Boosting Multi-Task Weak Learners with Applications to Textual and Social Data

Jean-Baptiste Faddoul, Boris Chidlovskii
Xerox Research Center Europe
Meylan, FRANCE
jean-baptiste.faddoul@xrce.xerox.com
boris.chidlovskii@xrce.xerox.com

Fabien Torre, Remi Gilleron
Lille university, INRIA LNE and LIFL
Lille, FRANCE
fabien.torre@univ-lille3.fr
remi.gilleron@univ-lille3.fr

***Abstract*—Learning multiple related tasks from data simultaneously can improve predictive performance relative to learning these tasks independently. In this paper we propose a novel multi-task learning algorithm called *MT-Adaboost*: it extends Ada boost algorithm [1] to the multi-task setting; it uses as multi-task weak classifier a multi-task decision stump. This allows to learn different dependencies between tasks for different regions of the learning space. Thus, we relax the conventional hypothesis that tasks behave similarly in the whole learning space. Moreover, MT-Adaboost can learn multiple tasks without imposing the constraint of sharing the same label set and/or examples between tasks. A theoretical analysis is derived from the analysis of the original Adaboost. Experiments for multiple tasks over large scale textual data sets with social context (Enron and Tobacco) give rise to very promising results.**

***Keywords*-Multi-task learning; Boosting; Decision trees; Textual and social data.**

I. Introduction

The standard methodology in machine learning is to learn one task at a time when large problems are broken into small, reasonably independent sub-problems that are learned separately and then recombined in a proper way. The multi-task learning attempts to extend this methodology [2]; its goal is to improve the performance of related tasks by learning a model which is able to represent the common knowledge across tasks.

Previous works on multi-task learning were essentially focused on neural networks, k-nearest neighbors [2], and support vector machines [3], [4], where the common knowledge is explicitly expressed as a shared part of the tasks. Similarly, probabilistic models have been adapted to the multi-task setting by introducing shared priors [5], [6].

Most of the existing techniques make some convenience assumptions, like sharing the same label set and/or examples between tasks. Moreover, they are often grounded on the hypothesis that related tasks tend to behave similarly in the whole learning space.

Unfortunately, dissimilar tasks might hurt the performance similarly to introducing noise in data and making global relatedness assumption turns to be too strong in real situations. Moreover this relatedness may show up different degrees or even different signs in different regions of the "learning space". It is therefore important that the multi-task learner determines the relatedness of tasks, learns its different degrees and accommodates the inductive bias accordingly.

In this paper we promote the multi-task learning using the boosting principle in order to achieve the above mentioned goals. It is worth to note that boosting has been already used in multi-task learning for face verification [7]. Following the ideas different from ours, it learns a set of boosted classifiers and is based on a probabilistic model where a multinomial variable indicates how much each boosted classifier contributes to each task. The learning algorithm involves Expectation-Maximization (EM) to learn both the multinomial random variables as well as the classifiers. Dai et al. [8] developed a variation of AdaBoost [1] that can incorporate training data from a different distribution than the test set. Instead of learning multiple models, their approach lowers the weights of data points (from the other distribution) that are not representative for the test set.

In this paper we present a novel multi-task learning approach called *MT-Adaboost*. It extends Adaboost algorithm [1] to the multi-task setting. It uses a new multi-task weak classifier which is a multi-task decision stump (*MT-Stump*). An MT-stump has multiple levels. Each level is a decision stump for one task. Thus an MT-stump defines a partition of the instance space dependent on the tasks. MT-stumps allow to learn the relatedness between tasks in different regions of the instance space. The main contribution of this paper is to relax the previous constraints on sharing labels/examples and global relatedness hypothesis.

We first validate our approach on synthetic data sets. Then we consider two datasets which are large scale textual documents together with social features (Enron and Tobacco). We consider different binary tasks and we show that MT-Adaboost outperforms, for both tasks, state-of-the -art algorithms (SVM and Adaboost).

The rest of the paper is organized as follows. Our approach is presented and analyzed in Section II. Then, Section III explains the experimental settings and reports the evaluation results. Section IV concludes the paper and presents future work.

