Results from a Deep Learning and Semi-Supervised Feature Learning Competition

Anonymous Author(s)
Affiliation
Address
email

Abstract

We present results from a recent large-scale semi-supervised feature learning competition, which attracted twenty-nine teams and 238 total submissions. The learning task was drawn from a real world task in malicious url classification. This was a large scale binary classification task, with a sparse feature space of one million features, and training data sets of 50,000 labeled examples and one million unlabeled examples. Participants were to learn a method of projecting the given high dimensional training and test data into a space of no more than 100 features that would be useful for supervised prediction.

We expected to find that the best performing methods would make extensive use of the unlabeled data in addition to the labeled training data, and that the best methods would utilize deep learning or other sophisticated feature learning approaches. Surprisingly this turned out not to be the case: relatively straight-forward supervised learning approaches using random forests out-performed all other methods.

1 Introduction

Deep learning has recently been one of the most active and exciting areas of current machine learning research. However, some results suggest that deep learning methods may be overkill for common classification and learning tasks. One striking example of this was reported by Coates *et al.*, in which it was found that a relatively simple k-means clustering method out-performed more complex deep learning methods on CIFAR-10 and NORB data sets for image classification [6]. Another example is by Li, who found that boosted decision trees surpass deep learning on a version of the MNIST data set that had been artificially hardened to favor deep learning approaches [11]. Finally, it is a common piece of "folk wisdom" in the machine learning community that tuning deep learning methods requires significant skill and experience, making them difficult to apply in practice.

In light of these observations, we wished to gain more knowledge about the practical utility of deep learning and semi-supervised feature learning methods for real-world problems. We organized an open competition[1] around deep learning and semi-supervised feature learning to see how well various methods fared when applied by many different teams and researchers. The competition was hosted on `kaggle.com`, and was publicized by announcements on major blogs, mailing lists, and by direct emails to many of the top research groups working in deep learning. The competition attracted twenty-nine teams and 238 total submissions over a period of three weeks. Competitors were motivated both by the promise of acknowledgement in this summary paper and by a prize of 500 USD for the first place team.

The results of this competition were surprising to us. The top performing method used relatively straight-forward variations of random forests and used the unlabeled data only to inform feature se-

[1] http://www.kaggle.com/c/SemiSupervisedFeatureLearning

lection. Methods such as bagging, boosting, and supervised linear predictors all did well. Some explicitly semi-supervised feature learning methods also did well, including k-means clustering where the learned features were used to augment the original feature space, kernel projections using unlabeled data, and transfer learning. More explicit feature learning methods such as deep learning, matrix factorization, pure clustering, and feature selection all fared generally more poorly. In general, methods that attempted to make more use of the unlabeled data fared worse than those that made less use of it, even though the amount of unlabeled data was twenty times greater.

The remainder of this report is organized as follows. The details of the experimental setup, including data set generation, competitor tasks, and evaluation methods are given in Section 2. We briefly review related studies and competitions in Section 3, and then give detailed reporting on the methods applied and their results in Section 4. Our discussion in the final section explores questions around the performance of deep learning methods on this task.

2 Experimental Setup

The learning task for this competition was drawn from the `url-reputation` data set by Ma *et al.* [12]. Each example in this data set corresponds to a url, which is to be classified as a `malicious` positive or a `non-malicious` negative. The data contains both sparse boolean features, largely drawn from anonymized lexical url content, and real-valued features from host characteristics [12]. We chose this data set because it is publicly available and has a large number of examples and features (for academic standards). Additionally, the data was from a real-world classification task and was collected in a time-ordered manner. These qualities mirror those that are found in practice in many large-scale deployed classification systems.

