A Privacy Guard Service

Xinfeng Ye Fan Wang
Department of Computer Science
The University of Auckland
Auckland, New Zealand
xinfeng@cs.auckland.ac.nz

Abstract—**Mobile devices are changing the way that people conduct their daily businesses and carry out their works. More and more people are using mobile devices to view personal or work-related information in public places, e.g. café, train, etc. This type of information might contain sensitive data, e.g. personal information, trade secrets, etc. As high-resolution cameras are widely used and public places are always crowded, viewing information in public carries the risk of having the screen filmed by cameras or being peeked by the near-by people. This paper proposes a privacy guard service to help people to safe-guard their sensitive data while viewing information in public. The service uses machine learning and natural language processing techniques to identify sensitive information in documents. The service automatically converts the sensitive information to meaningless strings. Text-to-speech code is embedded behind the symbols. The code enable the users to listen to the content of the original sensitive information by using an earphone when the users tap on the symbols. As a result, users are able to view their sensitive data in public without worrying about privacy leaks due to visual observation attacks.**

Keywords- privacy, natural language processing, visual observation attack, shoulder surfing, machine learning

I. INTRODUCTION

Mobile devises are changing the way that people conduct their daily business and carry out their work. More and more people are using mobile devices to view personal or work-related information in public places, e.g. café, train, etc. A survey showed that 76% of the people being surveyed prefer to work outside their offices [9]. A lot of information viewed in public contain sensitive materials, e.g. personal information, commercial secrets, etc. Therefore, viewing information in public carries the risk of leaking sensitive data if the data are observed by authorized entities.

There are two forms of visual observation attacks, i.e. shoulder-surfing and photographing. Shoulder-surfing means obtaining information by looking over a victim's shoulder. A survey showed that 85% of IT professionals admitted to seeing sensitive information on screen that they were not authorized to view [10]. Another survey found that 80% people admitted to read over a stranger whilst on public transport, in coffee shops or in shared places [2]. Thus, mobile devices users should regard shoulder-surfing as a serious threat.

In photographing attack, an attacker photographs the contents of a screen and extracts information from the image. There are more than 2 billion smartphones in the world [20]. This means billions of cameras are operating around us. Many metropolitan areas operate thousands of high-resolution surveillance cameras. A lot of these cameras are insure [17]. This means criminals can easily obtain the images captured by the cameras. As information displayed on the screens of mobile devices can be captured by various types of cameras, and sensitive information can be extracted from the captured images, photographing attack is a serious threat to the security of information.

Countering the threats of visual observation attacks has started attracting people's attentions in recent years [10]. However, not much research has been carried out in this area. Existing research has studied how to hide sensitive information in form-based data (i.e. data entered through a form) [14]. The scheme identified the sensitive information by checking the labels of the text boxes in a form. In real-life, most of the information accessed by users do not appear as form-based data. Therefore, the method in [14] only has limited applicability for countering visual observation attacks. This paper proposed a privacy guard service that helps users to identify and hide sensitive information in their documents. The sensitive information (a) are replaced with meaningless symbols, and (b) can be listened through an earphone when users tap on the symbols. It allows users to view documents in public without the fear of leaking sensitive information through visual observation attacks. The main contributions of this paper are:

(a) The scheme proposed using natural language processing (NLP) techniques to identify sensitive information. Instead of solely relying on keywords to locate sensitive information, the keywords are used as hints for recognizing sensitive information, and the NLP techniques are used to identify the words that are likely to reveal sensitive information. This approach allows the sensitive information to be identified more comprehensively.

(b) The scheme proposed the use of machine learning technique to classify files for improving the accuracy in identifying sensitive information. A word might have different significance when it appears in different types of files, e.g. financial report, health document.

Classifying files into different categories and giving each category its own keywords set enable the sensitive information in the files to be identified more accurately.

The rest of this paper is organized as follow. §II introduces some technologies used in the paper. §III presents the details of the proposed scheme. The evaluation of the scheme and the related work are described in §IV and §V respectively. Conclusions are given in §VI.

II. Relevant Techniques

A. Naïve Bayes classifier

Naive Bayes model is a probabilistic learning method [19]. In naïve Bayes, events are classified into n categories. C represents the set of all the categories. c_i (where $1 \le i \le n$) denotes an event in C. Each event has a set of features which is represented by a feature vector $\boldsymbol{f} = (f_1, f_2, \dots, f_m)$ where f_j $(1 \le j \le m)$ is the value of the j^{th} feature of the event. The naïve Bayes model assumes that the features f_j $(1 \le j \le m)$ are independent of each other. The naïve Bayes method is used to compute the conditional probability that a given event with features $(f_1, f_2, \dots, f_m)$ is an event in category c_k, i.e. $P(c_k|f_1, f_2, \dots, f_m)$. According to Bayes' theorem and the independence of f_j $(1 \le j \le m)$, the following can be obtained:

$$P(c_k|f_1, f_2, \dots, f_m) \propto P(c_k) \prod_{i=1}^{m} P(f_i|c_k)$$

where $y \propto x$ means y is directly proportional to x, $P(c_k)$ is the probability of events in category c_k occurs, and $P(f_i|c_k)$ is the probability that feature f_i is present given the event is in category c_k.

