Learning Not to Try Too Hard

Bill McDowell*
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
wmcdowel@cs.cmu.edu

Noah A. Smith
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
nasmith@cs.cmu.edu

1 Introduction

Discriminative learning algorithms are often motivated by their ability to trade off among different kinds of prediction mistakes with different costs. The cost of a mistake is usually taken to be fully defined by the task, i.e., human system designers are trusted to encode this knowledge prior to learning. Information about the inherent ease of avoiding some errors vs. others is generally not taken into account. Closely related to this, and critically important in domains where the data is constructed by humans, is the problem that the outputs in the training data may be unreliable. For example, if training data is produced by asking humans to label instances, and two labels are insufficiently well defined for human labelers to distinguish them, then a learner might be forgiven for conflating them.

We consider situations where human intuition about relative costs of different errors is insufficient. In a margin-based linear modeling framework, we propose a method for incorporating **learning of the cost function** alongside learning of the model. Our approach introduces explicit estimates of the "ease" of avoiding each type of error (for a particular model family). For error types that are "just too hard," our model is offered the possibility of giving up in favor of making other, less challenging predictions more accurately

[MODIFY THIS. –BM] Our experiments with text classification show scenarios where the method achieves performance improvements over a strong baseline.

2 Background and Notation

In a prediction problem, let $\mathcal{X}$ denote the input space, $\mathcal{Y}$ denote the output space, and assume N training instances $\{(x_1, y_1), \ldots, (x_N, y_N)\}$. We assume a linear model and prediction function:

$$\hat{y} = \operatorname*{argmax}_{y \in \mathcal{Y}} \left(f(x, y; \mathbf{w}) \triangleq \mathbf{w}^\top \mathbf{g}(x, y) \right) \tag{1}$$

where $\mathbf{w} \in \mathbb{R}^D$ are the parameters to be learned and $\mathbf{g} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^D$ is the feature vector function. We will let $\mathcal{M} = \{f(\cdot, \cdot; \mathbf{w}) \mid \mathbf{w} \in \mathbb{R}^D\}$ denote the model family under consideration, given a fixed choice of $\mathbf{g}$.

Our approach, which assumes $\mathcal{Y}$ is categorical, is based on the soft margin formulation of multiclass support vector machines [1–3]. Tsochantaridis et al. [4] and Taskar et al. [5] generalized this framework to allow for differences in costs between different kinds of mistakes, as found when $\mathcal{Y}$ is structured. Let the cost function $\Delta : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ be such that $\Delta(y, y')$ is the cost of predicting y when the correct label is y'. We use the "margin rescaling" variant the multiclass SVM:

$$\min_{\boldsymbol{\xi} \geq 0, \mathbf{w}} \frac{\lambda}{2} \|\mathbf{w}\|_2^2 + \frac{1}{m} \sum_{i=1}^{N} \xi_i \quad \text{s.t.} \quad \forall i, \forall y \in \mathcal{Y} \setminus \{y_i\}, f(x_i, y_i; \mathbf{w}) - f(x, y, \mathbf{w}) \geq \Delta(y, y_i) - \xi_i \tag{2}$$

*Alternative email: forkunited@gmail.com

This objective seeks $\mathbf{w}$ that minimizes misclassifications while maximizing the margin between correct and incorrect instances. Further, the more incorrect an (x, y) pair is, the greater the margin should be. This problem is often transformed into an unconstrained one corresponding to direct minimization of the regularized average hinge loss:

$$\min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|_2^2 + \sum_{i=1}^{N} -f(x_i, y_i; \mathbf{w}) + \max_{y \in \mathcal{Y}} f(x_i, y; \mathbf{w}) + \Delta(y, y_i) \tag{3}$$

We introduce some notation for errors. We let $\mathcal{S} \subseteq 2^{\mathcal{Y} \times \mathcal{Y}}$ be a collection of prediction error classes that exhausts $\mathcal{Y}^2$ (i.e., $\bigcup_{S \in \mathcal{S}} S = \mathcal{Y}^2$); the error classes need not be mutually exclusive. We let $e_S \in \mathbb{R}$ denote an estimate of the "ease" with which a learner searching in $\mathcal{M}$ can successfully avoid errors in class S. Then we let:

$$\Delta(y, y') = \sum_{S \in \mathcal{S} : (y, y') \in S} e_S = \mathbf{e}^\top \mathbf{s}(y, y') \tag{4}$$

where $\mathbf{e}$ is a vector of the e_S and $\mathbf{s}$ is a binary vector of length $\mathcal{S}$ indicating which error class(es) each possible confusion in $\mathcal{Y} \times \mathcal{Y}$ belongs to.

