Blind Steganalysis for Digital Images using Support Vector Machine Method

Marcelinus Henry Menori
School of Electrical Engineering and Informatics
Bandung Institute of Technology
Bandung, Indonesia
henrymenori@yahoo.com

Rinaldi Munir
School of Electrical Engineering and Informatics
Bandung Institute of Technology
Bandung, Indonesia
rinaldi@informatika.org

***Abstract*— Blind steganalysis is a method used to detect whether there is a hidden message in a media without having to know the steganography algorithm behind it. Digital image is converted into features using feature extraction algorithm subtractive pixel adjacency matrix. A model is built based on the resulting features using machine learning method support vector machine. The support vector machine method has different kernel configuration options, which are linear, polynomial, and Gaussian. The model that has been built then undergoes a testing to measure the accuracy performance in message detection and message length estimation. From the model testing, it is obtained that the accuracy in message detection shows good result while the accuracy in message length estimation does not. Highest accuracy is obtained with polynomial kernel.**

Keywords—blind steganalysis, support vector machine, digital images, feature extraction

I. INTRODUCTION

Steganography is a technique related to hiding message in a particular media. Common media used are image, video, audio, and text. The hidden message is usually in the form of text, but it does not rule out the possibility that it is another type of media, like image and audio. Image is the most frequently used media for message hiding. Image consists of pixels, each of which contains color bit. The common steganography technique is spatial domain method, i.e. hiding message in an image by replacing some of the bits in the image. This method is quite popular because it only slightly changes the original image and the hidden message has a fairly large size.

Nowadays, steganography has grown so much that more people start to think about how to reverse the process. The technique to do so is called steganalysis. Steganalysis can be divided into two, targeted steganalysis and blind steganalysis. Targeted steganalysis is done by reversing the steganography algorithm so that it can be seen whether there is a message hidden in an image. However, targeted steganalysis needs information about what algorithm used in the message hiding. Blind steganalysis is developed because not every image is known for its message hiding method. This method is not always accurate, but very useful if we do not have any information about the steganography algorithm used. Besides detecting whether there is a hidden message, blind steganalysis is also developed to detect important attributes such as message length and algorithm used to reach the goal of steganalysis, which is figuring out the content of the hidden message.

Blind steganalysis technique can be combined with machine learning to obtain better result. Steganalysis requires machine learning method in the form of binary classification to determine whether a digital image contains hidden message. The machine learning method that has form of binary classification is Support Vector Machine [4]. Support vector machine is also very good for classification with numerous features.

II. THEORY

A. Steganalysis

Steganalysis is technique to detect whether there is a hidden message in a media. The insertion of information on a particular will alter the characteristics of the media so that it can be detected through several techniques.

Steganalysis methods can be divided into two.

1. *Specific Technique Steganalysis* (*Targeted Steganalysis*)

Steganalysis for certain steganography method such as LSB (Least Significant Byte). This method has great accuracy if the method used in steganography is the same as one in the steganalysis.

2. *Blind steganalysis*

This steganalysis method does not emphasize on a specific, but for all steganography methods. This method is used by analyzing changes in pixel bit or by statistic.

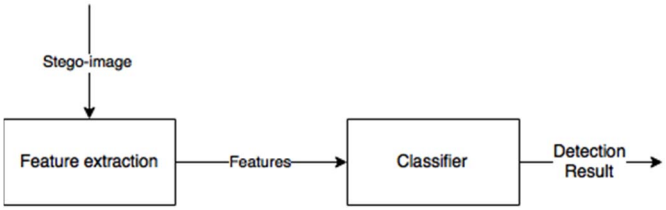

Figure 1.Workflow of steganalysis process

Steganalysis process starts from a stego-image that is suspected to have hidden message. From the stego-image, features are extracted using feature extraction method that has been defined. Then, the gained features are put into a classifier

978-1-5090-3840-4/16/$31.00 ©2016 IEEE

to determine whether the image contains hidden message. Another result of the classifier can be the length of the message or the steganography algorithm used to do the message hiding. Steganalysis process is shown in Figure 1.