The main task in this competition was to learn a feature space that was both compact and informative for classification. The procedures for participants were as follows:

- **Learn 100 Features.** The task was to learn a useful representation of at most 100 features. Contestants were allowed to use or ignore any of the labeled and unlabeled data for feature learning. Any method of feature learning was allowed, including unsupervised methods, semi-supervised methods, fully supervised methods, transfer learning, deep learning, feature selection, sparse coding, and matrix factorization.
- **Transform the Data.** Once a feature representation had been learned, it was used to transform both the 50,000 training data examples and the 50,000 test data examples into the new feature space.
- **Train a Model on the Transformed Data.** To standardize training, contestants used the widely available `libsvm` package [5] to train a model on the transformed training data with the provided training labels. Parameter settings for the linear SVM were fixed, with $C = 1$. (Some contestants used the faster `liblinear` package [7] with equivalent results.) The goal of this was to provide a fair comparison across feature learning methods with similar number of features. However, note that there would have been no benefit for a contest to try and cheat by using a more sophisticated learning algorithm at this stage, as the resulting prediction could have been used as a new learned feature.
- **Evaluate Effectiveness.** The trained model was then applied to the transformed test data. Contestants submitted both the prediction and the transformed features for each test example. This way, results could be verified using cross validation with the learned features.

2.1 Data Sets

The `url-reputation` data set contains roughly 2.4M labeled examples, divided into 120 total days. Data from the first 100 days was used to create the unlabeled data set, by picking 1M examples without replacement and randomizing the order. Class labels were removed and replaced with a dummy value of 0 rather than +1 or -1.

The training data was selected from the next 10 days of data. We picked 50,000 examples without replacement and randomized the order. We also injected class label noise into the data, to help obscure the source of the data and to make the classification problem both more difficult and more

realistic. This was done by randomly flipping 10% of the labeled negatives to positives, and randomly flipping 3% of the positives to negatives. We confirmed that the injected class label noise did indeed degrade performance of a simple linear SVM, compared to the performance using the training labels without the injected noise, and hoped that this would help to increase the benefit of using the unlabeled data.

The test data was selected from the final 10 days of data. We picked 50,000 examples without replacement and randomized the order. The labels for this data had no added noise and were kept hidden from the contestants until the end of of the competition.

In order to obscure the source of this data set for the public competition, we first projected the 3.2M original features down to a space of 1M features using the feature hashing technique [16]. We tested and confirmed that this did not significantly alter predictive performance of a simple linear SVM. We then sampled to change the class distribution from roughly balanced class distribution to roughly 10% positive and 90% negative.

2.2 Evaluation

The primary evaluation metric that we report in this paper is AUC Loss, computed as (1-AUC) where AUC is the Area under the ROC curve [8]. This metric has a statistical interpretation as the probability that a randomly sampled positive example will be wrongly predicted as being more likely negative than a randomly sampled negative example.

Each team was permitted to submit up to two entries per day during the three week competition. A public leader-board showed the performance of the best entry for each team, using AUC as the evaluation metric on 30% of the test data. The remaining 70% of the test data was kept for final private evaluation at the end of the competition, to ensure that teams were not able to "game" the leader-board for information.

3 Related Studies

There have been several related studies and competitions in recent years to evaluate the effectiveness of various forms of deep learning and feature learning. However, this competition is unique in that it aims to solve a single classification task, is relatively large-scale, and includes both a large amount of labeled data and a very large amount of unlabeled data. We believe this competition to be representative of a range of real-world tasks for deployable systems, especially for classification of data on the web where unlabeled data abounds but (potentially noisy) labeled data is also available.

The most closely related recent competition was the ICML 2011 unsupervised and transfer learning challenge [10]. This differed from our competition in several respects. First, the task was set up such that feature learning was done in a completely unsupervised fashion, while our competition provided both labeled and unlabeled data for feature learning. Second, in the prior challenge the classification task for transfer learning was not known in advance, while in our competition the binary classification task was known. Thus, the prior challenge posed to evaluate feature learning with little fore-knowledge of the task to be solved, while our competition sought methods that worked well when there was a specific learning problem at hand.

Two other important related studies were mentioned in the introduction: experimental comparisons between deep learning and simpler learning methods by Coates *et al.* [6] and by Li [11].

4 Competition Results

In this section, we detail the results of the competition. The complete distribution of scores from all 238 submissions are given in Figure 1. We selected a subset of 66 total submissions to report on in more detail in Table 1 and Table 2, choosing representative submissions from each of the twenty-nine teams and attempting to cover all of the methods that were employed. Most teams reported details of the methods used; where explicit reports were not given we have made good-faith efforts to infer the methods from file names and submission notes.

