Steganalysis with Mismatched Covers: Do Simple Classifiers Help?

Ivans Lubenko
Department of Computer Science
University of Oxford
Parks Road
Oxford OX1 3QD, UK
ivans.lubenko@cs.ox.ac.uk

Andrew D. Ker
Department of Computer Science
University of Oxford
Parks Road
Oxford OX1 3QD, UK
adk@cs.ox.ac.uk

ABSTRACT

Modern steganalysis can be incredibly sensitive and accurate, but only in an artificial setting in which the training data is from the exact same image source as the data being tested. When faced with mismatched training data, performance degrades, often substantially. Some recent publications have advocated the use of very simple classifiers for steganalysis. It is folklore from the machine learning literature that simple classifiers may be more robust to variations between training and testing data, and this paper investigates whether this is true for steganalysis also. Simple classifiers are compared with the classic complex Kernel Support Vector Machine, faced with training data which is mismatched with the testing data. A real-world source of images is used.

Categories and Subject Descriptors

D.2.11 [**Software Engineering**]: Software Architectures—*information hiding*; I.5.2 [**Pattern Recognition**]: Design Methodology—*classifier design and evaluation*

Keywords

Steganalysis, Heterogeneous Data, Machine Learning, Large Data, Perceptron

1. INTRODUCTION

Steganalysis has come a long way as a science, since the early literature of the 1990s. At that time, authors might prove the accuracy of their steganalysis algorithm by testing on a few dozen images, or perhaps a hundred. Now, with the introduction of standard databases (e.g. [2]), all reputable publications use at least a few thousand. A particular advancement has been the introduction of machine learning steganalysis using large feature sets, and some of the rigor from the machine learning field (including cross-validation, stratification, and hyperparameter optimization) has been introduced.

However, there remains a sense in which the testing of steganalysis is, commonly, artificial. It is the use of the same source for training and testing data: perfectly reasonable for laboratory conditions, but not very realistic for the use of steganalysis in the wild. It would be very kind of a steganographer to supply training sets to the steganalyst, and we know now [17] that typical image statistics are highly dependent on the camera and preprocessing operations which created the cover. Thus practical applications of steganalysis may be *mismatched*, i.e. the training data source is not identical to the test data source. Only a few publications have tested steganalysis in such a scenario [1, 4, 8, 10, 12, 18], and in each case the error rate of the detector was greater, with some results showing an apparently significant difference and others a dramatic difference. This suggests that modern steganalysis is not very robust to mismatched training and testing sources.

Some recent steganalysis methods [13, 16] have advocated the use of very simple classifiers for steganalysis. It is folklore from the machine learning literature that simple classifiers may be more robust to difficult training and testing data and this paper investigates how simple classifiers, compared with the classic complex Kernel Support Vector Machine, fare with training data which is mismatched with the testing data. A real-world source of images is used.

The paper is structured as follows. Section 2 reviews the combination of simple classifiers and large data in application to steganalysis. Section 3 explains the design of our experiments to measure robustness to cover source mismatch, and section 4 displays and interpret the results. The paper is concluded in section 5.

2. SIMPLE CLASSIFIERS, LARGE DATA

The Kernel Support Vector Machine (KSVM) is common throughout steganalysis literature. If the steganalysis problem is defined as binary classification, between cover and stego images of fixed payload size, and the testing and training data comes from a single homogeneous source, this is a rational decision. That is because complex classifiers such as KSVM make best use of limited training data and can model difficult (nonlinear) decision boundaries. And, implicitly or explicitly, most authors do indeed set up experiments which use a single homogeneous source. We call this a *fully-matched* data experiment.

MM&Sec'12, September 6–7, 2012, Coventry, United Kingdom.

A more realistic testing scenario would involve *mismatched* (non-homogeneous) data. A classifier's ability to learn from mismatched data can be tested by either cross-training on a collection of homogeneous sources (i.e. training on some sources and testing on others), or by training and testing on a heterogeneous source. Previous work [18] showed that KSVM yielded significantly lower performance in cross-training tests. In similar tests using heterogeneous data [16] it was outperformed by near-linear ensemble classifiers and was only able to match their performance when trained on a very large data set, which is not generally tractable (for details refer to [16]). Similarly disappointing results were observed in [1, 8, 12]

There are two potential explanations for the reduced detection rates on mismatched data: a) a complex classifier overfits the source; b) training a good model for heterogeneous data requires larger training sets. The first explanation comes from the idea that mismatched data requires the classifier to be more general and the reduced detection rates demonstrate that KSVM fails at generalising to the test cover source. The second explanation follows from [16], where KSVM required as much data as simple classifiers to perform as well. Explanations a) and b) are likely to be related and training KSVM on larger mismatched data may yield a more general model. However, KSVM and other complex classifiers are not always possible to train on larger data sets, which makes it difficult to test this relationship.