II. MT-ADABOOST

For sake of clarity, we now consider two supervised classification tasks T_1 and T_2 and we postpone the n-tasks case until the conclusion. Let $\mathcal{X}$ be the instance space, the task T_1 (and similarly for T_2) is defined as follows. D_1 is a distribution over $\mathcal{X}$, let $f_1 : \mathcal{X} \to Y_1$ be a target function, given a sample $S_1 = \{(x, f_1(x)) \mid x \in \mathcal{X}\}$, find an hypothesis function h_1 which minimizes $error(h) = Pr_{x\sim D_1}[h_1(x) \neq f_1(x)]$. We will consider binary classification tasks.

We propose to learn the tasks simultaneously. We formalize the multi-task learning problem as follows. We suppose a distribution D over $\mathcal{X} \times \{1, 2\}$. The intuitive idea is to choose an instance together with a task. The classification task we consider is: given a sample $S_1 \cup S_2$, find an hypothesis $h : \mathcal{X} \to Y_1 \times Y_2$ which minimizes $error(h) = Pr_{<x,i>\sim D}[h_i(x) \neq f_i(x)]$, where $h_i(x)$ is the i-th component of $h(x)$ and $i \in \{1, 2\}$. The error is a combination of errors for the two original tasks. The way errors are combined depends on the distribution D, i.e. on distributions D_1 and D_2 and how the two tasks are related. Our formalization of multi-task problems will allow to adapt Adaboost in Section II-B. But, before, let us present the weak learners we consider.

A. Multi-task Weak Classifiers

We generalize stumps for multi-task problems. Recall that stumps are one-level decision trees, i.e. a stump is defined by a test node and two prediction nodes. Thus, they define very efficient weak classifiers. Moreover, they allow to learn accurate strong classifiers when used in boosting algorithms [9].

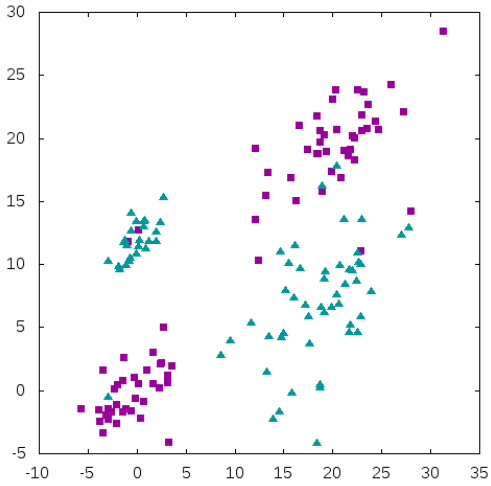
Let us consider the two classification tasks in Figure 1. They are defined over the same two attributes, but each one has different label set, and the samples are disjoint. The two tasks do not seem to be globally related, but in different regions of the space they show up different signs of strong relatedness.

Thus, we introduce multi-task-stumps (*MT-stumps*), which are able to capture the relatedness, even when it differs accross the different regions of the space. An MT-stump for two tasks has two levels. The first level is the root and it is a decision stump for one of the two tasks T_1 or T_2. The second level is located at the two prediction nodes of the first level. For each of the two prediction nodes, it is defined by a decision stump for the other task. An example of MT-stump, where task T_2 is chosen at the root, is shown in Figure 2. We do not give a formal definition, but it is easy to see that an MT-stump for two classification tasks over label sets Y_1 and Y_2 defines a function h from $\mathcal{X}$ into $Y_1 \times Y_2$. It is worth noting that the partition of the input space defined by an MT-stump depends on the two tasks.

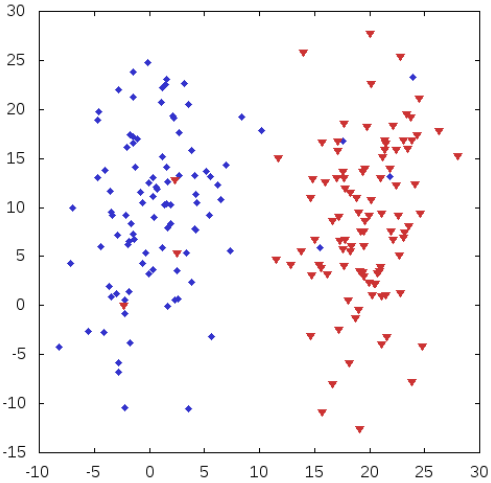
B. Formal MT-Adaboost

We consider the multi-task setting as defined above, with MT-stumps used as weak classifiers. A straightforward adaptation of the original Adaboost to multi-task learning leads to the algorithm MT-Adaboost presented in Figure 3.

