Exploring Multitask Learning for Steganalysis

Julie Makelberge and Andrew D. Ker

Oxford University Department of Computer Science, Parks Road, Oxford OX1 3QD, England.

ABSTRACT

This paper introduces a new technique for multi-actor steganalysis. In conventional settings, it is unusual for one actor to generate enough data to be able to train a personalized classifier. On the other hand, in a network there will be many actors, between them generating large amounts of data. Prior work has pooled the training data, and then tries to deal with its heterogeneity. In this work, we use multitask learning to account for differences between actors' image sources, while still sharing domain (globally-applicable) information. We tackle the problem by learning separate feature weights for each actor, and sharing information between the actors through the regularization. This way, the domain information that is obtained by considering all actors at the same time is not disregarded, but the weights are nevertheless personalized. This paper explores whether multitask learning improves accuracy of detection, by benchmarking new multitask learners against previous work.

Keywords: Steganalysis, Multitask Learning, Heterogeneous Data, Regularization

1. INTRODUCTION

The traditional setting for steganalysis usually considers whether one specific person has hidden data in a single object.[1] In practice, however, the use case of steganalysis can be seen as monitoring a large network that consists of mostly innocent actors and one or more guilty ones. Each actor will transmit multiple objects, and guilty actors will sometimes act innocently. The scenario of multiple objects was first proposed in 2006[2] and is called pooled steganalysis, and that of multiple actors was introduced in 2011.[3] The challenge is to make accurate classification of cover and stego objects, when the covers come from heterogeneous sources. One cannot train a classifier with data from a single source: heterogeneity of sources would cause a model mismatch. Most previous approaches deal with this problem by pooling training data from multiple sources.[4] Even then, to avoid mismatch completely, the steganalyst's opponent would have to provide training data. In all most reasonable situations, the detector must train on a source not identical to the suspect, and mismatch is inevitable.

A different way of dealing with this was proposed in Ref. 3. Here, the problem was redefined as a clustering problem in which the algorithm identifies two clusters: one with the innocent actors and one with (at most one) guilty actor. Another approach was proposed in Ref. 5, where this problem is dealt with by using simple classifiers, hoping that their classification rule is simple enough to generalize beyond known actors, whilst keeping the drop in accuracy as low as possible.

To replicate the multi-actor scenario for this paper, images were obtained from a leading social networking site. This work considers images generated by 26 actors who each transmit up to 200 images. The different actors in this scenario are the people uploading the images. By considering real world data, extra difficulties are also introduced: the image set will never be completely clean, there is the problem of double compression, double embedding and more processing artifacts. These difficulties capture the challenges of a real world scenario.

This work takes a new approach, in which the multitask machine learning technique is considered.[6] It explores whether the heterogeneity of sources can actually be an extra source of information rather than something to cast away. It does this by training a separate binary classifier for each actor while using regularization to include the domain information in each classifier to obtain a slightly more generalized model. This work explores whether multitask learning improves accuracy of detection, by benchmarking the new multitask learners[7] against comparable previous work.[8]

Further author information: (Send correspondence to ADK):
J. Makelberge: E-mail: julie.makelberge@cs.ox.ac.uk
A. D. Ker: E-mail: adk@cs.ox.ac.uk, Telephone: +44 1865 283530

1.1 Multi-Actor Steganalysis

There is a need to move steganalysis research toward a setting that resembles the real world more closely. Usually, when considering steganography and steganalysis, the Prisoners' Problem[1] presents formal definitions. In this setting, we have two prisoners, Alice and Bob, who are imprisoned in separate cells. The warden, Eve, has allowed them to communicate as long as they do not plan to escape and she can monitor the communications. The moment Eve notices that they are trying to escape; she will stop all communications between them and put them in solitary confinement. Consequently, Bob and Alice will resort to steganography to try to communicate covertly while Eve will use steganalysis to detect whether any secret messages are exchanged. However, in this setting, we consider only communications between two people. We can imagine that a practical application of steganalysis would be to monitor a large network where multiple people will be communicating with each other. Not all of them will be malicious, and even the actors that actually use steganography will not always send stego objects.