B. Support Vector Machine

Support Vector Machine (SVM) is a machine learning method based on binary classification. Binary classification is a method that divides data into two classes (binary) wherein each data will have class value +1 or -1. For each data, $(\vec{x}_i, y_i)$ with $i = 1 \ldots N$, $\vec{x}_i \in R^d$, and $y_i \in \{-1, +1\}$, binary classification $f(\vec{x}_i)$ results as follows.

$$y_i = \begin{cases} +1, & f(\vec{x}_i) \geq 0 \\ -1, & f(\vec{x}_i) < 0 \end{cases} \quad (1)$$

$\vec{x}_i$ stands for dataset, which is a collection of real numbers *a* as attribute, and *k* is the number of attributes in the data.

$$\vec{x}_i = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_k \end{pmatrix} \quad (2)$$

SVM method will form a support vector based on data that is closest to the separating hyper plane so that it will be formed one support vector for each class. This support vector will assist in classifying in determining confidence. If data is located in between the support vector, then the data is classified with lower confidence than one in below or above the support vector. Figure 2 shows the representation of support vector illustrated by the dotted line.

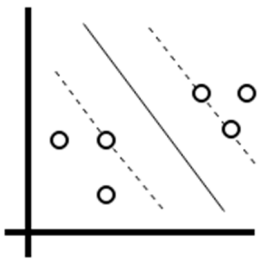

Figure 2. Support vector representation

Below are the steps to get hyper plane and support vector to form the model [5].

1. Every data has value α. This value represents the influence of the data on the hyper plane and support vector.

2. Calculate the value α for each data so that L_D gets maximal value with the following equation.

$$L_D(\alpha) \equiv \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1, j=1}^{n} \alpha_i \alpha_j y_i y_j k\left(\vec{x}_i \vec{x}_j\right) \quad (3)$$

With terms $\sum_{i=1}^{n} \alpha_i y_i = 0$ and $c \geq \alpha_i \geq 0$, where c is a determined constant. The kernel used is linear $k(\vec{x}_i, \vec{x}_j) = \vec{x}_i \cdot \vec{x}_j$,polynomial $k(\vec{x}_i, \vec{x}_j) = \left(\vec{x}_i \cdot \vec{x}_j\right)^2$, or Gaussian $k(\vec{x}_i, \vec{x}_j) = -\gamma \left\| \vec{x}_i - \vec{x}_j \right\|^2$

3. After L_D has maximal value, save every data that has value $\alpha > 0$. The data will be *support vector*.

4. Classify function is:

$$f(\vec{x}_d) = \sum_{i=1}^{ns} \alpha_i y_i \vec{x}_i \vec{x}_d + b \quad (4)$$

5. To do classification on a data *x*, equation (4) is used for calculation first. Then, equation (1) is used to do the classification.

To calculate the maximal value of L_D, several algorithms can be used, for instance, modified sequential minimal optimization [6].

Support vector machine is binary classifier, but it does not rule out the possibility to build multi-class model with this method. Multi-class model for *support vector* machine consists of several sub-models. Some way to build support vector machine multi-class model are one-against-all, one-against-one, and directed acyclic graph [4].

III. Related Works

Steganalysis method proposed by Siwei and Hany [7] uses color wavelet statistics as feature extraction and support vector machine as classifier. This method can be done in both spatial domain and transformation domain. However, the accuracy in spatial domain is lower than one in transformation domain for the feature extraction method is very close to transformation domain.

Steganalysis method proposed by Jiang [8] uses the combination of several methods. For feature extraction, filter is used in the form of shifting and also first order statistics. For classifier, ensemble classifier is used with addition of AdaBoost and bagging. This steganalysis method is done in spatial domain and is made to cope with the low accuracy in inserting text in a small size. But the feature extraction method used has a quite large dimension.

Steganalysis method proposed by Tomas [9] uses subtractive pixel adjacency matrix as feature extraction method and support vector machine as classifier. This steganalysis method is done in spatial domain. This method emphasizes on feature extraction and detection of suspicious pixel. Support vector machine method used is only based on Gaussian kernel.

IV. Proposed Method

To make a model to detect hidden message and estimate message length, first, we generate dataset and testset. Dataset is used for model training and testset is used for model testing. Digital images for dataset and testset are derived from internet

with various size and type. Work flow for proposed method is shown in Figure 3.