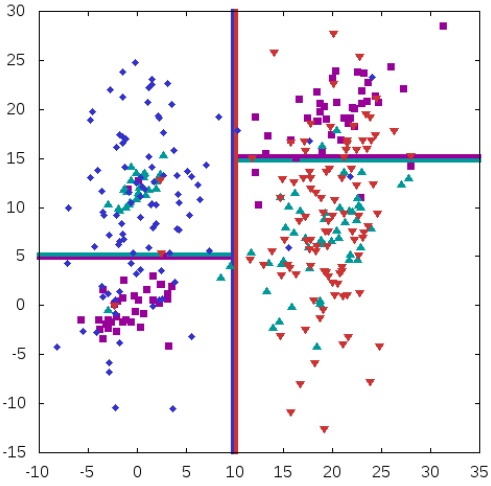
One difference from Adaboost is that we consider two samples S_1 and S_2 for the two tasks T_1 and T_2. We suppose



(a) Sample for the first task.



(b) Sample for the second task.



(c) The partition defined by an MT-stump. The first level of the MT-stump is a decision test for the second task. Its second level contains two decisions tests for the first task.

Fig. 1. Two tasks resolved by an MT-stump.

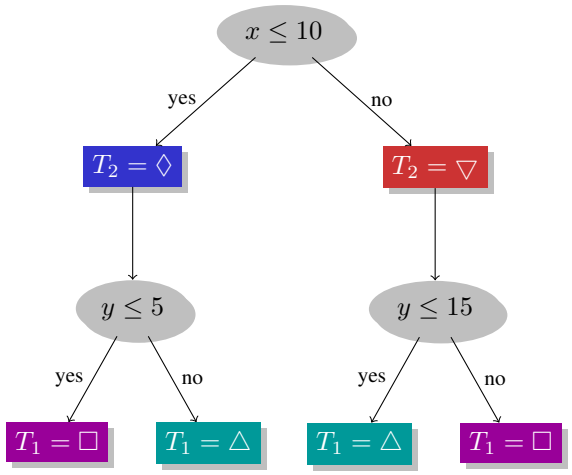


Fig. 2. An MT-stump: a stump for task T_2 at the root, two stumps for task T_1, one defined on the set of instances satisfying $x \leq 10$, the second one defined on the set of instances satisfying $x > 10$. For instance, let us consider an instance $a = (x = 12, y = 11)$, the MT-stump classifies a as $\triangle$ for T_1 and $\bigtriangledown$ for T_2.

that the sample S_i corresponds to examples drawn according to a distribution D over $\mathcal{X} \times \{1, 2\}$ with i as second component. Therefore, we consider, in MT-Adaboost, distributions over $S = S_1 \cup S_2$.

Another difference is that the error is computed in the multi-task setting. Indeed, for each t, an hypothesis h^t is an MT-stump, therefore h^t is a function from $\mathcal{X}$ into $Y_1 \times Y_2$. We recall that the true error of h^t w.r.t. target functions f_1 and f_2 has been defined as $error(h^t) = Pr_{<x,i>\sim D}[h_i^t(x) \neq f_i(x)]$, where $h_i^t(x)$ is the i-th component of $h^t(x)$, $i \in \{1, 2\}$, and D a distribution over $\mathcal{X} \times \{1, 2\}$. Now, given a distribution D^t over $S = S_1 \cup S_2$, the (empirical) error is defined as

$$\texttt{err}(h^t, S, D^t) = \sum_{e \in S : \texttt{pred}(h^t, e) \neq \texttt{class}(e)} D^t(e)$$

where $\texttt{pred}(h^t, e) = h_i^t(e)$ if $e \in S_i$ and $\texttt{class}(e)$ is the label of e in S_i. The (empirical) error $\epsilon^t = \texttt{err}(h^t, S, D^t)$ is the sum of the weighted error of the hypothesis h^t on the two sets S_1 and S_2 w.r.t. D^t.

