Stackelberg Games for Adversarial Prediction Problems

Michael Brückner
Department of Computer Science
University of Potsdam, Germany
mibrueck@cs.uni-potsdam.de

Tobias Scheffer
Department of Computer Science
University of Potsdam, Germany
scheffer@cs.uni-potsdam.de

ABSTRACT

The standard assumption of identically distributed training and test data is violated when test data are generated in response to a predictive model. This becomes apparent, for example, in the context of email spam filtering, where an email service provider employs a spam filter and the spam sender can take this filter into account when generating new emails. We model the interaction between learner and data generator as a Stackelberg competition in which the learner plays the role of the leader and the data generator may react on the leader's move. We derive an optimization problem to determine the solution of this game and present several instances of the Stackelberg prediction game. We show that the Stackelberg prediction game generalizes existing prediction models. Finally, we explore properties of the discussed models empirically in the context of email spam filtering.

Categories and Subject Descriptors

I.5.1 [**Pattern Recognition**]: Models—*statistical*; H.4.3 [**Information System Applications**]: Communications Applications—*electronic mail*

General Terms

Theory, Algorithms

Keywords

Adversarial Classification, Stackelberg Competition, Prediction Game, Spam Filtering

1. INTRODUCTION

A common assumption on which most learning algorithms are based is that training and test data are governed by identical distributions. However, in a variety of applications, the distribution that governs data at application time may be influenced by an adversary whose interests conflict those of the learner. Consider, for instance, the following three scenarios. In computer and network security, scripts that control attacks are engineered with botnet and intrusion detection systems in mind. Credit card fraudsters adapt their unauthorized use of credit cards—in particular, amounts charged per transactions and per day and the type of businesses that amounts are charged from—such as not to trigger alerting mechanisms employed by credit card companies. Email spam senders design message templates that are instantiated by nodes of botnets; templates are specifically designed to produce a low spam score with current spam filters. The domain of email spam filtering will serve as a running example throughout the paper. In all of these applications, assailants factor information about countermeasures that are being employed into the process of data generation.

The interaction between learner and data generators can be modeled as a game in which one player controls the predictive model whereas another exercises some control over the process of data generation. The adversary's influence on the generation of the data can be mathematically modeled as a transformation that is imposed on the distribution that governs the data at training time. The transformed distribution then governs the data at application time. The optimization criterion of either player takes as arguments both, the predictive model chosen by the learner and the transformation carried out by the adversary.

Typically, this problem is modeled under the worst-case assumption that the adversary desires to impose the highest possible costs on the learner. This amounts to a zero-sum game in which the loss of one player is the gain of the other. In this setting, both players can maximize their expected outcome by following a minimax strategy. El Ghaoui et al. [5] derive a minimax model for input data that are known to lie within some hyper-rectangles around the training instances. Their solution minimizes the worst-case loss over all possible choices of the data in these intervals. Lanckriet et al. [10] study the minimax probability machine. This classifier minimizes the maximal probability of misclassifying new instances for a given mean and covariance matrix of each class. Geometrically, this solution corresponds to a minimax strategy with hyper-ellipsoids around the training instances, rather than hyper-rectangles. Similarly, worst-case solutions to classification games in which the adversary deletes input features or performs arbitrary feature transformation have been studied [3, 6, 7, 14, 4].

Several applications motivate problem settings in which the goals of the learner and the data generator, while still conflicting, are not necessarily entirely antagonistic. For instance, a fraudster's goal of maximizing the profit made from exploiting phished account information is not the inverse of

KDD'11, August 21–24, 2011, San Diego, California, USA.

an email service provider's goal of achieving a high spam recognition rate at close-to-zero false positives. When playing a minimax strategy, one often makes overly pessimistic assumptions about the adversary's behavior and may not necessarily obtain an optimal outcome.

For games that do not exhibit the zero-sum property, a game-theoretic model has been studied that assumes both players to commit to their actions *simultaneously* [1]; that is, without information about the opponent's course of action. When the parameter space of the learner's model and the adversary's transformation and both players' loss functions satisfy specific criteria (*e.g.,* the loss functions have to be monotonic with distinct monotonicity and twice differentiable), then the prediction game has a unique Nash equilibrium that can be found by solving a compact optimization problem [1]. The Nash equilibrium is a combination of parameters for the predictive model and the adversary's transformation which has the property that neither player benefits by unilaterally deviating from it. For the learner, playing the Nash equilibrium instead of the minimax strategy is an optimal course of action under the following sufficient conditions: First, the adversary has to be trusted to behave rationally in the sense of maximizing their profit by playing a Nash strategy, too. If the learner plays the Nash equilibrium but the adversary deviates from that equilibrium, then both players may fare arbitrarily poorly. Secondly, a unique equilibrium needs to exist, since a combination of actions from two distinct equilibria may lead to an arbitrarily poor outcome for either player. Thirdly, the adversary must not have any information about the predictive model that the learner commits to before generating the data. In practice, this assumption can be violated when the adversary is able to probe the predictive model. If the adversary violates either of the above three conditions, no guarantees on the optimality can be given and, consequently, a learner may be ill-advised to play the Nash equilibrium.

In practice, a spam sender may follow heuristics derived from past experience and experiments with the filter. Such a setting in which both players act *non-simultaneously* can be modeled as a Stackelberg competition which allows one player—the follower—to be potentially fully informed about the move of the other player—the leader. We model adversarial learning as a Stackelberg competition in which the *learner acts as leader* by committing to a predictive model in the first step. The model is then disclosed to the follower—the data generator—who then gets to transform the input distribution.

Some authors [9, 12] study the case in which the *data generator acts as leader* and the learner as follower. This reflects a setting in which the adversary discloses how the future distribution will differ from the current distribution before the learner has to commit to a model, which contradicts the intuition of an adversarial model-building problem. When the data generator acts as leader and discloses the data transformation, the learner only has to solve a simple optimization problem in order to minimize the risk on the transformed data points.

The rest of this paper is organized as follows. Section 2 introduces the problem setting. We formalize the Stackelberg prediction game, derive an optimization problem to determine the Stackelberg equilibrium, and show how to employ kernel functions in Section 3. In Section 4, we present three instances of the SPG and discuss their relation to existing prediction models. We report on experiments on email spam filtering in Section 5; Section 6 concludes.

2. PROBLEM SETTING

We study prediction games between two players: The *learner* ($v = -1$) and an adversary, the *data generator* ($v = +1$). In our running example of email spam filtering, we study the competition between recipient and senders, not competition among senders. To this end, $v = -1$ refers to the recipient whereas $v = +1$ models the entirety of all legitimate and abusive email senders as a single, amalgamated player.