In this paper, we consider two prediction error classes, corresponding to unordered and ordered pairs of outputs. We denote them $\mathcal{S}^{\mathrm{u}}$ and $\mathcal{S}^{\mathrm{o}}$, respectively.

3 Cost Learning Model

Previous work assumes Δ follows intuitively from the prediction task. For example, in natural language dependency parsing, the number of words attached to the wrong parent (Hamming distance for the parse tree) is a sensible choice. We propose to parameterize Δ and learn its parameters jointly with $\mathbf{w}$. This learned cost function should encode distances between outputs from the perspective of the ease with which a model in the family $\mathcal{M}$ can distinguish between them. This joint learning setup is expected to be particularly useful when some classes of errors are difficult or impossible for a model in the class to resolve, due to unreliable annotations or an insufficient choice of features $\mathbf{g}$.

3.1 Ease

We desire a model that estimates prediction ease $\mathbf{e}$ while estimating predictive model parameters $\mathbf{w}$. We have used the term "ease" with respect to an arbitrary model in the family $\mathcal{M}$, but it is more sensible to consider the particular model we seek to estimate. We propose that, for error class S and a model with parameters $\mathbf{w}$, ease e_S should be inversely related to the number of of margin violations involving S that $f(\cdot, \cdot; \mathbf{w})$ makes in the training data assuming that argmax always gives a single label, breaking ties arbitrarily:

$$v_S(\mathbf{w}, \mathbf{e}) \triangleq \left|\left\{i \in \{1, \ldots, D\} \mid \left(y_i, \operatorname{argmax}_{y \in \mathcal{Y}} f(x_i, y; \mathbf{w}) + \mathbf{e}^\top \mathbf{s}(y, y_i)\right) \in S\right\}\right| \tag{5}$$

The intuition is that, when this set is large, it is because it is not easy for the model to shrink. Of course, we should also take into account that the distribution of the data may make some errors more frequent, inflating the size of the set in Eq. 5 even if S is "easy." Further, *infrequently* observed labels are generally expected to be harder to predict. Yet for an S that includes errors on a rarely occurring class, the set in Eq. 5 will necessarily be small, regardless of how easy it is. We therefore propose the following condition for e_S:

$$e_S = \max\left(0, 1 - \frac{v_S(\mathbf{w}, \mathbf{e})}{n_S}\right) \tag{6}$$

where n_S is a fixed, *a priori* upper bound on the count of S errors, v_S. This has the desirable property that if $v_S \geq n_S$, i.e., S is too difficult to shrink, then ease e_S goes to zero and the model is allowed to give up on S. It also keeps $e_S \in [0, 1]$, ensuring interpretability of the ease relative to the maximum possible value of $e_S = 1$.

3.2 Objective

Our approach is a modification to the SVM objective in Eq. 3; it is a joint optimization of $\mathbf{w}$ and $\mathbf{e}$:

$$\min_{\mathbf{e}\geq 0, \mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|_2^2 + \frac{1}{2}\|\mathbf{e}\|_{\mathbf{n}}^2 - \mathbf{e}^\top \mathbf{n} + \sum_{i=1}^{N} -f(x_i, y_i; \mathbf{w}) + \max_{y\in\mathcal{Y}} f(x_i, y; \mathbf{w}) + \mathbf{e}^\top \mathbf{s}(y, y_i) \tag{7}$$

where $\mathbf{n}$ is the vector of upper bounds on prediction error frequencies and $\|\mathbf{e}\|_{\mathbf{n}}^2 = \sum_{S\in\mathcal{S}} n_S e_S^2$. The changes amount to (1) including $\mathbf{e}$ as a free variable and (2) regularizing it with a quadratic penalty (second term in Eq. 7) and a linear penalty (third term in Eq. 7). The linear penalty selects which e_S should be nonzero—equivalently, are not impossibly difficult. Setting $e_S = 0$ amounts to giving up on S errors.[1]