MT-Adaboost is the original Adaboost adapted to the multi-task setting. Therefore MT-Adaboost inherits all properties of the original Adaboost. For instance, we have proved that, for all D^1 over $S = S_1 \cup S_2$, the training error (empirical error w.r.t. D^1) decreases exponentially with the number of boosting iterations because the training error is at most

$$exp\left(-2\sum_{t=1}^{T}(0.5 - \epsilon^t)^2\right)$$

Now, it remains to define a weak learning algorithm for MT-stumps in the multi-task learning setting, i.e. a learner that provides hypothesis with error strictly less than 0.5.

C. Weak Learning MT-stumps

Let us consider a distribution D' over $S = S_1 \cup S_2$, a weak learning algorithm L must, given as input S and D', output an MT-stump h such that the empirical error $\texttt{err}(h, S, D')$ is strictly less than $\frac{1}{2}$.

First, we show that there always exists an MT-stump satisfying this condition. For this, let us consider an MT-stump, and let us consider the *dual* MT-stump obtained by inverting the predicted class at each prediction node of the original MT-stump. It is easy to show that if an MT-stump has an error ϵ, its dual has the error $1 - \epsilon$. Which means that we can always find an MT-Stump with error strictly inferior to $\frac{1}{2}$.

So it is possible to define a weak learner, called `Exhaustive-MTS-Learner`, as follows: enumerate all possible MT-stumps and keep the one with the lowest error. To enumerate all MT-stumps, the learner loops on the two tasks (each of them can be used at the root of the tree), then it loops on the three stumps to be defined in an MT-stump, loops on the possible features and loops on the possible values for each attribute. Let us denote by a the number of features and by v the maximal number of values, then the number of tests is av, the number of stumps for one task is at most $2av$, the number of MT-stumps with a fixed order on the two tasks is at most $(2av)^3$, and last the number of MT-stumps is at most $2(2av)^3$. Let us note that all MT-stumps can be produced only one time before boosting, and then, at each boosting step, `Exhaustive-MTS-Learner` only evaluates each MT-stump w.r.t. the current distribution.

When `Exhaustive-MTS-Learner` does not apply because the number of the MT-stumps is too large for the algorithm to be efficient, we use randomization. We

Fig. 3. MT-Adaboost

Require:
- T *the number of boosting iterations*,
- S_1 and S_2 *samples for the two tasks*, $S = S_1 \cup S_2$
- `init` *initializes the distribution* D *over* S,
- L *a weak learner that returns an MT-stump given as input* S *and a distribution* D *over* S,
- `class` *returns the label of an example*,
- `pred` *with input an MT-stump, returns the label w.r.t. task* i *for* $e \in S_i$,
- `err` *a function that computes the error of a MT-stump w.r.t.* S *and a distribution* D *over* S.

$D^1 = \texttt{init}(S)$ {*initialize distribution*}
for $t = 1$ to T **do**
  $h^t = L(S, D^t)$ {*train the weak learner*}
  $\epsilon^t = \texttt{err}(h^t, S, D^t)$ {*calculate the error*}
  $\alpha^t = \ln(\frac{1-\epsilon^t}{\epsilon^t})$ {*calculate hypothesis weight*}
  for $e \in S$ **do** {*update the distribution*}
    if $\texttt{class}(e) = \texttt{pred}(h^t, e)$ **then**
      $D^{t+1}(e) = \frac{D^t(e).\exp(-\alpha^t)}{Z^t}$
    else
      $D^{t+1}(e) = \frac{D^t(e).\exp(+\alpha^t)}{Z^t}$
    end if
  end for
end for
return Classifier $h(x) = \texttt{argmax}_{(y_1,y_2)} \sum_{t=1}^{T} \alpha^t h^t(x)$