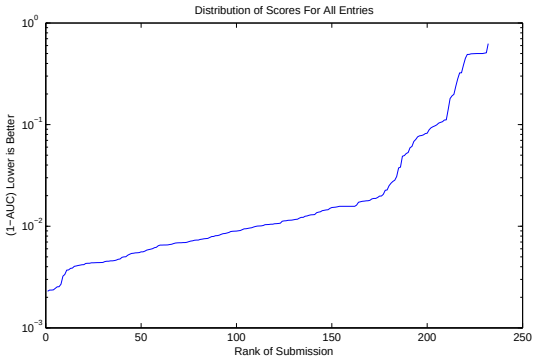
ID	(1-AUC)	TEAM	NOTES
1	0.0023 (0.0018-0.0030)	Hamner/Cukierski	9 supervised random forests, using supervised and unsupervised feature selection
2	0.0024 (0.0019-0.0029)	Hamner/Cukierski	5 supervised random forests, using supervised feature selection
3	0.0032 (0.0026-0.0040)	Hamner/Cukierski	one supervised random forest, using supervised feature selection
4	0.0037 (0.0034-0.0049)	Glenngariff	supervised L1 logistic regression on original features plus 200 k-means features
5	0.0040 (0.0038-0.0053)	Ekip	RankBoost using grid learners as weak learners using random kernel-mapped features
6	0.0041 (0.0036-0.0054)	Glenngariff	supervised L2 logistic regression on original features
7	0.0043 (0.0042-0.0058)	Eyert	supervised neural network with one hidden layer of 100 nodes, linear activation
8	0.0044 (0.0037-0.0056)	Glenngariff	supervised logistic regression with strong L1 regularization
9	0.0044 (0.0042-0.0059)	Eyert	supervised neural network with one hidden layer of 100 nodes
10	0.0044 (0.0038-0.0052)	Chenguang	bagging over supervised logistic regression
11	0.0044	Ekip	RankBoost using grid learners as weak learners using orthogonal kernel-mapped features
12	0.0044	Vincent	
13	0.0045	Ekip	RankBoost using decision stumps as weak learners using orthogonal kernel-mapped features
14	0.0046	Newman1	bagging over supervised logistic regression
15	0.0046	New Idea	bagging over supervised linear classifier
16	0.0048	Glenngariff	tfidf reweighting, sparse L1 regularization for linear model
17	0.0049	Falloutboy	alternating structural optimization (transfer learning plus SVD) 200 auxiliary classifiers
18	0.0050	Vsh	semi-supervised label propagation using random forests
19	0.0050	Falloutboy	alternating structural optimization (transfer learning plus SVD) 300 auxiliary classifiers
20	0.0054	Glenngariff	linear ranker plus graph-kernel smoothing using unlabeled data
21	0.0054	New Idea	bagging over supervised lasso learner
22	0.0055	Hamner/Cukierski	semi-supervised self training using one random forest
23	0.0059	New Idea	bagging
24	0.0060	Falloutboy	same as **17**, but without the auxiliary learners and adding in unlabeled test data
25	0.0061	Boom	
26	0.0062	Glenngariff	strong L1 regularization for feature pre-selection used to learn linear model
27	0.0064	Falloutboy	Gradient Boosted Regression Trees (GBRT) trained on 69 selected dense features
28	0.0067	Pongo	stochastic k-mediods on unlabeled data with k=1000; RBF kernel mapping; SVD to 100 features
29	0.0068	Eyert	supervised neural network with one hidden layer of 100 nodes
30	0.0069	Pongo	stochastic k-mediods on unlabeled data with k=1000; RBF kernel mapping; SVD to 100 features
31	0.0070	Chenguang	supervised logistic regression
32	0.0072	Pongo	stochastic k-mediods on unlabeled data with k=1000; RBF kernel mapping; SVD to 100 features
33	0.0073	Cobb	
34	0.0074	Hippocrates	lasso plus SVD
35	0.0075	Pongo	stochastic k-mediods on unlabeled data with k=1000; RBF kernel mapping; SVD to 100 features

Table 1: **Competition Results, Part 1.** These are results for a representative subset of methods and approaches applied during the competition. 95% confidence intervals are shown for top results.