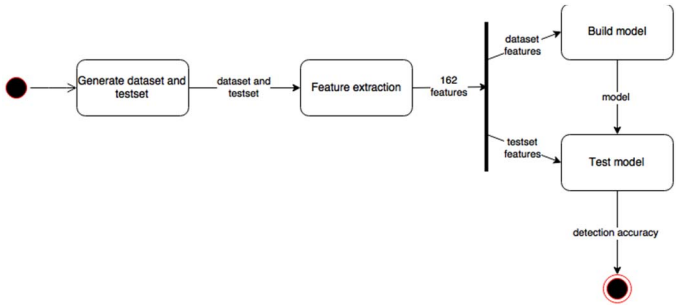

Figure 3. Workflow for proposed method

Dataset and testset are cover-image and stego-image from the digital images collected before. Digital images in dataset and testset are converted into same size and type, which is 800x600 pixel in bitmap type. For dataset, stego-image is obtained by inserting message in size of 0.05, 0.15, 0.25, 0.35, 0.45, 0.55, 0.65, 0.75, 0.85, and 0.95 bpp. For testset, stego-image is obtained by inserting message in random size that ranges from 0.01 to 0.99 bpp. The insertion of the message into digital images uses steganography tools, Steghide [10], Four Pixel [11], and Nine Pixel [12].

Feature extraction phase is a process to obtain important features from digital image in dataset and testset. Feature extraction algorithm used is subtractive pixel adjacency matrix proposed by Tomas [9]. The algorithm is as follows.

1. Calculate difference matrix D in images with width m pixel and height n pixel for right direction. The equation for difference matrix is

$$D_{i,j}^{\rightarrow} = I_{i,j} - I_{i,j+1} \qquad (5)$$

subject to $i \in \{1,\ldots,m\}$ and $j \in \{1,\ldots,n-1\}$. $I_{i,j}$ is pixel value at pixel (i, j). For color images, pixel value is average of red, green, and blue value. Example for calculating difference matrix in images with 4x4 pixel size can be seen in Figure 4.

5	3	3	3
2	1	2	5
5	4	3	2
1	3	7	4

⇨

2	0	0
1	-1	-3
1	1	1
-2	-4	3

Figure 4. Difference matrix for 4x4 pixel size image

2. Calculate value for Markov chain with equation (6)

$$M_{u,v}^{\rightarrow} = \Pr\left(D_{i,j+1}^{\rightarrow} = u \mid D_{i,j}^{\rightarrow} = v\right) \qquad (6)$$

subject to $u,v \in \{-4,\ldots,4\}$. If $\Pr\left(D_{i,j}^{\rightarrow} = v\right) = 0$ then $M_{u,v}^{\rightarrow} = \Pr\left(D_{i,j+1}^{\rightarrow} = u \mid D_{i,j}^{\rightarrow} = v\right) = 0$. There will be 81 values of Markov Chain. Example for calculating Markov Chain in Fig. 4

$$M_{0,0}^{\rightarrow} = \Pr\left(D_{i,j+1}^{\rightarrow} = 0 \mid D_{i,j}^{\rightarrow} = 0\right) = 1$$

$$M_{1,1}^{\rightarrow} = \Pr\left(D_{i,j+1}^{\rightarrow} = 1 \mid D_{i,j}^{\rightarrow} = 1\right) = \frac{2}{3}.$$

3. Repeat step one and two for the other direction
4. Simplify the features with equation (7) and (8)

$$F_{1,\ldots,81} = \frac{1}{4}\left[M^{\rightarrow} + M^{\leftarrow} + M^{\downarrow} + M^{\uparrow}\right] \qquad (7)$$

$$F_{82\ldots,162} = \frac{1}{4}\left[M^{\nearrow} + M^{\swarrow} + M^{\searrow} + M^{\nwarrow}\right] \qquad (8)$$

so, there will be 162 features for each image.

Training model with support vector machine requires dataset features as input. From the collection of features, two models will be built. The first model is used to determine whether the digital image contains hidden message, while the other model is used to estimate the hidden message length. The first model has linear, polynomial, and RBF as kernel configuration options, while the second model has additional configuration which is multi-class options, consists of one-against-all, one-against-one, and directed acyclic graph. Model learning with support vector machine method uses modified sequential minimal optimization algorithm proposed by Cao [6].

Testing phase requires the model that has been built before as input. The model is tested with dataset and testset to measure its performance. Each testing results in an accuracy value.