generate at random n MT-stumps and keep the hypothesis with the lowest error. This algorithm is called `Stochastic-MTS-Learner`. The random generation process of MT-stumps is defined as follows. We first pick up randomly a task, then based on a randomly chosen feature we construct a stump for this task which is the first level in the MT-Stump, see Figure 2. For the second level, we sample a task, then two random stumps are sampled for this task, one will be placed under the first decision leaf of the first level stump and the second under the second leaf, and so on for the other levels. It can be proved that `Stochastic-MTS-Learner` generates with high probability an MT-stump with error inferior to $\frac{1}{2}$. Indeed, we proved that if an MT-stump has an error ϵ, its dual has the error $1-\epsilon$. Therefore, half of the MT-stumps have an error lower than 0.5. Consequently, we fail to obtain a weak hypothesis among n randomly produced MT-stumps with probability less than $(\frac{1}{2})^n$. Thus `Stochastic-MTS-Learner` outputs an MT-stump which is a weak classifier with high probability. Also `Stochastic-MTS-Learner` can be modified such that, if no weak classifier is found among the n MT-stumps, the generation process continues until a weak MT-stump is produced.

Many optimization techniques coming from decision tree induction are possible but will not be discussed in this paper.

III. Experiments

In the following, we describe in detail the datasets, the preprocessing applied on them before reporting our experimental comparisons between MT-Adaboost and two single task learning methods SVM and Adaboost with Stumps.

A. Datasets

Our approach has been validated on one synthetic dataset and two real datasets. The synthetic one is a mixture of $2D$ multivariate Gaussians. It has two binary tasks T_1 and T_2[1]:

- T_1: composed of four Gaussians two for each class, each Gaussian has 50 examples.
- T_2: composed of one Gaussian per class, each Gaussian has 1000 examples.

As shown in Figure 1, T_1 is not linearly separable, whereas T_2 is. A percentage of 1% of noise is introduced for T_1 and 5% for T_2.

Second, we consider two large scale textual datasets namely Enron and Tobacco. They are used for the TREC (Text REtrieval Conference) legal track challenge[2]. Lawsuits involving companies and/or individuals have huge collections of documents varying from hard copy official documents to emails. A group of lawyers are engaged to mine those collections of millions of documents in order to decide which ones are *responsive* for a certain lawsuit. Case mining is costly, time consuming and critical since a single document might have an impact on the lawsuit. This kind of legal document collections is not easily available and even if they were, they would require considerable annotation efforts due to their huge size. To the best of our knowledge two real datasets of this kind are available: Enron and Tobacco, they have been used in the TREC legal track challenge for the last five years.

Enron dataset [3] contains all e-mails sent and received by some 150 accounts of the top management of Enron and spanning a period of several years, the total number of messages is about 250,000. Almost no censorship has been applied to the contents, resulting in a vast variety of subjects ranging from business related topics to strictly personal messages. It is no wonder that the Enron Corpus opened up completely new possibilities to the research [10], [11] and [12].

Tobacco dataset is based on documents released under the tobacco Master Settlement Agreement" (MSA). The MSA settled a range of lawsuits by the Attorneys General of several US states against seven US tobacco organizations (five tobacco companies and two research institutes). The University of California San Francisco (UCSF) Library, with support from the American Legacy Foundation, has created a permanent repository, the Legacy Tobacco Documents Library (LTDL), for tobacco documents [10]. The collection is based on a snapshot, generated between November 2005 and January 2006, of the MSA sub-collection of the LTDL. After reformatting OCR and metadata; and filtering documents with formatting problems; the collection consists of 6,910,192 document records in the form of XML elements [13].

B. Preprocessing

Enron: Annotations of the Enron dataset come from two different sources. The first is from the Department Of Justice of the United States DOJ [4], which has published a list of responsive emails used in the trials against the two CEO's of Enron. This set along with a manually annotated non-responsive emails constitute a binary classification task, *Responsive* Vs. *Non-responsive*, with total of 372 emails. The second annotated set comes from students of Berkeley University. Emails in this set are annotated by topic, for an average of 250 emails per topic. Five topics are used inn our experiments: *Business*, *Legal*, *Influence*, *Arrangement* and *Personal*. Since the two sets are small, and they share a common knowledge (ex. a personal email is not likely to be a responsive email), so learning them simultaneously would be advantageous. It should be noted, that those two sets are disjoint, i.e., there are no examples provided with both annotations. Since the task of discovering responsive documents is of interest for litigation applications, we consider it always in the experiments and we try to learn it with another task that comes from the topic annotated set: *Legal* Vs. *Personal*, *Business* Vs. *Arrangement* and *Influence* Vs. *Personal*.

