CHAPTER 2

THEORETICAL FOUNDATION

2.1 Theoretical Foundation

2.1.1 Spam 2.0 [6]

Web 2.0 is a form of a website that emphasizes on user generated content, user collaboration, and user communication; it emerges in the era when Internet has become more and more accessible to people [8]. As it gets popular and intensively used in World Wide Web, it faces a challenge: Spam 2.0. This is a new type of spam that attacks Web 2.0 platforms such as online forums, wikis, and blogs. The main difference between Spam 2.0 and the other spamming methods is that Spam 2.0 is hosted by legitimate websites, while the others require the spammers to host a media to bridge the communication between spammers and real users by themselves (for example email, internet chat, phone message). Legal websites are continuously being targeted by Spam 2.0 which then resulting damage to the website’s reputation and the spam getting high search engine rank [6]. Web 2.0 applications usually strive to get a high ranking in search engine so that they can attract more users. The number of users is very significant for this kind of website as the contents are user-generated. Spam contents that are posted in the website will indirectly get high search engine rank too. Users (web visitors) are also disadvantaged that they get unsolicited contents which sometimes go beyond advertising or marketing campaign: phising. How the Spam 2.0 works differently to the previous spamming methods require a new study to analyze and handle the problem.
The figure above describes Spam 2.0 key entities. From the left side, it can be seen that the spammers consist of Auto Submitters, Web Spambots, and Human Spammers. Auto Submitter is a tool which is used by spammers to quickly distribute spam content by automating process of sending form data. Web robots have been used to help human to do repeated actions more quickly, example of positive usage is web robots in search engine. However, they are being misused by some people to spread spam contents. Email spambots collect email addresses throughout the Internet and send unsolicited mails, while Web Spambots disseminate spam in Web 2.0 applications. People can also become a source of Spam 2.0; they are paid for spreading spam contents.

Web 2.0 applications consist of Comment System, Online Discussion Boards, Wikis, and Social Networking Platform. Content items are divided into two categories: application specific and website specific. Application specific content items are features
which are possessed by Web 2.0 applications that were mentioned before. Website specific content items exist only in specific websites, such as Facebook.

Table 2.1 Content Items (Source: [6])

<table>
<thead>
<tr>
<th>Features</th>
<th>Comment</th>
<th>Forum</th>
<th>Wiki</th>
<th>Social Networking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comment</td>
<td>post, poll, messaging, profile, attachment, signature</td>
<td>article content, tag, reference</td>
<td>profile, comment, tag, messaging, testimonial, attachment,</td>
<td></td>
</tr>
<tr>
<td>content, trackback</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2.1 shows content item contents in Web 2.0 applications. Spammers make use of these features in order to spread spam contents. It can be seen that the spam can be in many forms, including profile in forum and social networking sites.

There are two types of Spam 2.0 countermeasures: content specific and source specific. The content specific countermeasures try to observe and analyze the pattern of spam content, while source specific ones focus on detecting or preventing the actors from doing their actions.

This thesis will emphasize on bots as the source of Spam 2.0. Source-based countermeasure will be proposed in form of bot detection program.
2.1.2 Web Navigation

A research implied that web navigation is a user activity of jumping from a webpage to another webpage. In concrete, this is an act of sending HTTP request to a page in a website. The request contains user data such as requested page, request method, request time, query string, etc. [7]. Web navigation data can be used in source-specific Spam 2.0 countermeasure technique to differentiate between human and bot, assuming that they have distinct behavior in Web 2.0 applications.

2.1.3 Session ID

Session ID refers to a string that is used in web applications to track the state of a user [9]. It uniquely identifies each session in a website. Example of session ID: 420398e72fbc188899391848cc2c3634. A session contains information about current state of the user, stored in the web server. The client web browser holds the session ID in form of cookie [10]. Shopping cart in e-commerce websites is a common example of session usage, in which the session contains products selected by the customer with its respective quantity. In this thesis, session ID will be used as a unique identifier of each user for tracking web navigation behavior.

