Bayesian Clustering for Email Campaign Detection

Peter Haider HAIDER@CS.UNI-POTSDAM.DE
Tobias Scheffer SCHEFFER@CS.UNI-POTSDAM.DE
University of Potsdam, Department of Computer Science, August-Bebel-Strasse 89, 14482 Potsdam, Germany

Abstract

We discuss the problem of clustering elements according to the sources that have generated them. For elements that are characterized by independent binary attributes, a closed-form Bayesian solution exists. We derive a solution for the case of dependent attributes that is based on a transformation of the instances into a space of independent feature functions. We derive an optimization problem that produces a mapping into a space of independent binary feature vectors; the features can reflect arbitrary dependencies in the input space. This problem setting is motivated by the application of spam filtering for email service providers. Spam traps deliver a real-time stream of messages known to be spam. If elements of the same campaign can be recognized reliably, entire spam and phishing campaigns can be contained. We present a case study that evaluates Bayesian clustering for this application.

1. Introduction

In model-based clustering, elements $X = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\}$ have been created by an unknown number of sources; each of them creates elements according to its specific distribution. We study the problem of finding the most likely clustering of elements according to their source

$$C^* = \arg\max_C P(X|C), \tag{1}$$

where C consistently partitions the elements of X into clusters $C = \{c_1, \ldots, c_m\}$ that are mutually exclusive and cover each element.

Appearing in *Proceedings of the 26th International Conference on Machine Learning*, Montreal, Canada, 2009.

Computing the likelihood of a set X under a clustering hypothesis C requires the computation of the joint likelihood of mutually dependent elements X_c within a partition c. This is usually done by assuming latent mixture parameters θ that generate the elements of each cluster and imposing a prior over the mixture parameters. The joint likelihood is then an integral over the parameter space, where individual likelihoods are independent given the parameters:

$$P(X_c) = \int \prod_{\mathbf{x} \in X_c} P(\mathbf{x}|\theta) P(\theta) d\theta. \tag{2}$$

For suitable choices of $P(\mathbf{x}|\theta)$ and $P(\theta)$, this integral has an analytic solution. In many cases, however, the space $\mathcal{X}$ of elements is very high-dimensional, and the choice of likelihood and prior involves a trade-off between expressiveness of the generative model and tractability regarding the number of parameters. If the elements are binary vectors, $\mathcal{X} = \{0,1\}^D$, one extreme would be to model $P(\mathbf{x}|\theta)$ as a full multinomial distribution over $\mathcal{X}$, involving 2^D parameters. The other extreme is to make an independence assumption on the dimensions of $\mathcal{X}$, reducing the model parameters to a vector of D Bernoulli probabilities. No remedy for this dichotomy is known that preserves the existence of an analytic solution to the integral in Equation 2.

Our problem setting is motivated by the application of clustering messages according to campaigns; this will remain our application focus throughout the paper. Filtering spam and phishing messages reliably remains a hard problem. Email service providers operate *Mail Transfer Agents* which observe a stream of incoming messages, most of which have been created in bulk by a generator. A generator can be an application that dispatches legitimate, possibly customized newsletters, or a script that creates spam or phishing messages and disseminates them from the nodes of a botnet. Mail Transfer Agents typically blacklist known spam and phishing messages. Messages known to be spam can be collected by tapping into botnets, and by harvesting emails in spam traps. Spam traps are email addresses published invisibly on the web that

have no legitimate owner and can therefore not receive legitimate mail. In order to avoid blacklisting, spam dissemination tools produce emails according to probabilistic templates. This motivates our problem setting: If all elements that are generated in a joint campaign can be identified reliably, then all instances of that campaign can be blacklisted as soon as one element reaches a spam trap, or is delivered from a known node of a botnet. Likewise, all instances of a newsletter can be whitelisted as soon as one instance is confirmed to be legitimate.

While text classification methods are frequently reported to achieve extremely high accuracy for spam filtering under laboratory conditions, their practical contribution to the infrastructure of email services is smaller: they are often applied to decide whether accepted emails are to be delivered to the inbox or the spam folder. The vast majority of all spam delivery attempts, however, is turned down by the provider based on known message and IP blacklists. Text classifiers are challenged with continuously shifting distributions of spam and legitimate messages; their risk of false positives does not approach zero sufficiently closely to constitute a satisfactory solution to the spam problem.

This paper makes three major contributions. Firstly, we develop a generative model for a clustering of binary feature vectors, based on a transformation of the input vectors into a space in which an independence assumption incurs minimal approximation error. The transformations can capture arbitrary dependencies in the input space while the number of parameters stays reasonable and full Bayesian inference remains tractable. Secondly, we derive the optimization problem and algorithm that generates the feature transformation. Finally, we present a large-scale case study that explores properties of the Bayesian clustering solution for email campaign detection.