Enron documents are multi-modal, they have textual information and social implicit information come from the underlying social network which is composed users connected

[1]Figure 1 shows a simplified example of this dataset.

[2]http://trec-legal.umiacs.umd.edu/

[3]http://www.cs.cmu.edu/~enron/

[4]http://www.usdoj.gov/enron/

through their emails communications. Textual features we use are 2200, each of which represents a cluster of semantically similar words, e.g., trip, journey and travel. The value of such a feature is the number of occurrences words of its cluster occurred in a given document.

In addition to textual features, we extract a set of commonly used features to represent key properties of actors (for more details please see [14]). To translate the properties of actors to properties of e-mails, set of 37 features is constructed to represent each message. A message is represented by three sets of 12 features, the first is the properties of the sender, the second is the average of the properties of all receivers and the third is the properties of the most prominent receiver (i.e. with the highest social score). The last feature is the number of receivers.

Tobacco: Each document is annotated by multiple labels. In total there are 70 possible labels with an average of 10.000 documents per label. To construct multi-task setting from multi-label, we chose 4 labels and construct from their documents two binary tasks:

- Task1: *Smoking and lung cancer Vs. Marketing strategy.*
- Task2: *Nicotine and addiction Vs. Advertisements.*

It should be noted that each chosen document has one and only one of the four labels, i.e., there is no intersection between tasks' documents. For Tobacco, each document is represented by bag-of-words features. We have 40200 distinct words in the chosen collection, for each word there is a feature which has value 1 if the word exists in the document and 0 otherwise.

C. Algorithms and settings

For our algorithm, we define $\texttt{init}(S)$, where $S = S_1 \cup S_2$, as the distribution D^1 such that, for each $e \in S$, $\mathcal{D}^1(e) = \frac{1}{4.|S_i^k|}$ where S_i^k contains examples of class (k) in task T_i. This corresponds to sampling with probability $\frac{1}{2}$ one of the two tasks T_i, then sampling with probability $\frac{1}{2}$ one of the two classes (k) of the selected task and finally sampling uniformly at random in the set S_i^k.

`Stochastic-MTS-Learner` is used in all the experiments as the weak learner for MT-Adaboost, with $n = 1000$ (n is the number of randomly sampled MT-stumps at each step of boosting).

To evaluate the performance of MT-Adaboost we compare it with two state of the art Single Task Learning methods, Support Vector Machines SVM and Adaboost. To run SVM we use *LibLinear* software[5]. Which is an SVM implementation with linear kernel. Choosing linear kernel for our datasets was not arbitrary, but it was motivated by a practice in the literature to use them when dealing with large scale data sets where we have already huge number of dimensions that makes the data likely to be linearly separable without the need to project it to higher dimensional spaces, that would complicate the calculations, and does not bring a significance improvement. Adaboost is used with decisions stumps as the weak learner since they are the closest to our algorithm.

[5]http://www.csie.ntu.edu.tw/~cjlin/liblinear/

	Error rate	StdDev
T_1 SVM	9.698	1.06
T_1 Adaboost	9.999	1.87
T_1 MTAdaboost	**2.299**	0.44
T_2 SVM	8.674	1.33
T_2 Adaboost	8.193	1.39
T_2 MTAdaboost	**8.125**	0.96