There there are two issues with multi-actor steganalysis that motivate this paper. First of all, even if we assume that training data is supplied for each actor, there needs to be a way to not lose the information we obtain by knowing the source of each image. However, there might not be enough data to generate a fully personalized model for each actor. Multitask learning deals with this by incorporating the domain information in a personalized model. Secondly, if we have training data that corresponds to known actors and we want to generalize to unseen actors. While this paper does not deal with this problem directly, the idea of investigating domain information is useful when considering this issue.

In this paper, we will train a binary classifier for every actor $a \in \{1 \dots A\}$ that has $N^{(a)}$ labelled training instances $(x_n^{(a)}, t_n^{(a)})$, where each $t_n^{(a)} \in \{0, 1\}$ is a label and each $x_n^{(a)}$ is a D-dimensional feature vector, $x_n^a \in R^D$. These binary classifiers will be personalized for each actor whilst still using the domain information to obtain a slightly more generalized model, using a multitask machine learning technique.

1.2 Multitask Learning

In general, when considering machine learning algorithms, complex problems are broken down into simpler problems, which are then learned individually. However, this disregards the fact that these problems tend to naturally be related. In effect, a lot of the *domain information* – information common to all problems – is lost this way. Humans tend to use past experiences and knowledge when learning new things. For example, we can imagine that it would be easier to learn how to play the piano, if we already know how to play the organ. This concept can be extended to machine learning and it is what is called *transfer learning*.

Multitask Learning (henceforth MTL) is a subfield of transfer learing that was introduced by Caruana in Ref. 6. He argues that MTL is a collection of ideas, techniques and algorithms that improve generalization by considering the domain information of related tasks. Since then, multiple algorithms and techniques were proposed. Baxter introduced a theoretical background as to why MTL works well in Ref. 9. He saw MTL as an automated way of learning the best possible inductive bias. He assumed that every problem is part of some environment that constitutes of related problems. Therefore, he extended the Vapnik-Chevonenkis dimensions (VC-dimensions) and statistical learning theory for generalization bounds to the MTL setting.

There are many types of Multitask Learning (MTL), and for more information about the different methods we refer to Ref. 10. In this paper, we consider the domain adaptation scenario. The problem that arises here is that the distribution in one test domain is different from another. A spam filter illustrates this concept best: the filter should consider the generic solution obtained by taking into account the information of all users, but it should be personalized for each individual user's email corpus as well. Ref. 11 describes such a spam filter. The method that is employed here is mainly parameter transfer, which is the most refined method. It assumes that related tasks should share some parameters or priors, and share information through regularization.

2. METHOD

2.1 Logistic Regression

Since this is the first time the technique is applied to the field of steganalysis, the simplest form of multitask learning is used. We base it on logistic regression, which has been applied to steganalysis[8] but is not common.

It is also against simple logistic regression that we will benchmark the multitask algorithm.

Logistic regression is a well researched machine learning technique. It models the probability that an instance was generated from a certain class. It is typically used for classification by picking the most likely class. If we consider the binary classification problem, we assume class C_1 to be the the class of stego objects and C_2 to be the class of cover objects.

Logistic regression attempts to model the probability that instance $\mathbf{x}$ comes from class C_1 or C_2, using an unconstrained weight vector $\mathbf{w}$, by the model:

$$\log \frac{P(C_1 \,|\, \mathbf{x})}{P(C_2 \,|\, \mathbf{x})} = \mathbf{w}^T \mathbf{x}.$$

From this we can deduce the form of $P(C_k \,|\, \mathbf{x})$:

$$P(C_k \,|\, \mathbf{x}) = \frac{1}{1 + exp(\mathbf{w}^T \mathbf{x})} = \sigma(\mathbf{w}^T \mathbf{x}).$$

