Learning when to try hard at learning

Anonymous Author(s)
Affiliation
Address
email

[PROBABLY NEED A BETTER TITLE. –BM]

[NOAH, I PUT YOUR NAME SECOND BECAUSE YOUR NAME SEEMS TO BE LAST ON ALL OF YOUR RECENT PAPERS. I DON'T CARE WHICH OUR NAMES IS FIRST THOUGH, SO FEEL FREE TO SWAP IF YOU WANT. –BM]

Abstract

[I HAVE TROUBLE WRITING THIS PART. –BM]

1 Introduction

[GENERAL IDEAS ABOUT COST LEARNING ABSTRACTED AWAY FROM THE SVM DETAILS IN THE NEXT SECTION...? –BM]

2 Background

[I INTRODUCED SVMS THROUGH THE QUADRATIC PROGRAMMING FORMULATION BECAUSE IT SEEMED EASIER TO SUMMARIZE MARGIN MAXIMIZATION ACROSS ALL THE RELEVANT SVM VARIANTS THAT WAY WHILE STAYING TRUE TO THE LITERATURE... BUT MAYBE I SHOULD HAVE JUST KEPT IT IN THE FORM OF AN UNCONSTRAINED OPTIMIZATION PROBLEM FOR CONTINUITY WITH THE REST OF THIS PAPER? –BM]

[CITE SOME PAPER ON USING BINARY SVMS FOR MULTICLASS –BM]

[CITE MULTICLASS SVM –BM]

[CITE SOMETHING FOR MARGIN MAXIMIZATION –BM]

[CITE STRUCTURED SVM –BM]

[CITE EXAMPLE OF UNRELIABLE OUTPUT LABELS –BM]

[ALSO CITE TASKAR ET AL FOR 'MARGIN RESCALING'? –BM]

[MIGHT WANT TO GIVE SOME MOTIVATING EXAMPLES? NOT SURE IF THAT'S NECESSARY OR NOT. ONE EXAMPLE OF MULTICLASS DOMAIN, AND ONE EXAMPLE WHERE COST FUNCTIONS ARE USED IN STRUCTURED DOMAINS –BM]

[MENTION EXAMPLES OF MEASURES OF 'DIFFICULTY'? CITE. –BM]

For the multiclass classification problem, we are given a set of m labelled training data instances $D = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_m, \mathbf{y}_m)\} \subset \mathcal{X} \times \mathcal{Y}$ where each example $\mathbf{x}_i$ is assigned label $\mathbf{y}_i$, and we want to learn a function $f : \mathcal{X} \to \mathcal{Y}$ which gives the correct label for any input taken from $\mathcal{X}$. Past work has developed a multiclass support vector machine (SVM) which generalizes the concept of margin maximization employed by classical SVMs for binary classification tasks. In particular, assuming that $\mathbf{g} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^k$ computes a featurized representation of an input-output pair, the multiclass SVM gives a score to a labelled instance $(\mathbf{x}, \mathbf{y})$ as:

$$F(\mathbf{x}, \mathbf{y}; \mathbf{w}) = \mathbf{w}^\top \mathbf{g}(\mathbf{x}, \mathbf{y}) \tag{1}$$

And makes predictions according to:

$$\hat{f}(\mathbf{x}; \mathbf{w}) = \underset{\mathbf{y}' \in \mathcal{Y}}{\operatorname{argmax}} F(\mathbf{x}, \mathbf{y}'; \mathbf{w}) \tag{2}$$

Where the feature weights $\mathbf{w}$ are given by the solution to the following soft-margin maximizing quadratic program:

$$\begin{aligned} &\min_{\xi \geq 0, \mathbf{w}} \big(\frac{\lambda_2}{2} \|\mathbf{w}\|_2^2 + \frac{1}{m} \sum_{i=1}^{m} \xi_i \big) \\ &\text{s.t. } \forall i : \forall \mathbf{y} \in \mathcal{Y} \backslash \mathbf{y}_i : F(\mathbf{x}_i, \mathbf{y}_i; \mathbf{w}) - F(\mathbf{x}_i, \mathbf{y}; \mathbf{w}) \geq 1 - \xi_i \end{aligned} \tag{3}$$

This quadratic program chooses weights that minimize the number of misclassified instances while simultaneously trying to increase the margin between the scores of correct and incorrect labels, and the λ_2 hyper-parameter determines the relative importance of these two goals.[1]