The paper is structured as follows. We present the Bayesian clustering model in Section 2, and an optimization problem and algorithm for transforming dependent features into independent features in Section 3. Section 4 discusses the estimation of prior parameters, Section 5 develops a sequential clustering algorithm based on Bayesian decisions. Section 6 reports on empirical results in our motivating application. We review related work in Section 7. Section 8 concludes.

2. Bayesian Clustering for Binary Features

In general, a clustering hypothesis C entails that the likelihood of the dataset factorizes into the likelihoods of the subsets X_c of elements in the clusters $c \in C$. Elements within a cluster are dependent, so the likelihood of each element depends on the preceding elements in its cluster, as in Equation 3.

$$\begin{aligned} P(X|C) &= \prod_{c \in C} P(X_c) \\ &= \prod_{c \in C} \prod_{i:\mathbf{x}^{(i)} \in c} P(\mathbf{x}^{(i)} | \{\mathbf{x}^{(j)} \in c : j < i\}) \end{aligned} \quad (3)$$

The crucial part is modeling the probability $P(\mathbf{x}|X')$ of a binary feature vector $\mathbf{x}$ given a set of elements X'. A natural way is to introduce latent model parameters θ and integrate over them as in Equation 4.

$$P(\mathbf{x}|X') = \int_\theta P(\mathbf{x}|\theta) P(\theta|X') d\theta \quad (4)$$

Modeling θ as a full joint distribution over all 2^D possible feature vectors, with D being the dimensionality of the input space $\mathcal{X}$, is intractable.

Let ϕ_e be independent binary features and let vector $\phi(\mathbf{x})$ be a representation of $\mathbf{x}$ in the space of independent features ϕ. In order to streamline the presentation of the clustering model, we postpone the rationale and construction of the feature transformation to Section 3. Under the assumption that attributes in the space ϕ are independent, the model parameters can be represented as a vector of Bernoulli probabilities, $\theta \in (0,1)^E$, and we can compute $P(\mathbf{x}|\theta)$ as $\prod_{e=1}^{E} P(\phi_e(\mathbf{x})|\theta_e)$. Furthermore, we impose a Beta prior on every component θ_e with parameters α_e and β_e. Since the Beta distribution is conjugate to the Bernoulli distribution, we can now compute the posterior over the model parameters analytically as in Equation 5, where $\#_e = |\{\mathbf{x}' \in X' : \phi_e(\mathbf{x}') = 1\}|$.

$$\begin{aligned} P(\theta|X') &= P(X'|\theta) \frac{P(\theta)}{P(X')} \\ &= \frac{\prod_{e=1}^{E} \prod_{\mathbf{x} \in X'} P(\phi_e(\mathbf{x})|\theta_e) P_{Beta}(\theta_e|\alpha_e, \beta_e)}{\int P(X'|\theta') P(\theta') d\theta'} \\ &= \prod_{e=1}^{E} P_{Beta}(\theta_e|\alpha_e + \#_e, \beta_e + |X| - \#_e) \end{aligned} \quad (5)$$

The integral in Equation 4 then has the analytic solution of Equation 6:

$$\begin{aligned} P(\mathbf{x}|X') = &\prod_{e:\phi_e(\mathbf{x})=1} \frac{\alpha_e + \#_e}{\alpha_e + \beta_e + |X'|} \\ &\prod_{e:\phi_e(\mathbf{x})=0} \frac{\beta_e + |X'| - \#_e}{\alpha_e + \beta_e + |X'|}. \end{aligned} \quad (6)$$

For a single element, independent of all others, the probability term simplifies to

$$P(\mathbf{x}) = \prod_{e:\phi_e(\mathbf{x})=1} \frac{\alpha_e}{\alpha_e+\beta_e} \prod_{e:\phi_e(\mathbf{x})=0} \frac{\beta_e}{\alpha_e+\beta_e}. \quad (7)$$

Furthermore, the joint probability of an interdependent set X' in one cluster can be computed as

$$\begin{aligned} P(X') &= \prod_{i:\mathbf{x}^{(i)}\in X'} P(\mathbf{x}^{(i)}|\{\mathbf{x}^{(j)} \in X' : j<i\}) \\ &= \prod_{e=1}^{E} \frac{\prod_{k=1}^{\#_e}(\alpha_e+k-1)\prod_{k=1}^{|X'|-\#_e}(\beta_e+k-1)}{\prod_{k=1}^{|X'|}(\alpha_e+\beta_e-1)} \\ &= \prod_{e=1}^{E} \frac{B(\alpha_e+\#_e, \beta_e+|X'|-\#_e)}{B(\alpha_e,\beta_e)}, \end{aligned}$$

where B denotes the Beta function.

3. Feature Transformation

In this section we will present a method of approximating a distribution over high-dimensional binary vectors that allows analytical integration over the induced model parameters. The idea is to find a mapping into another space of binary vectors where the dimensions are treated independently of each other, such that the divergence between the original distribution and the approximate distribution defined in terms of the mapped vectors is minimal.