In the *past*, the data generator $v = +1$ produced a sample $D = \{(x_i, y_i)\}_{i=1}^n$ of n training instances $x_i \in \mathcal{X}$ with corresponding class labels $y_i \in \mathcal{Y} = \{-1, +1\}$. These object-class pairs are drawn according to a training distribution with density function $p(x, y)$. By contrast, *future* object-class pairs, produced by the data generator at application time, are drawn from some test distribution with density $\dot{p}(x, y)$ which may differ from $p(x, y)$.

The task of the learner $v = -1$ is to select the parameters $\mathbf{w} \in \mathbb{R}^m$ of a predictive model $h(x) = \operatorname{sign} f_{\mathbf{w}}(x)$ implemented in terms of a generalized linear decision function $f_{\mathbf{w}} : \mathcal{X} \to \mathbb{R}$ with $f_{\mathbf{w}}(x) = \mathbf{w}^\mathsf{T}\phi(x)$ and feature mapping $\phi : \mathcal{X} \to \mathbb{R}^m$. The learner's theoretical *costs* at application time are given by

$$\theta_{-1}(\mathbf{w}, \dot{p}) \;=\; \sum_{\mathcal{Y}} \int_{\mathcal{X}} c_{-1}(x, y)\ell_{-1}(f_{\mathbf{w}}(x), y)\dot{p}(x, y)\mathrm{d}x,$$

where weighting function $c_{-1} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ and loss function $\ell_{-1} : \mathbb{R} \times \mathcal{Y} \to \mathbb{R}$ detail the *weighted loss* $c_{-1}(x, y)\ell_{-1}(f_{\mathbf{w}}(x), y)$ that the learner incurs when the predictive model classifies instance x as $h(x) = \operatorname{sign} f_{\mathbf{w}}(x)$ while the true label is y. The positive class- and instance-specific weighting factors $c_{-1}(x, y)$ with $\mathbb{E}[c_{-1}(\mathbf{X}, \mathbf{Y})] = 1$ specify the importance of minimizing the loss $\ell_{-1}(f_{\mathbf{w}}(x), y)$ for the corresponding object-class pair (x, y). For instance, in spam filtering, the correct classification of non-spam messages can be business-critical for email service providers while failing to detect spam messages runs up processing and storage costs, depending on the size of the message.

The data generator $v = +1$ can modify the data generation process for future instances. In practice, spam senders update their campaign templates which are disseminated to the nodes of botnets. Formally, the data generator transforms the training distribution with density p to the test distribution with density $\dot{p}$. The data generator incurs *transformation costs* by modifying the data generation process which is quantified by $\Omega_{+1}(p, \dot{p})$. This term acts as a regularizer on the transformation and may implicitly constrain the shift that can be imposed on the distribution, depending on the nature of the application that is to be modeled. For instance, the email sender may not be allowed to alter the training distribution for non-spam messages, or to modify the nature of the messages by changing the label from *spam* to *non-spam* or vice versa. Additionally, changing the training distribution for spam messages may run up costs depending on the extent of distortion inflicted on the informational payload.

The theoretical *costs* of the data generator at application time are the sum of the expected prediction costs and the transformation costs,

$$\theta_{+1}(\mathbf{w}, \dot{p}) = \sum_{\mathcal{Y}} \int_{\mathcal{X}} c_{+1}(x, y)\ell_{+1}(f_{\mathbf{w}}(x), y)\dot{p}(x, y)\mathrm{d}x + \Omega_{+1}(p, \dot{p}).$$

In analogy to the learner's costs, $c_{+1}(x, y)\ell_{+1}(f_{\mathbf{w}}(x), y)$ quantifies the loss that the data generator incurs when instance x is labeled as $h(x) = \operatorname{sign} f_{\mathbf{w}}(x)$ while the true label is y. The weighting factors $c_{+1}(x, y)$ with $\mathbb{E}[c_{+1}(\mathbf{X}, \mathbf{Y})] = 1$ express the significance of (x, y) from the perspective of the data generator. In our example scenario, this allows to reflect that costs of correctly or incorrectly classified instances may vary greatly across different physical senders that are aggregated into the amalgamated player.

Since the theoretical costs of both players depend on the test distribution, they can, for all practical purposes, not be calculated. Hence, we focus on a regularized, empirical counterpart of the theoretical costs based on the training sample D. The empirical counterpart $\hat{\Omega}_{+1}(D, \dot{D})$ of the data generator's regularizer $\Omega_{+1}(p, \dot{p})$ penalizes the divergence between training sample $D = \{(x_i, y_i)\}_{i=1}^n$ and a perturbated training sample $\dot{D} = \{(\dot{x}_i, y_i)\}_{i=1}^n$ that would be the outcome of applying the transformation that translates p into $\dot{p}$ to sample D. The learner's cost function, instead of integrating over $\dot{p}$, sums over the elements of the perturbated training sample $\dot{D}$. The players' empirical cost functions can still only be evaluated after the learner has committed to parameters $\mathbf{w}$ and the data generator to a transformation from training to test density function, but this transformation need only be represented in terms of the effects that it will have on the training sample D. The transformed training sample $\dot{D}$ must not be mistaken for test data; test data will be generated under $\dot{p}$ at application time after the players have committed to their actions.

The empirical costs incurred by the predictive model h with parameters $\mathbf{w}$ and the shift from p to $\dot{p}$ amount to

$$\hat{\theta}_{-1}(\mathbf{w}, \dot{D}) = \sum_{i=1}^{n} c_{-1,i}\ell_{-1}(f_{\mathbf{w}}(\dot{x}_i), y_i) + \rho_{-1}\hat{\Omega}_{-1}(\mathbf{w}), \quad (1)$$

$$\hat{\theta}_{+1}(\mathbf{w}, \dot{D}) = \sum_{i=1}^{n} c_{+1,i}\ell_{+1}(f_{\mathbf{w}}(\dot{x}_i), y_i) + \rho_{+1}\hat{\Omega}_{+1}(D, \dot{D}), \quad (2)$$

where we have replaced the weighting terms $\frac{1}{n}c_v(\dot{x}_i, y_i)$ by constant cost factors $c_{v,i} > 0$ with $\sum_i c_{v,i} = 1$. The learner's regularizer $\hat{\Omega}_{-1}(\mathbf{w})$ in (1) accounts for the fact that $\dot{D}$ does not constitute the test data itself, but is merely a training sample transformed to reflect the test distribution and then used to learn the model parameters $\mathbf{w}$. The trade-off between the empirical loss and the regularizer is controlled by each player's regularization parameter $\rho_v > 0$ for $v \in \{-1, +1\}$.