V. EXPERIMENT

Model training is done in maximal iteration 3000, constant c 100, two classes for message detection, which are yes and no, and five classes for message length estimation, which are very low (<0.2 bpp), low (0.2 bpp – 0.4 bpp), medium (0.4 bpp – 0.6 bpp), high (0.6 bpp – 0.8 bpp), and very high (>0.8 bpp).

Model testing process is divided into four steps, testing for message detection in grayscale images, testing for message length estimation in grayscale images, testing for message detection in color images, and testing for message length estimation in color images. The accuracy of message detection and message length estimation is calculated with following equation.

$$\text{accuracy} = \frac{\text{number of correct classified images}}{\text{number of images in testset}} \qquad (9)$$

Table 1. Testing result for message detection in grayscale images

Kernel	Accuracy with Dataset	Accuracy with Testset
linear	67.25%	63%
polynomial	75.25%	73%
Gaussian	69.5%	59%

Table 2 Testing result for message detection in color images

Kernel	Accuracy with Dataset	Accuracy with Testset
linear	54.75%	57%
polynomial	64.25%	61%
Gaussian	51.5%	50%

Table 3 Testing result for message length estimation in grayscale images

Kernel	Multi-class	Accuracy with Dataset	Accuracy with Testset
linear	one-against-all	5.2%	4%
linear	one-against-one	28.5%	24.8%
linear	directed acyclic graph	29.15%	24.8%
polynomial	one-against-all	6.45%	6.8%
polynomial	one-against-one	39.4%	33.6%
polynomial	directed acyclic graph	38.15%	30%
Gaussian	one-against-all	7.95%	6%
Gaussian	one-against-one	30.95%	25.2%
Gaussian	directed acyclic graph	30.55%	24.4%

Table 4 Testing result for message length estimation in color images

Kernel	Multi-class	Accuracy with Dataset	Accuracy with Testset
linear	one-against-all	3.15%	2%
linear	one-against-one	21.35%	23.6%
linear	directed acyclic graph	22.3%	21.6%
polynomial	one-against-all	0%	0%
polynomial	one-against-one	26.35%	23.6%
polynomial	directed acyclic graph	26.05%	22%
Gaussian	one-against-all	5.25%	4%
Gaussian	one-against-one	20.9%	20.8%
Gaussian	directed acyclic graph	21.05%	20.8%

The testing result for message detection in grayscale images shows that configuration with polynomial kernel has the highest accuracy with both dataset and testset. Polynomial kernel configuration results better because polynomial is more flexible than linear and more rigid than Gaussian. Plane created by polynomial kernel is not a flat plane, but wavy as it adjusts to the data, while plane created by linear kernel is more rigid that it does not really match with the data. Plane created by Gaussian kernel is too flexible that it turns the model to overfit. Overfit is the state when the resulting model is too dependent on dataset that it does not create model that is more common. The result for message detection in color digital image testing also shows that configuration with polynomial kernel has the highest accuracy. A common model is needed so that noise in data does not affect the model.

The result for message length estimation in grayscale digital image testing shows that configuration with polynomial kernel and multi-class one-against-one has the highest accuracy with both dataset and testset. The message length estimation in color digital image testing also shows the same result. Testing result shows that polynomial kernel still better than two other kernels.

The comparison of accuracy result from testing in grayscale images and color images can be seen in Figure 5.

The comparison shows that accuracy of grayscale images testing is generally higher than the accuracy of color images testing. This happens due to the difference in feature extraction for grayscale images and color images. In grayscale images, the difference between each pixel can be easily calculated because it only contains one value in range 0-255, while in color images, each pixel contains three values each for red, green, and blue. Changes in the value of the pixel to another pixel near it become more difficult to detect because values each pixel contains are converted to grayscale value. For example, red, green, and blue values for one pixel are 12, 15, and 120, while for the other pixel the values are 13, 12, and 122. The difference of pixel values is being calculated and resulting in a zero value, it happens because both pixels have same grayscale value.

Accuracy for message length estimation is not good enough because the feature extraction method used does not match. Subtractive pixel adjacency matrix only indicates probability value that changes. As a result, if there is a hidden message, the hidden message can be known using support vector machine method. Subtractive pixel adjacency matrix does not show how much the pixel changes so it is not suitable for hidden message length estimation.