Given a dataset $(\mathbf{x}_i, t_i)$, $i = 1, \ldots, N$ of N labelled training instances where each $t_i \in \{0, 1\}$ (t_i is zero or one according to the class) and each $\mathbf{x}_i$ is a D-dimensional feature vector, $\mathbf{x}_i \in R^D$, we can define an error function we wish to minimize when learning the weights. It corresponds to the negative logarithm of the likelihood:

$$\mathcal{L}(\mathbf{w}) = -\sum_{i=1}^{N} \log P(C_1 \,|\, \mathbf{x}_i)^{t_i} (1 - P(C_1 \,|\, \mathbf{x}_i))^{1-t_i}.$$

No closed-form solution is known for this minimization problem. Therefore, the minimization is solved through optimization by, for example, using a pseudo-Newtonian algorithm. To avoid overfitting, we also add regularization by incorporating a zero mean Gaussian prior over the weights that is governed by the hyperparameter γ. This is the most common prior for log-linear models. It pushes the weights down to zero and penalizes larger weights quadratically.

$$\mathcal{L}(\mathbf{w}) = -\frac{\sum_{d=1}^{D} w_d^2}{\gamma^2} + \sum_{i=1}^{N} \log P(C_1 \,|\, \mathbf{x}_i)^{t_i} (1 - P(C_1 \,|\, \mathbf{x}_i))^{1-t_i}.$$

The best value for γ can only be found by trial and this process is called parameter optimization. It is performed using either a line search or some other exhaustive technique.

In this paper, we will compare two models to the multitask algorithm. First of all, we obtain a model in the traditional way, i.e. by pooling the data. This mean that we disregard who generated the images and use all the data to train one logistic regression binary classifier. From this point on, we call this the *pooled model*. Second, we will obtain a model by training a binary classifier for each actor individually, assuming that training data is available for every actor. We call this model the *personalized model*.

2.2 Multitask Logistic Regression

For the introduction of multitask learning to the field, we have chosen to use the simplest form of multitask logistic regression. We follow the multitask learner of Ref. 7.

In the setting of multitask learning logistic regression, every actor $a \in 1, \ldots, A$ has $N^{(a)}$ labelled training instances $(\mathbf{x}_n^{(a)}, t_n^{(a)})$ where each $t_n^{(a)} \in \{0, 1\}$ and each $\mathbf{x}_n^{(a)}$ is a D-dimensional feature vector. In this case we will use an italic font to indicate we are talking about an element of the vector and index the elements with $i \in 1, \ldots, D$. When we talk about which training instance we are considering, we will index it with $n \in 1, \ldots, N^{(a)}$.

To extend the simple logistic regression algorithm, we start by considering a parameter matrix in stead of a parameter fector. Let $\mathbf{w}^{(a)}$ indicate the weights associated with actor a and W the complete parameter matrix with each $\mathbf{w}^{(a)}$ as rows:

$$W = \begin{pmatrix} w_1^{(1)} & \cdots & w_D^{(1)} \\ \vdots & \ddots & \vdots \\ w_1^{(A)} & \cdots & w_D^{(A)} \end{pmatrix}.$$

Now the most intuitive way to extend the likelihood function would be to sum over the likelihood of all the actors:

$$\mathcal{L}(W) = -\sum_{a=1}^{A}\sum_{n=1}^{N^{(a)}} \log(P(t_n^{(a)}, | \mathbf{x}_n^{(a)})) = -\sum_{a=1}^{A}\sum_{n=1}^{N^{(a)}} t_n^{(a)} \log \sigma({\mathbf{w}^{(a)}}^T \mathbf{x}_n^{(a)}) + (1 - t_n^{(a)}) \log(1 - \sigma({\mathbf{w}^{(a)}}^T \mathbf{x}_n^{(a)})).$$

However, simply minimizing $\mathcal{L}(W)$ is equivalent to training a logistic regression classifier for each actor separately (the personalized model); there is no transit of information between them. To make this a multitask learner, we introduce a new parameter: the vector $\mathbf{u}$, which is a D-dimentional vector as well. This vector will be calculated as a regularized combination of all the weights and is what will introduce the exchange of information. We impose a Gaussian prior on the $\mathbf{u}$ with mean 0 and variance γ_1^2, and enforce that each i-th row of W follows a Gaussian prior with u_i as its mean and with variance γ_2^2. The posterior likelihood becomes

$$\mathcal{G}(W) = \mathcal{L}(W) + \mathcal{R}(W)$$

where

$$\mathcal{R}(W) = \sum_{j=1}^{D} \left(\frac{u_j^2}{\gamma_1^2} + \frac{1}{\gamma_2^2} \sum_{a=1}^{A} (w_j^{(a)} - u_j)^2 \right).$$