Simpler classifiers are naturally less prone to overfitting. If a simple classifier is equipped with an online learning function, then it can also learn from unlimited data. There is very little research into using simple classifiers for steganalysis. It has been shown that simple classifiers can achieve equivalent [16] or even better (empirical results from [11, 20, 21]) performance than a complex classifier. The idea is drawn from the "Large Data" approach, which requires a large features set and a large training data set. We employ this idea with the aim to build better steganalysis detectors for mismatched data.

Training from heterogeneous data naturally calls for using as much data as possible, providing that data is diverse. Diversity of steganalysis training data is likely to be proportional to its size. To train a good model from heterogenous data we propose to use the "Large Data" approach, where a classifier is trained on millions of training examples. In order to be able to learn from truly large data (at least a million training examples) we need to employ simple online classification algorithms that provide a) quick learning and b) learning within a constant amount of memory.

Online algorithms are capable of processing one example at a time (and do not require full training set in memory) and therefore allow for unlimited training. The learning is based on an update rule, which updates the decision boundary if an example is misclassified. In some cases one example at a time can contribute to the update like the vector addition in the perceptron algorithm. But there exist more complicated updates that utilise information about previous update steps (such as online gradient descent). Naturally only one pass through data is performed, which was proved [3] to be sufficient for optimal learning from a data set.

2.1 Rich Features

A large training set is only a partial requirement for the Large Data approach. The second requirement is high dimensionality. The advantage of learning in high dimensions is the simplification of decision boundaries that are required to separate the data. In many dimensions there is an increased potential of linear separability, which works to the advantage of the simple models that are necessary for processing large data. There are two approaches to learning in high dimensional space. The first approach is implicit through the use of the kernel trick. The second approach is to explicitly model the problem in the large feature space. For the latter, new high-dimensional feature sets are required.

At the point of writing not many truly large feature sets are available for steganalysis. CCC-300 [13] is one example of a rich feature set in JPEG domain. It was first introduced as part of an ensemble algorithms framework and was shown [16, 13] to produce models with improved accuracy when compared to smaller feature sets on no-shrinkage (ns-) F5. We will use this feature set in our experiments for this paper.

2.2 Average Perceptron

The Average Perceptron is a perceptron learner with a weight vector averaging step which allows for better stability of the algorithm. It was first used for steganalysis in [16], where it demonstrated a varying performance as a standalone classifier, but proved to be useful in the role of base learner in the ensemble. It created an ensemble capable of online learning, and improved both its speed and accuracy.

The Average Perceptron allows for online learning and is also capable for very fast training because of its simple update rule which can be expressed as the follows [7]:

$$w_i = w_{i-1} + x_i t_i$$
$$w_{avg} = w_{avg} + w_i$$

This update of the weight vector and the average weight vector occurs every time a new input example x_i is misclassified ($y(x_i) \neq t_i$) by the current average weight vector, where y is the decision function:

$$y(x) = sign(w_{avg}^T x)$$

In this paper we will use the Average Perceptron in the ensemble setting (details in subsection 3.2.3).

2.3 Ensemble Classifiers

The main principle behind ensemble classifiers is to achieve high classification rates on complex problems via a combination of simple and often linear classifiers. If certain conditions are satisfied, the resulting model is guaranteed [6] to be more accurate than models from the individual simple base learners.

The first of these conditions is every models trained by base learner must have accuracy over 50%. Another condition is for the models to be sufficiently diverse. There are different strategies for ensuring the diversity. It is possible to train base learners on randomly sampled subsets of the training set; each subset used to train one base model. This way the learners train from different data and are likely to produce different decision boundaries. AdaBoost [6] is one example of an advanced ensemble classifier that uses this strategy. Another strategy is to perform slicing of the feature set (picking random feature subsets for each base learner). This way each base learner has its own representation of the same training data.

The base learners are often simple classifiers with linear decision boundaries, however when they are combined in an ensemble decision function, they produce a piecewise-linear decision boundary. Their outputs are often combined using a simple majority vote, but there also exist more advanced strategies involving weighing the base learner's contribution based on its error.

Ensemble classifiers were first used for steganalysis in [13]. This version of ensemble classifier consists of multiple base learners which learn from the same data each using different (sub-)features. The sub-features are random slices from the large feature set. As the result the ensemble allows for training a classifier on *large features* in reduced time.

An ensemble classifier will be online-capable [16] if it uses online algorithms as base learners. Using the Average Perceptron as the base learner allowed us to train the ensemble on large data sets.

3. EXPERIMENTAL DESIGN

In our experiments we will look at homogeneous and heterogeneous data. In the former the data comes from a homogeneous source and allows for fully matched training and testing sets of images. Heterogeneous data can be either mismatched or partially-matched[1]. In our experiments we will focus on tests using fully-matched and mismatched data.