2.1.4 Web Spambot Behavior [4]

A research to capture and analyze web spambot behavior has been done. Its authors built a program, which was named HoneySpam 2.0, specially designed for trapping web
spambots. The program consisted of Navigation Tracking Component that recorded incoming request information, such as the remote address used, requested webpage, and the browser identity. It also incorporated Form Tracking Component for capturing form interaction data (mouse and keyboard activity, form load and submission). The HoneySpam 2.0 was implemented in a web application and deployed into 6 different hosting servers. A message was put in the web application so that human users did not use it. In this way they could be sure that only spambots visited the website.

From the experiment result, several web spambot behaviors were concluded:

1. **Reach Target Websites by Using Search Engines**
   
   The higher search engine rank a website has, the higher possibility that web spambots will come.

2. **Register New Account Rather Than Using the Existing One**
   
   Web spambots prefer to create new account for spamming rather than using the one they already made before. In this way the spammers do not have to be afraid of their account getting banned.

3. **Low Webpage Navigation Count and Revisit Rates**
   
   Most web spambots have very low webpage hits (< 5 times) and visit the website only once.

4. **Distribute Spam Quickly**
   
   Web spambots visit a website and disseminate spam contents very quickly before they go to other websites.
5. **Not Using the Target Website’s Web Form**

The research found that there was no web form interaction and it was believed that the spambots used their own form to submit contents.

6. **Generate Usernames**

Web spambots generate usernames so that they could create as many user accounts as possible.

This thesis will consider the characteristics mentioned above for detecting bot activities.

### 2.1.5 Classifier

Classifier is a function that accepts input values (features) and predicts to which class the data belongs to. Before doing its task, a classifier must be trained first by a set of labeled data (training set) [11]. The classifier will observe and analyze the training set so that it can classify unseen data (the test set). If it can generalize the training set to correctly identify test set, the classifier is considered good. When training a classifier, there is a challenge of overfitting and underfitting. Overfitting happens when the classifier model is too complex that it fits noise data. On the other hand, underfitting occurs when the classifier fails to model the complex data set. Overfitting is dangerous that the classifier’s prediction will not conform to the training data [12]. One phenomena of overfitting is that the classifier performs well with training set, but not in the test set. Few factors that contribute to overfitting [12], [13]:

- Training set size is too small
- Training set is too noisy
• Feature size is too large
• Incorrect parameter selection

One of the solutions to overfitting problem is to do cross validation, which will be discussed in section 2.1.9.

This thesis needs to use classifier for the task of differentiating human and bot, which means this is a binary classification problem (there are only two classes). The author will experiment with three well-known binary classifiers: Support Vector Machine (SVM), Naïve Bayes (NB), and k-Nearest Neighbor (kNN). A research concluded that SVM is good in theory; however NB and kNN are also no less as they have fair performance and much lighter than SVM [14].

2.1.5.1 Support Vector Machine

Support Vector Machine (SVM) is a kernel based classifier, which calculate dot products of the data [15]. There are 4 basic kernels in SVM: linear, polynomial, radial basis function (RBF), and sigmoid. Each kernel has different parameters that can be tuned to improve the classifier’s performance [16]:

- **Linear**: $K(x_i, x_j) = x_i^T x_j$
- **Polynomial**: $K(x_i, x_j) = (γx_i^T x_j + r)^d$, $γ > 0$
- **Radial basis function (RBF)**: $K(x_i, x_j) = \exp\left(-γ\|x_i - x_j\|^2\right)$, $γ > 0$
- **Sigmoid**: $K(x_i, x_j) = \tanh(γx_i^T x_j + r)$

Polynomial, RBF, and sigmoid belong to nonlinear kernel. Among these three kinds of nonlinear kernel, [16] suggest to try using RBF instead of polynomial or sigmoid because it is considered better in overall. With this in mind, the author
will only consider using linear and RBF kernel for classification task in the proposed program.

Figure 2.2 Linear SVM (Source: [15])

Figure 2.2 shows a linear SVM that separates two classes with a separating hyperplane (the thick line in the middle). The two lines on the left and right are the decision boundaries that will determine which data belongs to which class. The margin width is determined by Cost (C) parameter.

Figure 2.3 Different values of cost parameter (Source: [15])
As can be seen in Figure 2.3, different values of C affect width of the margin of the separating hyperplane. The higher value of C leads to narrow area of margin, while lower value will enlarge the margin width.