We have used a two-stage prior here. The vector $\mathbf{u}$ represents global weights (which, at the optimum, are simply a function of the local weights) which is the *domain information*. The prior on $\mathbf{u}$ is akin to the prior used to regularize conventional logistic regression. At the second stage, the deviation between the personalized weights and the global weights is then also assumed to come from a Gaussian distribution.

We want the value of $\mathbf{u}$ where the equation is at is minimum. Since $\mathcal{R}(W)$ is the only part of $\mathcal{G}(W)$ that depends on $\mathbf{u}$, we can find an expression for $\mathbf{u}$ depending on W by:

$$u_i = \underset{u}{\operatorname{argmax}} \left(\frac{u^2}{\gamma_1^2} + \frac{\sum_{a=1}^{A} (w_i^{(a)} - u)^2}{\gamma_2^2} \right)$$

which gives us

$$u_i(W) = \frac{\gamma_1^2 \sum_{a=1}^{A} w_i^{(a)}}{\gamma_2^2 + A\gamma_1^2}.$$

This means that the vector $\mathbf{u}$ will actually fall away during minimization since we express it using W. This phenomenon happens often in multitask learning techniques. As with the simple logistic regression model, the optimization has no closed-form solution due to the nonlinearity of the logistic sigmoid function, and the parameters γ_1 and γ_2 are found using a grid search procedure.

We must note that in this setting, learning is equivalent to optimizing weights. For the pooled model, we optimize a single problem with dimension D, for the personalized model this is A optimizations with dimension D, and for the multitask learner this is a single optimization problem with dimension AD. Bear in mind that each update step of a quasi-Newton algorithm like BFGS is at least quadratic in the dimensionality, and number of updates needed will also grow with the dimensionality, so optimizing the multitask model is much slower than personalized or pooled models. For example, the parameter optimization for 26 actors and 81-dimensional features took over 260 days of core-time on a on a 32 core cluster.

3. EXPERIMENTAL DESIGN AND RESULTS

3.1 Hypothesis

There is a distinct need for steganalysis research to move toward a real world setting, since a use case of steganalysis is to monitor social networks that get used by a lot of actors. As we have seen, multitask learning could be a step in the right direction since it deals well with the heterogeneity of sources, while still avoiding model mismatch. We will test the following hypotheses:

Hypothesis 1: There is a statistically significant increase in accuracy of detection when using the multitask logistic regression algorithm rather than pooling the data and applying simple logistic regression method to it. This statistical significant difference is with respect to the percentage of correctly classified images and a 95% confidence level.

Hypothesis 2: There is a statistically significant increase in accuracy of detection when using the multitask logistic regression algorithm rather than training a separate logistic regression model for each actor. This statistical significant difference is with respect to the percentage of correctly classified images and a 95% confidence level.

3.2 Experimental Design

We will assume that we are given training data from the same actors that we test against, for now putting aside the model mismatch problem which could arise with personalized detectors. We wish to compare multitask logistic regression against simple logistic regression steganalysis. We use a real-world dataset of images from a social networking site. The people uploading the pictures are considered to be the actors, as they will be using different cameras. This makes available hundreds of actors each with up to 4000 JPEG cover images. We create stego data by embedding using nsF5 (our implementation simulates the F5 embedding operation with a fixed embedding efficiency of $e = 2$). In this paper, we used two datasets. Each set is divided into a training set (80% of images) and a testing set (20% of images), stratified by actor and class.