When using the naïve Bayes model to classify text into different categories (e.g. financial article, health report, etc.), the set of categories that the documents might belong is C. The features of the document (i.e. $(f_1, f_2, \dots, f_m)$) are the words that appear in the document. That is, f_j $(1 \le j \le m)$ corresponds to a word. $P(c_k)$ is the probability that the contents of a document is about topic c_k. For example, if c_k is finance and $P(c_k)$ is 40%, it means that 40% of the documents are in the finance category. $P(f_i|c_k)$ represents the probability that word f_i appears in a document of topic c_k. For example, if f_i is "loss", c_k is "finance" and $P(f_i|c_k)$ is 20%, it means that 20% of the financial report contains the word "loss".

In document classification, the goal is to find the best topic category for a document. This means that, for a given a document, the classifier wants to find the topic c_k (where $1 \le i \le n$) such that $P(c_k|f_1, f_2, \dots, f_m)$ has the maximum value. That is, the classifier finds c_k (where $1 \le i \le n$) such that $P(c_k) \prod_{i=1}^{m} P(f_i|c_k)$ has the maximum value. That is,

$$c_k = arg \max_{c_j \in C} P(c_j|f_1, f_2, \dots, f_m) \quad (*)$$

$$= arg \max_{c_j \in C} P(c_j) \prod_{i=1}^{m} P(f_i|c_k)$$

where $arg\ max$ is the arguments of the maxima function.

The classifier is first trained using the supervised training method. $P(c_k)$ and $P(f_i|c_k)$ $(1 \le k \le n$ and $1 \le i \le m)$ are computed by the classifier during training. Later, when the classifier is given a document to classify, the classifier finds the topic c_k with maximum $P(c_k) \prod_{i=1}^{m} P(f_i|c_k)$; and, c_k is regarded as the topic of the document.

B. Natural Language Processing (NLP)

An English sentence consists of a *complete subject* and a *complete predicate*. The complete subject tells whom or what the sentence is about. The complete predicate tells what the subject does. A sentence might have an *object* that is a noun in the complete predicate. The object receives the action of the action verb in the predicate. For example, consider sentence "The company sold its airport holdings". In this sentence, "The company" is the subject, "sold its airport holdings" is the complete predicate, "sold" is the action verb.

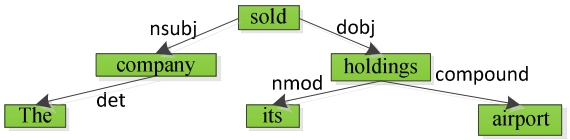

Figure 1 The Dependency Tree of S1

Natural language processing (NLP) is a range of techniques that enable computers to analyse and to understand human languages [3, 4]. Universal dependency has been widely used in NLP tasks [6]. It labels the dependencies (i.e., grammatical relations) between individual words [7]. Dependencies are triplets of the form: (*dependency relation*, *governor, dependent*) [8]. The dependencies in a sentence can be represented as a tree. In the tree, an arrow is drawn from the governor to its dependent. The dependency relation is given as the label of the arrow. For example, the dependency tree of sentence "The company sold its airport holdings" is shown in Figure 1. In the dependency tree, the action verb "sold" is the root of the tree. The Stanford Parser [7] can be used to generate the dependency tree of a sentence.

III. Privacy Guard Service

A. An Overview of the System

The system supports the processing of text and html files. The main entities in the system are shown in Figure 2. Users subscribe to the service of the privacy guard provider. When a user wants to access a file containing sensitive information in public, the user provides the URL of the file to the service provider. The service provider retrieves the file from the URL, analyses the file to identify the sensitive information in the file, converts the sensitive information to a symbol and the JavaScript code that allows the sensitive information to be heard in voice, and sends the processed file back to the user. It is assumed that the OAuth 2.0 protocol [12] is used to allow the service provider to gain access to the user's file from file storage provider. When the user receives the file,

the sensitive information have been converted to meaningless symbols. The JavaScript code behind the symbols convert the original sensitive information to voice. To listen to the original information, a user just taps on the symbols. If the user uses an earphone to listen to the information being read out, the sensitive information will not be revealed to the people that are in close proximity of the user.

Keywords are used as hints for identifying sensitive information. The system processes documents that cover a wide range of topics, e.g. financial, health, etc. A word might have different significance when it is used in articles of different topics. For example, the two sentences below are from a financial report and a health document respectively.