In our analysis, we estimate the transformation costs by the average squared ℓ^2-distance between x_i and $\dot{x}_i$ in feature space,

$$\hat{\Omega}_{+1}(\dot{D}, D) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2}\|\phi(\dot{x}_i) - \phi(x_i)\|^2. \quad (3)$$

The learner's regularizer $\hat{\Omega}_{-1}$ penalizes the complexity of the predictive model $h(x) = \operatorname{sign} f_{\mathbf{w}}(x)$. For our analysis, we consider Tikhonov regularization which, for linear decision functions $f_{\mathbf{w}}$, reduces to the squared ℓ^2-norm of $\mathbf{w}$,

$$\hat{\Omega}_{-1}(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2. \quad (4)$$

Note that either player's empirical costs $\hat{\theta}_v(\mathbf{w}, \dot{D})$ depend on both players' actions. The concept of an *optimal* choice of model parameters $\mathbf{w}$ regardless of the adversary's choice of a data transformation is therefore not well-defined. In the following section, we will refer to the Stackelberg model which identifies the concept of an optimal move of the leader which minimizes $\hat{\theta}_{-1}$ over $\mathbf{w}$ under the assumption that the follower will react by minimizing $\hat{\theta}_{+1}$ over $\dot{D}$ given the parameters $\mathbf{w}$ chosen by the leader.

3. STACKELBERG PREDICTION GAME

We model the prediction game as a Stackelberg competition; we refer to the resulting model as the *Stackelberg prediction game* (SPG). A Stackelberg game is one of the simplest dynamic games: In the first stage, the *leader*—in our case, the learner—decides on a predictive model $h(x) = \operatorname{sign} f_{\mathbf{w}}(x)$ with parameters $\mathbf{w}$. In the second stage, the data generator, who plays the part of the *follower*, observes the leader's decision and chooses a transformation that changes the distribution of *past* instances into the distribution of *future* instances. In this scenario, the learner has to commit to a set of parameters unilaterally whereas the data generator can take the model parameters $\mathbf{w}$ into account when preparing the data transformation.

The optimality of a *Stackelberg equilibrium* which we will now introduce rests on the assumption that the follower—the data generator—will act rationally in the sense of choosing a transformation that minimizes the resulting costs $\hat{\theta}_{+1}$ given the disclosed $\mathbf{w}$. To reach minimal costs given $\mathbf{w}$, the data generator has to identify a sample $\dot{D}$ that constitutes a global minimum of the cost function $\hat{\theta}_{+1}(\mathbf{w}, \dot{D})$. There may be several global minima with identical values of the cost function; in general, the data generator has to identify any element $\dot{D}$ from the set of optimal responses to $\mathbf{w}$,

$$\dot{\mathcal{D}}_{\mathbf{w}} = \left\{ \{(\dot{x}_i, y_i)\}_{i=1}^n : \{\dot{x}_i\}_{i=1}^n \in \operatorname*{argmin}_{\dot{x}'_1, \ldots, \dot{x}'_n \in \mathcal{X}} \hat{\theta}_{+1}\left(\mathbf{w}, \{(\dot{x}'_i, y_i)\}_{i=1}^n\right) \right\}.$$

Identifying an element $\dot{D} \in \dot{\mathcal{D}}_{\mathbf{w}}$ amounts to solving a regular optimization problem because $\mathbf{w}$ can be observed before $\dot{D}$ has to be chosen. A Stackelberg equilibrium is now identified by backward induction. Assuming that the data generator will decide for any $\dot{D} \in \dot{\mathcal{D}}_{\mathbf{w}}$, the learner has to choose model parameters $\mathbf{w}^*$ that minimize the learner's cost function $\hat{\theta}_{-1}$ for any of the possible reactions $\dot{D} \in \dot{\mathcal{D}}_{\mathbf{w}}$ that are optimal for the data generator:

$$\mathbf{w}^* \in \operatorname*{argmin}_{\mathbf{w} \in \mathbb{R}^m} \max_{\dot{D} \in \dot{\mathcal{D}}_{\mathbf{w}}} \hat{\theta}_{-1}(\mathbf{w}, \dot{D}). \quad (5)$$

An action $\mathbf{w}^*$ that minimizes the learner's costs and a corresponding optimal action $\dot{D} \in \dot{\mathcal{D}}_{\mathbf{w}^*}$ of the data generator are called a *Stackelberg equilibrium*. The Stackelberg equilibrium is a special case of a subgame perfect equilibrium which is an extension of the Nash equilibrium for games that are played non-simultaneously.

3.1 Finding a Stackelberg Equilibrium

Equation 5 establishes a hierarchical mathematical program—specifically, a *bilevel optimization problem*—with upper-level objective $\hat{\theta}_{-1}$ and lower-level objective $\hat{\theta}_{+1}$.

$$\min_{\mathbf{w}\in\mathbb{R}^m} \max_{\forall i\,:\,\dot{x}_i\in\mathcal{X}} \hat{\theta}_{-1}(\mathbf{w}, \{(\dot{x}_i, y_i)\}_{i=1}^n) \tag{6}$$

$$\text{s.t.} \qquad \{\dot{x}_i\}_{i=1}^n \in \operatorname*{argmin}_{\dot{x}'_1,\ldots,\dot{x}'_n\in\mathcal{X}} \hat{\theta}_{+1}(\mathbf{w}, \{(\dot{x}'_i, y_i)\}_{i=1}^n) \tag{7}$$

Bilevel programs are intrinsically hard to solve. Even the simplest instance in which all constraints and objectives are linear is known to be NP-hard [8]. The main difficulties arise from the constraints $\dot{x}'_i \in \mathcal{X}$ of the lower-level optimization problem which generally render constraint (7) of the upper-level optimization problem to be non-differentiable in $\mathbf{w}$, even if $\hat{\theta}_{+1}$ is continuously differentiable in $\mathbf{w}$ and $\dot{x}'_i$ for $i = 1, \ldots, n$.

Numerous approaches that address bilevel programs have been studied, for instance, based on gradient descent, penalty function, and trust-region methods; see, for instance, [2] for a detailed survey. Commonly, these methods reformulate the optimization problem into a mathematical program with equilibrium constraints. In this, the lower-level optimization problem is replaced by its Karush-Kuhn-Tucker (KKT) conditions. The resulting optimization problem with equilibrium constraints can be solved approximately by relaxing the complementary conditions [15]. However these methods do not necessarily converge to a (local) optimum and are applicable to small problems only.

That is why we focus on a special case of the above bilevel program. The following theorem reformulates the lower-level optimization problem into an unconstrained problem such that constraint (7) becomes continuously differentiable in $\mathbf{w}$. This requires the feature space induced by mapping ϕ, but not necessarily the input space $\mathcal{X}$, to be unrestricted and the data generator's loss function $\ell_{+1}(z, y)$ to be convex and continuously differentiable in $z \in \mathbb{R}$.