For the first dataset, we used 81-dimensional "calibrated Markov features" (the same 81 features which were symmetrized from the original 324 Markov features in Ref 12). There are 26 actors with 200 images (we also tested with 50 and 100 images per actor), using both an 0.1 bpnc and 0.2 bpnc (bits per non-zero coefficient) payload. For the second set, we use state-of-the-art 20-dimensional features that were obtained by partly-supervised dimensionality reduction[13] from 7850-dimensional CF^* features of Ref. 14. There are 26 actors (a different 26 from the first data set), 90 images per actor, and the payload was 0.1 bpnc.

Plain logistic regression, regularized with a Gaussian prior, is used as a base case for comparison of results. As in Ref. 8, it has been turned into a binary classifier by choosing the most likely class. We have implemented a model using pooled training data and a personalized model that learns a classifier for each actor separately. These models are compared to the multitask logistic regression model. In all cases we use BFGS as the numerical minimizer for the weights: this is nontrivial for the multitask learner, which is optimizing over many dimensions (one per actor and feature).

We will not compare the MTL model to algorithms other than logistic regression (state-of-the-art steganalysis uses Support Vector Machines[12] or ensembles of simple linear classifiers[4]) because this will not answer the question as to whether MTL can bring benefits to steganalysis of heterogeneous data. The results would be confused by differences between the classifiers.

The hyperparameters of these models were optimized using a line- or gridsearch with 5-fold cross validation. The idea of a line search is very simple: we try each option in $\{e^x \,|\, x \in \{-20, -19, \ldots, 19, 20\}\}$. This idea can be extended to a grid search for the multitask learner. Instead of a one-dimensional hyperparameter space, we now explore a two-dimensional space. We started with a very coarse grid for the multitask learner and refined the grid in the direction of the optimum, ensuring that the optimum was in the interior of the space searched. Cross validation was performed on the training set and overall accuracy will be reported on the testing set.

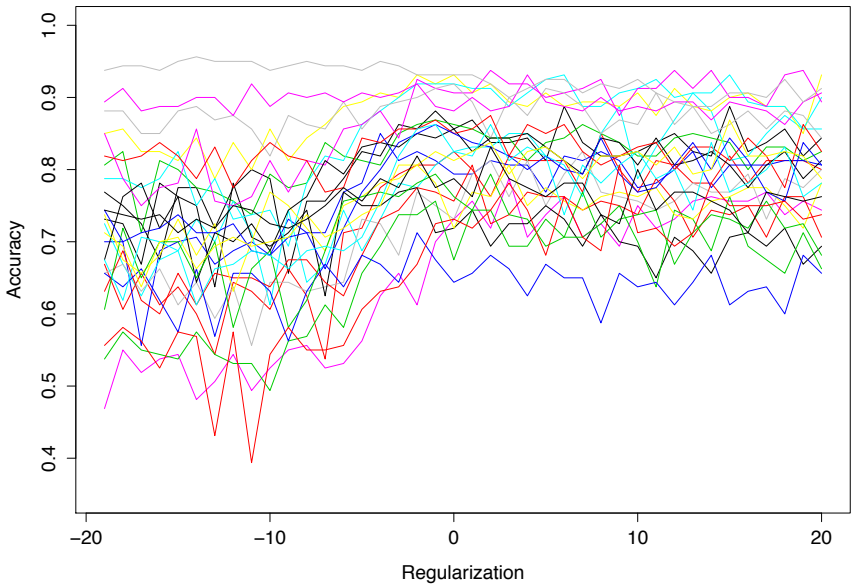

Figure 1. Results of the linesearch for the regularization parameter, training personalized learners on the 26 actors, 100 images per actor dataset. The lines show the accuracy for each actor given the different parameters. The values should be interpreted as e^x. We can see that the results differ greatly per actor.

3.3 Results

We begin by looking at the classification accuracy for personalized logistic regression, as the hyperparameter γ varies. This is displayed in Figure 1, and we can see that different actors do not need the same regularization parameter. Some of the actors prefer very high regularization whilst other prefer almost no regularization whatsoever. This seems to confirm our suspicion that a pooled classifier is forced to learn suboptimally.