1. The operational loss for 2013 is $10 million.
2. The lack of Zinc in dietary intake causes hair loss.

The word "loss" might be regarded as a sensitive word in the financial report as it associates with commercial secret while it might not be considered as important in a health document. Therefore, different keywords should be used when recognizing sensitive information in documents of different topics. The system classifies documents into several topic categories. A set of predefined keywords is given to each category. Users can modify the keywords sets to suit their needs when configuring the system.

The system uses a naive Bayes classifier for determining the category to which a document belongs. When a document is submitted to the system, the system first uses the classifier to determine the topic presented in the document. Then, the system analyses the document to identify sensitive information using the set of keywords for the identified topic and the universal dependency relations between the words in the document. The system rewrites the document by replacing the identified sensitive information with meaningless symbols and JavaScript code.

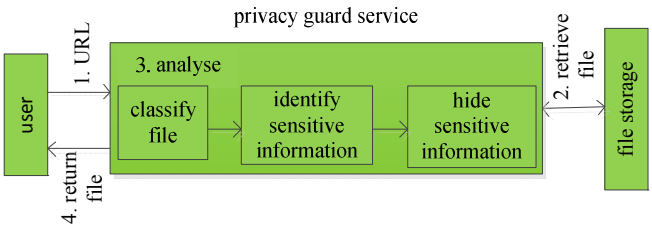

Figure 2 The Conceptual Diagram of the System

B. Topic Category and Keywords Selection

As explained in §III.A, in order to identify the sensitive information more accurately, documents need to be classified into different topics to allow them to be analyzed using the relevant set of keywords. The proposed system only focused on four topic categories, i.e. personal, finance, health, and government. More categories can be easily added to the system.

The personal category covers the documents that contain people's private information such as age, address, etc. The finance category refers to documents that contain financial status information, e.g. gain, budget, etc. Documents in the health category have health-related information, e.g. weight, diagnosis, etc. The government category are the documents on government affairs, e.g. security, emergency, etc.

The system provides a predefined set of keywords for each topic category. First, a small number of keywords for each category are chosen. These keywords are selected according to human knowledge about each topic and common sense. For example, in a financial report, the word "loss" might associate with some sensitive information. Thus, word "loss" is picked as a keyword for the finance topic. A word has many synonyms. People might use the synonyms of the keywords when writing documents. Therefore, WordNet is used to expand the keywords sets. WordNet is a database for the English language. It groups English words into sets of synonyms called synsets [22]. The synsets of each keyword in the initial keywords sets are retrieved for considering being added to the keywords set. Thus, the initially chosen keywords can be regarded as the seeds for further expanding the keywords sets.

The keywords being used in identifying sensitive information depend on the needs of the users. For example, some users might regard the gender information as sensible while others might be happy to freely share that information with others. Therefore, the system provides an interface that allows the users to modify the keywords sets.

C. Classifying Documents

In order to use the relevant keyword set when analyzing a document, the document needs to be classified into a topic category first. Using machine learning techniques to classify text has been studied for many years [19]. The naive Bayes classifier has been proven to be a very effective method for classifying texts [13]. Thus, the proposed system uses the naïve Bayes classifier of the NLTK platform [15] to classify a document before analyzing the contents of the document.

To train the classifier, 30 documents were used for each of the four categories. First, the words that are used as features are chosen. Before the feature words are chosen, the documents are processed to remove the stop words and to find the stem of the words. *stop words* are the words that are not useful in identifying the characteristics of the documents, e.g. "the", "to", etc. A word might have many inflected variants. For example, the following two sentences have the same meaning; and, word "value" means the worth of the house in both sentences. Therefore, instead of treating "value" and "valued" as different features of the documents, only the stem of the word (i.e. "value") should be regarded as a feature of the documents. The Porter stemmer of the NLTK platform is used to obtain the stem of the words in the documents.

- The value of the house is $1 million.
- The house is valued at $1 million.

After removing the stop words and obtaining the stems of the words in the documents, the top 2000 most frequently used words are collected from the documents that are used for training the classifier. These words are used as the

features of the documents. The presence/absence of these words in each of the training documents and the category of the corresponding document are recorded.

A record for each file is created. The record is a tuple $[\boldsymbol{f}, category]$ where $\boldsymbol{f}$ is the feature vector and $category$ is the category to which the file is classified. These records of training files are passed to the classifier during the training. Function **FindFeatures** sets the value of the feature vector of a document. Function **GetFeatures** shows the steps in setting $[\boldsymbol{f}, category]$ for each training document.

