots, x_{1n}$. In this study, we will consider the following NMF to detect cross-site scripting attacks.

$$\begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \end{pmatrix}.$$

4. Detection of Cross-site Scripting

4.1 Cross-site Scripting Attack

Cross-site scripting attack is one of web application attacks, and this attack is used for phishing. Cross-site scripting attacks

come from the HTTP request, form fields on web page or cookies. If users get caught in a trap that attackers set up, then the data of the trap are sent to web application. And when the data come back to user, the attack is executed. The following are specific sample of cross-site scripting attacks.

l_1 ¡IMG SRC=”javascript:alert('XSS');”¿
l_2 perl -e 'print ”¡SCRIPT¿alert(ẌSS)¡/SCRIPT¿”;' ¿ out
l_3 ¨alert('XSS');//
l_4 '¡STYLE¿BODY-moz-binding:url(”http://ha.ckers.org/xssmoz.xml#xss”) ¡/STYLE¿

The above 4 sample will be used for the detection test later. Moreover, the following 5 normal sample will be also used for the detection test.

l_5 123-4567
l_6 '2010nen11gatsu10nichi
l_7 1-2-3RoppongiMinatoku
l_8 [graph:id:text:text(:image)]
l_9 —!text—

4.2 Proposed Detection Algorithm

Firstly, let us define $x_{11}, x_{12}, \cdots, x_{1n}$. We define symbols as follows:

l_i : input string ($j = 1, 2, \cdots$)
$|l_j|$: strength of l_j
L : set of l
s_i : character in L ($i = 1, 2, \cdots$)
$|s_i|$: total number of s_i in L

Let

$$x_i = \left[1000 \cdot \frac{|s_i|}{\sum_{j=1}^{J} |l_j|}\right].$$

Here, $[y]$ is the greatest integer that is less than or equal to y. We multiplied 1000 to get double digits integer. We collected 30 cross-site scripting attacks from [11] [12] and generated 50 normal sample. Then, we obtained the following Table 1. Here,

Table 1 Characters in Cross Site Scripting Attacks

Variable	Characters	Value
x_1	” (double quotation mark)	36
x_2	< (less than sign)	25
x_3	> (grater than sign)	25
x_4	/ (slash)	24
x_5	' (single quotation mark)	22
x_6	space	21
x_7	) (right parenthesis)	19
x_8	((left parenthesis)	19
x_9	= (equal)	19
x_{10}	(yen sign)	18
x_{11}	; (semicolon)	12
x_{12}	- (hyphen)	31
x_{13}	| (vertical bar)	20
x_{14}	% (percent)	18
x_{15}	+ (plus)	11
x_{16}	* (asterisk)	11
x_{17}	& (ampersand)	5
x_{18}	{ (left brace)	3
x_{19}	} (right brace)	3
x_{20}	[(left bracket)	3
x_{21}	] (right bracket)	3
x_{22}	? (question mark)	1

normal sample are composed of the input of name, address, e-mail address, phone number, html grammar and Wiki grammar. Characters $s_1, \cdots, s_{11}$ and $s_{12}, \cdots, s_{22}$ occurred frequently in our collected attack sample, respectively.

Secondly, let us define a_{11} and a_{12}. We define a_{11} and a_{12} as attack feature element and normal feature element, respectively. We compute a_{11} and a_{12} in the following way.

$$a_{11} = \left[\frac{x_1 + \cdots + x_{11}}{10}\right]$$
$$a_{12} = \left[\frac{x_{12} + \cdots + x_{22}}{10}\right]$$

We multiplied $\frac{1}{10}$ to get double digits integer.