Tables 1 and 2 shows the accuracy the models obtained, with optimal regularization, with the simple logistic regression model using pooled data, personalized for each actor, and the multitask logistic regression model. They are for 0.1 bpnc and 0.2 bpnc payloads respectively. To compare the results of the different models, we use a Z-test. This is a statistical test that indicates whether the difference between the means of two distributions (or the accuracy of two classifiers) is statistically significant. We obtain a z-score and if this value does not lie within the interval $[-1.96, 1.96]$, we can reject the null hypothesis that states that the models have the same performance (with a confidence level of 95%).

This reveals a minor flaw in our initial experimental design: the sensitivity of the Z-test is dependant on the amount of testing data. So in order to make the results comparable, we always report results on the 100 images per actor test set, regardless of the size of the training set. The test set is still always disjoint from the training set. For the 81 features dataset, the testing set therefore contains $2 \cdot 20 \cdot 26 = 1040$ images; for the 20 features dataset, the testing set contains 936 images.

Tables 1 and 2 show a statistically significant improvement of the multitask model over the pooled model for the 100 and 200 images per actor sets when considering a payload of 0.1bpnc, and for the 50 and 100 images per actor sets when considering a payload of 0.2bpnc. When we compare the multitask model to the personalized model, we can see statistically significant improvements for the 100 images per actor sets with both payloads. Table 3 shows the accuracy and z-scores for the models using the 20 features dataset. Here, we can see a statistically significant improvement of the multitask model over both the personalized and pooled models. These results show that, depending on the amount of data, one should chose a different model. If there are

81-dimensional Markov features, 26 actors, 0.1 bpnc payload			
	50 images/actor	100 images/actor	200 images/actor
Pooled	81.44	81.15	81.52
Personalized	80.87	82.14	85.09
Multitask	83.37	85.58	85.19
z-score multitask > pooled	1.15	2.37*	2.01*
z-score multitask > personalized	1.48	2.03*	0.06

Table 1. Accuracy (%) of the various classifiers measured on the testing set for 100 images per actor. Scores for a Z-test are also shown. $*$ indicates significance at the 95% level.

81-dimensional Markov features, 26 actors, 0.2 bpnc payload			
	50 images/actor	100 images/actor	200 images/actor
Pooled	84.04	94.71	94.23
Personalized	94.62	94.62	95.58
Multitask	95.58	96.54	95.67
z-score multitask > pooled	8.70*	2.03*	1.50
z-score multitask > personalized	1.02	2.13*	0.10

Table 2. Accuracy (%) of the various classifiers measured on the testing set for 100 images per actor. Scores for a Z-test are also shown. $*$ indicates significance at the 95% level.

20-dimensional reduced CF^* features, 26 actors, 0.1 bpnc payload	
	90 images/actor
Pooled	92.09
Personalized	91.67
Multitask	94.55
z-score multitask > pooled	2.12*
z-score multitask > personalized	2.46*

Table 3. Accuracy (%) of the various classifiers measured on the testing set for 90 images per actor. Scores for a Z-test are also shown. $*$ indicates significance at the 95% level.