GetFeatures($TDocs$)

1 input: $TDocs$ is the set of documents that are used for training the classifier
2 output: $FRec$ is a set of tuples $[FV, category]$. Each tuple records the information extracted from a file. FV is the feature vector of the file. $category$ is the category of the file. $Words$ is the set containing all the feature words
3 **for each** doc **in** $TDocs$ **do**
4 $doc' \leftarrow remove\_stop\_words(doc)$
// replace each word with the stem of the word
5 $doc'' \leftarrow replace\_with\_stem(doc')$
// replace the original documents with the modified
// documents to allow the stems of the words to be
// used as feature words
6 $TDocs \leftarrow TDocs - \{doc\} \cup \{doc''\}$
7 **end_do**
// initialize the return value
8 $FRec \leftarrow \emptyset$
9 let $Words$ be the set that contains the 2000 most frequently used words in all the files in $TDocs$
// set the value of the feature vector for each file
10 **for each** doc **in** $TDocs$ **do**
// set the feature vector for document doc
11 $FV \leftarrow$ FindFeatures($Words, doc$)
// $doc.category$ is the topic category of file doc. It
// is manually assigned to doc when doc is selected
// as a training file.
12 $FRec \leftarrow FRec \cup \{[FV, doc.category]\}$
13 **end_do**
14 **return** $FRec,\ Words$

FindFeatures($Words$, doc)

1 input: $Words$ is a set of feature words
doc is a document
2 output: FV is a mapping relation that corresponds to the feature vector with each element corresponds to a feature word
3 let FV be a mapping relation such that
$FV: Words \rightarrow \{true, false\}$
// set the value of each element in the feature vector
4 **for each** w **in** $Words$ **do**
// The value of the vector element corresponding to
// word w is set to true if file doc contains w;
// otherwise, the element is set to false.
5 **if** w **in** doc **then** $FV[w] \leftarrow true$
6 **else** $FV[w] \leftarrow false$
7 **fi**
8 **end_do**
9 **return** FV

Here is an example showing how feature vectors are set. For simplicity, it is assumed that only four words, i.e. diagnose, drug, margin and profit, are used as feature words for the documents. Thus, each feature vector consists of four elements, i.e. (f_1, f_2, f_3, f_4). The four elements f_1 to f_4 correspond to words diagnose, drug, margin and profit respectively. If the word that corresponds to f_j $(1 \leq j \leq 4)$ appears in the document, f_j is set to true. Otherwise, f_j is false.

This example shows how the features of two files are recorded. The two files are in finance and health category respectively. Due to the constraints on the length of the paper, only some of the sentences in the files are given.

- Some sentences in the financial file: Gross margin decreased significantly to 23.76%. Operating profit turned negative in 2015, at -98,
- Sentence in the health report: Erica was first diagnosed with a thyroid condition at the age of 21.

According to **FindFeatures** (lines 4 to 8), when checking the first file, since "margin" and "profit" both appear in the file, the elements corresponding to these words in the feature vector, i.e. f_3 and f_4, are set to $true$. The elements corresponding to the words that do not appear in the file, i.e. f_1 and f_2, are set to $false$. Hence, the vector for the finance file is $(false, false, true, true)$. Since the first file is in the finance category, the tuple representing the features and the category of the file is $[(false, false, true, true), finance]$.

For the second file, after each word is substituted with the stems of the words, word "diagnosed" is replaced with its stem "diagnose" (line 5 and 6 of **GetFeatures**). Since "diagnose" is the only feature word appearing in the file and it corresponds to f_1 in the feature vector, the feature vector of the second file is $(true, false, false, false)$. Since the second file is in category health, the tuple representing the second file is $[(true, false, false, false), health]$.

Set $FRec$ contains tuples to be fed to the naïve Bayes classifier to compute $P(c_k)$ and $P(f_i|c_k)$ in formula (*) of §II.A. Once the training is completed, the classifier can be used to predict the category of a given document based on formula (*). Function **Predict** describes the steps in using the classifier to predict the category of a document.

Predict($doc, Words$)

1 input: doc is the document to be classified
$Words$ is the set of feature words
2 output: the category to which doc belongs
// replace each word with the stem of the word
3 $doc \leftarrow$ replace_with_stem(doc)
// set the feature vector for document doc

4 $FV \leftarrow \text{FindFeatures}(Words, doc)$
// use the naïve Bayes classifier to find the category of doc
5 **return** $Naive\_Bayes(FV)$

D. Identifying Sensitive Information

Once the category of a document is identified, the keywords assigned to the category are used to match the words in the document to recognise the sensitive information. As the document might contain the inflected variants of the keywords, when checking whether a word in the document matches a keyword, the stem of the word is used in the checking. For example, assume that "value" is a keyword. When matching the words in sentence "The house is valued at $1 million." for keywords, word "valued" would be mapped to its stem, i.e. "value", first. As a result, "valued" would be identified as matching a word in the keywords set. Thus, "valued" will be marked as a sensitive word.