![Figure 2.3](image1.png)

Figure 2.4 Nonlinear SVM with different values of gamma parameter (Source: [15])

When using nonlinear SVM, the separating hyperplane is determined by one or more parameters other than C. Figure 2.4 shows how nonlinear SVM, in this case with RBF kernel, affected by Gaussian parameter (gamma). SVM with this kernel is sensitive to kernel width. If it is too large, the model will underfit; too small and it will overfit [17]. In summary, the performance of SVM depends on the value of its parameters. They should be carefully selected to achieve the best performance and avoid overfitting (or underfitting). [18] suggested the advantages of SVM:

- The ability to model complex, real-world problems
- Good performance for data set with a lot number of attributes, given only small of training data
2.1.5.2 Naïve Bayes

The main idea of Bayesian approach is calculating whether a belief should be changed given new evidence \[19\]. The Bayes rule \[20\]:

\[
P(R = r|e) = \frac{P(e|R = r) P(R = r)}{P(e)}
\]

\(P(R = r \mid e)\) denotes the probability that random variable \(R\) has value \(r\) given evidence \(e\). Bayes classifiers based their calculations on this formula. Naïve Bayes comes with an idea to simplify calculation by assuming conditional independence between variables.

Naïve Bayes classifier assumes that all of the features are independent of class label. One might think that this classifier will not perform well given the simplistic assumption it has. However, it was reported to perform well even when the assumption is false (the data is not independent on each other) \[21\]. As probabilistic classifier, Naïve Bayes has the following advantages \[22\]:

- Ability to reject classification
  Probability output can be used as indicator to measure about how confident the prediction is. If the confidence level is low, there is an option not to classify.

- Working with unbalanced data
  Scaled likelihood trick: train the classifier with balanced data, and then convert the posterior calculation into likelihood.
• Combination of different feature sets

It is possible for the classifier to learn from two different feature sets separately and then combine the output.

2.1.5.3 K-Nearest Neighbor

k-Nearest Neighbor (kNN) is a classifier that determine a data belongs to which class, by looking at its neighbors (other data). The k value denotes how many neighbors should be considered when doing classification. In its kNN classification consists of two steps: first, distance metric is used to choose the nearest neighbors. Second, a weighted voting is done among the neighbors to determine the class label [23]. The two important parameters for kNN is the value of k, to determine number of nearest neighbors used, and the distance function. Some of the known distance metrics are Euclidean, Manhattan, Minkowski, Earth Mover distance [23].

![Figure 2.5 Example of 3-Nearest Neighbor Classification](Source: [23])
[23] mentioned some advantages of kNN classifier:

- Easy to implement
- There are noise reduction technique that can be applied for kNN
- Analysis of the neighbors can be used as explanation of the classifier’s output

2.1.6 Framework for Bot Detection in Web 2.0 Applications [7]

![Diagram of the framework](image-url)

In their previous work, a group of researchers proposed a framework to be used in Web 2.0 applications as bot detection technique, shown in Figure 2.6. The framework consists of three main components: Tracker, Extractor, and Classifier. Tracker records navigation information such as IP address, username, session details, and the requested webpage. Extractor does two kinds of function: transaction identification for grouping requests,
action extraction to derive actions based on information recorder by Tracker component. An action was defined as a group of webpages requested by user in order to do certain functionality in a website, such as registering new account. The last and the most important component in the framework is Classifier. This component has the responsibility of judging whether a session is generated by human or bot. First, it constructs the input vector based on feature set and then outputs the result (whether it is human or bot). This framework will be a baseline for the author to build the bot detection program.