It is an interesting question how one defines a homogeneous source. What physical situation leads to it is dependent on the feature set used. For our purposes we view it as a single cluster that forms in the proposed feature space: it may arise from a single camera or multiple cameras with the same compression quality factor or it could be highly specific to given data. For details of how we simulated matched and mismatched data see Section 3.1.

It is difficult to compare the complex classifiers to online classifiers on both fully-matched and mismatched data: fully-matched experiments are not possible in the online setting because it is difficult to find a single homogenous source with large enough number of images. As such, we can only train online algorithms on heterogeneous steganalysis data and test on heterogeneous data composed of homogeneous subsets.

3.1 Realistic Data Set

Because we are interested in the accuracy of steganalysis in realistic conditions, we use a real-world set of cover images. They were obtained by following links between publicly-visible photos published on a leading social networking site. Restricting the search to users who identified with the Oxford University network, we obtained over 4 million images, each identified with the (anonymised) user who uploaded it. We will treat these different users (also called "actors") as different sources, though we should also be aware that individual users may themselves have multiple cameras or use variable settings. As such the individual actors themselves are a mixture, but are likely to be homogeneous or near-homogeneous. We emphasize that only publicly-available photos were downloaded, and that no personally-identifiable information was retained.

The images were not filtered, except for the removal of very small images. As well as non-natural images uploaded to the social networking site, there is unknown preprocessing probably including resizing and JPEG re-compression, so they are challenging for steganalysis, much more so than the laboratory conditions of a single camera providing single-compressed covers. The only way in which our images are homogeneous is their approximate size (about 1 megapixel) and final JPEG compression factor (quality 85).

Note that these images are not identical to those used in our prior work [16]. Although they come from the same 4 million image set, in this work we have selected 1 711 actors who uploaded at least 500 images each (exactly 500 are selected from each actor). For fully-matched training, we selected 26 actors who uploaded at least 4 000 images each (again, exactly 4 000 are selected from each). We choose these parameters to make it possible to train complex classifiers on data from a few actors, or even just one.

One actor's image set is near-homogeneous but image sets from different actors are heterogenous between themselves. This provides us with a good test bed for cross-training experiments.

3.2 Classifiers Used

To be in-line with experiments in [16] we use the following three classifiers:

3.2.1 Kernel Support Vector Machine (KSVM)

Because of its previously extensive use in steganalysis, KSVM was chosen as the benchmark-setting classifier on this data, as in [13, 15, 16]. The popularity of Kernel Support Vector Machines spawns from the promise of notably high classification accuracy that can be achieved on small data using the kernel trick. KSVM is a large margin classifier and therefore provides theoretical grounds [7] behind finding an optimal model for the available data. This makes it very well suited to typical steganalysis set of problems - two-class steganalysis of fully-matched homogeneous data. It is however often difficult to adapt KSVM to other problems such as multiclass steganalysis [15] or scale it for use with larger data sets or larger feature sets (see [16] and the discussion in the next section).

For all experiments in this paper we adapt a popular open-source implementation of kernel SVM, the libsvm library [5]. The hyperparameters for the kernel SVM were optimised using cross-validated grid search on a held out subset of the training data sets.

3.2.2 Ensemble Fisher Linear Discriminant (EFLD)

Ensemble classifiers were shown to match the accuracy of kernel SVMs in tests on binary steganalysis problems of matched [13] and heterogeneous [16] data. In [16] it achieved the same accuracy as the kernel SVM requiring same number of training samples to peak, 20 000. This was a one-off experiment using KSVMs because it required over ten days to train and over 50GB of RAM. In general KSVM cannot be effectively tested on such large data. The EFLD had smaller computing time and memory requirements and because with the additional training data (up to 400 000 tested) it showed no further improvement in accuracy, we can make an assumption that no more than 20 000 samples will be required to train a good EFLD model in these experiments.

Here we adapt the algorithm from the original ensemble classifier built for steganalysis from [13]. It is composed of k Fisher Linear Discriminant base learners and randomly

[1]When some but not enough for a full training run of the test source's images were made available for training.

slices the feature set into k sets of subfeatures of size L. The final decision is made by the majority vote of the base learners. Both hyperparameters require optimisation - the number of base learners k has the potential of varying the variance of the resulting ensemble and the size of subfeature vectors L works as a tradeoff between accuracy and diversity of base the learners. These hyperparameters can be found using an automated procedure. Kodovsky proposed [14] a forward search based on the out-of-bag error, which we also adapt here.

3.2.3 Online Ensemble Average Perceptron (OEAP)

OEAP follows the structure of the EFLD classifier, with the Average Perceptron (introduced in Section 2.3 above) replacing the FLD as the base learner. Average Perceptron adds the capability of online learning. In [16] this classifier has outperformed, albeit not statistically significantly, the other two algorithms on the problem of detecting nsF5 in heterogeneous data. It will use the same parameters as the EFLD.