THEOREM 1. *Let the leader's cost function $\hat{\theta}_{-1}$ and the follower's cost function $\hat{\theta}_{+1}$ be defined as in* (1) *and* (2) *with regularizers $\hat{\Omega}_{-1}$ and $\hat{\Omega}_{+1}$ defined as in* (4) *and* (3)*, respectively. Let feature mapping $\phi : \mathcal{X} \to \mathbb{R}^m$ be surjective, let the data generator's loss function $\ell_{+1}(z, y)$ be convex and continuously differentiable with respect to $z \in \mathbb{R}$ for any fixed $y \in \mathcal{Y}$. Now let weight vector $\mathbf{w}^* \in \mathbb{R}^m$ and factors $\tau_1^*, \ldots, \tau_n^* \in \mathbb{R}$ be a solution of the optimization problem*

$$\min_{\mathbf{w},\forall i\,:\,\tau_i} \quad \sum_{i=1}^n c_{-1,i}\ell_{-1}\left(f_{\mathbf{w}}(x_i) + \tau_i\|\mathbf{w}\|^2, y_i\right) + \frac{\rho_{-1}}{2}\|\mathbf{w}\|^2 \tag{8}$$

$$\text{s.t.} \qquad \forall i : 0 = \tau_i + \frac{n}{\rho_{+1}} c_{+1,i}\ell'_{+1}\left(f_{\mathbf{w}}(x_i) + \tau_i\|\mathbf{w}\|^2, y_i\right).$$

Then the Stackelberg prediction game in Equation 6 attains an equilibrium at $(\mathbf{w}^*, \dot{D}^*)$ with $\dot{D}^* = \{(\dot{x}_i^*, y_i)\}_{i=1}^n$ and $\dot{x}_i^* \in \{\dot{x} \in \mathcal{X} : \phi(\dot{x}) = \phi(x_i) + \tau_i^*\mathbf{w}^*\}$.

Proof. Constraint 7 says that $\{\dot{x}_i^*\}_{i=1}^n$ has to be a solution of the restricted optimization problem

$$\min_{\forall i\,:\,\dot{x}_i\in\mathcal{X}} \sum_{i=1}^n c_{+1,i}\ell_{+1}(\mathbf{w}^\mathsf{T}\phi(\dot{x}_i), y_i) + \frac{\rho_{+1}}{n}\sum_{i=1}^n \frac{1}{2}\|\phi(\dot{x}_i) - \phi(x_i)\|^2.$$

As the objective as well as the constraints are entirely defined in terms of $\dot{\mathbf{x}}_i^* = \phi(\dot{x}_i^*)$, this condition is equivalent to enforcing $\{\dot{\mathbf{x}}_i^*\}_{i=1}^n$ to be a solution of the unrestricted optimization problem

$$\min_{\forall i\,:\,\dot{\mathbf{x}}_i\in\mathbb{R}^m} \sum_{i=1}^n c_{+1,i}\ell_{+1}(\mathbf{w}^\mathsf{T}\dot{\mathbf{x}}_i, y_i) + \frac{\rho_{+1}}{n}\sum_{i=1}^n \frac{1}{2}\|\dot{\mathbf{x}}_i - \phi(x_i)\|^2. \tag{9}$$

This solution is uniquely defined for any fixed $\mathbf{w}$ as loss function $\ell_{+1}(z, y)$ is required to be convex in z, and consequently in $\dot{\mathbf{x}}_i$, and the term $\|\dot{\mathbf{x}}_i - \phi(x_i)\|^2$ is quadratic in $\dot{\mathbf{x}}_i$ and therefore strictly convex for any fixed $\phi(x_i)$. Given $\mathbf{w} \in \mathbb{R}^m$ and minimizer $\dot{\mathbf{x}}_i^* \in \mathbb{R}^m$, the set $\dot{\mathcal{X}}_{\mathbf{w}}^i = \{\dot{x}^* \in \mathcal{X} : \phi(\dot{x}^*) = \dot{\mathbf{x}}_i^*\}$ contains all instances $\dot{x}^*$ which correspond to the optimally transformed instance in feature space $\dot{\mathbf{x}}_i^*$. Since ϕ is surjective, $\dot{\mathcal{X}}_{\mathbf{w}}^i$ is guaranteed to be non-empty, and consequently, for any solution $\{\dot{\mathbf{x}}_i^*\}_{i=1}^n$, there exist at least one corresponding set of instances $\{\dot{x}_i^*\}_{i=1}^n$. As ϕ is not required to be a bijective mapping, there may exist multiple instances $\dot{x} \in \dot{\mathcal{X}}_{\mathbf{w}}^i$ which are optimal in the sense of minimizing the data generator's loss. However, since all of these instances share the same feature representation $\dot{\mathbf{x}}_i^*$, the inner maximization of the upper-level optimization problem in (6) vanishes,

$$\min_{\mathbf{w}\in\mathbb{R}^m} \max_{\forall i\,:\,\dot{x}_i\in\dot{\mathcal{X}}_{\mathbf{w}}^i} \hat{\theta}_{-1}(\mathbf{w}, \{(\dot{x}_i, y_i)\}_{i=1}^n) = \min_{\mathbf{w}\in\mathbb{R}^m} \sum_{i=1}^n c_{-1,i}\ell_{-1}\left(\mathbf{w}^\mathsf{T}\dot{\mathbf{x}}_i^*, y_i\right) + \frac{\rho_{-1}}{2}\|\mathbf{w}\|^2, \tag{10}$$

where $\{\mathbf{x}_i^*\}_{i=1}^n$ is the solution of Optimization Problem 9. Since 9 is convex, this constraint can be replaced by its complementary conditions which are given by $\nabla_{\dot{\mathbf{x}}_i}\hat{\theta}_{+1}(\mathbf{w}, \dot{D}) = 0$ for $i = 1, \ldots, n$ where

$$\nabla_{\dot{\mathbf{x}}_i}\hat{\theta}_{+1}(\mathbf{w}, \dot{D}) = c_{+1,i}\ell'_{+1}(\mathbf{w}^\mathsf{T}\dot{\mathbf{x}}_i^*, y_i)\mathbf{w} + \frac{\rho_{+1}}{n}(\dot{\mathbf{x}}_i - \phi(x_i)).$$

The mapped instance $\dot{\mathbf{x}}_i^*$ that satisfies the i-th complementary condition is given by

$$\dot{\mathbf{x}}_i^* = \phi(x_i) + \tau_i\mathbf{w} \tag{11}$$

with

$$\begin{aligned} \tau_i &= -\frac{n}{\rho_{+1}} c_{+1,i}\ell'_{+1}\left(\mathbf{w}^\mathsf{T}\dot{\mathbf{x}}_i^*, y_i\right), \\ &= -\frac{n}{\rho_{+1}} c_{+1,i}\ell'_{+1}\left(\mathbf{w}^\mathsf{T}\phi(x_i) + \tau_i\mathbf{w}^\mathsf{T}\mathbf{w}, y_i\right), \\ &= -\frac{n}{\rho_{+1}} c_{+1,i}\ell'_{+1}\left(f_{\mathbf{w}}(x_i) + \tau_i\|\mathbf{w}\|^2, y_i\right). \end{aligned} \tag{12}$$