Most importantly, Eq. 7 is minimized at $\mathbf{e}$ that matches Eq. 6. To see this, we can find the subdifferential ∂C of the function C minimized in Eq. 7, and consider the optimized ease values $\mathbf{e}^*$ for which $\mathbf{0} \in \partial C(\mathbf{w}^*, \mathbf{e}^*)$. ∂C is given by:

$$\begin{aligned}\partial C(\mathbf{w}, \mathbf{e}) = &\frac{\lambda}{2}\nabla(\|\mathbf{w}\|_2^2) + \frac{1}{2}\nabla(\|\mathbf{e}\|_{\mathbf{n}}^2) - \nabla(\mathbf{e}^\top \mathbf{n}) + \\ &\sum_{i=1}^{N} -\nabla(f(x_i, y_i; \mathbf{w})) + \mathbf{Co}(\nabla F_{\max}^{\Delta}(x_i, y_i; \mathbf{w}, \mathbf{e}))\end{aligned} \tag{8}$$

Where addition, subtraction, and ∇ are overloaded to perform sums, subtractions, and gradients over sets, $\mathbf{Co}(\cdot)$ gives the convex hull of a set, and $F_{\max}^{\Delta}$ is defined as:

$$F_{\max}^{\Delta}(x, y; \mathbf{w}, \mathbf{e}) = \{f(x, \hat{y}; \mathbf{w}) + \mathbf{e}^\top \mathbf{s}(y, \hat{y}) | \hat{y} \in \operatorname*{argmax}_{y'\in\mathcal{Y}} f(x, y'; \mathbf{w}) + \mathbf{e}^\top \mathbf{s}(y, y')\} \tag{9}$$

We are only interested in the value of $\mathbf{e}$, so we can consider Eq. 8 reduced to a simpler subdifferential with respect to a single component e_S of $\mathbf{e}$:

$$\partial_{e_S} C(\mathbf{w}, \mathbf{e}) = n_S e_S - n_S + \sum_{i=1}^{N} \mathbf{Co}(\frac{\partial F_{\max}^{\Delta}(x_i, y_i; \mathbf{w}, \mathbf{e})}{\partial e_S}) \tag{10}$$

Given that $\mathbf{0} \in \partial C(\mathbf{w}^*, \mathbf{e}^*)$, Eq 10 leads to:

$$e_S^* \in 1 - \frac{\sum_{i=1}^{N} \mathbf{Co}(\frac{\partial F_{\max}^{\Delta}(x_i, y_i; \mathbf{w}^*, \mathbf{e}^*)}{\partial e_S})}{n_S} \tag{11}$$

If we allow the argmax in Eq. 9 to arbitrarily break ties as in Section 3.1, and we constrain $e_S^* \geq 0$ as in Eq. 7, then Eq. 11 suggests that e_S^* takes on the value given by Eq. 6 when the objective is minimized. So our objective chooses values of $\mathbf{e}$ according to our intuitions from Section 3.1.

3.3 Constants n

The appropriate choice for the normalization vector $\mathbf{n}$ in Eq. 7 depends on the prediction classes in $\mathcal{S}$ and the types of bias we seek to avoid in estimating $\mathbf{e}$. For $\mathcal{S}^{\mathrm{u}}$ and $\mathcal{S}^{\mathrm{o}}$, we are most concerned with unbalanced marginal distributions over labels. Let c_y be the frequency of the label y in the training data, $|\{i \mid y_i = y\}|$. We propose two choices of $\mathbf{n}$, both based on the training data:

1. **Logical n**: an upper bound on $v_S(\cdot,\cdot)$ based on frequencies in the training data. For $\mathcal{S}^{\mathrm{u}}$, let $n_{S_{\{y,y'\}}} = c_y + c_{y'}$. For $\mathcal{S}^{\mathrm{o}}$, let $n_{S_{y,y'}} = c_y$ where $S_{y,y'}$ corresponds to an erroneous label of y' in place of the correct y.

[1] The meaning of 'giving up' is unclear. It seems that setting $e_S = 0$ actually amounts to giving up on margin maximization, but not error minimization with respect to S, but we don't work this out in detail. **[NOAH, DO YOU UNDERSTAND THIS DIFFERENTLY? –BM]**

2. **Expected** $\mathbf{n}$: an upper bound calculated by assuming that our learner can perform better than a random classifier that uses label proportions observed in the training data. For $\mathcal{S}^{\mathrm{u}}$, let $n_{S_{\{y,y'\}}} = 2c_y c_{y'}/N$. For $\mathcal{S}^{\mathrm{o}}$, let $n_{S_{y,y'}} = c_y c_{y'}/N$.

The **Logical** choice will tend to dramatically overestimate the maximum count of each prediction error, but we might choose it over **Expected** if we believe that the random classifier would not give a baseline rate at which our model is biased to predict certain labels by the label distribution independent of the inputs. A third option, not explored here, might use a more sophisticated model to estimate bounds on error counts.

