A Study of SVM Kernel Functions for Sensitivity Classification Ensembles with POS Sequences

Graham McDonald  
University of Glasgow  
Glasgow, Scotland, UK  
G.McDonald1@research.gla.ac.uk

Craig Macdonald  
University of Glasgow  
Glasgow, Scotland, UK  
Craig.Macdonald@glasgow.ac.uk

Nicolás García-Pedrajas  
University of Córdoba  
Córdoba, Spain  
npedrajas@uco.es

Iadh Ounis  
University of Glasgow  
Glasgow, Scotland, UK  
Iadh.Ounis@glasgow.ac.uk

ABSTRACT

Freedom of Information (FOI) laws legislate that government documents should be opened to the public. However, many government documents contain sensitive information, such as confidential information, that is exempt from release. Therefore, government documents must be sensitivity reviewed prior to release, to identify and close any sensitive information. With the adoption of born-digital documents, such as email, there is a need for automatic sensitivity classification to assist digital sensitivity review. SVM classifiers and Part-of-Speech sequences have separately been shown to be promising for sensitivity classification. However, sequence classification methodologies, and specifically SVM kernel functions, have not been fully investigated for sensitivity classification. Therefore, in this work, we present an evaluation of five SVM kernel functions for sensitivity classification using POS sequences. Moreover, we show that an ensemble classifier that combines POS sequence classification with text classification can significantly improve sensitivity classification effectiveness (+6.09% F2) compared with a text classification baseline, according to McNemar’s test of significance.

1 INTRODUCTION

Freedom of Information (FOI) laws state that government documents should be open to the public. However, many government documents contain sensitive information, such as confidential information. Therefore, FOI laws exempt sensitive information from release and government documents must be sensitivity reviewed prior to release, to identify and close any sensitivities. However, with the introduction of born-digital documents, such as email, the volume of documents has increased and document creation processes have become less structured. Hence, traditional sensitivity review processes are not viable for digital sensitivity review.

Automatic classification techniques can potentially be adapted to assist the digital sensitivity review process and reduce the time taken to review documents. McDonald et al. [12] showed that sensitivities relating to information supplied in confidence could be captured in the grammatical structure of documents, by representing the documents as sequences of Part-of-Speech (POS) n-grams [11]. For example, sensitivities relating to information supplied in confidence are often recounts of dialogues or actions and, therefore, can contain strings such as “an informer gave him”, “the ambassador said she” or “a detainee showed us”. These strings can all be represented by the POS sequence DT NN VB PR, or subsequently as a sequence of POS n-grams, e.g. as POS 2-grams DTNN NVB VBPR. McDonald et al. [12] showed that the frequencies of certain POS sequences can be an indicator of potential sensitivity.

Representing documents by an abstraction, such as the POS tags they contain, has an additional attractive by-product. In effect, a document’s tokens (POS n-grams) can be viewed as a sequence of symbols from an alphabet, rather than terms from a vocabulary and, hence, gives rise to the possibility of developing techniques based on sequence classification [18]. Sequence classification has been shown to be effective in fields such as Bioinformatics (e.g. classifying protein sequences) and Cyber-Security (e.g. intrusion detection), in addition to Information Retrieval (IR) tasks (e.g. bot detection from query log sequences). An intrinsic component of sequence classification is selecting a classification kernel function that is suitable for the classification task being attempted, for example, sequence-similarity kernels such as the Spectrum kernel [10].

Our contributions in this work are two-fold. Firstly, we present a thorough evaluation of five SVM kernel functions for POS sequence classification of sensitive information that would be exempt from release under UK FOI laws. Secondly, we show that a weighted majority vote ensemble classifier that combines POS sequence classification with text classification can significantly improve sensitivity classification (+6.09% F2) compared to a text classification baseline, according to McNemar’s test of significance.

This paper is structured as follows: Section 2 discusses prior work; Section 3 provides an overview of the kernel functions that we deploy; We present our experimental setup in Section 4 and our results in Section 5; Concluding remarks follow in Section 6.