Tsochantaridis et al further generalized the margin maximization to make sense for domains where some prediction mistakes are more costly than others (especially domains where $\mathcal{Y}$ contains a large number of structured outputs). They introduce a function $\Delta : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ that determines the cost $\Delta(\mathbf{y}, \hat{\mathbf{y}})$ of predicting label $\hat{\mathbf{y}}$ when then correct label is $\mathbf{y}$. The $\Delta(\mathbf{y}, \hat{\mathbf{y}})$ is larger for more inaccurate $\hat{\mathbf{y}}$ predictions, and $\Delta(\mathbf{y}, \mathbf{y}) = 0$, giving no cost for a correct prediction. Tsiochantaridis et al incorporate the cost function into the SVM learning by modifying the multiclass SVM quadratic program through 'margin re-scaling' as:[2]

$$\begin{aligned} &\min_{\xi \geq 0, \mathbf{w}} \big(\frac{\lambda_2}{2} \|\mathbf{w}\|_2^2 + \frac{1}{m} \sum_{i=1}^{m} \xi_i \big) \\ &\text{s.t. } \forall i : \forall \mathbf{y} \in \mathcal{Y} \backslash \mathbf{y}_i : F(\mathbf{x}_i, \mathbf{y}_i; \mathbf{w}) - F(\mathbf{x}_i, \mathbf{y}; \mathbf{w}) \geq \Delta(\mathbf{y}_i, \mathbf{y}) - \xi_i \end{aligned} \tag{4}$$

Intuitively, this margin re-scaling quadratic program allows the model to choose margins between between scores of $\mathbf{y}_i$ and $\hat{\mathbf{y}}_i$ proportional to the prediction cost $\Delta(\mathbf{y}_i, \hat{\mathbf{y}})$. This choice biases the model toward making low cost predictions over high cost predictions.

Whereas previous work assumes that we hand-select domain-specific cost functions that seem like good measures of the distances between labels, we propose learning this function from the data jointly with the prediction function f. Similarly to the hand-selected cost functions, the learned cost functions should reflect some notion of the distance between two labels, but as seen from the perspective of the model and its features rather than from the perspective of the person engineering the model. In other words, the value of $\Delta(\mathbf{y}, \hat{\mathbf{y}})$ should be proportional to the ease with which the model discriminates between $\mathbf{y}$ and $\hat{\mathbf{y}}$ given its features. This choice of cost gives the model the freedom to shift its weights so that there are wider margins between labels which it can easily discriminate, and smaller margins between labels which it has difficulty discriminating. Such a policy is particularly useful when there are classes of incorrect predictions that are difficult or impossible for the model to systematically resolve due to unreliably annotated output labels or a choice of features that is insufficient for the task.

[1]This SVM and all other models in this paper can include non-regularized bias terms which we omit from our descriptions for readability, but these biases were included in the implementations we used for our experiments.

[2]Tsiochantaridis et al also introduce an alternative 'slack re-scaling' technique, but the present paper mainly builds off of the 'margin re-scaling' approach.

There are several ways that we might quantify the ease with which the model discriminates between two labels, but for now, we will keep the notion of 'easiness' fuzzy while establishing its relation to the cost function. Basically, we propose that an incorrect prediction $\hat{\mathbf{y}}$ where the actual label is $\mathbf{y}$ can fall into some number of 'incorrect prediction classes', and the overall cost of the incorrect prediction is the sum over the easiness of resolving (shrinking) each of these classes. More formally, let $\mathcal{S} \subseteq 2^{\mathcal{Y}^2}$ be a collection of incorrect prediction classes that collectively exhausts $\mathcal{Y}^2$. Assume that the 'easiness' with which a model of type $\mathcal{M}$ (e.g. SVM) given features $\mathbf{g}$ and data D resolves incorrect prediction class $S \in \mathcal{S}$ is given by $\mathcal{E}(S, \mathcal{M}, \mathbf{g}, D) \in \mathbb{R}$. Then, we assume that the cost is given by:

$$\Delta(\mathbf{y}, \hat{\mathbf{y}}) = \sum_{S \in \mathcal{S}} \mathcal{E}(S, \mathcal{M}, \mathbf{g}, D) \mathbb{1}((\mathbf{y}, \hat{\mathbf{y}}) \in S) = \mathcal{E}^\top \mathcal{S}(\mathbf{y}, \hat{\mathbf{y}}) \tag{5}$$