4 Experiments

We implemented the multiclass SVM (Eq. 3) and variations of our method, which we refer to as normalized cost learning (NCL; Eq. 7), using stochastic gradient descent (SGD) with a learning rate determined by AdaGrad [6][7]. We ran SGD for 50 passes over the data, using a random permutation each time. We observed that during the last ten passes, accuracy varied by < 0.01 and fewer than 10% of predictions changed.[2]

We ran several experiments to compare these models on standard text classification datasets/tasks, using conventional training/test splits. 10% of the training set is used in each case to perform a grid search for λ over 16 values in $[10^{-6}, 50]$, choosing the one that gives the best accuracy, and then fixing λ and training on the whole training set.[3]

We consider six variations of NCL, varying the prediction error sets ($\mathcal{S}^{\mathrm{u}}$; $\mathcal{S}^{\mathrm{o}}$) and normalization constants ($\mathbf{1}$, i.e., none; logical; expected).

4.1 Datasets

We considered two datasets with relatively large output label sets: 20 Newsgroups (20NG; 20 category labels corresponding to newsgroups) [4] and Reuters-21578 (R52; 52 topic labels). [5] We followed [8] in preprocessing the text corpora (including downcasing, removing symbols, etc.) and let features in $\mathbf{g}$ correspond to tfidf for unigrams [9].

The 20NG dataset consists of 18,846 documents, sorted by date, with 60% used for training. Though the categories are roughly uniformly distributed, the topics vary greatly in their relatedness, following a hierarchical labeling scheme (e.g., *rec.autos* and *rec.motorcycles* are likely more closely related than either to *sci.space*). This offers a way to measure the effectiveness of NCL at learning "ease": the less closely related two categories are, the greater the ease in learning to distinguish them.

The R52 dataset contains 9,100 documents; we use the ModApte split (70% training). The label distribution is skewed, with 43% of documents assigned to *earn* and 37 topics receiving fewer than 50 examples.

4.2 Results

Table 1 shows the micro-averaged accuracies on the Reuters and 20 newsgroups tasks for the SVM baseline model and versions of NCL with different choices of normalization constants $\mathbf{n}$ and incorrect prediction classes $\mathcal{S}$. We had expected that NCL would generalize better on each task, giving

[2] We were originally running for 150 iterations for previous buggy versions, but we adjusted down to 50 iterations after we realized that these experiments were going to give a negative result. It's extremely unlikely that increasing the number of iterations to 150 will change the current conclusions, but if this work is ever published, we might want to rerun other versions of the algorithm with a larger number of iterations–just to be safe.

[3] We also ran experiments on synthetic data sets, but the results for those experiments are similar to the results for the text-classification data sets, and so they are not included in this document. Specifically, the synthetic data set results show at most minor performance gains on some versions of the data although the cost function seems to always be learned appropriately. See the code documentation at `https://github.com/forkunited/CostFunctionLearning/` for more detail.

[4] `http://qwone.com/~jason/20Newsgroups`

[5] `http://www.csmining.org/index.php/r52-and-r8-of-reuters-21578.html`.

Table 1: Micro-averaged accuracies of different learners.

Learner	20NG full	20NG clusters	R52
SVM	0.834	0.920	0.945
NCL: $\mathcal{S}^{\mathrm{o}}$, none	0.834	0.919	0.946
NCL: $\mathcal{S}^{\mathrm{u}}$, none	0.832	0.921	0.945
NCL: $\mathcal{S}^{\mathrm{o}}$, logical	0.836	0.920	0.948
NCL: $\mathcal{S}^{\mathrm{u}}$, logical	0.830	0.919	0.948
NCL: $\mathcal{S}^{\mathrm{o}}$, expected	0.830	0.920	0.949
NCL: $\mathcal{S}^{\mathrm{u}}$, expected	0.820	0.911	0.946

increased accuracy, but unfortunately, the accuracy of each version of NCL is always approximately the same as the accuracy of the SVM, which suggests that the learned cost-scaled margins did not allow NCL to generalize better than the SVM's constant margin. Nevertheless, we show in the next section that NCL learned cost functions that intuitively represent the distance or ease of distinguishing each pair of labels.

4.3 Hierarchy and Ease

especially in the **clusters** column where only the misclassifications in easier prediction classes are penalized.

The hierarchical structure of the 20 Newsgroups topics encodes a notion of distance between topics as distance within the hierarchy. As noted above, we expect that more distant topics (in the hierarchy) should correspond to greater ease in distinguishing between them.