2 RELATED WORK

Most of the existing literature on automatically identifying sensitive information has addressed the task of masking personal data [2, 5]. However, sensitive information in government documents is more
wide-ranging than personal information and can include, for example, issues of confidentiality or international relations. Gollins et al. [6] posited that IR technologies could assist the digital sensitivity review process. They also noted that some sensitivities, such as international relations, can pose more of a risk due to the potential effect of accidental release. Hence, there is a need for automatic techniques for classifying these more wide-ranging types of sensitivity.

Text classification has been shown to be an effective approach as a basis for automatic sensitivity classifiers [1, 13]. Text classification relies on there being a specific set of terms, for which their distribution can be a reliable indicator of the class that is to be identified. However, as Gollins et al. [6] noted, sensitivity arises not only from the terms in a document but also from the context in which they appear and, therefore, sensitivity classification must go beyond term features (and text classification). In this work, we focus on combining text classification with sequence classification techniques for sensitivity identification.

McDonald et al. [12] showed that Part-of-Speech (POS) n-gram sequences can be effective for identifying supplied in confidence sensitivities. They adopted an approach from Lioma and Ounis [11], who showed that more frequent POS n-grams in a collection are likely to bear more content. McDonald et al. used the distributions of POS n-grams in sensitive and non-sensitive text to measure the sensitivity load of text sequences. Differently from the work of [12], in this work, we use POS sequences to study different SVM kernel functions for sensitivity classification. Moreover, we investigate methods of ensemble learning for effectively combining POS sequence classification with text classification for sensitivity.

Ensemble classification [3] methods combine the decisions from a committee of individual classifiers with a view to improving the overall classification performance. The simplest, but often most effective, of these approaches combines the predictions from the committee classifiers by viewing each classifier’s prediction as a vote for the class of a document [9]. Another popular approach, namely stacking [17], is to learn a separate (meta-learner) combiner function from the predictions of the committee classifiers. In this work, we investigate weighted voting and stacking ensembles for combining sequence and text classification for sensitivity classification.

### 3 SVM KERNEL FUNCTIONS

As previously stated in Section 1, an intrinsic component of a new sequence classification task is to identify a suitable kernel function for the task. Therefore, in this section we provide an overview of the kernel functions and classifier that we deploy for POS sequence classification for sensitivity.

Support Vector Machines (SVM) [16] are a type of supervised learning algorithm that learn a linear separating hyperplane between two classes within a vector space. SVM achieves this by solving a dual optimisation problem on a set $S$ of training instance vectors, $x_i$, with corresponding class labels, $y_i$, where $i = 1...m$, $x_i \in \mathbb{R}^n$ and $y_i \in \{\pm1\}$. The SVM optimisation aims to 1) maximise the distance between the hyperplane and the closest instances in either of the classes, and, 2) minimise the classification error. The resulting optimisation problem, $\text{Maximise } \Sigma_i a_i - \frac{1}{2} \Sigma_i \Sigma_j a_i a_j y_i y_j \langle x_i, x_j \rangle$, requires learning the optimal weights, $a_i$ for $i = 1...m$, where $a_i \geq 0$. Since this optimisation problem relies on the inner products $\langle x_i, x_j \rangle$, which can be viewed as a distance measure, this component of the optimisation can be substituted by a kernel function, $K(x_i, x_j)$, that computes a measure that is selected for the classification task.

The linear kernel, defined as $K_{\text{linear}}(x_i, x_j) = x_i^T x_j$, is the simplest kernel. However, $K_{\text{linear}}$ has desirable properties in that it is very fast to train and does not tend to over-fit the learned model to $S$ when $|x|$ is very large [7]. For non-linearly separable data, a more suitable kernel is the Gaussian kernel, $K_{\text{gaussian}}(x_i, x_j) = \exp\left(-\frac{|x_i - x_j|^2}{2\sigma^2}\right)$, where $\sigma$ is a parameter that determines the width of the Gaussian function, i.e. the region of influence for an instance in vector space. A properly tuned Gaussian kernel will always be able to learn the optimal decision of a linear kernel [8], yet tuning $\sigma$ can be expensive and does not guarantee obtaining a better model.