ID	(1-AUC)	TEAM	NOTES
36	0.0085	Chenguang	minibatch k-means plus PCA, replace last two features with supervised linear models
37	0.0086	Deepblue	supervised linear regression
38	0.0089	Chenguang	minibatch k-means to 500 features for labeled and unlabeled data; PCA to 100 features
39	0.0089	Vsh	semi-supervised label propagation using random forests
40	0.0090	Swedish Chef	SVD on real valued features, PCA on binary features
41	0.0091	Falloutboy	alternating structural optimization (transfer learning plus SVD) 400 auxiliary classifiers and GBRT training
42	0.0092	New Idea	use 99 high variance features plus lasso
43	0.0095	Eyert	supervised neural network with one hidden layer of 100 nodes
44	0.0101	Ekip	similarity function was learned with a neural network with 2 layers directly on sparse data
45	0.0104	Busi2012	random forest on test data
46	0.0107	TeamSMRT	feature selection based on spearman rank correlation plus variance
47	0.0113	image_doctor	mini batch k-means
48	0.0113	Benchmark	mini batch k-means
49	0.0114	Clueless	
50	0.0115	Clueless	
51	0.0126	Hippocrates	100 high variance features
52	0.0128	Jeremy Howard	
53	0.0129	TeamSMRT	100 high variance features
54	0.0130	Chris De Vries	kmeans++
55	0.0139	Hippocrates Team	principle component analysis
56	0.0153	TeamSMRT	top 100 features with largest linear correlation to labeled data
57	0.0172	image_doctor	mini-batch k-means
58	0.0226	TeamSMRT	top 100 features with most non-zero entries in the test set
59	0.0250	Glenngariff	random projections
60	0.0489	IDEAL	deep learning (details unspecified)
61	0.0682	Clueless	self organizing maps; unsupervised
62	0.0715	Clueless	self organizing maps; unsupervised
63	0.0776	IDEAL	deep learning (details unspecified)
64	0.0822	Chris De Vries	k-means clustering data weighted with Pearson correlation
65	0.0889	Mblum	
66	0.1035	Ogrisel	combination of PCA and k-means, using tf-idf weighting

Table 2: **Competition Results, Part 2.** These are results for a representative subset of methods and approaches applied during the competition.

Figure 1: **Distribution of (1-AUC) Results Across All Submissions.** This graph shows the (1-AUC) results for each of the 238 submitted results, in ranked order from best (left) to worst (right). Note the log scale on the y-axis.



Because many teams employed different methods for different submissions, we will discuss the results on a per-method basis rather than a per-team basis. For brevity, we refer to submissions by the id number **N** given in Table 1 and Table 2. For example, **2** refers to the submission by team Hamner/Cukierski using five supervised random forests. (Note that id number is not equivalent to the absolute rank of the submission; see Figure 1 to compare ranks for different 1-AUC scores.)

4.1 Random Forests

The first-place team of Benjamin Hamner and William Cukierski employed several variants of random forests [4] for the top results in **1** and **2**, training on labeled data only. (Confidence intervals for these results overlap with each other, but do not overlap with **4**.) These submissions used multiple random forests, each trained on different subsets of the feature space selected with different supervised and un-supervised feature selection methods which deliberately avoided including only the most informative features. Cukierski suggests that this acted as a "poor man's regularization" and protected against over-fitting. Random forests without this form of feature selection were less successful in **45**.

4.2 Clustering

Clustering remains a popular approach, as seen by the large number of submissions using k-means and its variants. The most successful of these was by Tanguy Urvoy of team Glennfariff in **4**, in which 200 k-means features were learned on the unlabeled and labeled data used to augment the original feature space to train a sparse logistic regression model. Approaches that used clustering to learn a feature space of 100 features directly using unsupervised clustering via k-means (or k-mediods used by Fabio Aiolli of Pongo) were less successful, as shown in **28**, **30**, **32**, **35**, **36**, **38**, **47**, **48**, **54**, **57**, **64**, and **66**.

4.3 Boosting

Boosted trees have been shown to out-perform deep learning even on very hard data sets [11]. In this competition, Artem Sokolov of the Ekip team used boosting to achieve strong results. Best results with this approach used grid-based learners (decision trees of depth two) in conjunction with kernel-mapped features in **5** and **11**. Decision stumps were used in **13**.

4.4 Random Projections

The approach of the Ekip team used an initial kernel mapping step, sampling 5000 unlabeled examples as a basis and projecting each example into the space defined by the kernel similarity with each of the basis examples [9]. They then used random projections to reduce this space to 100 dimensions using a method from [1]. This gave strong results when combined with boosting. Classical random projections were tried in **59**, but were not especially effective.