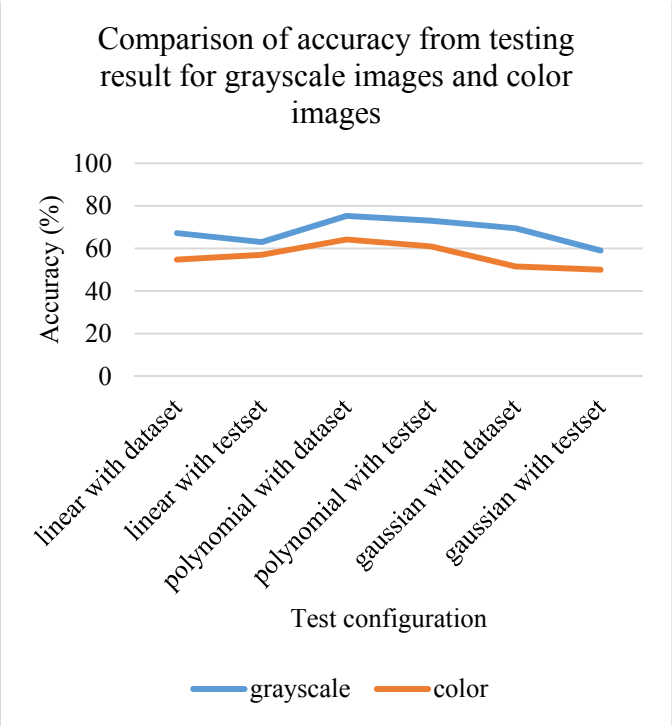

Figure 5 Comparison of accuracy from testing result for grayscale images and color images.

VI. CONCLUSION

Blind steganalysis with support vector machine method can be implemented to detect hidden message and estimate hidden message length in digital image.

The model that is built with the support vector machine method has a quite good result in detecting the hidden message with an accuracy of 73% for grayscale image and 61% for color image. The hidden message length estimation in digital image has poor result. The highest accuracy achieved is equal to 33.6% for grayscale image and 23.6% for color digital image. Configuration with polynomial kernel is better than the other two, linear and Gaussian.

ACKNOWLEDGMENT

Author would like to thank Dr. Ir. Rinaldi Munir, M.T. as the supervisor for the guidance and valuable knowledge during this research.

REFERENCES

[1] Hussain, M., & Hussain, M. (2013). A survey of image steganography techniques.

[2] Chandramouli, R., Kharrazi, M., & Memon, N. (2003, October). Image steganography and steganalysis: Concepts and practice. In International Workshop on Digital Watermarking (pp. 35-49). Springer Berlin Heidelberg.

[3] Fridrich, J., Goljan, M., Hogea, D., & Soukal, D. (2003). Quantitative steganalysis of digital images: estimating the secret message length. Multimedia systems, 9(3), 288-302.

[4] Hsu, C. W., & Lin, C. J. (2002). A comparison of methods for multiclass support vector machines. IEEE transactions on Neural Networks, 13(2), 415-425.

[5] Schölkopf, B., & Smola, A. J. (2002). Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT press.

[6] Cao, L. J., Keerthi, S. S., Ong, C. J., Zhang, J. Q., Periyathamby, U., Fu, X. J., & Lee, H. P. (2006). Parallel sequential minimal optimization for the training of support vector machines. IEEE Transactions on Neural Networks, 17(4), 1039-1049.

[7] Lyu, S., & Farid, H. (2004, June). Steganalysis using color wavelet statistics and one-class support vector machines. In Electronic Imaging 2004 (pp. 35-45). International Society for Optics and Photonics.

[8] Yu, J., Zhang, X., & Li, F. (2015). Spatial steganalysis using redistributed residuals and diverse ensemble classifier. Multimedia Tools and Applications, 1-13.

[9] Pevny, T., Bas, P., & Fridrich, J. (2010). Steganalysis by subtractive pixel adjacency matrix. IEEE Transactions on Information Forensics and Security, 5(2), 215-224.

[10] http://steghide.sourceforge.net/

[11] Liao, X., Wen, Q. Y., & Zhang, J. (2011). A steganographic method for digital images with four-pixel differencing and modified LSB substitution. Journal of Visual Communication and Image Representation, 22(1), 1-8.

[12] Swain, G. (2014). Digital image steganography using nine-pixel differencing and modified LSB substitution. Indian Journal of Science and Technology, 7(9), 1444-1450.