By substituting $\langle x_i, x_j \rangle$ with a kernel function, we effectively create a feature map, $\phi$, which maps an instance, $x$, to a new (possibly higher dimensional) space. For the linear and Gaussian kernels, $\phi$ is implicit within the dot products defined in the functions. Often, however, kernels explicitly define this mapping as the input to the kernel function. String kernels operate on finite subsequences of strings and the Spectrum kernel [10] is a simple string kernel defined by its map $\phi$ over all sub-sequences in an alphabet $A$. For a given alphabet $A$, $|A| = 1$, a document’s feature map, $\phi(x) = (\phi(a(x))_{a \in A}^k$, is the frequency weighted set of all contiguous subsequences of length $k \geq 1$, that the document contains, i.e. its k-spectrum, and where $\phi(a(x))$ is the frequency of $a$ in $x$. The Spectrum kernel is then defined as $K_{\text{spectrum}}(x, y) = \langle \phi(x), \phi(y) \rangle$.

One limitation of the Spectrum kernel is that it is constrained to exact matches when calculating the similarity of instances. The Mismatch Kernel [4] addresses this by allowing for a pre-defined number of mismatched symbols within sequences. For a given sequence $a = a_1..a_k$, $a \in A$, $N(k, m)(a)$ is the set of all $k$-length sequences, $b = b_1..b_k$, $b \in A$ that differ from $a$ by at most $m$ mismatches. The Mismatch kernel’s feature map is then defined as $\phi_{k,m}(a) = (\phi(\beta(a)))_{\beta \in A}$, where $\phi_{\beta}(a) = 1$ if $\beta \in N(k, m)(a)$, else $\phi_{\beta}(a) = 0$. From this feature map, the $(k,m)$-mismatch kernel is defined as $K_{k,m}(x, y) = \langle \phi_{k,m}(x), \phi_{k,m}(y) \rangle$.

Finally, the Smith-Waterman kernel, $K_{SW}$, is based on the Smith-Waterman sequence similarity algorithm [15]. Unlike the kernels presented thus far, it is not strictly a kernel function, since it does not satisfy certain mathematical conditions, e.g. it is not always positive definite. However, in this work, we test its effectiveness as a kernel function for POS n-gram sequence classification.

For complex sequence classification tasks, a single SVM kernel may not provide an optimal solution. One method of addressing this is to combine multiple simpler kernels as a hybrid kernel, with the aim of considering multiple aspects of an instance vector. We hypothesise that different types of kernels will identify different aspects of sensitivity and, therefore, in this work, we evaluate two hybrid kernels that are a linear combination of the scores from two simpler kernels, namely Spectrum+Linear and Spectrum+Gaussian.

### 4 EXPERIMENTAL SETUP

#### Collection:
Our test collection is 3801 government documents that have been sensitivity reviewed by government reviewers. All documents that contain any sensitivity relating to Personal Information or International Relations FOI exemptions were labeled as sensitive. All other documents were labeled not-sensitive, resulting in a binary classification task with 502 sensitive and 3299 not-sensitive...
documents. We perform a 5-fold Cross Validation and randomly
down-sample non-sensitive documents to balance the training data.

**Baseline:** As a baseline we use a text classification approach with
binary bag-of-words features. For this approach, we use SVM with
a linear kernel and $C = 1.0$, since these *default* parameters are
known to be effective for text classification [7, 14] and can provide
a strong baseline for sensitivity classification [1, 13].

**Sequence Classification:** For the POS sequence representations,
following [11, 12], we use the TreeTagger\(^1\) part-of-speech tagger
to POS tag documents using a reduced set of 15 POS tags. We then
create separate $n$-gram sequence representations of the collection,
resulting in individual $n$-gram sequence collections for $n = \{1...10\}$.

Table 1 presents the number of observed unique tokens in the alpha-
bet, $A$, for each size of $n$. For the linear and Gaussian kernels, we
represent documents as token frequency vectors. For the Spectrum,
Mismatches and Smith-Waterman kernels, we count the frequency of
$k$ length sub-sequence matches in a pair of documents. We train
a separate committee classifier for each size of $n$-gram sequence,
resulting in $n$ votes per kernel as input to the ensemble approaches.