TABLE I
ERROR RATE RESULTS FOR THE TWO SYNTHETIC TASKS, T_1 AND T_2

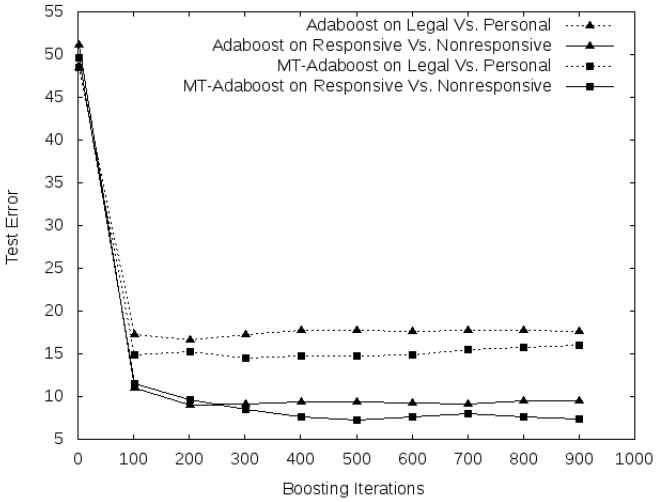


Fig. 4. Test error curves for Adaboost and MT-Adaboost on two Enron tasks.

Both MT-Adaboost and Adaboost has only one parameter to tune which is the number of boosting iterations T. By doing Cross Validation on each pair dataset / algorithm, we chose the value at which the error tends to be stable. For both algorithms we chose, $T = 25$ for Gaussian data, $T = 300$ for Enron and $T = 150$ for Tobacco. In the next section we will show empirically that the test (generalization) error converges to a minimum value with a small variance, after certain number of iterations, which means that empirically the algorithm does not overfit even when using large number of iterations. In [1] similar behavior has been observed.

D. Results

We measure the error rate (percentage of misclassified examples) and the standard deviation on the test set.

Gaussian data: As shown in Table I, MT-Adaboost outperforms Adaboost and SVM on both tasks with a significant improvement on T_1 which is more difficult than T_2.

Enron: MT-Adaboost has got lower error rate than SVM and Adaboost on Enron tasks. Results are shown in Tables II, III, and IV In Figure 4 we show empirically that MT-Adaboost reduces the test error more than Adaboost on both tasks, and as Adaboost, MT-Adaboost's error becomes stable after a certain number of iterations, which means that it is also robust against over-fitting.

Tobacco: Similarly, MT-Adaboost outperforms Adaboost

	Error rate	StdDev
T_1 SVM	19.835	1.96
T_1 Adaboost	18.059	0.87
T_1 MTAdaboost	**17.294**	0.44
T_2 SVM	18.66	1.21
T_2 Adaboost	13.622	1.09
T_2 MTAdaboost	**8.757**	1.46

TABLE II
TWO ENRON TASKS. T_1: LEGAL VS PERSONAL, T_2: RESPONSIVE VS NON RESPONSIVE

	Error rate	StdDev
T_1 SVM	28.644	0.86
T_1 Adaboost	27.556	0.83
T_1 MTAdaboost	**24.183**	1.26
T_2 SVM	18.66	1.21
T_2 Adaboost	13.622	1.09
T_2 MTAdaboost	**8.108**	0.54

TABLE III
TWO ENRON TASKS. T_1: BUSINESS VS ARRANGEMENT , T_2: RESPONSIVE VS NON RESPONSIVE

and SVM for the first task of Tobacco, and gives comparable error rate for the second task. Results shown in Table V.

IV. MULTI-TASK LEARNING

We now discuss the multi-task case with over 2 tasks. It should be noted that MT-stumps with n levels can not be considered for complexity issues when n is large. Thus, we consider MT-stumps with two levels as weak hypotheses. Such a two level MT-stump abstains for tasks not appearing in the MT-stump. Consequently, we have considered the original Adaboost algorithm for weak hypotheses that abstain proposed by Freund and Schapire in [15]. We have defined a weak learner searching for a MT-stump level by level. While preliminary experiments are very promising, further investigations are required. Indeed, a more thorough theoretical study of MT-AdaBoost with hypotheses that abstain is needed. This is because different choices for updating weights are possible depending on how the error for hypotheses that abstain is defined. Also, we have to prove that our weak learner can find an hypothesis with an error rate lower than 0.5. Last, we must design more experiments.