There are many possible choices for prediction classes $\mathcal{S}$, but in this paper, we focus on the following choices:

$$\mathcal{S}_{[\mathcal{Y}]^2} = \{S_{\{\mathbf{y},\mathbf{y}'\}} | \mathbf{y}, \mathbf{y}' \in \mathcal{Y}\} \text{ where } S_{\{\mathbf{y},\mathbf{y}'\}} = \{(\mathbf{l}, \mathbf{l}') | (\mathbf{l} = \mathbf{y} \wedge \mathbf{l}' = \mathbf{y}') \vee (\mathbf{l} = \mathbf{y}' \wedge \mathbf{l} = \mathbf{y})\} \tag{6}$$

$$\mathcal{S}_{\mathcal{Y}^2} = \{S_{(\mathbf{y},\mathbf{y}')} | \mathbf{y}, \mathbf{y}' \in \mathcal{Y}\} \text{ where } S_{(\mathbf{y},\mathbf{y}')} = \{(\mathbf{y}, \mathbf{y}')\} \tag{7}$$

$\mathcal{S}_{[\mathcal{Y}]^2}$ contains a prediction class for every unordered pair of labels, and $\mathcal{S}_{\mathcal{Y}^2}$ contains a prediction class for every non-ordered pair. We might choose $\mathcal{S}_{[\mathcal{Y}]^2}$ when factoring the cost function if we believe it is equally easy to resolve misclassifications of $\mathbf{y}$ as $\mathbf{y}'$ and misclassifications of $\mathbf{y}'$ as $\mathbf{y}$. Otherwise, if we suspect these two incorrect prediction types to differ in easiness, then we might choose $\mathcal{S}_{\mathcal{Y}^2}$.

3 A cost learning model

We desire a model which approximates $\mathcal{E}(S, \mathcal{M}, \mathbf{g}, D)$ (for some notion of 'easiness') to estimate prediction costs while simultaneous learning feature weights $\mathbf{w}$. Our proposal follows the intuition that the easiness $\mathcal{E}(S, \mathcal{M}, \mathbf{g}, D)$ of prediction class S for model class $\mathcal{M}$ is related to the size of the set:

$$S_{\mathcal{M},\mathbf{g},D} = \{(\mathbf{x}_i, \mathbf{y}_i) | (\mathbf{y}_i, \hat{f}^{\Delta}_{\mathcal{M},\mathbf{g},D}(\mathbf{x}_i; \mathbf{w}_{\mathcal{M},\mathbf{g},D})) \in S \text{ and } (\mathbf{x}_i, \mathbf{y}_i) \in D\} \tag{8}$$

Where $\mathbf{w}_{\mathcal{M},\mathbf{g},D}$ are the learned feature weights, and $\hat{f}^{\Delta}_{\mathcal{M},\mathbf{g},D}$ is the model's prediction function augmented with the cost:

$$\hat{f}^{\Delta}_{\mathcal{M},\mathbf{g},D}(\mathbf{x}_i; \mathbf{w}) = \underset{\mathbf{y} \in \mathcal{Y}}{\operatorname{argmax}} \Big(F(\mathbf{x}_i, \mathbf{y}; \mathbf{w}) + \Delta(\mathbf{y}_i, \mathbf{y}) \Big) \tag{9}$$

The size of $S_{\mathcal{M},\mathbf{g},D}$ is the number of training examples with margin violations in class S. If the size of this set is large, we might infer that the model has trouble shrinking it, and so it's not 'easy'. This might lead us to conclude that $S_{\mathcal{M},\mathbf{g},D}$ tends to shrink with 'easiness'. However, for many data sets and choices of $\mathcal{S}$, the size of each $S_{\mathcal{M},\mathbf{g},D}$ can be inherently biased by the data independently of 'easiness'. For example, for $S_{\{\mathbf{y},\mathbf{y}'\}} \in \mathcal{S}_{[\mathcal{Y}]^2}$, the size of $S_{\{\mathbf{y},\mathbf{y}'\},\mathcal{M},\mathbf{g},D}$ is biased by the number of examples in the training data which have output labels $\mathbf{y}$ and $\mathbf{y}'$–if there are few training examples of labels $\mathbf{y}, \mathbf{y}' \in \mathcal{Y}$, then the size of $S_{\{\mathbf{y},\mathbf{y}'\},\mathcal{M},\mathbf{g},D}$ will necessarily be small relative to other prediction classes. Furthermore, we expect output labels which occur infrequently in the training data to be more difficult for the model to predict correctly, so this will lead to the size of $S_{\{\mathbf{y},\mathbf{y}'\},\mathcal{M},\mathbf{g},D}$ increasing with the 'easiness' of $S_{\{\mathbf{y},\mathbf{y}'\}}$ which is opposite the conclusion that that we draw if we think of the size of $S_{\{\mathbf{y},\mathbf{y}'\},\mathcal{M},\mathbf{g},D}$ as increasing due to the model's difficulty in shrinking it. In general, this suggests that if we want the size of $S_{\mathcal{M},\mathbf{g},D}$ to vary with easiness, we need to normalize it to account for properties of the training data that introduce irrelevant biases.