Finally, let us define $b_{11}, \cdots, b_{2,22}$ in the following way. Let L_A and L_N be the set of attack sample and the set of normal sample, respectively. For $i = 1, 2, \cdots, 22$, we compute

$$x_i^{(a)} = \left[1000 \cdot \frac{|s_i|}{\sum_{l \in L_A} |l|}\right] + \epsilon$$
$$x_i^{(n)} = \left[1000 \cdot \frac{|s_i|}{\sum_{l \in L_N} |l|}\right] + \epsilon,$$

where $\epsilon = 0.001$. By adding ϵ, we got $x_i^{(a)} > 0$ and $x_i^{(n)} > 0$. From Theorem 2, we can see that the dimension of $I(V)$ becomes 0 by adding the following equations:

$$\frac{b_{1i}}{b_{2i}} = \frac{x_i^{(a)}}{x_i^{(n)}}.$$

Therefore, we can obtain unique decomposition by computing the above equations.

4.3 Learning Process

In the learning process of our proposed model, characters are extracted to detect attacks. From our collected sample, we extracted 11 characters $s_1, s_2, \cdots, s_{11}$ that well appear in attack sample, and 11 characters $s_{12}, s_{13}, \cdots, s_{22}$ that well appear in normal sample in the following way. We call $\{s_1, s_2, \cdots, s_{11}\}$ (resp. $\{s_{12}, s_{13}, \cdots, s_{22}\}$) attack feature (resp. normal feature) in this paper. The character s_i is classified attack feature (resp. normal feature) if $x_i^{(a)} \geq x_i^{(n)}$ (resp. $x_i^{(a)} < x_i^{(n)}$).

4.4 Detection Rule

By using the process of Section 4.1, we can compute the following two matrices from the given data $\{x_{11}, x_{12}, \cdots, x_{1n}\}$.

$$A = \begin{pmatrix} a_{11} & a_{12} \end{pmatrix}$$
$$B = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \end{pmatrix}.$$

Firstly, we explain on the role of the matrix A. The element a_{11} of A is determined by appearance frequency of attack feature $s_1, s_2, \cdots, s_{11}$. Similarly, a_{12} of A is determined by appearance frequency of normal feature $s_{12}, s_{13}, \cdots, s_{22}$. Therefore, it may be said that input l is attack (resp. normal) if $a_{11} > a_{12}$ (resp. $a_{11} < a_{12}$).

On the other hand, it would appear that the first row of B (resp. the second row of B) shows the feature of attack (resp. the feature

of normal). If $a_{11} = a_{12}$, we detect in the following process. Let

$$d_A(b_{1i}, b_{2i}) = \begin{cases} 1 & (b_{1i} \geq b_{2i}) \\ 0 & (b_{1i} < b_{2i}), \end{cases}$$

for $i = 1, 2, \cdots, 11$, and

$$d_N(b_{1i}, b_{2i}) = \begin{cases} 1 & (b_{1i} \leq b_{2i}) \\ 0 & (b_{1i} > b_{2i}), \end{cases}$$

for $i = 12, 13, \cdots, 22$. Then, we detect l as attack (resp. normal) if $b_A \geq b_N$ (resp. $b_A < b_N$), where

$$b_A = \frac{\sum_{i=1}^{11} d_A(b_{1i}, b_{2i})}{11}$$
$$b_N = \frac{\sum_{i=12}^{22} d_N(b_{1i}, b_{2i})}{11}$$

4.5 Simulation Result

Here, we summarize the result of the detection test. In this simulation, we extracted attack feature and normal feature from 30 attack sample and 50 normal sample, and we prepared 4 attack sample and 5 normal sample for the detection test shown in Section 5.1.

To show the effectiveness of our proposed detection method, we compared our proposed method with the detection method of Naive Bayes classifier. Let $P_A(s_i)$ and $P_N(s_i)$ be the probability that s_i appears in attack and normal, respectively. We compute $P_A(s_i)$ and $P_N(s_i)$ in the following way. For $i = 1, 2, \cdots, 22$, let

$$P_A(s_i) = \frac{x_i^{(a)} + 0.001}{\sum_{j=1}^{J}(x_i^{(a)} + 0.001)}$$
$$P_N(s_i) = \frac{x_i^{(n)} + 0.001}{\sum_{j=1}^{J}(x_i^{(n)} + 0.001)}.$$

By adding 0.001, $P_A(s_i)$ and $P_N(s_i)$ become always positive value. Assume an input l including $\{s_{i_1}, s_{i_2}, \cdots, s_{i_K}\}$. We detect l as attack (resp. normal) if $\prod_{k=1}^{K} P_A(s_{i_k}) \geq \prod_{k=1}^{K} P_N(s_{i_k})$ (resp. otherwise).