A straightforward approach would be to employ a model that captures dependencies between small sets of attributes only and assumes independence otherwise. Instead of working with independence assumptions, we construct a search space of transformations. This space is complete in the sense that any possible interaction of attributes can be reflected in the newly constructed attributes. These attributes are constructed such that their product approximates the true distribution as closely as possible.

The Bayesian clustering model introduced in Section 2 requires us to infer the probability of a feature vector $\mathbf{x}$ given a set of feature vectors X' it depends on. We therefore want to approximate $P(\mathbf{x}|X')$ by a quantity $Q_\phi(\mathbf{x}|X')$, where ϕ is a mapping from the original vector space $\mathcal{X}=\{0,1\}^D$ to the image space $\mathcal{Z}=\{0,1\}^E$. We define Q_ϕ as a product over independent probabilities for each output dimension, as in Equation 8.

$$Q_\phi(\mathbf{x}|X') = \prod_{e=1}^{E} P(\phi_e(\mathbf{x})|\phi_e(X')) \quad (8)$$

By its definition as a product over probabilities, quantity $Q_\phi(\mathbf{x}|X)$ is always non-negative; however, it does not necessarily sum to one over the event space of possible inputs $\mathbf{x}$. Since Q_ϕ serves as an approximation of a probability in the Bayesian inference, it is desirable that $\sum_{\mathbf{x}} Q_\phi(\mathbf{x}|X) \le 1$ for all X. Moreover, the natural measure of approximation quality – the Kullback-Leibler divergence – is only motivated for measures that add up to at most one and may be maximized by trivial solutions otherwise. Note that the sum does not have to be exactly 1, since an extra element $\bar{\mathbf{x}}$ with $P(\bar{\mathbf{x}}) = 0$ can be added to $\mathcal{X}$ that absorbs the remaining probability mass, $Q_\phi(\bar{\mathbf{x}}|X') \stackrel{\text{def}}{=} 1 - \sum_{\mathbf{x}\in\mathcal{X}} Q_\phi(\bar{\mathbf{x}}|X')$.

Normalization of $Q_\phi(\mathbf{x}|X)$ is intractable, since it would require explicit summation of Equation 8 over all 2^D possible input elements. We therefore have to define the space of possible transformations such that after any transformation Q_ϕ is guaranteed to sum to at most one. By Theorem 1 (see Appendix), this holds for all injective transformations.

Every mapping from $\mathcal{X}=\{0,1\}^D$ to the $\mathcal{Z}=\{0,1\}^E$ can be represented as a set of E Boolean functions, and every Boolean function can be constructed as a combination of elementary operations. Therefore we can define the search space as the set of all concatenations of elementary Boolean transformations ψ that preserve injectiveness. The choice of which elementary transformations to use is driven by the practical goal that ϕ also preserves sparseness. The following two elementary transformations are injective, sufficient to generate any Boolean function, and preserve sparsity:

$$\begin{aligned} &\psi^x_{ij}((\dots,x_i,\dots,x_j,\dots)^\top) = (\dots,x_i,\dots,x_i\neq x_j,\dots)^\top \\ &\psi^a_{ij}((\dots,x_i,\dots,x_j,\dots)^\top) \\ &\qquad = (\dots,x_i\wedge x_j,\dots,x_i\wedge\neg x_j,\neg x_i\wedge x_j,\dots)^\top. \end{aligned}$$

Every ψ replaces two features by Boolean combinations thereof, leaving every other feature untouched.

For any set of elements X, the quantity $Q_\phi(\mathbf{x}|X)$ should minimize the Kullback-Leibler divergence from the true distribution $P(\mathbf{x}|X)$. Hence, the optimization criterion (Equation 9) is the expected KL divergence between $Q_\phi(\mathbf{x}|X)$ and $P(\mathbf{x}|X)$ over all X.

$$\phi^* = \arg\min_\phi \mathop{E}_{\mathbf{X}\sim P(\mathbf{x})} [KL(P(\cdot|X)||Q_\phi(\cdot|X))] \quad (9)$$

$$= \arg\min_\phi \mathop{E}_{\mathbf{X}\sim P(\mathbf{x})} \left[\sum_{\mathbf{x}\in\mathcal{X}} P(\mathbf{x}|X)\log\frac{P(\mathbf{x}|X)}{Q_\phi(\mathbf{x}|X)}\right] \quad (10)$$

$$= \arg\max_\phi \mathop{E}_{\mathbf{X}\sim P(\mathbf{x})} \left[\sum_{\mathbf{x}\in\mathcal{X}} P(\mathbf{x}|X)\log Q_\phi(\mathbf{x}|X)\right] \quad (11)$$

$$= \arg\max_\phi \mathop{E}_{\mathbf{X}\sim P(\mathbf{x}),\mathbf{x}\sim P(\mathbf{x}|X)} [\log Q_\phi(\mathbf{x}|X)]. \quad (12)$$

Equation 10 expands the definition of the KL divergence; Equation 11 replaces minimization by maximization and drops the term $P(\mathbf{x}|X)\log P(\mathbf{x}|X)$ which is constant in ϕ. We approximate the expectation in Equation 12 by the sum over an empirical sample S, and obtain