The **full** column under 20 newsgroups shows the accuracies computed over the full set of 20 category labels, and the **clusters** column shows the accuracies computed where all categories from a topic cluster shown in Figure **??** are treated as a single label.

Note that NCL does not take advantage of any information about the hierarchy.

Table 1 includes accuracies computed when nearby topics in the hierarchy are are collapsed into single topics, either at the second level (i.e., into **[?? –NAS]** categories) or the first (i.e., into **[?? –NAS]** categories). The advantages of NCL are nearly as great for these collapsed tasks as for the primary one, meaning that it is better than the SVM at distinguishing topics that are farther apart in the hierarchy, and therefore has learned a notion of ease that relates to hierarchy distance.

[FIX THE ABOVE PARAGRAPH WHEN GET NEW NUMBERS –BM]

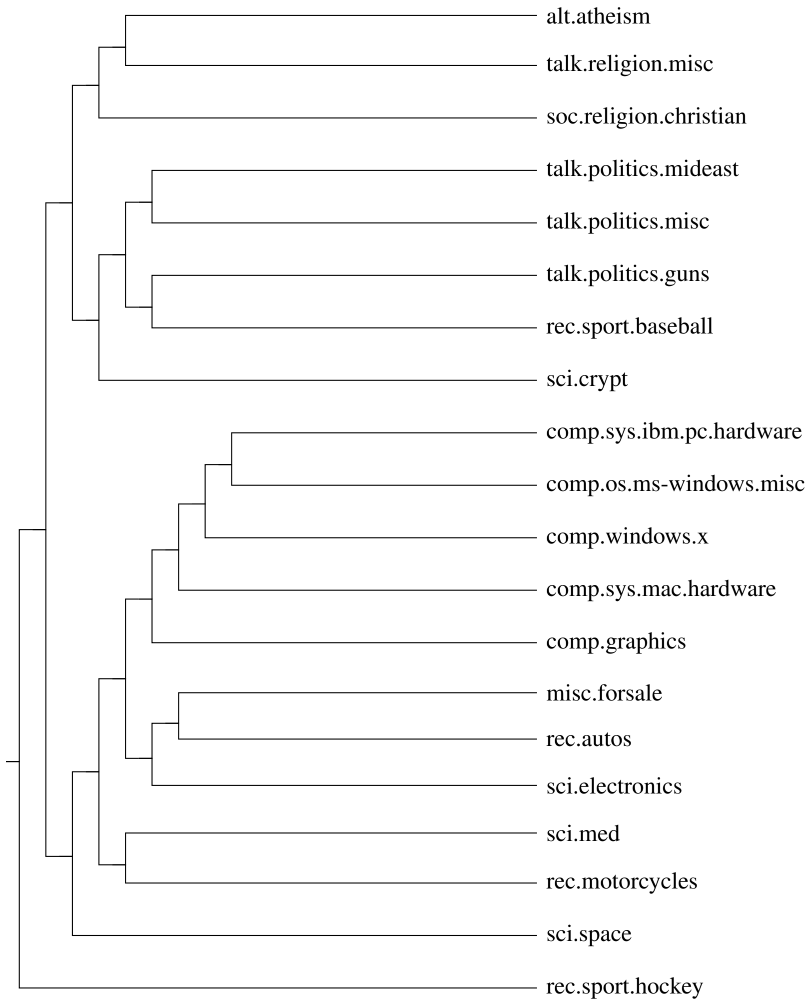
We checked this directly by using $\mathbf{e}$ to reconstruct the hierarchy by applying a hierarchical clustering algorithm.

[CITE HIERARCHICAL CLUSTERING. BE SPECIFIC ABOUT WHICH TYPE OF HIERARCHICAL CLUSTERING IS USED. –BM]