**Ensemble Classification:** For the ensemble approaches, we com-
bine the predictions of the text classification $P_t$ with the predictions
of $n$ sequence classifiers $P_n$, resulting in $n + 1$ document features
$f$, $f \in \{P_t, P_n\}$. We test four combination methods. Firstly, in
Weighted Majority Vote (WMV), to predict a document’s class, $P_n$
is assigned a weight $w$ for each fold and the document’s overall pre-
diction score is calculated as $\frac{\sum f \cdot w}{n + 1}$. The remaining three
combination methods are *stacking* approaches. These require an
intermediate step where $P_t$ and $P_n$ are predictions for a validation
set for each of the 5-fold Cross Validation folds. $P_t$ and $P_n$ are then
concatenated and the resulting $n + 1$ predictions (per document) are
used to train the combiner. We test three classifiers as combiners,
namely Logistic Regression (LR), SVM and Random Forests (RF).

**Classification and Parameters:** We use scikit-learn\(^2\) and extend
LibSVM\(^3\) with the Spectrum, Mismatch and Smith-Waterman ker-
nels. Parameter values for the sequence and combinator classifiers
are selected by 10-fold Cross Validation on training and validation
sets respectively, for each of the 5-fold Cross Validation folds. We
vary SVM’s $C$ parameter exponentially in the range $[0.001,10000]$, and
similarly for $\gamma$ parameters in $[0.0001,10]$. For sequence classifi-
cation, sub-sequences are varied for $k = \{3, 6, 9, 12\}$. For ensemble
combinators: for WMV, we test $w = \{1...10\}$; for LR we select $L_t$ as
our loss function and vary $C$ in the same range as for SVM; for RF,
we test number of trees $t = \{100, 250, 500, 750, 1000\}$. We optimise
for area under the Receiver Operating Characteristic curve (auROC).

**Metrics:** We select auROC as our main metric for measuring kernel
effectiveness, since it is calculated over all decision thresholds for a
classifier. Additionally, we report precision (P), True Positive Rate
(TPR), True Negative Rate (TNR), $F_1$, $F_2$ and Balanced Accuracy
(BAC) metrics. We report statistical significance using McNemar’s
non-parametric test, with $p < 0.001$. Significant improvements com-
pared to the text classification baseline are denoted by $+$ in Table 4.

---

\(^1\)http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/
\(^2\)http://scikit-learn.org/
\(^3\)https://cise.ntu.edu.tw/~cjlin/libsvm/

### Table 1: The total unique POS $n$-gram tokens in each collection representation.

<table>
<thead>
<tr>
<th>Unique Tokens</th>
<th>1-gram</th>
<th>2-gram</th>
<th>3-gram</th>
<th>4-gram</th>
<th>5-gram</th>
<th>6-gram</th>
<th>7-gram</th>
<th>8-gram</th>
<th>9-gram</th>
<th>10-gram</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>209</td>
<td>1877</td>
<td>11408</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>51238</td>
<td>172109</td>
<td>441251</td>
<td>888837</td>
<td>1465215</td>
<td>2052063</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 2: Results for the best performing size of $n$-gram for stand-alone POS sequence classification, according to the area under the ROC curve (auROC).