2.1.7 Web Spambot Detection by Utilizing Action Time and Action Frequency

[24]

A study about web spambot detection concluded that action time and action frequency can be good feature sets for human-bot classification, with an assumption that web spambots act differently to human in Web 2.0 websites. The paper proposed a framework for detecting web spambot based on web usage data [24]:
It can be seen that Figure 2.7 is very similar to Figure 2.6. Indeed, these two frameworks were proposed by the same researchers. It seems that the framework in Figure 2.7 is based on the one in Figure 2.6 with slight modifications. Web Usage Tracking module is basically the same as Tracker in Figure 2.6, that records navigation data. Module Data Preparation consists of three components: Data Cleaning, Transaction Identification, and Dwell Time Completion. Data Cleaning component removes irrelevant web usage data, such as the researchers and web crawler sessions. Transaction Identification determines a set of requested webpages in a session. Dwell Time Completion calculates how long a user is viewing a webpage.
From Module Feature Measurement, it can be seen that the proposed framework used Action Time and Action Frequency as feature sets. Action Time means how long users need to do a task; while Action Frequency is how many times an action is done. The Classification Module uses SVM as the classifier to determine which session is generated by spambot and which one is generated by human.

The web spambot detection framework was implemented in an online forum. The result was quite good in terms of accuracy (93.18% for action time and 96.23% for action frequency).

2.1.8 Link Obfuscation to Detect Web Bots [25]

A study suggested that some of bots tried to mimic human navigation behavior by clicking random links. The researchers introduced a technique called Decoy Link Design Adaptation (DLDA) to detect this kind of bot. The main idea was to protect links in a website by generating decoy/invalid links that might be clicked by bots. CSS (Cascading Style Sheet) and Javascript were the main technologies used to do this task. There are four types of decoy links mentioned in the paper:

- Multiple Link: creating multiple invalid links and position them off the page so that human will not be able to see.
- Tag Cloud: creates many links and the largest is the valid one.
- Tab Link: decoy links in form of tab menu with text and color different to the valid ones.
- Shadow Link: decoy links that are identical to the valid ones in terms of the text contained, and placed below the valid links.
The researchers suggested Multiple Link and Shadow Link as better than Tag Cloud and Tab Link, as they did not confuse users.

The research use session timeouts as a measure of how many bots are trapped by the decoy links. This technique worked well that many bots could be detected and no human users clicked on the invalid links. The drawback was a noticeable webpage load time overhead introduced when too many decoy links were to be generated (although the more decoy links generated, the more probable bots got trapped). It was concluded that 10 links is the most optimal amount for detecting bots.

2.1.9 K-Fold Cross Validation

The basic understanding of cross validation is that it is a technique to evaluate performance of a classifier by dividing data into two parts: training data and validation data. K-fold cross validation is a variant of cross validation that divides the overall data equally into k folds. There will be k iterations of cross validation with different fold used as validation data while the other k-1 folds are used for training [26].

![3-fold Cross Validation](image)

Figure 2.8 3-fold Cross Validation (Source: [26])
Figure 2.7 exhibits an example of 3-fold cross validation. The dark grey rectangles represent training data and the light grey ones represent validation data. 10-fold cross validation is considered as the most commonly used of k-fold cross validation and it is also suggested by data mining community because of its likeliness to be able to generalize to the whole data (10-fold cross validation uses 90% data for training its predictions) [26].

2.1.10 Matthew’s Correlation Coefficient

Matthew’s Correlation Coefficient (MCC) is a tool to determine the how good a binary classifier in terms of correlation between observation and prediction. The formula is as follows:

\[
Matthew's \ Correlation \ Coefficient = \frac{(TP \cdot TN) - (FP \cdot FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}
\]

TP = True Positives

TN = True Negatives

FP = False Positives

FN = False Negatives

The result of the calculation is between -1 and +1, showing correlation between observation and prediction. +1 indicates total agreement, -1 indicates total disagreement and 0 if the predictions are random [27].
2.1.11 Data Normalization

Normalization or scaling refers to an action of mapping data value into a new value with specified range. This is very important, especially for SVM. Without normalization, features/attributes with large value range will dominate the others with smaller value range. Doing normalization before classification takes care of the problem and it can also ease the calculation process. Value range of [-1, 1] or [0, 1] is recommended [16].

2.2 Research Methodology

The methodology of the thesis was as follows:

- Literature research about bot detection techniques.
- Build the web navigation recording program and implement it in a web application.
- Collect data from the web navigation recording program.
- Do parameter selection in order to achieve the optimal performance of the tested classifiers.
- Do 10 fold cross validation with each classifier (SVM, NB, and kNN) using the collected data.
- Select the classifier with best 10 fold cross validation accuracy to be implemented in the web application.