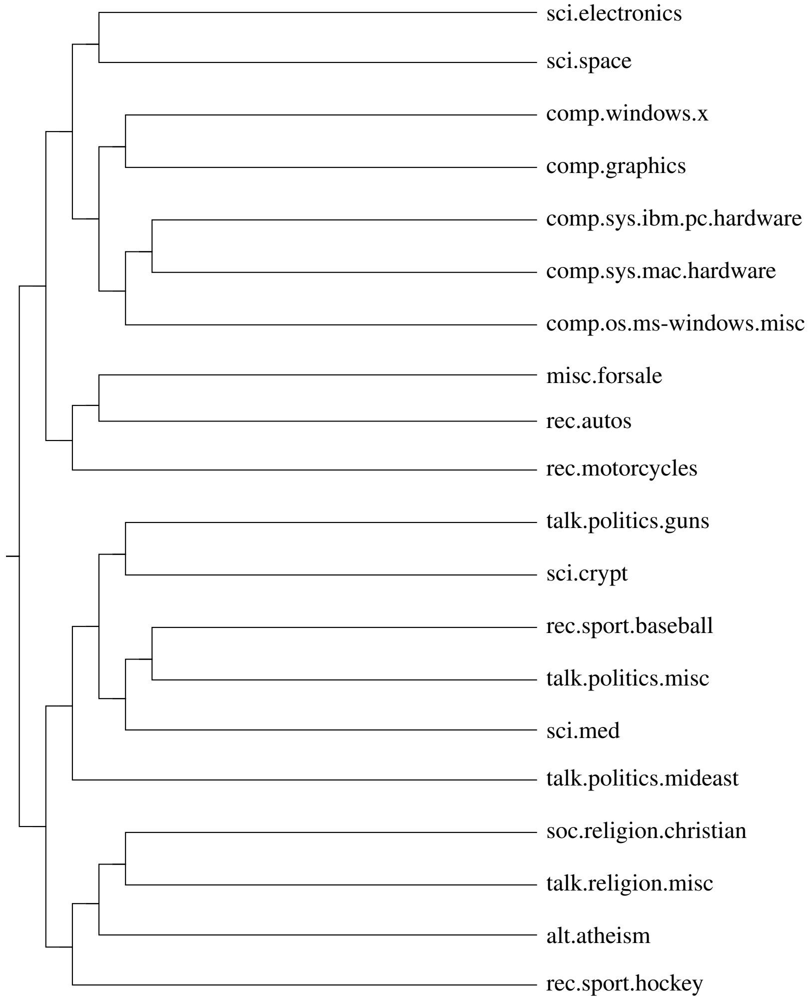
[ADD HIERARCHY FIGURE. –BM]

[ADD MEASUREMENTS OF SIMILARITY BETWEEN HIERARCHIES –BM]

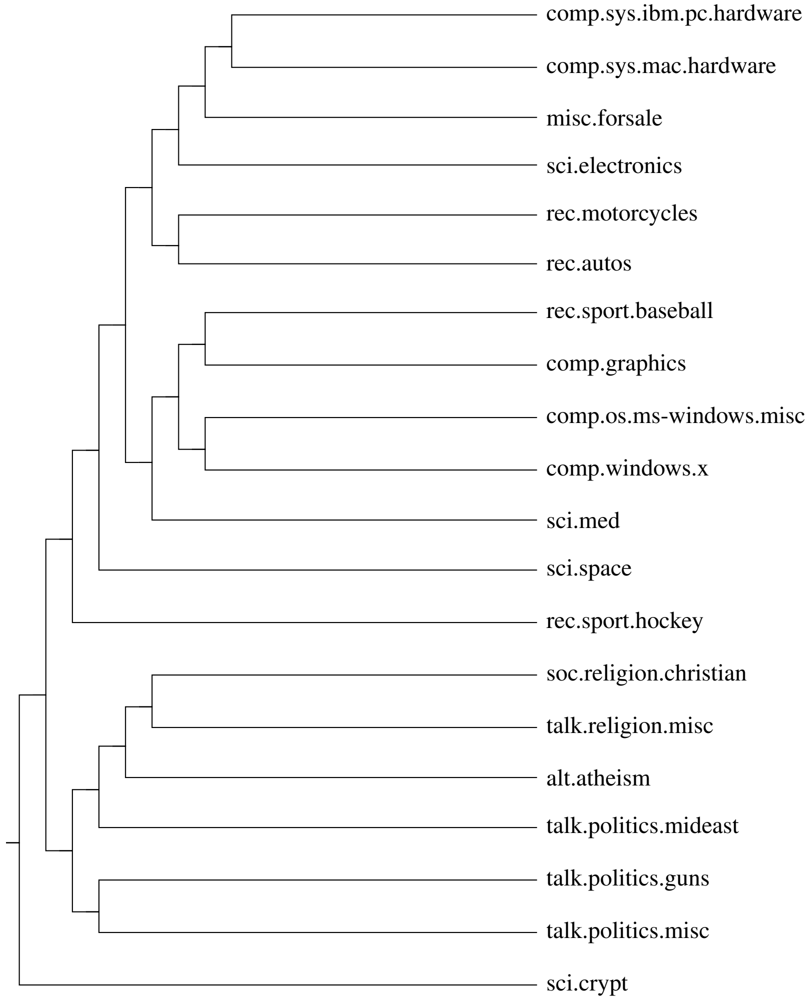
None: (.456)