4.5 Bagging

The bagging approach takes multiple random subsets of the training data, and trains one supervised model with each [3]. This approach was successful as a natural way to do supervised feature learning, and appeared to reduce the adverse effect of class label noise. Methods **1** and **2** can be seen as bagging variants with random forests. More classical bagging was done with linear classifiers in other submissions, including Chenguang Zhu's bagged logistic regression approach **10**, as well as **14**, **15**, **21**, and **23**.

4.6 Supervised Linear Classifiers

Supervised linear classifiers worked well on this data, even with no modifications. For this form of approach, submissions tended to include a single "feature", which was the output of the supervised classifier. Variants included logistic regression and linear support vector machines. Some teams used L1 (or lasso) regularization to encourage sparsity, as in **8**, **16**, and **26**. Others used L2 regularization to smooth paramters, as in **6**.

4.7 Deep Learning

The most successful deep learning method in this competition was a purely supervised neural network with a single hidden layer, using linear activation thresholds in **7** by Le Hai Son of the Eyert team. Variants of this approach were also used in **9**, **29**, and **43**, perhaps illustrating the difficulty of tuning such approaches. Team IDEAL applied deep learning methods in **60** and **63**; this team remained anonymous and did not provide further details. Team Clueless applied Self-Organizing Maps, using a dual-layer torodial model with the second layer trained in a supervised manner in **49**, with variants of this approach in **50**, **61**, and **62**.

4.8 Transfer Learning

In transfer learning, auxiliary models are trained using the values of specific features as labels, and the outputs of these models are used as features. Peter Prettenhofer applied this approach using Alternating Structural Optimization [2] in two stages, first learning 200 auxiliary models [13] on features that had been selected using a strongly L1 regularized linear model, and then projecting the 200-dimensional representation down to 100 dimensions using SVD in **17** and **24**.

4.9 Prediction Smoothing and Label Propogation

Approach **20** made use of the unlabeled data by performing graph-kernel smoothing as a prediction regularizer [14]. A semi-supervised label propagation approach using random forests was used by Vivek Sharma in **18** and **36**, in which a supervised random forest (trained using the FEST package[2]) repeatedly labeled unlabeled data, and those unlabeled data points that were labeled with high confidence were added to the set of training data to re-train the supervised learner. A single iteration variant of this approach was also used in **22**, using a trained random forest to label all unlabeled data and then re-training on the entire labeled and artificially labeled data.

4.10 Matrix Factorization

A number of submissions used matrix factorization methods to reduce dimensionality as part of a chain of methods, first using clustering or transfer learning to reduce to between 200 and 1000 features and then using SVD or PCA to reduce to 100. This approach was used in **17**, **24**, **28**, **30**, **32**, **35**, **36**, **38**, and **66**. A small number of submissions used matrix factorization in isolation, including Christopher Hefele's approach **40** as team Swedish Chef and **51** by Hippocrates Team.

4.11 Feature Selection Methods

A number of teams attempted to reduce the dimensionality of the space to 100 features simply by picking informative features from the original feature space. Various methods included picking

[2] http://www.cs.cornell.edu/∼nk/fest/

features based on variance in **42**, **51**, and **53**, using correlation with the known labels in **46** and **56**, and features with high rates of non-zero entries **58**.

5 Discussion

How well did deep learning methods perform in this competition? This depends to a degree on what we mean by "deep".

John Langford wrote an insightful blog post[3] in 2006, classifying decision trees as deep learners because of their ability to learn complex decision surfaces from simple input features. In this light, we can view the success of random forests as a win for deep learning. Similarly, Langford classified feature learning methods as inherently "deep". The strong results from the Ekip team, using kernel mappings and random projections to learn a feature space score a win here as well.

If we view deep learning as strictly Restricted Boltzman Machines and Autoencoders, then we do not have evidence that these methods did well. The source of this lack of results is currently unclear. It may be that these methods were simply not tried (other than by team IDEAL). Cukierski posits that the scale of the unlabeled data set may have been prohibitive for such complex methods within the timeframe of the competition. There is some evidence for this: only 23% of users who downloaded the competition data submitted a result.