If a sentence contains a keyword, it is likely that some sensitive information can also be revealed by some words that are associated with the keyword in the sentence. For example, assume that "value" is a keywords; and consider sentence "The house is valued at $1 million.". Clearly, if "value" is sensitive, the subject of the sentence (i.e. "house") and the worth of the house (i.e. "$1 million") are both sensitive information. Therefore, only hiding the words that match keywords is insufficient. The words associated with the keywords should also be hidden when the sentence is displayed. In this example, when the sentence is shown to the user, the sentence should appear as "The ~~house~~ is ~~valued~~ at ~~$1 million~~.". The strikethrough words represent the sensitive information that are replaced by symbols.

Universal dependency is used to discover the sensitive information associated with the keywords. As keywords might appear in different parts of a sentence, the sentences have to be analysed differently according to the positions of the keywords in the sentences. The discussion here only show some major cases. A complete discussion can be found in [21].

1) The keyword is the action verb of a sentence.

Examples of this case are the two sentences below:

S1. The company sold its airport holdings.

S2. The house is valued at $1 million.

Assume that "sell" and "value" are keywords. For this type of sentences, the keyword describes what the subject does. Thus, if the subject does a sensitive action, the identity of the subject might be sensitive information. Hence, the proposed scheme treats the subject as sensitive information for this type of sentences. "The company" and "The house" are the subjects of the two sentences respectively. From Figure 1 and 3, it can be seen that the verb of a sentence is the root of the dependency tree. In active voice sentences (e.g. S1), the relationship between the verb and the subject is nsubj as shown in Figure 1. For passive voice sentences (e.g. S2), the relationship between the verb and the subject is nsubpass as shown in Figure 3. Therefore, the subject of the sentences can be identified using these relationships.

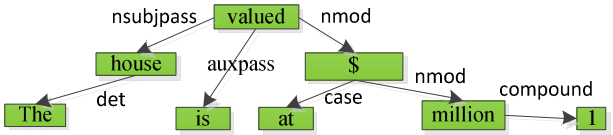

Figure 3 The Dependency Tree of S2

In an active voice sentence, the object of the sentence (i.e. "its airport holdings" in S1) is acted on by the verb. The object is rooted in the subtree that has a dobj relation with the verb. In a passive voice sentence, the subtree that has the nmod relation with the verb contains the description about how the verb affects the subject. If the verb is a sensitive word, the effect caused by the verb might also be sensitive information. Thus, the information in the subtrees of the verb should be marked as sensitive.

The prepositions, determiners (e.g. "with", "at", "the", etc.) and auxiliary verbs (e.g. "be", "have", etc.) normally do not reveal any sensitive information. Thus, there is no need to hide them. They can be identified through the "case", "det", "auxpass" and "aux" relations in the dependency tree as shown in Figures 1 and 3. Therefore, after being processed by the system, the two sentences will appear as below:

S1'. The ~~company~~ ~~sold~~ ~~its airport holdings~~.

S2'. The ~~house~~ is ~~valued~~ at ~~$1 million~~.

2) Keywords are nouns in the object of a sentence.

Examples of this case are the sentences below:

S3. The doctor gave the patient an antithyroid drug.

S4. The project costs one million dollars.

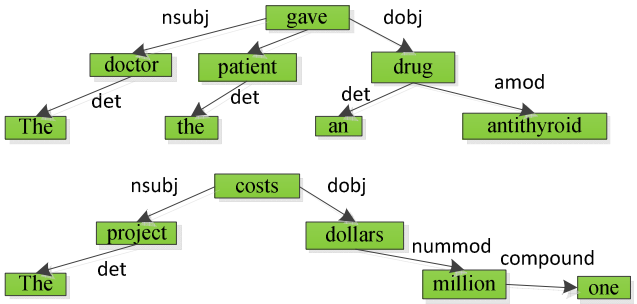

Figure 4 The Dependency Trees of S3 and S4

Assume the keywords are "drug" and "dollar". The dependency trees of the two sentences are in Figure 4. The adjectival modifier of a word might contain valuable information about the word. In S3, "antithyroid" is the adjectival modifier of "drug". They would reveal the type of the drug. Thus, the adjectival modifier of a keyword should also be regarded as sensitive. The compound relation combines two words to form a compound word. In S4, "one" and "million" form a compound word. Together, they are the numeric modifier (i.e. one million) of word "dollar". If "dollar" is a sensitive word, the phrase (i.e. "one million") that modifies the meaning of "dollar" is likely to be sensitive as well. Therefore, after being processed by the system, the two sentences will appear as below:

S3'. The doctor gave the patient an ~~antithyroid~~ ~~drugs~~.

S4'. The project costs ~~one million~~ ~~dollars~~.

3) Keywords are nouns in the subject of a sentence.

An example of this case is as below:

S5. Gross margin decreased significantly.

Assume "margin" is the keyword. The word that forms a compound word with a keyword normally provides some hints on the keyword. For example, given the compound word "Gross xxx", it is relatively easy to guess that "xxx" is "margin". This means only hiding the keyword is insufficient. Thus, the word that has a compound relation with a keyword should be marked as sensitive. Hence, "Gross" which has a compound relation with keyword "margin" should be marked as a sensitive word.