When replacing $\dot{\mathbf{x}}_i^*$ by (11) in the upper-level Optimization Problem 10 and enforcing Equation 12, Optimization Problem 8 follows. Hence, a solution $\mathbf{w}^*$ of (8) with corresponding $\tau_1^*, \ldots, \tau_n^*$ is also a solution of (6) with $\dot{x}_i^* \in \dot{\mathcal{X}}_{\mathbf{w}^*}^i = \{\dot{x} \in \mathcal{X} : \phi(\dot{x}) = \phi(x_i) + \tau_i^*\mathbf{w}^*\}$. □

The objective as well as the constraints of the optimization problem in Theorem 1 are generally not jointly convex in $\mathbf{w}$ and $\tau_1, \ldots, \tau_n$. However, under the assumptions of the following proposition, a locally optimal solution can still be found efficiently by standard SQP solvers.

PROPOSITION 1. *Let loss function $\ell_{-1}(z, y)$ be twice continuously differentiable and loss function $\ell_{+1}(z, y)$ be convex and thrice continuously differentiable with respect to $z \in \mathbb{R}$ for any fixed $y \in \mathcal{Y}$. Then, a point satisfying the KKT conditions of the optimization problem in Equation* (8) *can be obtained by* sequential quadratic programming *(SQP) methods.*

The objective as well as the constraints in (8) are twice continuously differentiable with respect to $\mathbf{w}$ and τ_i for $i = 1, \ldots, n$. Hence, the corresponding complementary conditions are continuously differentiable which is a sufficient condition to apply SQP methods; this proves Proposition 1.

3.2 Applying Kernels

Theorem 1 states that a Stackelberg equilibrium with parameter vector $\mathbf{w} \in \mathbb{R}^m$ can be obtained by solving the optimization problem in (8) which requires an explicit feature representation $\phi(x_i)$ of the training instances. However, in some applications, such a feature mapping is unwieldy or even not existing. Instead, one is often equipped with a kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ which measures the similarity between two instances. Generally, kernel function k is assumed to be a positive-semidefinite kernel such that it can be stated in terms of a scalar product in the corresponding reproducing kernel Hilbert space; *i.e.,* $\exists \phi$ with $k(x, x') = \phi(x)^\mathsf{T} \phi(x')$. Making use of the representer theorem [13], we can now express weight vector $\mathbf{w}$ as a linear combination of the mapped training instances; that is,

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i \phi(x_i) \tag{13}$$

where feature mapping ϕ is implicitly defined by kernel k. When substituting $\mathbf{w}$ in (8) by (13), the squared norm of $\mathbf{w}$ and decision function $f_\mathbf{w}$ can be completely expressed in terms of the kernel,

$$\|\mathbf{w}\|^2 = \sum_{j,k=1}^{n} \alpha_j \alpha_k k(x_j, x_k), \tag{14}$$

$$f_\mathbf{w}(x_i) = \sum_{j=1}^{n} \alpha_j k(x_i, x_j). \tag{15}$$

Hence, the optimization problem in (8) can be reformulated into an optimization problem over $\tau_1, \ldots, \tau_n \in \mathbb{R}$ and the dual weights $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$ without the need of an explicit feature mapping ϕ. However, inferring an optimal transformed sample $\dot{D}^*$ still requires the knowledge of an explicit mapping ϕ and its inverse ϕ^{-1}. Of course, this is not a restriction as we are interested in the predictive model $f_\mathbf{w}$ rather than the transformed sample $\dot{D}^*$.

Note that for computational reasons, it may be advisable to first construct an explicit feature mapping from the kernel matrix and then to train the Stackelberg model in the primal. For instance, we can employ the kernel PCA map[1]

$$\phi : x \mapsto \mathbf{K}^{-\frac{1}{2}} \left[k(x, x_1), \ldots, k(x, x_n)\right]^\mathsf{T}, \tag{16}$$

where $\mathbf{K}$ denotes the kernel matrix with $\mathbf{K}_{ij} = k(x_i, x_j)$. Within our experiments (presented in Chapter 5) where we use linear kernels, we study all three variants: Computing the model in input space, computing the kernelized version, and computing the PCA map-induced variant. Even though all variants yield the same solution, using an explicit PCA mapping is generally fastest for reasonable n.

[1]Matrix $\mathbf{K}^{-\frac{1}{2}}$ can be computed directly from the eigenvalue decomposition of the kernel matrix $\mathbf{K}$; in case it is singular we use the pseudo-inverse of $\mathbf{K}^{\frac{1}{2}}$.

4. INSTANCES OF THE SPG

By the choice of ℓ_v, distinct instances of the Stackelberg prediction game (SPG) can be identified which, to some extent, generalize existing prediction models such as the *SVM for invariances* [14] and the *SVM with uneven margins* [11].

4.1 SPG with Worst-Case Loss

The *SPG with worst-case loss* is an instance of the Stackelberg prediction game that is characterized by an antagonicity of the (weighted) empirical costs of learner and data generator; that is, the data generator employs the loss function

$$\ell_{+1}^{\text{wc}}(z, y) = -\ell_{-1}(z, y)$$

and cost factors $c_{+1,i} = c_{-1,i}$. Loss functions ℓ_{+1}^{wc} and ℓ_{-1} cannot both be convex at the same time—except for an inappropriate linear function—and so the requirements of either Theorem 1 or Proposition 1 are violated. As we cannot apply Theorem 1, we consider the original optimization problem (Equations 6-7). We substitute ℓ_{+1}^{wc} and $c_{+1,i}$ in the objective (Equation 2) of the lower-level optimization problem

$$\min_{\forall i \,:\, \dot{x}_i \in \mathcal{X}} \sum_{i=1}^{n} c_{+1,i} \ell_{+1}^{\text{wc}}\left(f_\mathbf{w}(\dot{x}_i), y_i\right) + \frac{\rho_{+1}}{n} \sum_{i=1}^{n} \frac{1}{2} \|\phi(\dot{x}_i) - \phi(x_i)\|^2$$

which decouples into n maximization problems

$$\max_{\dot{x}_i \in \mathcal{X}} c_{-1,i} \ell_{-1}(f_\mathbf{w}(\dot{x}_i), y_i) - \frac{\rho_{+1}}{n} \frac{1}{2} \|\phi(\dot{x}_i) - \phi(x_i)\|^2. \tag{17}$$