Optimization Problem 1. *Over the set of concatenations of elementary transformations, $\phi \in \{\psi_{ij}^x, \psi_{ij}^a\}^*$, maximize*

$$\sum_{\mathbf{x}\in S} \log Q_\phi(\mathbf{x}|S\setminus\{\mathbf{x}\}).$$

The sum of log-probabilities can be calculated as

$$\begin{aligned}&\sum_{\mathbf{x}\in S} \log Q_\phi(\mathbf{x}|S\setminus\{\mathbf{x}\})\\ =&\sum_{e=1}^{E} \#_e^S \log(\alpha_0 + \#_e^S - 1) - |S|\log(\alpha_0+\beta_0+|S|-1)\\ &+ (|S| - \#_e^S)\log(\beta_0 + |S| - \#_e^S - 1),\end{aligned}$$

where $\#_e^S = |\{\mathbf{x}' \in S : \phi_e(\mathbf{x}') = 1\}|$, and α_0, β_0 are the parameters of the Beta prior.

Optimization Problem 1 is non-convex, so we apply a greedy procedure that iteratively adds the next-best transformation starting with the identity transformation, as detailed in Algorithm 1.

Algorithm 1 Greedy Transformation Composition

```
φ^0 ← id
for t = 1 ... do
  ψ̂ ← arg max_ψ Σ_{x∈S} Q_{φ^{t-1}∘ψ}(x|S \ {x})
  if Σ_{x∈S} Q_{φ^{t-1}∘ψ̂}(x|S \ {x}) < Σ_{x∈S} Q_{φ^{t-1}}(x|S \ {x})
  then
    return φ^{t-1}
  else
    φ^t ← φ^{t-1} ∘ ψ̂
  end if
end for
```

4. Parameter Estimation

In the following section, we will derive a closed-form estimator for parts of the parameters of the prior $P(\theta)$. The decisions whether to merge an element $\mathbf{x}$ with a set X' depend strongly on the prior parameters via $P(\mathbf{x})$ in Equation 7 and $P(\mathbf{x}|X')$ in Equation 6.

Heller and Ghahramani (2005) derive an EM-like algorithm that maximizes the data likelihood by iteratively finding the best clustering and then performing gradient descent on the prior parameters. This approach is computationally very expensive, since the likelihood function is not convex and the entire dataset needs to be re-clustered in each iteration.

We overcome these problems by using an alternative parametrization of the Beta distribution. This allows us to estimate half of the parameters from an unclustered set of training examples S; the other half of the parameters is pooled into a single value and adjusted by a grid search on tuning data.

We re-parametrize the Beta priors as $\alpha_e = \mu_e\sigma$ and $\beta_e = (1-\mu_e)\sigma$, where the μ_e are the prior means and σ is the common precision[1] parameter. The probability of an element *not given* any other elements of the same cluster does not depend on the prior precisions, only on the means. Hence, the means have a stronger impact on the resulting partitioning.

Imposing a Beta-distributed hyperprior on μ_e with parameters $\alpha_0 > 1$ and $\beta_0 > 1$ we can compute the Maximum-A-Posteriori estimate of the means as

$$\begin{aligned}\mu_e &= \arg\max_\mu P(\mu|S) = \arg\max_\mu \prod_{\mathbf{x}\in S} P(\phi_e(\mathbf{x})|\mu)P(\mu)\\ &= \arg\max_\mu P_{Beta}\left(\mu|\alpha_0 + |\{\mathbf{x}\in S:\phi_e(\mathbf{x})=1\}|,\right.\\ &\qquad\left.\beta_0 + |\{\mathbf{x}\in S:\phi_e(\mathbf{x})=0\}|\right)\\ &= \frac{\alpha_0 + |\{\mathbf{x}\in S:\phi_e(\mathbf{x})=1\}| - 1}{\alpha_0+\beta_0+|S|-2}.\end{aligned}$$

5. Sequential Bayesian Clustering

In this section we discuss the task of inferring the most likely partitioning of a set of emails and present our model-based sequential clustering algorithm.

Brute-force search over the entire space of possible partitionings in Equation 1 requires the evaluation of exponentially many clusterings and is therefore intractable for reasonable numbers of emails. A more efficient approach would be to perform Markov chain Monte Carlo sampling methods like in (Williams, 2000), which yields not only the Maximum-A-Posteriori partitioning, but samples from the posterior distribution.

Approximate agglomerative clustering algorithms (Heller & Ghahramani, 2005) are more efficient. Since in practice emails have to be processed sequentially, and decisions whether an email belongs to a spam campaign cannot be revised after delivering it, we adopt the sequential clustering algorithm of Haider et al.

[1] The precision parameter of a Beta distribution is inversely related to its variance.