A sentence normally describes the actions taken by the subject of the sentence. If the subject is sensitive, the action taken by the subject is also likely to be sensitive. In S5, the action verb "decreased" reflects the financial status of the company. Thus, the action verb should be regarded as sensitive. The adverbial modifier (i.e. "significantly") of the action verb in S5 shows the severity of the action; and, it has an important impact on the stock price of the company. Hence, the adverbial modifier of a sensitive verb should be treated as sensitive as well. After processing, the sentence being displayed should be:

S5'. ~~Gross margin decreased significantly~~.

E. Handling Tables

Some documents use tables to display data. Normally, tables have row and column headers that indicate the types of data being displayed in the table. If the headers are in the keywords set, the data in the corresponding rows or columns are regarded as sensitive data. For example, assume that "asset" is in the keyword set. According to the row header, all the numbers appearing in the row with header "Assets" are treated as sensitive data.

Two-year Financial Overview	2014	2013
Assets	18.7	15.4
Admission rate	6.7%	6.3%

The system currently supports tables appearing in html file. By analysing the html table tags, it is possible to associate row and column headers with their corresponding data.

F. Processing Steps

The pseudo code below describes the steps in analysing a document for sensitive information. As explained in §III.D, the subtrees containing the subject of a sentence can be obtained by finding the subtrees that have nsubj and nsubjpass relation with the root of the dependency tree (line 7). The other subtrees form the predicate of the sentence (line 8). If the action verb is a keyword, according to §III.D.1, the subject as well as the predicate should be marked as containing sensitive information (lines 9 to 11). The same thing happens if the subject contains a keyword as explained in §III.D.3 (lines 12 to 14). If a sensitive word is a noun in the predicate of a sentence, as discussed in § III.D.2, only the word and the words associated with the word need to be marked as sensitive (lines 15 to 17).

```
    IdentifySensitive(doc, Keywords)
1   input: doc is a document
           Keywords is the set of keywords
2   output: doc is the document with all the sensitive
            words being marked
3   let Sen be a set containing all the sentences in doc
4   // process each sentence in document doc
    for each s in Sen do
5   // use Stanford parser to generate the dependency tree
      d_tree ← parser(s)
6     let r be the root of the dependency tree d_tree
7     let Sub be the set of words in the subtree containing
        the subject of a sentence
8     let Pre be the set of words in the subtrees
        containing the predicate of a sentence
      // check whether the stem of the dependency tree's
      // root (i.e. the action verb of the sentence) is a
      // keyword
9     if stem(r) in Keywords then
10      Mark_Sensitive(r)
11    fi
      // check whether the stem of any word in the subject
      // of the sentence is a sensitive word
12    if (∃w: Sub. (stem(w) ∈ Keywords)) then
13      Mark_Sensitive(r)
14    fi
      // check whether the stem of any word in the
      // predicate of the sentence is a sensitive word
15    if (∃w: Pre. (stem(w) ∈ Keywords)) then
16      Mark_Sensitive(w)
17    fi
18  end-for-each // s in Sen
19  let Tables be the set of all the tables in doc
    // check each table for sensitive information
20  for each t in Tables do
21    let RowH and ColumnH be the set of the
        headers of rows and columns respectively
22    for each h in RowH/ColumnH do
        // check whether the header is a sensitive word
23      if h ∈ Keywords then
24        mark all the data in h's row/column as sensitive
25      fi
26    end-for-each // h in RowH/ColumnH
27  end-for-each // t in Tables
```

Mark_Sensitive is a recursive function for marking sensitive words in a sentence. In **Mark_Sensitive**, set *NERel* contains the universal dependency relations that link a node in the dependency tree to a word that is unlikely to reveal any sensitive information. Examples of these nonessential relations are: coordinating conjunction, case marking, determiner, auxiliary verbs, etc. Typical words associated with these types of relations are: "and", "by",

"the", "is", etc. The function propagates the marking to all the nodes in the tree rooted at a given node n.

```
    Mark_Sensitive(n)
1   input: n is a node in the dependency tree
2   let children(n) denote the set of n's children in the
        dependency tree
3   for each c in children(n) do
4       let rel be the dependency relation between n and c
        // If c might reveal sensitive information, mark c as
        // sensitive and propagate the marking to the
        // subtree rooted at c.
5       if ¬(rel ∈ NERel) then
6           label n as sensitive;  mark_sensitive(n)
7       fi
8   end_for_each
```

G. Accessing Sensitive Information

Once the sensitive words in a document are marked. They are converted to meaningless symbols. Text-to-speech code is embedded behind the symbols. The code enable the users to listen to the content of the original sensitive information through an earphone when the users tap on the symbols. The project uses the ResponsiveVoice Text-To-Speech library [18] for converting text to voice. The library is only 14KB. The code behind the symbols are JavaScripts that call a text-to-voice conversion routine of the library.