3.2.4 Selection of Hyperparameters

Most machine learning classifiers have hyperparameters. A hyperparameter is a user-defined parameter to the classifier's learning function that manifests how it trains its model. A hyperparameter can for example be structural, like the initial number of clusters in clustering algorithms, or penalising, like the cost parameter in the SVM. Using hyperparameters we can often balance between fitting an accurate model and one that overfits. In SVM with Gaussian kernel a very small value of γ would force it to fit the data very closely producing a very complex and irregular decision boundary; whilst large values may yield models that are too general.

We use selective grid search to optimise the misclassification cost C and the kernel width γ for KSVM. The EFLD's hyperparameters L and k are optimised using the forward search described above. We reuse these values for L and k in OEAP.

3.2.5 Default versus Adaptive Thresholds

Training a classifier involves learning a good model through the use of optimised hyperparameters. Sometimes it also involves tuning the model to the data based on its error on the training set. Many classifiers involve a threshold; it is often part of its decision function. There is a default value for the threshold (often 0). By the means of adjusting this threshold it is possible to minimise the classifier's probability of misclassification on a particular data set:

$$P_E = min \frac{P_{FP} + P_{FN}}{2}$$

where P_{FP} is the probability of getting a false positive from the model in question and P_{FN} is the probability of getting a false negative.

There are advantages and disadvantages to using adapted threshold. The main idea behind it is to improve the classifier's accuracy on testing data but it can also lead to worse generalizability if the training set is not representative of the testing data. This condition cannot always be avoided, especially on the real world data which is unlikely to be fully-matched.

In the experiments for this paper the classifiers will be tested using both default and adapted thresholds. In KSVM the decision is formed by taking the *sign* function of the value returned by the model:

$$sign(a - T) = \begin{cases} 0 & \text{if } a < T \\ 1 & \text{if } a \geq T \end{cases}$$

where T is by default set to 0, but can also be positive or negative if adjusted.

The EFLD involves multiple thresholds. Firstly there is a threshold on the individual base learners; the FLD learns it from the training data (often by simply taking the average of the means of cover and stego training examples). Secondly, there is a threshold in the voting procedure, which manifests the proportion of votes required for a particular classification. Following [13] we will only adapt the threshold on EFLD's base learners.

However it is not possible to adapt the threshold on an online model using this simple procedure because both its set of trained examples and its decision function are continuously updated. In our experiment we will use the default threshold of zero for OEAP base learners.

3.3 Hypotheses Tested

The previous work [16] on using the large data techniques for steganalysis demonstrated the value of simple classifiers. As well as quick to train, they are more robust to overfitting then complex classifiers.

In this work we look at their robustness in the difficult cross-training setting. Our tests will be based around the following hypotheses:

Hypothesis 1: simple classifiers and large data are more robust than complex classifiers to training (cover) mismatch.

Hypothesis 2: if 1 holds then are they so robust that we can train on any suitably diverse mismatched training set, rather than aiming for a large enough matched training set?

Secondary questions are to study the absolute performance of the three classifiers, and how this is affected by the diversity of training data in the cross-trained setting.

3.4 Procedure Followed

To test these hypotheses our experiments will be based around two scenarios: using the fully-matched and the mismatched data. The fully-matched scenario requires images in the training and test sets to be drawn from the same source, so one actor will provide samples for the full training and test sets. The mismatched experiments require cross-training and therefore the training sets will be heterogeneous and contain images from multiple actors.

We start with two separate data sets: the fully-matched set $\boldsymbol{B}$ composed of 26 actors with a large set of 4 000 cover images each, and the mismatched set $\boldsymbol{D}$ which contains 1 711 actors with 500 cover images each.

The fully-matched experiments using set $\boldsymbol{B}$ are summarised in Algorithm 1.

ALGORITHM 1 (FULLY-MATCHED DATA).

For $\boldsymbol{B_i} \in \boldsymbol{B}$:
  Pick $T_r, T_e \in \boldsymbol{B_i}$,
    s.t. $|T_r| = 3\,000, |T_e| = 1\,000$ and $T_r \cap T_e = \emptyset$
  Create pairs of cover and stego images from T_r and T_e

Randomise order of T_r
Train on T_r keeping track of estimated threshold t^*
Test on T_e using t^* or default t

In these experiments the accuracy is measured on images T_e from the same actor $\boldsymbol{B_i}$ as the training images T_r. Training on T_r and testing on T_e involves creating cover and stego pairs of images from T_r and T_e respectively, i.e. stratified training. The embedding scheme is the no-shrinkage (ns-) F5, an improved version of the F5 algorithm that uses wet paper codes to remove the artefacts of shrinkage [9][2]. We use 0.05 bits per non-zero DCT coefficient for the payload size.