Algorithm 2 Model-based Sequential Clustering

$C \leftarrow \{\}$
for $t = 1 \dots n$ **do**
  $c_j \leftarrow \arg\max_{c \in C} P(\mathbf{x}^{(t)}|X_c)$
  if $P(\mathbf{x}^{(t)}|X_c) < P(\mathbf{x}^{(t)})$ **then**
    $C \leftarrow C \cup \{\{\mathbf{x}^{(t)}\}\}$
  else
    $C \leftarrow C \setminus \{c_j\} \cup \{c_j \cup \{\mathbf{x}^{(t)}\}\}$
  end if
end for
return C

(2007). This greedy incremental algorithm has the advantage of approximately finding the best campaign association of a new email in $O(n)$, where n is the number of previously seen emails, instead of taking $O(n^3)$ operations for performing a full agglomerative re-clustering.

Instead of a weighted similarity measure as in (Haider et al., 2007), our clustering model is based on a generative model. We replace the weighted sum over pairwise features by an integral over the model parameters of a cluster. This gives us the model-based sequential clustering in Algorithm 2.

In every step, the algorithm compares the hypotheses that the new element belongs to one of the existing clusters with the hypothesis that it forms its own cluster. The likelihoods of these hypotheses are calculated according to Equations 6 and 7. This greedy algorithm can be straightforwardly extended to using a non-uniform prior over clustering hypotheses, by replacing $P(\mathbf{x})$ with $P(\mathbf{x})P(C \cup \{\{\mathbf{x}^{(t)}\}\})$ and $P(\mathbf{x}^{(t)}|X_c)$ with $P(\mathbf{x}^{(t)}|X_c)P(C \setminus \{c_j\} \cup \{c_j \cup \{\mathbf{x}^{(t)}\}\})$.

6. Email Campaign Detection

In this section, we explore the behavior of the feature transformation procedure, and conduct a case study of the Bayesian clustering method for spam filtering.

In unsupervised clustering, there is no ground truth available that the output of the clustering algorithm can be compared with. Fortunately, our motivating application scenario – email spam containment – has a natural evaluation criterion: the contribution of the produced partitionings to accurate filtering.

We design an experimental setting that evaluates the Bayesian clustering solution and the feature transformation technique for the problem of detecting spam campaigns at an email service provider. Benchmark data sets such as the SpamTREC corpus are not suitable for our evaluation. A fair evaluation relies on a stream that contains realistic proportions of messages of mailing campaigns, in the correct chronological order. Benchmark corpora contain messages that have been received by users, and have therefore passed unknown filtering rules employed by the receiving server. Furthermore, benchmark data sets do not contain reliable time stamps whereas the actual chronological order is crucial.

Our experimental setting relies on a stream of spam messages received by the mail transfer agent of an email service provider. Between July and November 2008, we recorded a small fraction of spam messages, a total of 139,250 spam messages in correct chronological order. The messages have been tagged as spam because the delivering agent was listed on the Spamhaus IP block list which is maintained manually. We simulate the practical setting where one has a stream of verified spams, and one stream of unknown emails, by taking every other email from the set as training example. The rest is split into test examples (90%) and tuning examples. In order to maintain the users' privacy, we blend the stream of spam messages with an additional stream of 41,016 non-spam messages from public sources. The non-spam portion contains newsletters and mailing lists in correct chronological order as well as Enron emails and personal mails from public corpora which are not necessarily in chronological order. Every email is represented by a binary vector of 1,911,517 attributes that indicate the presence or absence of a word. The feature transformation technique introduces an additional 101,147 attributes.

6.1. Feature Transformation

In order to assess the capability of our feature transformation technique for approximating a high dimensional probability distribution, we train the transformation on an additional set S_1 of 10,000 older emails including spam and ham (*i.e.,* non-spam) messages in equal parts, and test on another set S_2 of emails of the same size. Since we cannot measure the Kullback-Leibler divergence from the true distribution directly, we measure the quantity $\frac{-1}{|S_i|} \sum_{\mathbf{x} \in S_i} \log Q_\phi(\mathbf{x}|S_i \setminus \{\mathbf{x}\})$, which is the average entropy of an email, given all other emails of the set. We compare the entropies on the training and test sets for the transformation found by the Greedy Transformation Composition algorithm to the entropy of the identity transformation. The identity transformation corresponds to an assumption of independent attributes in the input space.

In addition to the overall optimal transformation, we compute the optimal transformations ϕ^a and ϕ^x composed of only elementary transformations of the forms

ψ^a_{ij} or ψ^x_{ij}, respectively. Preliminary experiments showed that the choice of prior parameters α_0 and β_0 has negligible influence within reasonable ranges, so we report the results for $\alpha_0 = 1.1$ and $\beta_0 = 100$ in Table 1. We can see that including elementary transformations of the form ψ^x_{ij} decreases training entropy, but increases test entropy. The best transformation reduces test entropy compared to the identity transformation by about 33%. This shows that factorizing the probability over the dimensions in the image space yields a much better approximation than factorizing over the dimensions in the original feature space.