An equivalent formulation of (17) is given by

$$\max_{\dot{x}_i \in \mathcal{X}_i'} \ell_{-1}(f_\mathbf{w}(\dot{x}_i), y_i) \tag{18}$$

where $\mathcal{X}_i' = \{\dot{x} \in \mathcal{X} : c_{-1,i} = \frac{\rho_{+1}'}{n} \frac{1}{2} \|\phi(\dot{x}) - \phi(x_i)\|^2\}$ are feasible sets of transformed instances. The difference between both formulations is that in (18), regularization parameter ρ_{+1}' explicitly restricts the amount of transformation of each instance x_i. As now the inner maximization of the upper-level optimization problem in (6) can be stated in terms of the solution of the lower-level optimization problem, $\ell_{-1}(f_\mathbf{w}(\dot{x}_i^*), y_i)$, the entire bilevel optimization problem reduces to the following constrained minimization problem.

$$\min_{\mathbf{w}, \forall i \,:\, \xi_i} \sum_{i=1}^{n} c_{-1,i} \xi_i + \rho_{-1} \frac{1}{2} \|\mathbf{w}\|^2 \tag{19}$$

$$\text{s.t.} \quad \forall i : \xi_i \geq 0,\ \xi_i \geq \max_{\dot{x}_i \in \mathcal{X}_i'} \ell_{-1}(f_\mathbf{w}(\dot{x}_i), y_i) \tag{20}$$

If the lower-level maximization problem (20) has a unique solution for any fixed $\mathbf{w} \in \mathbb{R}^m$, then the above optimization problem can be solved by gradient descent where in each iteration the maximization problem in (20) has to be solved for the current iterate $\mathbf{w}^k$ (see, *e.g.,* [14]). In case the learner choses the hinge loss,

$$\ell_{-1}^{\text{h}}(z, y) = \max(0, 1 - yz), \tag{21}$$

the SPG with worst-case loss reduces to an instance of the *SVM for invariances* [14].

4.2 SPG with Linear Loss

A second instance of the Stackelberg prediction game is the *SPG with linear loss* in which the data generator employs a linear loss function,

$$\ell_{+1}^{\text{lin}}(z, y) = z,$$

which penalizes high decision values z independently of the class. This choice is appropriate, for instance, in email spam filtering where the data generator is purely interested in the delivery of an email x which becomes unlikely for large values of z, independently of the corresponding true class y.

For the linear loss that is continuously differentiable and convex, the constraints in (8) reduce to

$$\tau_i = -\frac{n}{\rho_{+1}} c_{+1,i} \tag{22}$$

for $i = 1, \ldots, n$. When choosing the hinge loss (21) for the learner and replacing τ_i in (8) by (22) we arrive at the following minimization problem.

$$\min_{\mathbf{w}, \forall i\,:\,\xi_i} \quad \sum_{i=1}^{n} c_{-1,i}\xi_i + \rho_{-1}\frac{1}{2}\|\mathbf{w}\|^2$$
$$\text{s.t.} \quad \forall i : \xi_i \geq 0,\ \xi_i \geq 1 - y_i \left(\mathbf{w}^\mathsf{T}\phi(x_i) - \frac{n}{\rho_{+1}} c_{+1,i}\|\mathbf{w}\|^2 \right)$$

The latter constraints can be reformulated to

$$y_i \mathbf{w}^\mathsf{T}\phi(x_i) \geq 1 + y_i\kappa_i - \xi_i$$

which amounts to the constraints of the *SVM with uneven margins* [11]. The only syntactic distinction is that $\kappa_i = \frac{n}{\rho_{+1}} c_{+1,i}\|\mathbf{w}\|^2$ is indirectly defined by ρ_{+1} and $c_{+1,i}$; however, for each choice of $\kappa_i \geq 0$ in the SVM with uneven margins, there exist appropriate parameters ρ_{+1} and $c_{+1,i}$ of an equivalent SPG with linear loss and vice versa.

Consider the special case of equal factors $c_{+1,i} = c_{+1,j}$, and consequently $\kappa = \kappa_i = \kappa_j$, for all $i, j = 1, \ldots, n$. Then the margin of negative instances becomes $1 - \kappa$ whereas the margin of positive instances is $1 + \kappa$. In our example of spam filtering, this goes with the intuition that the margin of spam instances that vary greatly has to be larger than the margin of non-spam instances that remain almost unmodified. This effect is stronger when the data generator's regularization parameter ρ_{+1} is small. By contrast, if ρ_{+1} goes to infinity, and consequently κ attains zero, then the SPG with linear loss reduces to the regular SVM.

4.3 SPG with Logistic Loss

Finally, this section introduces the *SPG with logistic loss*. This instantiation meets the preconditions of Theorem 1 and Proposition 1, and the resulting optimization criterion can be solved with standard tools. The learner may use any loss function that is convex and twice continuously differentiable (Equation 23 details the loss function used in our experiments) while the data generator uses the logistic loss

$$\ell^{\log}_{+1}(z, y) = \log\left(1 + e^{z}\right)$$

which again penalizes large decision values z. The rationale behind this loss function is that the data generator experiences costs when the learner blocks an event, *i.e.,* produces a high decision function value for an instance. For instance, a legitimate sender experiences costs when a legitimate email is erroneously blocked just like an abusive sender, also amalgamated into the *data generator*, experiences costs when spam messages are blocked. Cost function $\ell^{\log}_{+1}$ approaches zero for small values of the decision function. Now, the constraints in (8) resolve to $g_i(\mathbf{w}, \tau_i) = 0$ for $i = 1, \ldots, n$ with

$$g_i(\mathbf{w}, \tau_i) = \tau_i \left(1 + e^{-f_\mathbf{w}(x_i) - \tau_i\|\mathbf{w}\|^2}\right) + \frac{n}{\rho_{+1}} c_{+1,i}.$$

Functions $g_i(\mathbf{w}, \tau_i)$ are not jointly convex in $\mathbf{w}$ and τ_i. However, as they are smooth (*i.e.,* infinitely differentiable) in both arguments, their roots can be obtained efficiently and, consequently, the resulting optimization problem

$$\min_{\mathbf{w}, \forall i\,:\,\tau_i} \quad \sum_{i=1}^{n} c_{-1,i}\ell_{-1}\left(f_\mathbf{w}(x_i) + \tau_i\|\mathbf{w}\|^2, y_i\right) + \frac{\rho_{-1}}{2}\|\mathbf{w}\|^2$$
$$\text{s.t.} \quad \forall i : 0 = g_i(\mathbf{w}, \tau_i)$$

can be solved by standard SQP solvers.