The mismatched experiments will use images from set $\boldsymbol{D}$ and are described in Algorithm 2. In these experiments the online OEAP will be trained on large training data sets with over a million training samples. KSVM and EFLD on the other hand cannot be trained on training sets much larger than 6 000 and 20 000 samples respectively using the same rich features. However it was shown [16] that using the rich features is crucial for achieving high accuracy on non-homogeneous data. Therefore to ensure that the results are not affected by an unlucky choice of training actors $\boldsymbol{T}$ (6K, 20K $\ll$ 1M), we perform repeated experiments (iterations). There will be ten repeats with one training run and ten test actors each.

Algorithm 2 (Mismatched Data).

For iteration 1..10:
Split $\boldsymbol{D}$ into training actors $\boldsymbol{T}$ and testing actors $\boldsymbol{A}$
less diverse data:
Pick set of actors $\boldsymbol{T'} \subset \boldsymbol{T}$, $|\boldsymbol{T'}| = 6$ / 20 (SVM/EFLD)
$T_r = \bigcup \boldsymbol{T'}$ (all images from actors in $\boldsymbol{T'}$, 3 000/10 000)
more diverse data:
Pick $T_r \subset \bigcup \boldsymbol{T}$, s.t. $|T_r| =$ 3 000 / 10 000 (SVM/EFLD)
all data:
$T_r = \bigcup \boldsymbol{T}$
Pick $\boldsymbol{A'} \subset \boldsymbol{A}$, s.t. $|\boldsymbol{A'}| = 10$
$T_e = \bigcup \boldsymbol{A'}$ (half of images from each actor in $\boldsymbol{A'}$, 2 500)
Create pairs of cover and stego images from T_r and T_e
Randomise order of T_r
Train on T_r, keeping track of estimated threshold t^*
Test on T_e using t^* or default t

This algorithm includes two strategies for sampling from the set of training actors $\boldsymbol{T}$. In the first strategy, we pick $N = 6$ (SVM) or 20 (EFLD) actors at random and take M cover images from each, where $M = 500$. We call this sample *less diverse data*. In the second strategy we simply pick $N \times M$ images at random from all actors in $\boldsymbol{T}$. This is *more diverse data*. The two strategies produce training sets of equal sizes. The *more diverse data* set is similar to the heterogeneous training set used in [16], however unlike the training set in [16] it is mismatched and therefore excludes *any* images from the actors appearing in the testing sets (hence is still an example of cross-training). The *less diverse data* resembles the training previously seen in cross-training experiments in steganalysis e.g. in [12, 1, 4, 18]: it contains a small number of different homogeneous data sets.

[2]Note that we used the 2008 version of the nsF5 simulator; a newer version simulates a higher embedding efficiency.

For mismatched data experiments the accuracy is measured on images from individual actors in the set of test actors $\boldsymbol{A'}$. Each near-homogeneous actor yields 1 000 paired cover and stego samples from which we reserve 500 random paired samples for testing; 50 000 test samples in total.

Each classifier will produce an accuracy score for each of its randomly selected test actors as well as an overall total accuracy score in an experiment.

In the above experiments data normalisation occurs at each training run, with the normalisation parameters calculated from the training data and subsequently used to normalise both the training and the testing data.

4. RESULTS

We test our hypotheses based on two criteria of the classifiers' performance: their average accuracy and their stability. To test whether one classifier has better average accuracy than another, we compare the means of their accuracies on each tested actor. To test whether one classifier is more stable than another, we compare the variances of their accuracies on each tested actor. All comparisons are tested for statistical significance. In the fully-matched tests we use the same 26 actors and in the mismatched tests we take 100 random actors for every test.

The overview of classifiers' performance on both fully-matched (later "matched") and mismatched data is given in Table 3; here the minimum training error (MTE-) EAP is simply the best model of OEAP picked based on the lowest cumulative training error of the base learners on a 10 000 rolling samples (it typically peaked on approximately 400 000 examples). The figures clearly show the advantage that can be gained from using matched data over mismatched data for training and testing a classifier. The discrepancy in variability is especially revealed in the one-actor minima (the worst cases) achieved by both classifiers on matched and mismatched data respectively.

Tables 1 and 4 provide the results of the statistical tests. To compare the means the Student two-sample t-test is used, with Welch's degrees-of-freedom (df)-adjustment for unequal population variance [19]. For variances the F-ratio test is used.

For both EFLD and KSVM, matched data gives significantly better results than mismatched data, both in terms of average accuracy (μ) and reduced variability (σ). This is shown by the low values of the respective p-values from the first column of Table 1.

When sampling mismatched data it is better to use more diverse data than less diverse data as shown in the second column of Table 1. It does not provide advantage in terms of average accuracy ($p > 0.5$ for both KSVM and EFLD), but it does yield significantly reduced variability, which is also reflected in the improved one-actor minimum accuracies.