alt.atheism
talk.religion.misc
soc.religion.christian
talk.politics.mideast
talk.politics.misc
talk.politics.guns
rec.sport.baseball
sci.crypt
comp.sys.ibm.pc.hardware
comp.os.ms-windows.misc
comp.windows.x
comp.sys.mac.hardware
comp.graphics
misc.forsale
rec.autos
sci.electronics
sci.med
rec.motorcycles
sci.space
rec.sport.hockey

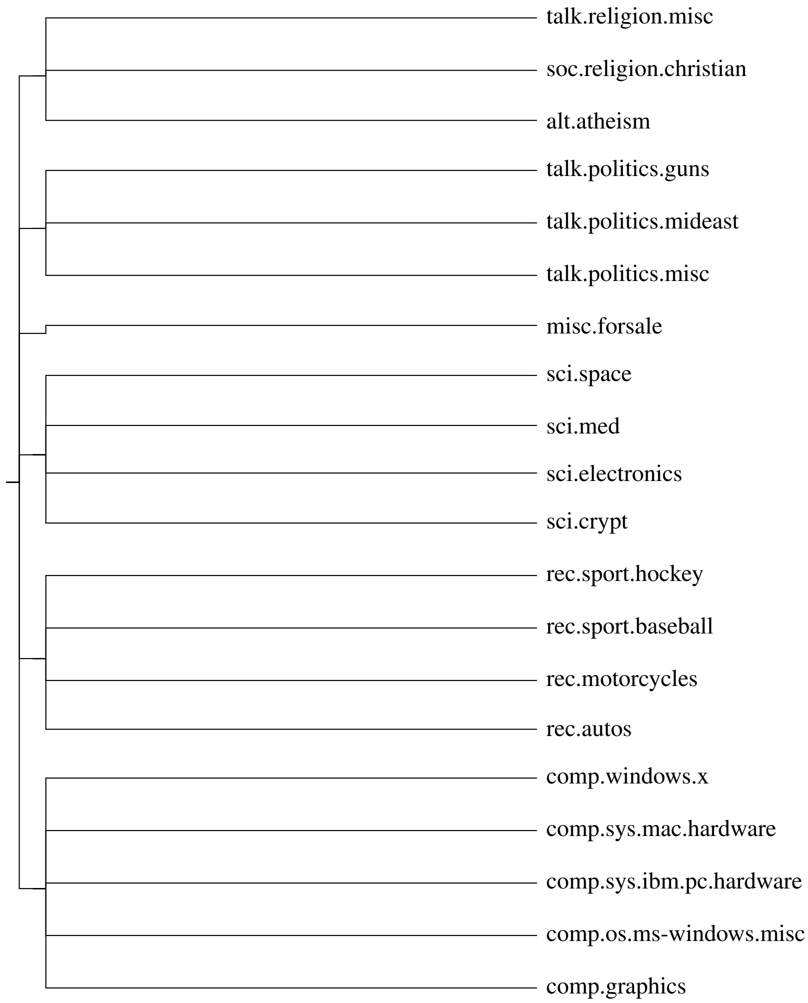
Logical: (.540)



Expected: (.245)


comp.sys.ibm.pc.hardware
comp.sys.mac.hardware
misc.forsale
sci.electronics
rec.motorcycles
rec.autos
rec.sport.baseball
comp.graphics
comp.os.ms-windows.misc
comp.windows.x
sci.med
sci.space
rec.sport.hockey
soc.religion.christian
talk.religion.misc
alt.atheism
talk.politics.mideast
talk.politics.guns
talk.politics.misc
sci.crypt

True:



5 Related Work

[ADD FOOTNOTE ABOUT RELATIONSHIP BETWEEN NORM IN OBJECTIVE AND MAHALANOBIS NORM (JUST FOR PEOPLE IN GROUP WHO LOOK AT THIS LATER) –BM] [NOT SURE THIS IS CRITICAL –NAS]

[HERE ARE SOME THINGS TO POSSIBLY WRITE ABOUT: –BM]

Self-paced learning [10]. [NOTE RELATIONSHIP BETWEEN THIS AND CURRENT OBJECTIVE –BM]

Curriculum Learning [11].