Table 2 shows the detection result of our proposed method and Naive Bayes method. The detection results of our proposed method and Naive Bayes method were 100%. To investigate the point of difference between our proposed method and Naive Bayes method, we prepared two obfuscation attack sample l_{10} and l_{11}. The following the part of the obfuscation attack sample.

```
$=~[];$={___:++$,$$$$:(![]+"")[$],
```

Table 2 Detection Result

Input	Proposed Method	Naive Bayes
l_1 (attack)	attack	attack
l_2 (attack)	attack	attack
l_3 (attack)	attack	attack
l_4 (attack)	attack	attack
l_5 (normal)	normal	normal
l_6 (normal)	normal	normal
l_7 (normal)	normal	normal
l_8 (normal)	normal	normal
l_9 (normal)	normal	normal

Table 3 Detection Result of obfuscation attack sample

Input	Proposed Method	Naive Bayes
l_{10} (attack)	attack	normal
l_{11} (attack)	attack	normal

Table 3 shows the detection result of obfuscation attack sample. Our proposed method could detect two obfuscation attack sample, but Naive Bayes method judged these sample as normal. From the result of NMF, we could see that the characters of single quote and double quote have important role to detect obfuscation attack sample.

5. Conclusions

In this paper, we investigated the algebraic property of NMF, and proposed the detection method of cross-site scripting attacks by using the algebraic property of NMF. Moreover, by comparing to the method of Naive Bayes, we showed the effectiveness of our proposed method. However, we could not collect sufficient amounts of attack sample. So, we need to collect sufficient amounts of attack sample, and to do same simulation of this study. This is our important future work.

References

[1] B. Murray and B. Murray, *Email Surveillance Using Non-negative Matrix Factorization*, Computational and Mathematical Organization Theory 11 (3), pp. 249-264 (2005)
[2] H. Kim and H. Park, *Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis*, Bioinformatics 23 (12), pp.1495-1502 (2007)
[3] K. Devarajan, *Nonnegative Matrix Factorization: An Analytical and Interpretive Tool in Computational Biology*, PLoS Computational Biology 4 (7) (2008)
[4] M. W. Berry, M. Browne, A. N. Langville, V. P. Pauca and R. J. Plemmons, *Algorithms and applications for approximate nonnegative matrix factorization*, Computational Statistics and Data Analysis (2006)
[5] D. D. Lee and H. S. Seung, *Learning the parts of objects by non-negative matrix factorization*, Nature 401(6755), pp.788-791 (1999)
[6] D. D. Lee and H. S. Seung, *Algorithms for Non-negative Matrix Factorization*, Advances in Neural Information Processing Systems 13: Proceedings of the 2000 Conference. MIT Press. pp. 556-562 (2001)
[7] S. Hotta and S. Miyahara, *An Initialization Method for Non-negative Matrix Factorization and Its Applications*, IEICE, PRMU, 102(652), pp.19-24 (2003)
[8] T. Matsuda, *Solution Space of Non Negative Matrix Factorization and Consideration of Feature Extraction on Web Application Attacks*, 2014 International Conference on Information Science, Electronics and Electrical Engineering (Accepted).
[9] J. Kim and H. Park, *Fast Nonnegative Matrix Factorization: An Active-set-like Method and Comparisons*, SIAM Journal on Scientific Computing 33 (6), pp.3261-3281 (2011)
[10] D. Cox, J. Little and D. O'Shea, *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*, Springer (1997)
[11] *An Autonomous Zone*, [Online]. Available: http://anautonomouszone.com/blog/xss-cheat-sheet
[12] *Cross-site-scripting (XSS) Tutorial*, [Online]. Available: http://www.veracode.com/security/xss