<table>
<thead>
<tr>
<th></th>
<th>N</th>
<th>P</th>
<th>TPR</th>
<th>TNR</th>
<th>$F_1$</th>
<th>$F_2$</th>
<th>BAC</th>
<th>auROC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear</td>
<td>5</td>
<td>0.2185</td>
<td>0.6355</td>
<td>0.6561</td>
<td>0.5223</td>
<td>0.4314</td>
<td>0.6043</td>
<td>0.6897</td>
</tr>
<tr>
<td>Gaussian</td>
<td>4</td>
<td>0.2070</td>
<td>0.6494</td>
<td>0.6214</td>
<td>0.3339</td>
<td>0.4550</td>
<td>0.6354</td>
<td>0.6520</td>
</tr>
<tr>
<td>Spectrum</td>
<td>1</td>
<td>0.1868</td>
<td>0.6374</td>
<td>0.5644</td>
<td>0.2909</td>
<td>0.4370</td>
<td>0.6109</td>
<td>0.6636</td>
</tr>
<tr>
<td>Mismatch</td>
<td>1</td>
<td>0.1847</td>
<td>0.4833</td>
<td>0.6006</td>
<td>0.2673</td>
<td>0.3387</td>
<td>0.5420</td>
<td>0.5415</td>
</tr>
<tr>
<td>Smith-Waterman</td>
<td>2</td>
<td>0.2266</td>
<td>0.6250</td>
<td>0.6006</td>
<td>0.3226</td>
<td>0.3024</td>
<td>0.4128</td>
<td>0.6476</td>
</tr>
<tr>
<td><strong>Hybrid</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spectrum+Linear</td>
<td>4</td>
<td>0.2266</td>
<td>0.6178</td>
<td>0.6656</td>
<td>0.3258</td>
<td>0.3834</td>
<td>0.6447</td>
<td>0.6779</td>
</tr>
<tr>
<td>Spectrum+Gaussian</td>
<td>2</td>
<td>0.2145</td>
<td>0.5995</td>
<td>0.6778</td>
<td>0.3251</td>
<td>0.4381</td>
<td>0.6388</td>
<td>0.6764</td>
</tr>
<tr>
<td><strong>Boosted</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Linear</td>
<td>1-6</td>
<td>0.3433</td>
<td>0.6074</td>
<td>0.5656</td>
<td>0.3178</td>
<td>0.4384</td>
<td>0.6881</td>
<td>0.7027</td>
</tr>
<tr>
<td>Gaussian</td>
<td>1-9</td>
<td>0.2290</td>
<td>0.5726</td>
<td>0.6511</td>
<td>0.2983</td>
<td>0.3933</td>
<td>0.4118</td>
<td>0.7031</td>
</tr>
<tr>
<td>Spectrum</td>
<td>1-3</td>
<td>0.1834</td>
<td>0.7416</td>
<td>0.6478</td>
<td>0.2829</td>
<td>0.3420</td>
<td>0.3912</td>
<td>0.6801</td>
</tr>
</tbody>
</table>

### Table 3: Fleiss’ $k$ agreement between the linear, Gaussian and Spectrum kernels for predictions on sensitive documents, i.e. True Positive or False Negative predictions.

<table>
<thead>
<tr>
<th></th>
<th>Lin-Gau-Spec</th>
<th>Lin-Gau</th>
<th>Lin-Spec</th>
<th>Gau-Spec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fleiss’ $k$</td>
<td>0.5312</td>
<td>0.7301</td>
<td>0.4502</td>
<td>0.4122</td>
</tr>
</tbody>
</table>

---

5 RESULTS

In this section, we first review the performance of each of the kernels as *stand-alone* classifiers for POS sequence classification without text features, before evaluating the combined *ensemble*
approaches, compared to the text classification baseline.

Table 2 presents the results for the *stand-alone* classifiers. The table
shows the best performing size of $n$-gram for each of the individ-
ual kernels and for two hybrid classifiers, namely *Spectrum+Linear*
and *Spectrum+Gaussian*, according to auROC. Additionally, Table 2
also presents the results of a simple boosting classification approach
where, for a specific kernel, we add the output from an $n$-gram clas-
sification as an additional feature for the $n + 1$-gram classification.

As shown in Table 2, the linear kernel achieves the best auROC score (0.6897). However, the Gaussian and Spectrum kernels per-
form competitively with the linear kernel, achieving 0.6820 and
0.6636 auROC respectively. Moreover, in sensitivity classification
the cost of mis-classifying a sensitive document is far greater than
that of mis-classifying a not-sensitive document and the highest $F_2$
(0.4550) and TPR (0.6574) scores are achieved by the Gaussian
and Spectrum kernels respectively. The Mismatch and Smith-Waterman
kernels perform less well, achieving 0.5415 and 0.6476 auROC re-
spectively. Therefore, in the remaining approaches, we focus on
the Spectrum and Gaussian kernels.