	Error rate	StdDev
T_1 SVM	16.644	0.86
T_1 Adaboost	14.143	1.15
T_1 MTAdaboost	**13.429**	1.03
T_2 SVM	18.66	1.21
T_2 Adaboost	13.838	0.7
T_2 MTAdaboost	**8.649**	1.25

TABLE IV
TWO ENRON TASKS. T_1: INFLUENCE VS PERSONAL , T_2: RESPONSIVE VS NON RESPONSIVE

	Error rate	StdDev
T_1 SVM	14.6	0.5
T_1 Adaboost	13.45	0.05
T_1 MTAdaboost	**12.45**	0.05
T_2 SVM	15.85	0.15
T_2 Adaboost	**15.92**	0.55
T_2 MTAdaboost	15.95	0.45

TABLE V
TWO TOBACCO TASKS. T_1: SMOKING AND LUNG CANCER VS. MARKETING STRATEGY T_2: NICOTINE AND ADDICTION VS. ADVERTISEMENTS

ACKNOWLEDGMENT

This work was partially supported by Ministry of Higher Education and Research, Nord-Pas de Calais Regional Council, FEDER through the *CPER 2007-2013*, the FRAGRANCE project and the LAMPADA project co-funded by the *French Association on Research - ANR*.

REFERENCES

[1] Y. Freund and R. Schapire, "A short introduction to boosting," *Japonese Society for Artificial Intelligence*, vol. 14, no. 5, pp. 771–780, 1999.

[2] R. Caruana, "Multitask learning," in *Machine Learning*, vol. 28, 1997, pp. 41–75.

[3] T. Evgeniou and M. Pontil, "Regularized multi–task learning," in *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2004, pp. 109–117.

[4] V. W. Zheng, S. J. Pan, Q. Yang, and J. J. Pan, "Transferring multi-device localization models using latent multi-task learning," in *AAAI'08: Proceedings of the 23rd national conference on Artificial intelligence*. AAAI Press, 2008, pp. 1427–1432.

[5] J.-M. Renders, É. Gaussier, C. Goutte, F. Pacull, and G. Csurka, "Categorization in multiple category systems," in *ICML'06: Proceedings of the 23rd International Conference on Machine Learning*, 2006, pp. 745–752.

[6] Q. Liu, X. Liao, H. L. Carin, J. R. Stack, and L. Carin, "Semisupervised multitask learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 6, pp. 1074–1086, 2009.

[7] X. Wang, C. Zhang, and Z. Zhang, "Boosted multi-task learning for face verification with applications to web image and video search," in *CVPR'09: Proceedings of the 22nd IEEE conference on Computer Vision and Pattern Recognition*, 2009, pp. 142–149.

[8] W. Dai, Q. Yang, G. R. Xue, and Y. Yu, "Boosting for transfer learning," in *ICML '07: Proceedings of the 24th international conference on Machine learning*. New York, NY, USA: ACM, 2007, pp. 193–200.

[9] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *ICML'96: Proceedings of the 13th International Conference on Machine Learning*, 1996, pp. 148–156.

[10] A. McCallum, X. Wang, and A. C. Emmanuel, "Topic and role discovery in social networks with experiments on enron and academic email," *J. Artif. Int. Res.*, vol. 30, no. 1, pp. 249–272, 2007.

[11] J. Shetty, "Discovering important nodes through graph entropy: The case of enron email database," in *KDD, Proceedings of the 3rd international workshop on Link discovery*, 2005, pp. 74–81.

[12] R. Bekkerman, A. Mccallum, and G. Huang, "Automatic categorization of email into folders: Benchmark experiments on enron and sri corpora," in *Technical Report, Computer Science department, IR-418*, pp. 4–6.

[13] D. Lewis and D. D. LewisConsulting, "Building a test collection for complex document information processing," in *ACM SIGIR'06: Proceedings of the 29th Annual International ACM Conference on Research and Development in Information Retrieval*, 2006, pp. 665–666.

[14] M. Hovelynck and B. Chidlovskii, "Multi-modality in one-class classification," in *WWW '10: Proceedings of the 19th international conference on World wide web*, 2010, pp. 441–450.

[15] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, Aug. 1997.