Confidence weighted learning [12].

Ed Hovy inter-annotator agreement cost [13]

Hidden variable learning by state splitting mentioned in `http://www.cs.cmu.edu/~nasmith/papers/career-proposal-2010.pdf` [14].

Finite state output encodings mentioned in `http://www.cs.cmu.edu/~nasmith/papers/career-proposal-2010.pdf` [15]

5.1 Future Work and Conclusions

[IS THERE ANYTHING ELSE THAT I FORGOT? –BM]

In future work, richer representations of prediction error types ($\mathcal{S}$) might be pursued. For example, classes might be constructed based on frequencies of classes, with the rarest labels forming a group. For structured output spaces such as natural language parsing, the domain might suggest groups of errors; post hoc analysis of $\mathbf{e}$ might, in turn, suggest ways to improve the model through feature engineering.[6] Our framework is easily extended to let these classes depend on the input or metadata as well, allowing very rich parameterizations of learnable cost functions. Recall that these classes need not be mutually exclusive.

Alternative ways to estimate $\mathbf{n}$ might also be considered, such as using a more sophisticated model to estimate bounds on error frequencies in the training set. More generally, characterizations of ease might be developed through alternate means, such as the stability measure from learning theory [16], which might offer insight into the generalizability of predictions involving a particular label.

We concede that our notion of "ease" merges several concepts that might be treated separately. These include the reliability of the labels in training data, the distinctiveness of the labels given the model family (choice of features), the learnability of each label given the number of instances it has in the training set, and the overall similarity of the training distribution to the "true" one. We believe it is an open theoretical question how these various notions might relate to learning guarantees.

[ADD RECOMMENDATION ABOUT DOING SYNTHETIC DATA EXPERIMENTS WITH MORE FEATURES –BM]

[GET RID OF NS –BM]

[THE 'REFERENCES' HEADING GIVEN BY THE BIBLIOGRAPHY COMMAND IS THE WRONG SIZE FONT. NEEDS TO BE THE SIZE OF A 'THIRD LEVEL HEADING'. HOW TO CHANGE THIS? –BM] [MAYBE THE PROBLEM COMES FROM USING NATBIB? –NAS]

References

[1] Vladimir N Vapnik. Statistical learning theory (adaptive and learning systems for signal processing, communications and control series), 1998.

[2] Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *The Journal of Machine Learning Research*, 2:265–292, 2002.

[3] Jason Weston and Chris Watkins. Multi-class support vector machines. Technical report, Citeseer, 1998.

[4] Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the twenty-first international conference on Machine learning*, page 104. ACM, 2004.

[5] Ben Taskar Carlos Guestrin Daphne Koller. Max-margin markov networks. 2003.

[6] G George Yin and Harold Joseph Kushner. *Stochastic approximation and recursive algorithms and applications*. Springer, 2003.

[7] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159, 2011.

[8] Ana Cardoso-Cachopo. Improving Methods for Single-label Text Categorization. PdD Thesis, Instituto Superior Tecnico, Universidade Tecnica de Lisboa, 2007.

[9] Man Lan, Chew Lim Tan, and Hwee-Boon Low. Proposing a new term weighting scheme for text categorization. In *AAAI*, volume 6, pages 763–768, 2006.

[10] M Pawan Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. In *NIPS*, volume 1, page 3, 2010.

[11] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. ACM, 2009.

[12] Mark Dredze, Koby Crammer, and Fernando Pereira. Confidence-weighted linear classification. In *Proceedings of the 25th international conference on Machine learning*, pages 264–271. ACM, 2008.

[13] Barbara Plank, Dirk Hovy, and Anders Søgaard. Learning part-of-speech taggers with inter-annotator agreement loss. In *Proceedings of EACL*, 2014.

[6] We note an interesting parallel to the *ceteris paribus* reasoning suggested by inspection of linear model weights $\mathbf{w}$; inspecting $\mathbf{e}$ shows, "all other things equal," a scaling of error types by ease-of-avoidance.

[14] Slav Petrov. *Coarse-to-fine natural language processing*. Springer, 2011.

[15] Edward Loper. *Encoding structured output values*. PhD thesis, University of Pennsylvania, 2008.

[16] Sayan Mukherjee, Partha Niyogi, Tomaso Poggio, and Ryan Rifkin. Learning theory: stability is sufficient for generalization and necessary and sufficient for consistency of empirical risk minimization. *Advances in Computational Mathematics*, 25(1-3):161–193, 2006.