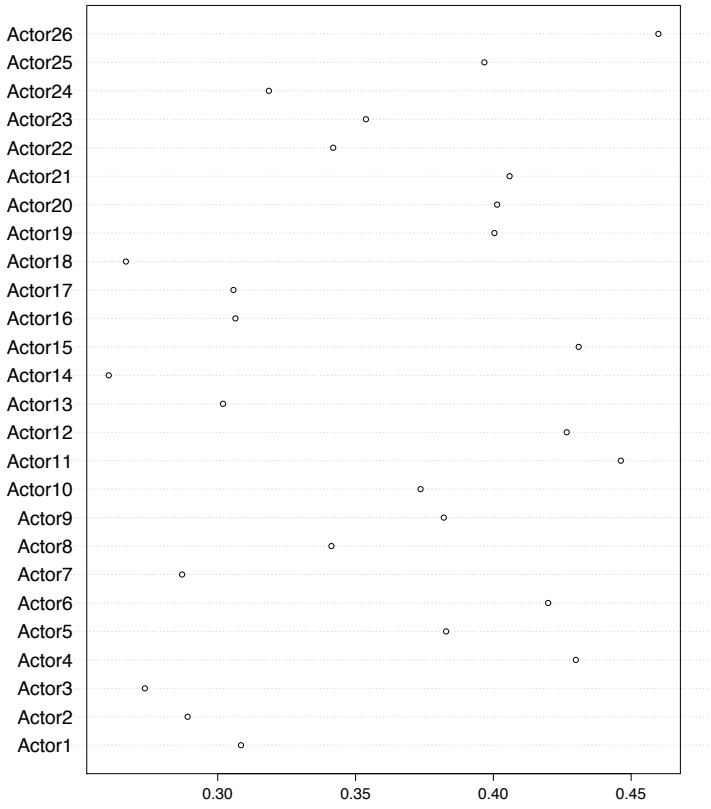

Figure 2. The cosines calculated between the weight vectors of the pooled model and each actor of the multitask model.

enough images per actor in the training set then a personalized learner can be trained. If there are too few images per actor then none of the classifiers learns enough information. But for intermediate sizes, the multitask learner is able to exploit the domain information and give improved performance.

To inspect whether the MTL model is learning significantly different weights for different actors, we computed the cosines* between the weight vectors of the *pooled* logistic regression model and the *individual* weights of the multitask model (for the 26 actors, 100 images per actor dataset). If very closely aligned, we would conclude that the multitask learner is doing little other than pooling the data, but Figure 2 shows that the cosines are in the range 0.25–0.5, showing weak alignment. We also computed the cosines between the *individual* weights of the personalized model and those for the *multitask* model. If very closely aligned, we would conclude that the multitask leaner is treating the actors separately and has not found any domain information. Figure 3 shows that there is some alignment, but for certain actors the learner has found a vector with quite a different direction. Some domain information has been found, and this accounts for the increased classification accuracy.

4. CONCLUSIONS

We have tried to apply an idea from transfer learning to steganalysis of heterogeneous data. Instead of discarding information about an image's origin, we learned a model for each actor (originator) and used regularization to transfer domain information between the models.

*A normalized correlation, obtained via the dot product in the usual way.

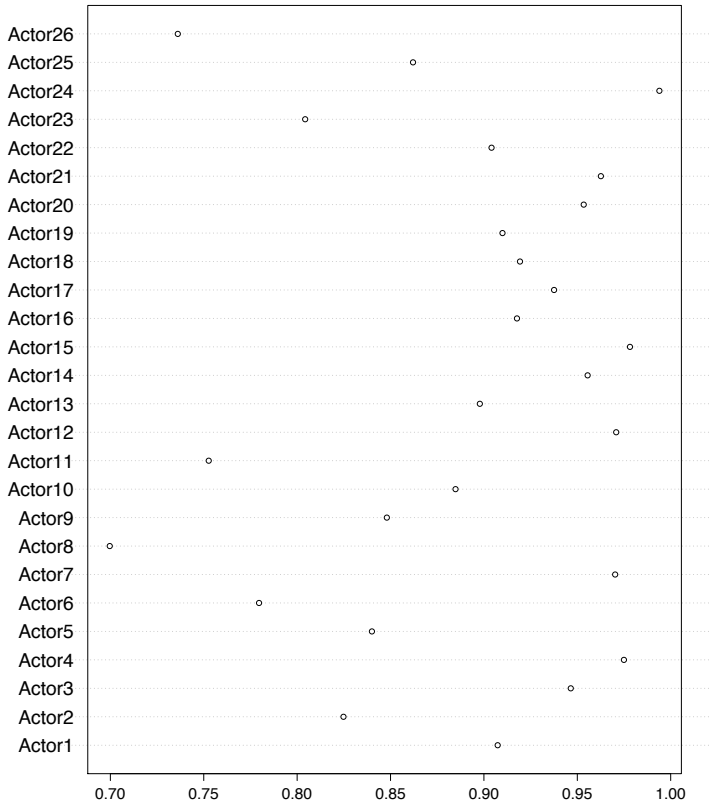

Figure 3. The cosines calculated between the weight vectors of each actor of the multitask and the personalized model.