IV. Evaluation

Experiments were carried out to evaluate the accuracy and the efficiency of the proposed scheme. In the experiments, the machine hosting the privacy guard service is Dell Latitude E6540 with a 2.5GHz Intel Core i5-2520M processor, 8GB memory and 64-bit Windows 7. 140 files were chosen from the NLTK corpus [15] and articles available on the Internet for the evaluation. The files are from four categories with 35 files for each of the four file categories. For each category, 30 files were randomly selected for training the naïve Bayes classifier and the other 5 files were used for evaluating the accuracy and the performance of the system. These file were manually checked to ensure that they contain sensitive information that are covered by the keywords. The size of each file has been trimmed to about 500 words.

A. The accuracy of the naïve Bayes classifier

After being trained with 120 files, 20 files were fed to the classifier for identifying the category of the files. Amongst the 20 files, two files were wrongly classified. This gave an accuracy rate of 90%. Thus, it appears that the classifier performs reasonably well for the task. For the two wrongly classified files, they both contain a fair number of keywords for multiple categories. This seemed to be the reason that the classifier was confused.

B. The effectiveness in identifying sensitive information

To ensure that there are sufficient files for each file category, the two files that were wrongly classified were replaced with two other files that can be correctly classified by the classifier. There are 10243 words (including numbers) in total in the 20 files used in the evaluation. Checking the file manually revealed that about 2711 words should be marked as sensitive. The system treated 2941 words as sensitive. Amongst the 2941 words, 2625 are the expected sensitive words (i.e. true positive); and 316 words are wrongly identified as sensitive words (i.e. false positive). This means that 86 expected sensitive words were not identified by the system (i.e. false negative).

An analysis of these false negative cases showed that they are mostly numbers in tables. The misidentification is due to the html file is not well formed. That is, the opening and the closing tags of rows, columns and the table do not always match. This makes the system unable to align the column titles and the data correctly. The false positive cases are due to the aggressiveness of the system in marking words as sensitive. For example, according to §III.D.3, the words in the predicates are marks as sensitive if the subject of a sentence contains a keyword. Sometimes, even if the subject has a keyword, the predicate does not necessarily to be sensitive information. For example, in "Grave's disease affects a small portion of adults.", word "disease" in the subject is a keyword, the predicate "affects a small portion of adults" does not need to be regarded as sensitive as it relates to a general knowledge.

The precision and recall of the system are as below. From the precision and the recall rates, it can be seen that the accuracy of the system is reasonably high.

$$precision = \frac{true\ positive}{true\ positive + false\ positive} = 89.26\%$$

$$recall = \frac{true\ positive}{true\ positive + false\ negative} = 96.7\%$$

C. The efficiency of the proposed scheme

The table below shows the time of processing one file. The reported time is the average time of processing 20 files. The processing time of each file was the average of 10 runs.

	classify	identify	parser 1	parser 2
time (ms)	314	140	4140	2856

"classify" is the time for classifying a document. "identify" includes the time for calculating the stems of words, searching for keywords in the document, and marking the sensitive words. "parser 1" is the time for using one Stanford parser to generate the dependency trees for all the sentences in a document. It can be seen that the execution is dominated by the time for generating the dependency trees of the sentences in a document. The creation of the dependency trees of different sentences are independent of each other. Thus, on a multicore machine, it is possible to concurrently run several instances of the Stanford parser using multiple threads. "parser 2" is the time for running two instances of the Stanford parser concurrently with each instance being given half of the sentences in the file. It can be seen that there is a 30% reduction in the execution time. Therefore, with more hardware available, it might be possible to bring the execution time down to a more acceptable level.

V. Related Work

Hiding critical information from public viewing has been used in many systems. In 1965, Xing first proposed not to print password on login screen to avoid sensitive data leaks [5]. Screen protectors [1] can narrow the viewing angle of screens; but the contents on the screen are still visible from certain angles. Honan defined the visual security problem and gave advices on how to tackle the problem [10]. The advices given by Honan are general principles and guidelines. They do not provide any concrete solutions to the problem.

Mitchell et al. developed a system for replacing sensitive data with meaningless symbols [14] when displaying form-based data. In [14], a cashtag is used to represent a sensitive data item. A table consisting of pairs of cashtag and its corresponding sensitive data is created. When the sensitive data need to be displayed, they are replaced by their corresponding cashtags. To understand the displayed information, users need to memorise the meaning of the cashtags. Unlike [14], the system in this paper does not require the users to remember the meaning of any symbol. Also, instead of using a fixed mapping table, the proposed system used NLP techniques to locate sensitive information. This allows the system to recognise sensitive data that have been overlooked by the users when defining the keywords sets. Different to [14], the proposed system is used to display full document instead of simple form-based data.