Table 1. Comparison of training and test entropies using different feature transformations.

Transformation	id	ϕ^*	ϕ^a	ϕ^x
Training entropy	1013.7	**574.1**	585.3	632.4
Test entropy	1007.1	687.5	**672.0**	720.2

6.2. Bayesian Clustering for Spam Filtering

Our evaluation protocol is as follows. We use a training window of 5,000 known spam messages, corresponding to a history of approximately 11 days. The training messages are partitioned using Algorithm 2. In each step, the clustering algorithm adds the chronologically next 100 known spam emails to the partitioning and removes the 100 oldest. We then classify the 100 next test messages. We use the odds ratio

$$\frac{\max_{X_{spam}} P(\mathbf{x}|X_{spam})}{P(\mathbf{x})}$$

as classification score, the maximum is over all spam clusters in the training window. Test messages are not added to the window of partitioned training messages.

A main difficulty in spam filtering is that ham emails can be very diverse, and it is unrealistic that one has training examples available from every region of the true distribution. We conduct experiments in two different settings that assess performance when ham emails from the test distribution are available and the performance without access to ham emails, respectively. In setting A, we train the feature transformation and parameters μ_e with 10,000 ham emails from the test distribution, and in setting B, we train on 10,000 spam messages instead.

As baseline for setting A, we use a Support Vector Machine that is trained in every step on the history of the last 5,000 spam and on the same 10,000 ham emails as the clustering method. Hence, the SVM baseline receives the same training data. In setting B, we use a one-class SVM, trained on the history of 5,000 spam messages. Additionally, we evaluate the benefit of the feature transformation by comparing with a clustering algorithm that uses the identity transformation.

An EM clustering procedure that uses a point estimates for the model parameters serves as an additional reference. We use a MAP estimate based on the same model prior used for the Bayesian model. EM requires the number of clusters to be known. We use the number of clusters that the Bayesian model identifies as input to the EM clustering.

We use two evaluation measures. Firstly, we measure the area under the ROC curve (AUC). Secondly, we use an evaluation measure that reflects the characteristics of the application more closely. An MTA has to be extremely confident when deciding to refuse a message for delivery from a contacting agent. We therefore measure the rate of true positives (spam messages identified as such) at a false positive rate of zero. We adjust the hyperparameters σ for the clustering model and C or ν for the standard SVM and one-class SVM, respectively, on the tuning set. We tune the parameters separately for optimal AUC, and for an optimal rate of true positives at a false positive rate of zero. Figure 1 shows ROC curves.

We can see that in the setting with ham emails available for training, the SVM outperforms the clustering-based filter in terms of AUC. In terms of the true positive rate at a false positive rate of zero, the clustering method outperforms the SVM classifier, by achieving a true positive rate of $0.945 \pm 9.1 \times 10^{-4}$ compared to $0.938 \pm 9.6 \times 10^{-4}$ of the SVM. The cluster-based filter shows its full strength in setting B, in the absence of non-spam training messages from the test distribution. Here, it achieves an AUC value of $0.992 \pm 2.6 \times 10^{-4}$ and a true positive rate of $0.749 \pm 1.7 \times 10^{-3}$, whereas the one-class SVM attains an AUC of $0.770 \pm 1.4 \times 10^{-3}$ and a true positive rate of $0.102 \pm 1.2 \times 10^{-3}$. That is, the Bayesian clustering method increases the true positive rate at zero false positives almost sevenfold, in the setting where no training emails from the distribution of the test hams are available. Clustering with the identity transformation as well as clustering with the EM algorithm performs worse in all settings than Bayesian clustering with the feature transformation. In setting A (ham messages from the test distribution available) with the parameters tuned for a high true positive rate at a false positive rate of zero, the EM algorithm achieves a true positive rate of only 0.04.