The mismatched data gives us a way to compare KSVM and EFLD to the online OEAP and MTE-EAP. The difference in their performance is summarised in Table 4. On diverse mismatched data, EFLD has significantly higher average performance than KSVM, but there is no significant difference in variability between them. This confirms our previous findings [16] that KSVM has the tendency to overfit the cover source and potentially requires as much data as EFLD to train a more general and more accurate model. This however cannot be effectively verified in a reasonable timeframe (see Table 2 for details).

Table 1: Comparison of training data

	Test	matched v more diverse	more diverse v less diverse
KSVM	$\mu_1 > \mu_2$	$t = 11.03$ $df = 62.80$ $p \ll 0.001$	$t = 0.54$ $df = 154.21$ $p > 0.5$
	$\sigma_1^2 < \sigma_2^2$	$F = 2.59$ $df = 99, 25$ $p < 0.05$	$F = 3.28$ $df = 99, 99$ $p \ll 0.001$
EFLD	$\mu_1 > \mu_2$	$t = 8.88$ $df = 57.20$ $p \ll 0.001$	$t = -0.29$ $df = 172.86$ $p > 0.5$
	$\sigma_1^2 < \sigma_2^2$	$F = 2.21$ $df = 99, 25$ $p < 0.05$	$F = 2.23$ $df = 99, 99$ $p \ll 0.001$

OEAP did not perform significantly differently to KSVM on average, but its final model has significantly higher variability. The results from these experiments are also shown in Figure 1, where OEAP's continuous performance can be seen (black). In this chart the average accuracy (dot) and the 75th quantile and 25th quantile (top and bottom whiskers) reflect the respective classifier's performance over all 100 mismatched test actors. It can be seen that OEAP's performance gets worse for larger data, which is the reason for the introduction of MTE-EAP (shown in green).

Column three in Table 4 shows that MTE-EAP is significantly more accurate on average than EFLD. It also however has significantly higher variance than EFLD, which is surprising and is largely due to one huge (the one-actor minimum value is 0.546) and two large outliers.

Table 2 shows the computing time budget required to train each classifier on mismatched data. The times are approximate and include disk read and normalisation and are based on the classifiers running concurrently[3] on an Intel Xeon 3.47GHz with 12 cores and 192GB of RAM. This machine was essential for running some of the more memory-intensive experiments, especially the KSVM.

Although it is possible that the classifiers' rank order (MTE-EAP>EFLD>KSVM) as shown in Table 4 simply reflects the size of training data, it is still a fair comparison if the training times are on the same order of magnitude. We have previously shown that the KSVM required over ten days to be trained on 20 000 samples with the same CCC-300 features, which would not be in line with the other two algorithms for a fair comparison.

The reason for MTE-EAP's improved accuracy is unknown based on the above results as well as our assumption that EFLD peaks at 20 000, given the two classifiers' similarity. Further tests may be required to verify if this assumption holds on this particular data. We have no explanation for why OEAP's final model's performance is not in line with its approximately best model - MTE-EAP and further experiments are required to fully understand this issue.

In summary, the above results give evidence in favour of the hypothesis that simple classifiers such as EFLD and MTE-EAP are more robust to mismatched data. It also informs the choice of training data for mismatched use-case: to reduce variability, more diverse training data is better. But classifiers trained on matched data still outperform, very significantly, all classifiers trained on mismatched data.

It was disappointing that the adaptive threshold exhibited a negative impact on KSVM's performance in all cases. We believe that the problem arises during hyperparameter optimisation. The grid search estimates their performance using cross-validation and the default threshold and there is a chance of this performance being misleading. Then the hyperparameters will produce a suboptimal model on the true training data which in our case overfits. But because the model overfits, the default threshold is bound to be the best on the training data for that model. The solution to this problem is to use adapted threshold during the hyperparameter optimisation or use the default value.

Table 2: Approximate classifiers' training times on mismatched data (includes disk read time and normalisation).

Classifier	Training Size	Training Time
KSVM	6 000	8 hrs
EFLD	20 000	18 hrs
OEAP & MTE-EAP	1 000 000	13 hrs

5. CONCLUSIONS

Our aim was to investigate, empirically, whether simple classifiers compare with complex classifiers, when faced with mismatched training data, and whether they are so robust that their performance does not degrade. We have shown that EFLD, which is quite simple, is more robust to mismatched data than the complex KSVM, and that the even

Table 4: Comparison of classifiers

Test	EFLD v KSVM	OEAP v KSVM	MTE-EAP v EFLD
$\mu_1 > \mu_2$	$t = 4.92$ $df = 197.99$ $p \ll 0.001$	$t = 0.50$ $df = 176.33$ $p > 0.5$	$t = 2.27$ $df = 175.56$ $p < 0.05$
$\sigma_1^2 < \sigma_2^2$	$F = 1.015$ $df = 99, 99$ $p > 0.5$	$F = 0.481$ $df = 99, 99$ $p < 0.001^*$	$F = 0.473$ $df = 99, 99$ $p < 0.001^*$

in these cases $^*\sigma_1^2 > \sigma_2^2$.