In this paper we tried a very simple form of MTL, using logistic regression, and tested against standard logistic regression in pooled and personalized varieties. For certain amounts of training data, where this is sufficient information to learn a model but insufficient information to train personalized models, we saw a statistically significant increase in classification accuracy.

We have not tested the MTL method against state of art detectors, because the comparison would reveal more about the ability of SVMs or ensemble classifiers to learn better than logistic regression (which is, after all, only a linear classifier). Our conclusion is that the principles of MTL can *in principle* bring benefit to difficult real-world cases of steganalysis, where there are many cover sources to learn, rather than to advocate a new steganalysis classifer. Further work is needed to see whether MTL methods can be used in contemporary steganalysis methods.

A limitation of the multitask learner is the dimensionality of its optimization problem. With A actors and features of dimensionality D, this particular learner optimizes a non-convex objective function over AD dimensions, which becomes very challenging for moderate values of A. With contemporary features sets having dimensionality in the thousands, this is a major drawback unless the features can be reduced.[13] There is hope that a hierarchical approach could make the optimizations more tractable.

ACKNOWLEDGMENTS

The authors thank Tomáš Pevný for helpful discussions. Some of this work formed part of the first author's MSc dissertation at the Department of Computer Science, Oxford University.

REFERENCES

[1] Simmons, G. J., "The prisoners' problem and the subliminal channel," in [*Advances in Cryptology: Proceedings of CRYPTO'83*], (1984).

[2] Ker, A. D., "Batch steganography and pooled steganalysis," in [*Proceedings of 8th Information Hiding Workshop*], *Lecture Notes in Computer Science* **4437**, 265–281, Springer (2007).

[3] Ker, A. D. and Pevnỳ, T., "A new paradigm for steganalysis via clustering," in [*IS&T/SPIE Electronic Imaging: Security, Steganography, and Watermarking of Multimedia Contents IX*], **7880**, 0U01–0U13 (2011).

[4] Lubenko, I. and Ker, A. D., "Going from small to large data in steganalysis," in [*IS&T/SPIE Electronic Imaging: Security, Steganography, and Watermarking of Multimedia Contents IX*], **8303**, 0M01–0M10 (2011).

[5] Lubenko, I. and Ker, A. D., "Steganalysis with mismatched covers: do simple classifiers help?," in [*Proceedings of 14th Multimedia and Security Workshop*], 11–18, ACM (2012).

[6] Caruana, R., Pratt, L., and Thrun, S., "Multitask learning," *Machine Learning* **28**, 41–75 (1997).

[7] Lapedriza, À., Masip, D., and Vitrià, J., "A hierarchical approach for multi-task logistic regression," in [*Proc. 3rd Iberian Conference on Pattern Recognition and Image Analysis*], *Lecture Notes in Computer Science* **4478**, 258–265, Springer (2007).

[8] Lubenko, I. and Ker, A. D., "Steganalysis using logistic regression," in [*IS&T/SPIE Electronic Imaging: Security, Steganography, and Watermarking of Multimedia Contents IX*], **7880**, 0K01–0K11 (2011).

[9] Baxter, J., "A model of inductive bias learning," *Journal of Artificial Intelligence Research* **12**, 149–198 (2000).

[10] Pan, S. J. and Yang, Q., "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering* **22**(10), 1345–1359 (2010).

[11] Bickel, S. and Scheffer, T., "Dirichlet-enhanced spam filtering based on biased samples," in [*Advances in Neural Information Processing Systems 19*], **19**, 161–168, MIT (2007).

[12] Pevnỳ, T. and Fridrich, J., "Merging Markov and DCT features for multi-class JPEG steganalysis," in [*IS&T/SPIE Electronic Imaging: Security, Steganography, and Watermarking of Multimedia Contents IX*], **6505**, 03–14 (2007).

[13] Pevný, T. and Ker, A. D., "The challenges of rich features in universal steganalysis," (2013). To appear in IS&T/SPIE Electronic Imaging: Media Watermarking, Security, and Forensics 2013.

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