WHYPER [16] and SUPOR [11] were developed to carry out privacy and security checks. Both schemes used the Stanford parser to parse the application descriptions into individual words to identify input fields that contain sensitive data. They do not need to study the relationship between the words. Unlike [11, 16], the scheme in this paper needs to analyze the dependencies between the words to identify the words that are associated with the sensitive words. [11] and [16] were developed for identifying and tracking sensitive data through applications. Different to [11] and [16], the system in this paper is for hiding sensitive information in the document.

VI. Conclusions

This paper proposed a privacy guard service that helps people to safe-guard their sensitive data while viewing information in public. The system uses keywords as hints for identifying sensitive data. To improve the accuracy of the service, instead of using a single keywords set, files are classified into several categories; and keywords are defined for each category. The system uses a naïve Bayes classifier to determine the category to which a document belongs. This allows the correct set of keywords to be used when analysing a document. When analysing a document, not only the keywords are identified, the words associated with the keywords that are likely to contain sensitive information are also recognized using NLP techniques. Experiments showed that the proposed system provides good accuracy in the identifying sensitive information. Future work will be focused on improving the processing speed of the system through parallel processing and widen the types of files that can be handled by the system.

References

[1] 3M, Privacy and Screen Protectors http://solutions.3m.com/wps/portal/3M/en_EU/Protectors, accessed on 2017-02-05

[2] 3M, Shoulder Surfing Survey. 2007. http://multimedia.3m.com/mws/media/463995O/privacy-filters.pdf, accessed on 2017-02-05

[3] S. Bandyopadhyay, S. K. Naskar and A. Ekbal, "Emerging Applications of Natural Language Processing: Concepts and New Research", IGI Global, Information Science Reference, 2013.

[4] E. Cambria and B. White, "Jumping NLP curves: A review of natural language processing research," IEEE Computational Intelligence Magazine, vol. 9, no. 2, pp. 48-57, 2014.

[5] P.A. Crisman, CTSS Programmers Guide, 2nd Ed., MIT Press, 1965

[6] M. de Marneffe et al. Universal Stanford dependencies: A cross-linguistic typology. LREC 2014: 4585-4592

[7] M. C. De Marneffe, B. MacCartney, and C. D. Manning, "Generating typed dependency parses from phrase structure parses," In Proc. 5th LREC, 2006, pp. 449-454.

[8] M. C. De Marneffe, "Stanford typed dependencies manual," 2008.

[9] FlexJobs https://www.flexjobs.com/blog/post/survey-76-avoid-the-office-important-tasks/, accessed on 2017-02-05

[10] Honan, Brian. "Visual Data Security White Paper", July 2012. BH Consulting & European Association for Visual Data Security. http://www.visualdatasecurity.eu/wp-content/uploads/2012/07/ Visual -Data- Security-White-Paper.pdf, accessed on 2017-02-05

[11] J. Huang, Z. Li, X. Xiao, Z. Wu, K. Lu, X. Zhang, and G. Jiang. SUPOR: Precise and Scalable Sensitive User Input Detection for Android Apps. In Pros. of the 24th USENIX Security Symposium, pp 977-992, 2015

[12] IETF, The OAuth 2.0 Authorization Framework, https://tools.ietf.org/html/rfc6749, accessed on 2017-02-05

[13] Christopher D. Manning, Prabhakar Raghavan & Hinrich Schütze (2008). Introduction to Information Retrieval. Cambridge University Press

[14] Michael Mitchell, An-I Andy Wang, and Peter Reiher. 2015. Cashtags: protecting the input and display of sensitive data. In Proceedings of the 24th USENIX Conference on Security Symposium (SEC'15), CA, USA, 961-976.

[15] Natural Language Toolkit, http://www.nltk.org/, accessed on 2017-02-05

[16] R. Pandita, X. S. Xiao, W. Yang, W. Enck, and T. Xie, "WHYPER: Towards Automating Risk Assessment of Mobile Applications," In Procs of the 22st USENIX conference on Security symposium. 2013.

[17] Pillai, Geetha. "Caught on Camera: You are Filmed on CCTV 300 Times a Day in London", International Business Times, March 2012. http://www.ibtimes.co.uk/britain-cctv-camerasurveillancewatch-london-big-312382, accessed on 2017-02-05

[18] Responsivevoice, https://responsivevoice.org/, accessed on 2017-02-05

[19] Russell, Stuart; Norvig, Peter (2003). Artificial Intelligence: A Modern Approach (2nd ed.). Prentice Hall.

[20] Smsglobal, Smartphone Ownership, Usage And Penetration By Country, http://thehub.smsglobal.com/smartphone-ownership-usage-and-penetration, accessed on 2017-02-05

[21] F. Wang, A Privacy Guard for Mobile Applications, MSc Thesis, The University of Auckland, New Zealand, 2017

[22] WordNet. https://en.wikipedia.org/wiki/WordNet. Accessed on 2017-02-08.