[3]In some cases the classifiers were competing for disk access.

simpler Ensemble Average Perceptron, if used in a mode selecting the model with minimum training error, is slightly more robust. Such results have been seen for a real-world source of images which contains over 1700 different sources, but more experiments would be needed to generalize it to different embedding methods and payload sizes.

It is possible that the discrepancy arises because the usual format for hyperparameter selection in KSVM favours overfitting the training source too closely. Overfitting the training source (as opposed to overfitting the training data) is at the heart of the problem of mismatched training data. Were the cross-validation modified, it might be possible to avoid this. It is a common theme of machine learning that regularisation must be done with some finesse. We also noted that if mismatched data is to be used, it is better if it is more diverse. Perhaps this is not surprising, but it is a lesson that training on a mixture of a small number of data sets is not the best way to proceed. A large number of smaller sets would be better. This points towards using big data sets for training, which inevitably leads to online learners and simple classifiers if the training is to be tractable.

Although the simple classifiers have more robustness to training mismatch, this is not to the extent of equalling the performance of fully-matched classifiers (KSVM or EFLD). Thus it is still in the interests of the steganalyst to obtain, if it is at all possible, a set of cover data from the same source as the steganographer under suspicion. However, it is difficult to understand how this would be realistic.

6. ACKNOWLEDGMENTS

The authors thank Tomáš Pevný for helpful discussions.

7. REFERENCES

[1] M. Barni, G. Cancelli, and A. Esposito. Forensics aided steganalysis of heterogeneous images. In *Acoustics Speech and Signal Processing*, Proc. IEEE ICASSP '10, pages 1690 –1693, 2010.

[2] P. Bas, T. Filler, and T. Pevný. Break Our Steganographic System: the ins and outs of organizing BOSS. In *Proc. 13th Information Hiding Workshop*, volume 6958 of *Springer LNCS*, pages 59–70, 2011.

[3] L. Bottou and Y. LeCun. Large scale online learning. In *Advances in Neural Information Processing Systems 16*, NIPS, Cambridge, MA, 2003. MIT Press.

[4] G. Cancelli, G. Doërr, I. Cox, and M. Barni. A comparative study of ±1 steganalyzers. In *Proc. IEEE Int. Workshop Multimedia Signal Processing*, pages 791 –794, 2008.

[5] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[6] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Proc. 13th International Conference on Machine Learning*, ICML, pages 325–332, 1996.

[7] Y. Freund and R. E. Schapire. Large margin classification using the perceptron algorithm. *Machine Learning*, 37:277–296, December 1999.

[8] J. Fridrich, J. Kodovský, V. Holub, and M. Goljan. Steganalysis of content-adaptive steganography in spatial domain. In *Proc. 13th Information Hiding Workshop*, volume 6958 of *Springer LNCS*, pages 102–117, 2011.

[9] J. Fridrich, T. Pevný, and J. Kodovský. Statistically undetectable JPEG steganography: dead ends challenges, and opportunities. In *Proc. 9th ACM workshop on Multimedia and Security*, ACM MM&Sec '07, pages 3–14, 2007.

[10] G. Gül and F. Kurugöllü. A new methodology in steganalysis: breaking highly undetectable steganograpy (HUGO). In *Proc. 13th Information Hiding Workshop*, volume 6958 of *Springer LNCS*, pages 71–84, 2011.

[11] A. Y. Halevy, P. Norvig, and F. Pereira. The unreasonable effectiveness of data. *IEEE Intelligent Systems*, 24(2):8–12, 2009.

[12] A. D. Ker and I. Lubenko. Feature reduction and payload location with WAM steganalysis. In *Media Forensics and Security XI*, volume 7254 of *Proc. SPIE*, pages 0A01–0A13, 2009.

[13] J. Kodovský and J. Fridrich. Steganalysis in high dimensions: fusing classifiers built on random subspaces. In *Media Watermarking, Security, and Forensics III*, volume 7880 of *Proc. SPIE*, pages 0L01–0L13, 2011.

[14] J. Kodovský, J. Fridrich, and V. Holub. Ensemble classifiers for steganalysis of digital media. *IEEE Transactions on Information Forensics and Security*, 7(2):432–444, 2012.

[15] I. Lubenko and A. D. Ker. Steganalysis using logistic regression. In *Media Watermarking, Security, and Forensics 2011*, volume 7880 of *Proc. SPIE*, pages 0K01–0K11, 2011.

[16] I. Lubenko and A. D. Ker. Going from small to large data in steganalysis. In *Media Watermarking, Security, and Forensics 2012*, volume 8303 of *Proc. SPIE*, pages 0M01–0M10. SPIE, 2012.

[17] T. Pevný. *Kernel Methods in Steganalysis*. PhD thesis, Binghamton University, SUNY, 2008.