5. EXPERIMENTAL EVALUATION

The goal of this section is to explore the relative strengths and weaknesses of the discussed instances of Stackelberg prediction games and existing baseline methods in the context of email spam filtering. We compare a regular *support vector machine* (SVM), *logistic regression* (LogReg), the *SVM for invariances* with feature scaling (Invar-SVM, [14]), *Nash logistic regression* (Nash, [1]), and the Stackelberg instances *SPG with worst-case loss* (SPG$^{\text{wc}}$, *cf.* Section 4.1), *SPG with linear loss* (SPG$^{\text{lin}}$, *cf.* Section 4.2), and the *SPG with logistic loss* (SPG$^{\log}$, *cf.* Section 4.3). For all Stackelberg instances we choose the logistic loss function

$$\ell^{\log}_{-1}(z, y) = \log\left(1 + e^{-yz}\right) \tag{23}$$

for the learner which is convex and smooth, and consequently satisfies Proposition 1. In the absence of prior knowledge on the instance-specific costs, we set $c_{v,i} = \frac{1}{n}$ for all $v \in \{-1, +1\}$, $i = 1, \ldots, n$ and train all methods in the PCA map induced feature space. To solve the nonlinear program of the SPG with logistic loss we use the Ipopt solver [16].

We use four email corpora detailed in Table 1: The first data set contains emails of an email service provider (*ESP*) collected between 2007 and 2010. The second (*Mailinglist*) is a collection of emails from publicly available mailing lists augmented by spam emails from Bruce Guenter's spam trap of the same time period. The third corpus (*Private*) contains newsletters and spam and non-spam emails of the authors. The last corpus is the NIST *TREC 2007* spam corpus. All emails are tokenized, converted into binary bag-of-word vectors, and sorted chronologically.

Table 1: Data sets used in the experiments.

data set	*instances*	*features*	*delivery period*
ESP	169,612	541,713	01/06/2007 - 27/04/2010
Mailinglist	128,117	266,378	01/04/1999 - 31/05/2006
Private	108,178	582,100	01/08/2005 - 31/03/2010
TREC 2007	75,496	214,839	04/08/2007 - 07/06/2007

Our evaluation protocol is as follows. We use the 4,000 oldest emails as training portion and set the remaining emails aside as test instances. We use the F-measure—that is, the harmonic mean of precision and recall—as evaluation measure and train all methods 20 times on a stratified subset of 200 spam and 200 non-spam messages sampled from the training portion. In order to tune the regularization parameters we perform a 5-fold cross validation on the training sample within each repetition of an experiment and for each method separately.

In the first experiment, we evaluate all methods *into the future* by processing the test set in chronological order. Each test sample is split into 20 disjoint subsets. We average

Figure 1: F-measure of predictive models. Error bars indicate standard errors.

the F-measure on each of those subsets over the 20 models (trained on different samples drawn from the training portion) for each method and perform a paired t-test.

Figure 1 shows that, for all data sets, the Stackelberg prediction games with linear loss and with logistic loss outperform the regular SVM and logistic regression that do not explicitly factor the adversary into the optimization criterion. On the ESP corpus, the SPG with linear loss is slightly better than the SPG with logistic loss whereas for the Mailinglist corpus the SPG with logistic loss outperforms the SPG with linear loss. On the TREC 2007 data set, most of the methods behave comparably with a slight advantage for the Nash logistic regression and the SPG instances with logistic loss and linear loss. The period over which the TREC 2007 data have been collected is very short; therefore we believe that the training and test instances are governed by nearly identical distributions. Consequently the game-theoretic models do not gain a significant advantage over logistic regression that assumes *iid* samples. For the other three data sets, the game-theoretical models outperform the *iid* baselines.

Table 2 shows aggregated results over all four data sets. For each point in each of the diagrams of Figure 1, we conduct a pairwise comparison of all methods based on a paired t-test at a confidence level of $\alpha = 0.05$. When a difference is significant, we count this as a win for the method that achieves a higher F-measure. Each line of Table 2 details the wins and, set in italics, the *losses* of one method against all other methods. The Stackelberg prediction game with logistic loss has more wins than it has losses against each of the other methods. The Stackelberg prediction game with linear loss has more wins than losses against each of the other methods except for the SPG with logistic loss and the Nash logistic regression. The ranking continues with the Invar-SVM, the SPG with worst-case loss, logistic regression, and the regular SVM which loses more frequently than it wins against all other methods.

To study the predictive performance as well as running time behavior with respect to the size of the data set, we train the baselines and the three SPG instances for a varying number of training examples. We report on the results for the representative ESP data set in Figure 2. Except for SPG^{wc}, the game models significantly outperform the trivial baseline methods SVM and logistic regression, especially for small corpus sizes. However, this comes at the price of considerably higher computational cost. For the game models, the Stackelberg instance SPG^{lin} clearly outperforms all reference methods with respect to efficiency. Though, the larger the size of the data set, the stronger the computational differences, where at the same time the discrepancy of the predictive performance diminishes.

The data generator's regularizer that we use in the experiments does not distinguish between modifications of spam and non-spam messages. In reality, most senders of legitimate messages do not deliberately change their writing behavior such as to bypass spam filters, perhaps with the exception of senders of legitimate newsletters who must be careful not to trigger filtering mechanisms. In a final exper-

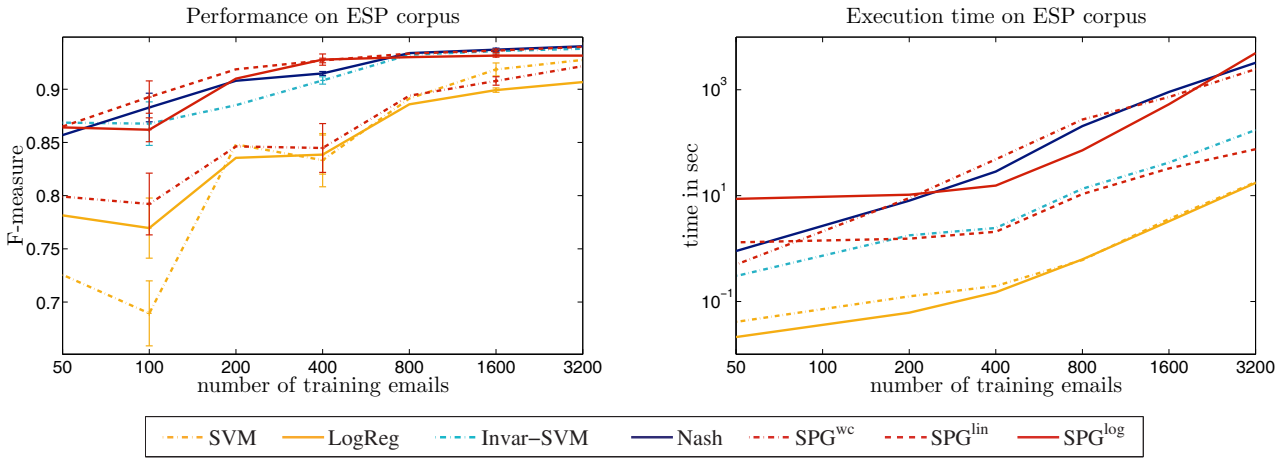

Figure 2: Predictive performance (left) and execution time (right) for varying sizes of the training data set.