3.1 A measure of 'easiness'

[MIGHT WANT TO DISCUSS OTHER POSSIBILITIES, OR GENERALLY REFER TO OTHER POSSIBILITIES IN FUTURE WORK –BM]

The above observations suggest the following as a possible measure of 'easiness':

$$\mathcal{E}(S, \mathcal{M}, \mathbf{g}, D) = \max\left(0, 1 - \frac{|S_{\mathcal{M},\mathbf{g},D}|}{n_{S,\mathcal{M},\mathbf{g},D}}\right) \tag{10}$$

Where $n_{S,\mathcal{M},\mathbf{g},D}$ is a normalization constant which gives the maximum possible value we expect for the size of $S_{\mathcal{M},\mathbf{g},D}$, accounting for irrelevant biases introduced by the data as discussed above. This measure is in $[0, 1]$, and it has the property that if $|S_{\mathcal{M},\mathbf{g},D}| \geq n_{\mathcal{M},\mathbf{g},D}$, then $\mathcal{E}(S, \mathcal{M}, \mathbf{g}, D) = 0$, indicating that $S_{\mathcal{M},\mathbf{g},D}$ is so difficult to shrink that its size is greater than our expected upper bound.

3.2 A cost learning objective

[CITE SELF-PACED LEARNING FOR INSPIRATION FOR NEW OBJECTIVE FUNCTION? –BM]

[ADD FOOTNOTE ABOUT RELATIONSHIP BETWEEN NORM IN OBJECTIVE AND MAHALANOBIS NORM –BM]

[IS THERE ANY MATH I SHOULD BE MORE EXPLICIT ABOUT? –BM]

We can modify the margin re-scaling SVM learning procedure given by quadratic program 4 to learn the cost function according to easiness measure 10. First, we transform the quadratic program into the equivalent unconstrained optimization problem:

$$\min_{\mathbf{w}} \frac{\lambda_2}{2}\|\mathbf{w}\|_2^2 + \sum_{i=1}^{m}\left(- F(\mathbf{x}_i, \mathbf{y}_i; \mathbf{w}) + \max_{\mathbf{y}\in\mathcal{Y}}\left(F(\mathbf{x}_i, \mathbf{y}; \mathbf{w}) + \Delta(\mathbf{y}_i, \mathbf{y})\right)\right) \tag{11}$$

And we modify this function to include the cost learning as:

$$\min_{\hat{\mathcal{E}}\geq 0, \mathbf{w}} \frac{\lambda_2}{2}\|\mathbf{w}\|_2^2 + \sum_{i=1}^{m}\left(-F(\mathbf{x}_i, \mathbf{y}_i; \mathbf{w}) + \max_{\mathbf{y}\in\mathcal{Y}}\left(F(\mathbf{x}_i, \mathbf{y}; \mathbf{w}) + \hat{\mathcal{E}}^\top \mathcal{S}(\mathbf{y}_i, \mathbf{y})\right)\right) - \hat{\mathcal{E}}^\top \mathbf{n} + \frac{1}{2}\|\hat{\mathcal{E}}\|_{\mathbf{n}}^2 \tag{12}$$

Where $\mathbf{n}$ is the vector of normalization constants, and $\|\hat{\mathcal{E}}\|_n^2 = \sum_{S\in\mathcal{S}} n_S \hat{\mathcal{E}}_S^2$.[3] Intuitively, this new objective function uses the $-\hat{\mathcal{E}}^\top \mathbf{n}$ term to select which $\hat{\mathcal{E}}_S$ should be non-zero (or correspondingly which S are not impossibly difficult), and the $\|\hat{\mathcal{E}}\|_{\mathbf{n}}^2$ term orders these non-zero $\hat{\mathcal{E}}_S$ relative to the normalization constants.