When evaluating the effectiveness of kernels, we are interested in
notable differences in the correctness of predictions for sensitive
documents. As shown in Table 3, there is substantial Fleiss’ $k$
agreement between the linear and Gaussian kernels, but only moderate
agreement between the Spectrum and linear or Gaussian kernels.
This is in line with our expectation that sequence-based kernels,
such as string kernels, can identify different features of sensitivity
than vector space kernels, such as linear or Gaussian. Therefore,
we select the Spectrum kernel as our base kernel for hybrid kernels.

As can be seen from Table 2, the hybrid kernels achieve 0.67 au-
ROC. This is slightly less than the 0.68 auROC achieved by the linear
and Gaussian kernels individually. However, in terms of balanced
accuracy, the hybrid kernels improve overall performance (0.6417)

---

\[\text{http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/}\]
\[\text{http://scikit-learn.org/}\]
\[\text{https://cise.ntu.edu.tw/~cjlin/libsvm/}\]
Table 4: Results for ensemble classification and TC baseline.

<table>
<thead>
<tr>
<th>Text Classification (TC)</th>
<th>F1</th>
<th>F2</th>
<th>BAC</th>
<th>auROC</th>
</tr>
</thead>
<tbody>
<tr>
<td>TC+POSLinear</td>
<td>0.2419</td>
<td>0.6075</td>
<td>0.6601</td>
<td>0.3453</td>
</tr>
<tr>
<td>TC+POSGaussian</td>
<td>0.2430</td>
<td>0.6083</td>
<td>0.6704</td>
<td>0.3530</td>
</tr>
<tr>
<td>TC+Spectrum</td>
<td>0.2431</td>
<td>0.6082</td>
<td>0.6601</td>
<td>0.3578</td>
</tr>
<tr>
<td>TC+Spectrum+Gaussian</td>
<td>0.2478</td>
<td>0.6174</td>
<td>0.6729</td>
<td>0.3591</td>
</tr>
<tr>
<td>TC+Gaussian</td>
<td>0.2211</td>
<td>0.6295</td>
<td>0.6626</td>
<td>0.3273</td>
</tr>
<tr>
<td>TC+Gaussian+Gaussian</td>
<td>0.2445</td>
<td>0.6464</td>
<td>0.6894</td>
<td>0.3540</td>
</tr>
<tr>
<td>Support Vector Machine (SVM)</td>
<td><strong>F1</strong></td>
<td><strong>F2</strong></td>
<td><strong>BAC</strong></td>
<td><strong>auROC</strong></td>
</tr>
<tr>
<td>TC+POSLinear</td>
<td>0.2461</td>
<td>0.6105</td>
<td>0.6675</td>
<td>0.3589</td>
</tr>
<tr>
<td>TC+POSGaussian</td>
<td>0.2358</td>
<td>0.6229</td>
<td>0.6893</td>
<td>0.3436</td>
</tr>
<tr>
<td>TC+Spectrum</td>
<td>0.2403</td>
<td>0.6256</td>
<td>0.6990</td>
<td>0.3463</td>
</tr>
<tr>
<td>TC+Spectrum+Gaussian</td>
<td>0.2433</td>
<td>0.6236</td>
<td>0.7076</td>
<td>0.3488</td>
</tr>
<tr>
<td>TC+Gaussian</td>
<td>0.2447</td>
<td>0.6371</td>
<td>0.7111</td>
<td>0.3519</td>
</tr>
<tr>
<td>TC+Gaussian+Gaussian</td>
<td>0.2462</td>
<td>0.6421</td>
<td>0.7093</td>
<td>0.3515</td>
</tr>
</tbody>
</table>

6 CONCLUSIONS

In this work, we presented a thorough investigation of five SVM kernel functions (Linear, Gaussian, Spectrum, Mismatch and Smith-Waterman) for sensitivity classification using Part-of-Speech n-gram sequences. We showed that an ensemble classification approach that combines text classification with sequence classification can significantly improve sensitivity classification effectiveness. Moreover, we found that combining linear kernel POS sequence classification with text classification by Weighted Majority Vote lead to the largest increase in sensitivity classification effectiveness (+ 6.09% F1), when compared to a text classification baseline.

REFERENCES