It is also possible that there was a self-selection bias in the reporting from particioants – teams with lower ranking results tended to give fewer or no details about their methods. The full data sets have now been released on the competition website;[4] we hope that the complete range of deep learning methods will be tested on this task and the results made public for comparison.

It is worthwhile to observe that open source software for this set of deep learning methods is behind in development compared to other learning methods such as SVM's. It is difficult for a non-expert to locate RBM or Autoencoder software that is freely available, is easy to use, and works at scale. We would like to encourage the community to work at making general purpose deep learning software available, so that these methods can be a ready tool for practical application.

Finally, we do have clear evidence that some obviously "non-deep" learning methods did do well. Supervised linear models gave strong performance, despite their simplicity and despite making no use of the vast amount of unlabeled data provided.

Tanguy Urvoy of team Glengariff conjectures that the success of simple supervised methods is because the classification task is fixed in advance. Invoking Vapnik's famous dictum [15], Urvoy notes that feature learning for a single classification task is solving a problem more general than the one that needs to be solved. His argument is that the best "feature space" in this case consists of a single feature showing the correct class label. We believe that Glengarriff's best submission, which combined simple supervised learning with additional features learned via unsupervised clustering. shows a practical approach in this situation. Semi-supervised feature learning may be best used not to replace the original feature space for fixed classification tasks, but to augment it.

Acknowledgments

We acknowledge and thank the `kaggle.com` team for their assistance in administering this competition, which was especially appreciated given the tight deadline in which we worked.

We would also like to take this opportunity to acknowledge and thank the participants in this competition. These are: Fabio Aiolli of `Pongo`, Will Cukierski, Chris De Vries, Benjamin Hamner, Christopher Hefele of `Swedish Chef`, Jeremy Howard of `Kaggle`, Peter Prettenhofer of `Falloutboy`, Matt Prior of `image_doctor`, Vivek Sharma of `Vsh`, Artern Solokov of `Ekip`, Le Hai Son of `Eyert`, Tanguy Urvoy of `Glengarriff`, and Chenguang Zhu. Also competing were the following anonymous teams: `newman1`, `Vincent`, `New Idea`, `Cobb`, `Hippocrates Team`, `deepblue`, `Busi2012`, `TeamSMRT`, `Aniket`, `george`, `mblum`, and `ogrisel`.

[3] http://hunch.net/?p=219

[4] http://www.kaggle.com/c/SemiSupervisedFeatureLearning

References

[1] D. Achlioptas. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *J. Comput. Syst. Sci.*, 66, June 2003.

[2] R. K. Ando and T. Zhang. A Framework for Learning Predictive Structures from Multiple Tasks and Unlabeled Data. *J. Mach. Learn. Res.*, 6, 2005.

[3] L. Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[4] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[5] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2, 2011.

[6] A. Coates, H. Lee, and A. Ng. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the 14th International Conference on Articial Intelligence and Statistics (AISTATS)*, 2011.

[7] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[8] T. Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[9] M. florina Balcan, A. Blum, and S. Vempala. Kernels as features: On kernels, margins, and low-dimensional mappings. In *In 15th International Conference on Algorithmic Learning Theory (ALT 04*, pages 79–94, 2004.

[10] I. Guyon, G. Dror, V. Lemaire, G. Taylor, and D. Aha. Unsupervised and transfer learning challenge.

[11] P. Li. Robust logitboost and adaptive base class (abc) logitboost. In *Conference on Uncertainty in Artificial Intelligence*, 2010.

[12] J. Ma, L. Saul, S. Savage, and G. Voelker. Identifying suspicious urls: an application of large-scale online learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, 2009.

[13] P. Prettenhofer and B. Stein. Cross-Lingual adaptation using structural correspondence learning. *ACM Trans. Intell. Syst. Technol.*, 3(1), Oct. 2011.

[14] A. Sokolov, T. Urvoy, and O. Ricard. Madspam consortium at the ecml/pkdd discovery challenge 2010. *Proceedings of the ECML/PKDD 2010 Discovery Challenge*, 2010.

[15] V. Vapnik. *The nature of statistical learning theory*. Springer Verlag, 2000.

[16] K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg. Feature hashing for large scale multitask learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1113–1120. ACM, 2009.