If the solution to objective 12 is at point that is differentiable with respect to $\hat{\mathcal{E}}_S$, then it's easy to show that $\hat{\mathcal{E}}_S$ has exactly the value given by equation 10. Otherwise, if the solution is at a non-differentiable point, then $\hat{\mathcal{E}}_S$ has a value that approximates equation 10 in a sensible way.

[FILL IN DETAILS ABOUT NON-DIFFERENTIABLE SOLUTIONS –BM]

3.3 Some 'easiness' normalization constants

[MENTION CHOOSING N USING BASELINE SVM ESPECIALLY IF THIS IS INCLUDED IN RESULTS. BUT OTHERWISE MAYBE PUT IT IN FUTURE WORK SINCE CHOOSING N BASED ON OTHER MODELS IS A GENERAL TOPIC TO EXPLORE –BM]

The appropriate choice for the normalization vector $\mathbf{n}$ in objective 12 depends on the prediction classes $\mathcal{S}$ and the type of irrelevant bias we are trying to remove from 'easiness' measure. For $\mathcal{S}_{[\mathcal{Y}]^2}$ and $\mathcal{S}_{\mathcal{Y}^2}$, we want to remove the bias introduced into the size of the prediction classes by non-uniform label distributions, as discussed at the beginning of section 3. Otherwise, the model will

[3] We abbreviate $\hat{\mathcal{E}}(S, \mathcal{M}, \mathbf{g}, D)$ and $n_{S,\mathcal{M},\mathbf{g},D}$ as $\hat{\mathcal{E}}_S$ and n_S for readability.

tend to over-estimate the costs of incorrect predictions involving labels that occur infrequently in the training data. The following are two plausible choices of $\mathbf{n}$ to achieve this goal. In each, assume that $D_{\mathbf{y}}$ is the set of training examples for which $\mathbf{y}$ is the correct label.

1. **Logical** $\mathbf{n}$: Choose n_S as an upper bound on $|S|$ that cannot possibly be violated given the training data. For prediction classes $S_{\mathbf{y},\mathbf{y}'} \in \mathcal{S}_{\mathcal{Y}^2}$, choose $n_{S_{\mathbf{y},\mathbf{y}'}} = |D_{\mathbf{y}}|$, and for prediction classes $S_{\{\mathbf{y},\mathbf{y}'\}} \in \mathcal{S}_{[\mathcal{Y}]^2}$, choose $n_{S_{\{\mathbf{y},\mathbf{y}'\}}} = \max\left(|D_{\mathbf{y}}|, |D_{\mathbf{y}'}|\right)$.
2. **Expected** $\mathbf{n}$: Choose n_S as the expected number of times a model makes a mistake in S if it predicts labels at random according to their distribution in the training data. For prediction classes $S_{\mathbf{y},\mathbf{y}'} \in \mathcal{S}_{\mathcal{Y}^2}$, choose $n_{S_{\mathbf{y},\mathbf{y}'}} = \frac{|D_{\mathbf{y}}||D_{\mathbf{y}'}|}{m}$, and for prediction classes $S_{\{\mathbf{y},\mathbf{y}'\}} \in \mathcal{S}_{[\mathcal{Y}]^2}$, choose $n_{S_{\{\mathbf{y},\mathbf{y}'\}}} = \frac{2|D_{\mathbf{y}}||D_{\mathbf{y}'}|}{m}$. Other variations of this idea might choose alternative models by which to compute the expectation, but model that makes predictions at according to the training data label distribution seems reasonable for getting rid of the label distribution bias.

In general, the **Logical** choice of $\mathbf{n}$ is an upper bound on the **Expected** choice. The **Logical** choice will tend to over-estimate the maximum size of each prediction class, but we might choose it over **Expected** if we have reason to believe that it is difficult to estimate the baseline rate at which the model is biased to predict certain labels by the label distribution independent of the inputs.

4 Experiments

[CITE PEGASOS FOR SGD LEARNING RATE –BM]

[ADD GRAPHS SHOWING CONVERGENCE OF SGD? –BM]

We implemented the multiclass SVM and normalized cost learning SVM (SVMCLN) using stochastic gradient descent (SGD) to approximate objectives 11 and 12. Our SGD implementation used a learning rate of $\frac{1}{\lambda_2 t}$ at time-step t. We ran several experiments to compare these models on standard text classification corpora. In each experiment, we compared the we used the standard train/test split given by the respective data corpus, while also randomly selecting a 10% subset of the training data to use as a dev set. We used the dev set to perform a grid search over 16 possible values ranging from 10^{-6} to 0.5×10^2 for the λ_2 hyper-parameter in each model. After the grid search, we fixed the best value for λ_2, and retrained on the full training data (including the dev set), evaluating the final performance of the model on the test set.