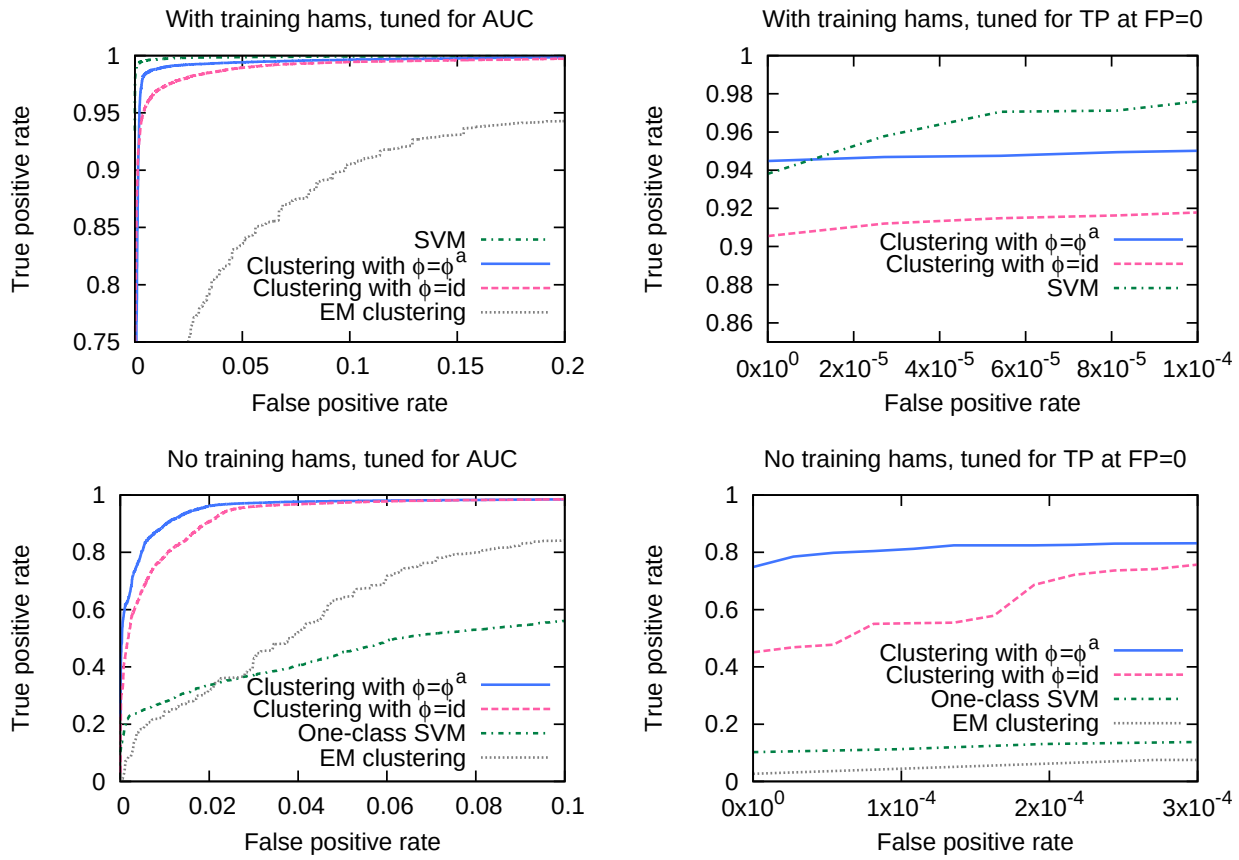

Figure 1. Evaluation of spam filtering performance.

7. Related Work

Previous work on Bayesian clustering explored in great detail the use of hierarchical priors for the cluster structure and algorithms for inference under such priors, for example (Williams, 2000), (Heller & Ghahramani, 2005), and (Lau & Green, 2007). These approaches focus on modeling hierarchical dependencies between elements, while modeling only low-level dependencies between the attributes within elements, such as Gaussian covariances. By contrast, we assume a uniform prior over the cluster structure, and instead focus on modeling arbitrary dependencies between binary attributes. We find that a non-uniform prior over partitionings is in fact not necessary, because properly taking the prior over mixture parameters $P(\theta)$ into account also prevents the trivial solution of assigning every element to its own cluster from being optimal.

Haider et al. (2007) devise a technique for mailing campaign detection that relies on training data that are manually clustered by campaigns. We find that the effort of manually partitioning training data into clusters is prohibitive in practice. Note that the effort of partitioning data is much higher than the effort of labeling data for classification because pairs of examples have to be considered.

Multi-way dependencies between attributes have been considered for instance by Zheng and Webb (2000) and Webb et al. (2005). They model the probability of an attribute vector as a product of conditional probabilities, such that each attribute can depend on multiple other attributes. If these approaches were to be used for Bayesian clustering, the number of mixture parameters would grow exponentially in the degree of dependencies. For our application, the high number of attributes renders these approaches infeasible.

Remedies for the problem of constantly changing distributions in the Spam filtering domain have been proposed in the area of adverserial learning. Teo et al. (2008) developed a formulation that allows to model test emails as modified versions of the training emails and optimize the classifier against the worst-case scenario of modifications. This approach leads to classi-

fiers that are more robust against changes of the distribution of spam emails, but still require the availability of recent spam and ham training data.

8. Conclusion

We devised a model for Bayesian clustering of binary feature vectors. The model is based on a closed-form Bayesian solution of the data likelihood in which the model parameters are integrated out. It allows for arbitrary dependencies between the input features, by transforming them into a space in which treating them as independent incurs minimal approximation error. We derived an optimization problem for learning such a transformation as a concatenation of elementary Boolean operations. In order to estimate the parameters of the prior from unlabeled data, we rewrite the parameters of the beta distribution in terms of mean values and a common variance. The mean values can be inferred in closed form from unlabeled data efficiently, the common variance constitutes a parameter that is adjusted on tuning data. We adapted a sequential clustering algorithm to use it with Bayesian clustering decisions. In a case study, we observed that the Bayesian clustering solution achieves higher true positive rates at a false positive rate of zero than an SVM. The benefit of the clustering solution is particularly visible when no non-spam training messages from the test distribution are available.