Table 2: Results of paired t-test over all corpora: Number of trials in which each method (row) has significantly outperformed each other *method (column)* vs. number of times it *was outperformed*.

method vs. *method*	*SVM*	*LogReg*	*Invar-SVM*	*Nash*	*SPG^{wc}*	*SPG^{lin}*	*SPG^{log}*
SVM	0:*0*	6:*44*	2:*64*	0:*72*	8:*50*	6:*54*	6:*69*
LogReg	44:*6*	0:*0*	3:*41*	0:*72*	0:*29*	6:*48*	5:*57*
Invar-SVM	64:*2*	41:*3*	0:*0*	6:*40*	39:*10*	20:*23*	18:*30*
Nash	72:*0*	72:*0*	40:*6*	0:*0*	57:*2*	33:*17*	14:*16*
SPG^{wc}	50:*8*	29:*0*	10:*39*	2:*57*	0:*0*	17:*46*	9:*48*
SPG^{lin}	54:*6*	48:*6*	23:*20*	17:*33*	46:*17*	0:*0*	10:*23*
SPG^{log}	69:*6*	57:*5*	30:*18*	16:*14*	48:*9*	23:*10*	0:*0*

iment, we want to study whether the Stackelberg model reflects this aspect of reality. Table 3 shows the average number of modifications—*i.e.,* word additions and deletions—performed by the sender per spam and per non-spam email depending on the sender's regularization parameter ρ_{+1} for fixed ρ_{-1}.

Table 3: Average number of word additions and deletions per instance for SPG^{log}.

ρ_{+1}	non-spam		spam	
	additions	*deletions*	*additions*	*deletions*
4	1.4	1.6	14.6	17.6
16	0.3	0.3	9.9	11.6
64	0.0	0.0	7.1	8.7
256	0.0	0.0	2.4	2.8
1024	0.0	0.0	0.8	0.9

As expected, the number of transformations increases inversely proportional to the regularization parameter. Even for equal cost factors $c_{v,i}$, non-spam messages are rarely modified because the interests of sender and recipient are coherent for legitimate messages.

6. CONCLUSIONS

We model adversarial prediction problems as a game in which a learner has to commit to a predictive model using past data whereas the data generator may choose a transformation function after the predictive model has been disclosed which then defines the test distribution. This model reflects applications such as the detection of network attacks and spam filtering in which an assailant can probe the filter. The cost functions of learner and data generator are generally conflicting but are not constrained to be perfectly antagonistic. Playing the Stackelberg equilibrium instead of a worst-case strategy based on a zero-sum model is advisable when the data generator can be assumed to behave rational in the sense of minimizing a cost function. However, in contrast to the Nash strategy, the Stackelberg model does not rely on the existence of a unique equilibrium and the assumptions that the adversary has no information about the predictive model and is able to identify and follow the equilibrial strategy.

We derived a compact optimization problem that determines the solution of the resulting Stackelberg prediction game. We showed that the Stackelberg model generalizes existing prediction models such as SVM with uneven margins and SVM for invariances. We evaluated spam filters resulting from a regular SVM, logistic regression, existing game-theoretical models, and three instances of the Stackelberg game on several spam-filtering data sets. The relative performance of the distinct game-theoretic models varies, but we observe that when compared to any other model, the Stackelberg model with logistic loss has more wins than it has losses against each of the baseline methods.

Acknowledgments

This work was supported by the German Science Foundation DFG under grant SCHE 540/12-1 and by STRATO AG.

7. REFERENCES

[1] M. Brückner and T. Scheffer. Nash equilibria of static prediction games. In *Advances in Neural Information Processing Systems*. MIT Press, 2009.

[2] B. Colson, P. Marcotte, and G. Savard. An overview of bilevel optimization. *Annals of Operations Research*, 153(1):235–256, 2007.

[3] O. Dekel and O. Shamir. Learning to classify with missing and corrupted features. In *Proceedings of the International Conference on Machine Learning*, pages 216–223. ACM, 2008.

[4] O. Dekel, O. Shamir, and L. Xiao. Learning to classify with missing and corrupted features. *Machine Learning*, 81(2):149–178, 2010.

[5] L. E. Ghaoui, G. R. G. Lanckriet, and G. Natsoulis. Robust classification with interval data. Technical Report UCB/CSD-03-1279, EECS Department, University of California, Berkeley, 2003.

[6] A. Globerson and S. T. Roweis. Nightmare at test time: robust learning by feature deletion. In *Proceedings of the International Conference on Machine Learning*. ACM, 2006.

[7] A. Globerson, C. H. Teo, A. J. Smola, and S. T. Roweis. *Dataset Shift in Machine Learning*, chapter An adversarial view of covariate shift and a minimax approach, pages 179–198. MIT Press, 2009.

[8] R. Jeroslow. The polynomial hierarchy and a simple model for competitive analysis. *Mathematical Programming*, 32:146–164, 1985.

[9] M. Kantarcioglu, B. Xi, and C. Clifton. Classifier evaluation and attribute selection against active adversaries. *Data Mining and Knowledge Discovery*, 22(1-2):291–335, 2011.

[10] G. R. G. Lanckriet, L. E. Ghaoui, C. Bhattacharyya, and M. I. Jordan. A robust minimax approach to classification. *Journal of Machine Learning Research*, 3:555–582, 2002.

[11] Y. Li and J. Shawe-Taylor. The SVM with uneven margins and chinese document categorization. In *Proceedings of the Pacific Asia Conference on Language, Information and Computation*, pages 216–227, 2003.

[12] W. Liu and S. Chawla. A game theoretical model for adversarial learning. In *ICDM Workshops*, pages 25–30. IEEE Computer Society, 2009.

[13] B. Schölkopf, R. Herbrich, and A. J. Smola. A generalized representer theorem. In *COLT: Proceedings of the Workshop on Computational Learning Theory, Morgan Kaufmann Publishers*, 2001.

[14] C. H. Teo, A. Globerson, S. T. Roweis, and A. J. Smola. Convex learning with invariances. In *Advances in Neural Information Processing Systems*. MIT Press, 2007.

[15] S. Veelken. *A New Relaxation Scheme for Mathematical Programs with Equilibrium Constraints: Theory an Numerical Experience*. PhD thesis, Technische Universität München, 2009.

[16] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106:25–57, 2006.