Due to the stochasticity of the optimization algorithm, we expected some noisiness in its convergence. In general, we ran SGD for 150 iterations over the random permutations of the full data. At this point, the accuracy and predictions of each model on the dev/test sets would be relatively stable. In particular, the accuracy would vary by at most 0.001 and fewer than at most 5% of the predictions would change across 10 full iterations over the training data.

4.1 Datasets

[CITE 20 NEWSGROUPS AND REUTERS R52 –BM]

[CITE HTTP://WWW.AAAI.ORG/PAPERS/AAAI/2006/AAAI06-121.PDF ON RELATIVELY LOW DIFFERENCE IN PERFORMANCE BETWEEN LOG(TF) AND TFIDF –BM]

[CITE HTTP://QWONE.COM/~JASON/WRITING/LOOCV.PDF FOR EXAMPLE USE OF LOG(TF) –BM]

[PREPROCESSED USING METHOD FROM HTTP://WEB.IST.UTL.PT/ACARDOSO/DOCS/2007-PHD-THESIS.PDF –BM]

We compared the performance of the models on the 20 Newsgroups and the Reuters-21578 (R52) data sets. We chose these two data sets because they both have a relatively large number of output labels, some of which might not be distinctive enough for the SVM to plausibly learn well, in which case we expected a gain in performance from SVMCLN.

Table 1: Micro-averaged Accuracies

Model	n	$\mathcal{S}$	20news	20news level 2	20news level 1	Reuters
SVM	N/A	N/A	0.7760	0.8008	0.8654	0.9213
SVMCLN	None	$\mathcal{Y}^2$	0.8008	0.8259	0.8752	0.9213
SVMCLN	None	$[\mathcal{Y}]^2$	0.8011	0.8257	0.8751	0.9194
SVMCLN	Logical	$\mathcal{Y}^2$	0.8024	0.8271	0.8769	0.9210
SVMCLN	Logical	$[\mathcal{Y}]^2$	0.8376	0.8631	0.9142	0.9159
SVMCLN	Expected	$\mathcal{Y}^2$	0.8303	0.8557	0.9092	0.9159
SVMCLN	Expected	$[\mathcal{Y}]^2$	0.8307	0.8558	0.9084	0.9171

For both of these data sets, the target labels $\mathcal{Y}$ are single document topics/categories, and the inputs $\mathcal{X}$ are documents. We preprocessed each text document from both corpora using the procedure given in [FILL IN] (convert to lower-case, remove symbols, etc), and then we computed $\mathbf{g}$ as normalized $\log$ unigram term-frequency features. In particular,

$$\mathbf{g}_{\mathbf{y}'}(\mathbf{x}, \mathbf{y}) = \mathbb{1}(\mathbf{y} = \mathbf{y}')\mathbf{f}(\mathbf{x}) \tag{13}$$

Where $\mathbf{f}(\mathbf{x})$ is a normalized vector whose elements are $\log(1 + tf(\mathbf{x}, v))$ when v is a unigram in the corpus vocabulary and $tf(\mathbf{x}, v)$ is the frequency of v in $\mathbf{x}$.

Below are some details specific to the 20 Newsgroups and Reuters corpora.

4.1.1 20 newsgroups

[I WILL FILL THIS IN. –BM]

4.1.2 Reuter 21578 (R52)

[I WILL FILL THIS IN. –BM]

4.2 Results

[I WILL FILL THIS IN. –BM]

5 Discussion

5.1 Related literature

[I HAVE TROUBLE WRITING THIS PART. –BM]

5.2 Future work

[THERE'S PROBABLY LOTS OF STUFF I DON'T KNOW ABOUT WHICH COULD GO HERE. HERE ARE SOME IDEAS THOUGH: –BM]

[INCORPORATE INPUT FEATURES INTO COST –BM]

[OTHER CHOICES FOR N –BM]

[OTHER MEASURES OF 'EASINESS' –BM]

[STRUCTURED VERSION –BM]

[ALTERNATING MINIMIZATION VERSION –BM]

[RELATIONSHIP BETWEEN MODEL 'DIFFICULTY' AND 'TASK' DIFFICULTY –BM]

[Characterize properties of data/features that determine whether cost learning is useful –bm]

References

[See comments in source for partial list of references –bm]