Acknowledgments

We gratefully acknowledge support from STRATO Rechenzentrum AG.

References

Haider, P., Brefeld, U., & Scheffer, T. (2007). Supervised Clustering of Streaming Data for Email Batch Detection. *Proceedings of the 24th International Conference on Machine Learning* (pp. 345–352).

Heller, K. A., & Ghahramani, Z. (2005). Bayesian hierarchical clustering. *Proceedings of the 22nd International Conference on Machine Learning* (pp. 297–304).

Lau, J., & Green, P. (2007). Bayesian Model-Based Clustering Procedures. *Journal of Computational and Graphical Statistics*, *16*, 526–558.

Teo, C., Globerson, A., Roweis, S., & Smola, A. (2008). Convex Learning with Invariances. *Advances in Neural Information Processing Systems*, *20*, 1489–1496.

Webb, G., Boughton, J., & Wang, Z. (2005). Not So Naive Bayes: Aggregating One-Dependence Estimators. *Machine Learning*, *58*, 5–24.

Williams, C. (2000). A MCMC approach to hierarchical mixture modelling. *Advances in Neural Information Processing Systems*, *12*, 680–686.

Zheng, Z., & Webb, G. (2000). Lazy Learning of Bayesian Rules. *Machine Learning*, *41*, 53–84.

Appendix

Theorem 1. *Let the implicit model parameter θ_e of the distribution $P(z_e)$ be Beta-distributed with parameters α_e and β_e for each $e \in \{1,\dots,E\}$. Then the quantity $Q_\phi(\mathbf{x}|X')$ as defined in Equation 8 sums to at most 1 for all X' iff ϕ is injective.*

Proof. First we show indirectly that from $\forall X' : \sum_{\mathbf{z}\in\mathcal{Z}} |\{\mathbf{x} : \phi(\mathbf{x}) = \mathbf{z}\}| \prod_{e=1}^{E} P(z_e|\phi_e(X')) \leq 1$ follows $\forall \mathbf{z} : |\{\mathbf{x} : \phi(\mathbf{x}) = \mathbf{z}\}| \leq 1$. Assume that there exists a $\mathbf{z}^* \in \mathcal{Z}$ with $|\{\mathbf{x} : \phi(\mathbf{x}) = \mathbf{z}^*\}| \geq 2$. Then choose an $\mathbf{x}^*$ with $\phi(\mathbf{x}^*) = \mathbf{z}^*$. Without loss of generality, let $\forall e : \mathbf{z}^*_e = 1$. Then set $n = \max_e \left(\sqrt[E]{0.5}(\alpha_e + \beta_e) - \alpha_e \right) / \left(1 - \sqrt[E]{0.5}\right) + 1$ and $X^* = \{\mathbf{x}^*, \dots, \mathbf{x}^*\}$ with $|X^*| = n$. It follows that

$$\prod_{e=1}^{E} P(z^*_e|\phi_e(X^*)) = \prod_{e=1}^{E} \int P(z^*_e|\theta_e) P(\theta_e|\phi_e(X^*)) d\theta$$

$$= \prod_{e=1}^{E} \frac{\alpha_e + n}{\alpha_e + \beta_e + n} > \prod_{e=1}^{E} \sqrt[E]{0.5} = 0.5,$$

and thus

$$\sum_{\mathbf{z}\in\mathcal{Z}} |\{\mathbf{x} : \phi(\mathbf{x}) = \mathbf{z}\}| \prod_{e=1}^{E} P(z_e|\phi_e(X^*))$$

$$\geq |\{\mathbf{x} : \phi(\mathbf{x}) = \mathbf{z}^*\}| \prod_{e=1}^{E} P(z^*_e|\phi_e(X^*)) > 1.$$

The opposite direction follows from the fact that $\forall X' : \sum_{\mathbf{z}\in\mathcal{Z}} \prod_{e=1}^{E} P(z_e|\phi_e(X')) \leq 1$, because $P(z_e|\phi_e(X'))$ is not an approximation, but a true Bayesian probability. Now we have

$$\forall X' : \sum_{\mathbf{x}\in\mathcal{X}} Q_\phi(\mathbf{x}|X') \leq 1$$

$$\Leftrightarrow \forall X' : \sum_{\mathbf{x}\in\mathcal{X}} \prod_{e=1}^{E} P(\phi_e(\mathbf{x})|\phi_e(X')) \leq 1$$

$$\Leftrightarrow \forall X' : \sum_{\mathbf{z}\in\mathcal{Z}} |\{\mathbf{x} : \phi(\mathbf{x}) = \mathbf{z}\}| \prod_{e=1}^{E} P(z_e|\phi_e(X')) \leq 1$$

$$\Leftrightarrow \forall \mathbf{z} : |\{\mathbf{x} : \phi(\mathbf{x}) = \mathbf{z}\}| \leq 1 \Leftrightarrow \phi \text{ is injective.}$$

□