[18] T. Pevný, P. Bas, and J. Fridrich. Steganalysis by subtractive pixel adjacency matrix. In *Proc. 11th ACM workshop on Multimedia and Security*, ACM MM&Sec '09, pages 75–84, 2009.

[19] J. Rice. *Mathematical Statistics And Data Analysis*. Duxbury Advanced Series. Thomson/Brooks/Cole, 2007.

[20] S. Tong and D. Koller. Restricted bayes optimal classifiers. In *Proc. 17th National Conference on Artificial Intelligence and 12th Conference on Innovative Applications of Artificial Intelligence*, AAAI/IAAI, pages 658–664. AAAI Press / The MIT Press, 2000.

[21] M. Zinkevich, M. Weimer, A. J. Smola, and L. Li. Parallelized stochastic gradient descent. In *Proc. 24th Neural Information Processing Systems*, Advances in Neural Information Processing Systems, pages 2595–2603, 2010.

Table 3: Table of results comparing classifiers' accuracy in matched and mismatched data.

Training set	Classifier	Threshold	Training Size	Testing Size	μ	σ	min	max
less diverse mismatched data	KSVM	default	6 000	100 actors × 500 images	0.804	0.071	0.508	0.904
		adaptive	6 000		0.722	0.066	0.500	0.862
	EFLD	default	20 000		0.838	0.058	0.512	0.906
		adaptive	20 000		0.838	0.058	0.512	0.906
more diverse mismatched data	KSVM	default	6 000	100 actors × 500 images	0.809	0.039	0.686	0.874
		adaptive	6 000		0.702	0.051	0.566	0.842
	EFLD	default	20 000		0.836	0.039	0.708	0.902
		adaptive	20 000		0.836	0.039	0.708	0.902
all mismatched data	OEAP	default	1 000 000	100 actors × 500 images	0.812	0.056	0.594	0.889
	MTE-EAP	default	1 000 000		0.851	0.056	0.546	0.935
matched data	KSVM	default	6 000	26 actors × 2000 images	0.876	0.024	0.831	0.935
		adaptive	6 000		0.746	0.011	0.730	0.771
	EFLD	default	6 000		0.892	0.026	0.852	0.952
		adaptive	6 000		0.892	0.026	0.853	0.952

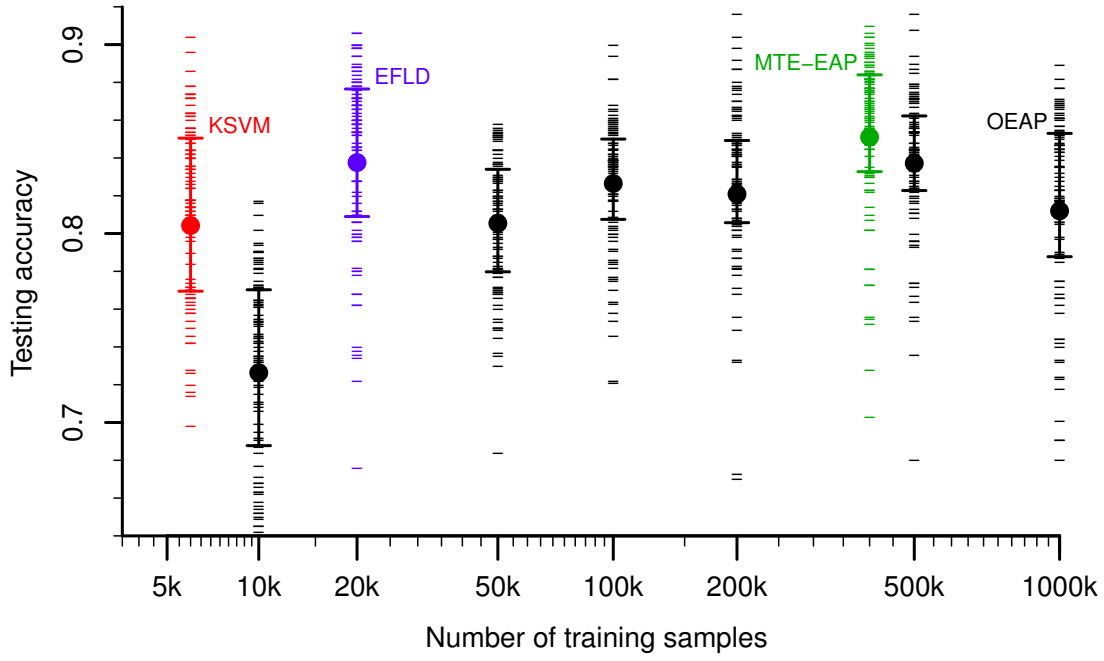

Figure 1: Mismatched data-trained classifiers' accuracy on 100 actors with the average highlighted in dot and 75th quantile (top) and 25th quantile (bottom) highlighted in whiskers, representing the stability/variation. Note the non-linear scale on the x-axis.