Linguistic Structured Sparsity in Text Categorization

Dani Yogatama
Language Technologies Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, USA
dyogatama@cs.cmu.edu

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

Abstract

We introduce three linguistically motivated structured regularizers based on parse trees, topics, and hierarchical word clusters for text categorization. These regularizers impose linguistic bias in feature weights, enabling us to incorporate prior knowledge into conventional bag-of-words models. We show that our structured regularizers consistently improve classification accuracies compared to standard regularizers that penalize features in isolation (such as lasso, ridge, and elastic net regularizers) on a range of datasets for various text prediction problems: topic classification, sentiment analysis, and forecasting.

1 Introduction

What is the best way to exploit linguistic information in statistical text processing models? For tasks like text classification, sentiment analysis, and text-driven forecasting, this is an open question, as cheap "bag-of-words" models often perform well. Much recent work in NLP has focused on linguistic feature engineering (Joshi et al., 2010) or representation learning (Glorot et al., 2011; Socher et al., 2013).

In this paper, we propose a radical alternative. We embrace the conventional bag-of-words representation of text, instead bringing linguistic bias to bear on *regularization*. Since the seminal work of Chen and Rosenfeld (2000), the importance of regularization in discriminative models of text—including language modeling, structured prediction, and classification—has been widely recognized. The emphasis, however, has largely been on one specific kind of inductive bias: avoiding large weights (i.e., coefficients in a linear model).

Recently, *structured* (or composite) regularization has been introduced; simply put, it reasons about different weights jointly. The most widely explored variant, group lasso (Yuan and Lin, 2006) seeks to avoid large ℓ_2 norms for *groups* of weights. Group lasso has been shown useful in a range of applications, including computational biology (Kim and Xing, 2008), signal processing (Lv et al., 2011), and NLP (Eisenstein et al., 2011; Martins et al., 2011; Nelakanti et al., 2013). For text categorization problems, Yogatama and Smith (2014) proposed groups based on sentences, an idea generalized here to take advantage of richer linguistic information.

In this paper, we show how linguistic information of various kinds—parse trees, thematic topics, and hierarchical word clusterings—can be used to construct group lasso variants that impose linguistic bias without introducing any new features. Our experiments demonstrate that structured regularizers can squeeze higher performance out of conventional bag-of-words models on seven out of eight of text categorization tasks tested, in six cases with more compact models than the best-performing unstructured-regularized model.

2 Notation

We represent each document as a feature vector $\mathbf{x} \in \mathbb{R}^V$, where V is the vocabulary size. x_v is the frequency of the vth word (i.e., this is a "bag of words" model).

Consider a linear model that predicts a binary response $y \in \{-1, +1\}$ given $\mathbf{x}$ and weight vector $\mathbf{w} \in \mathbb{R}^V$. We denote our training data of D documents in the corpus by $\{\mathbf{x}_d, y_d\}_{d=1}^D$. The goal of the learning procedure is to estimate $\mathbf{w}$ by minimizing the regularized training data loss:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \Omega(\mathbf{w}) + \textstyle\sum_{d=1}^D \mathcal{L}(\mathbf{x}_d, \mathbf{w}, y_d),$$

where $\mathcal{L}(\mathbf{x}, \mathbf{w}, y)$ is the loss function for document d and $\Omega(\mathbf{w})$ is the regularizer.

In this work, we use the log loss:

$$\mathcal{L}(\mathbf{x}_d, \mathbf{w}, y_d) = -\log(1 + \exp(-y_d \mathbf{w}^\top \mathbf{x}_d)),$$

Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, pages 786–796,
Baltimore, Maryland, USA, June 23-25 2014. ©2014 Association for Computational Linguistics

Other loss functions (e.g., hinge loss, squared loss) can also be used with any of the regularizers discussed in this paper.

Our focus is on the regularizer, $\Omega(\mathbf{w})$. For high dimensional data such as text, regularization is crucial to avoid overfitting.[1]

The usual starting points for regularization are the "lasso" (Tibshirani, 1996) and the "ridge" (Hoerl and Kennard, 1970), based respectively on the ℓ_1 and squared ℓ_2 norms:

$$\Omega_{las}(\mathbf{w}) = \lambda_{las}\|\mathbf{w}\|_1 = \lambda \textstyle\sum_j |w_j|$$
$$\Omega_{rid}(\mathbf{w}) = \lambda_{rid}\|\mathbf{w}\|_2^2 = \lambda \textstyle\sum_j w_j^2$$

Both methods disprefer weights of large magnitude; smaller (relative) magnitude means a feature (here, a word) has a smaller effect on the prediction, and zero means a feature has no effect.[2] The hyperparameter λ in each case is typically tuned on a development dataset. A linear combination of ridge and lasso is known as the elastic net (Zou and Hastie, 2005). The lasso, ridge, and elastic net are three strong baselines in our experiments.

3 Group Lasso

Structured regularizers penalize estimates of $\mathbf{w}$ in which collections of weights are penalized jointly. For example, in the group lasso (Yuan and Lin, 2006), predefined groups of weights (subvectors of $\mathbf{w}$) are encouraged to either go to zero (as a group) or not (as a group)—this is known as "group sparsity."[3]

The variant of group lasso we explore here uses an $\ell_{1,2}$ norm. Let g index the G predefined groups of weights and $\mathbf{w}_g$ denote the subvector of $\mathbf{w}$ containing weights for group g:

$$\Omega_{glas}(\mathbf{w}) = \lambda_{glas} \textstyle\sum_{g=1}^{G} \lambda_g \|\mathbf{w}_g\|_2,$$

where λ_{glas} is a hyperparameter tuned on a development data, and λ_g is a group specific weight. Typically the groups are non-overlapping, which offers computational advantages, but this need not be the case (Jacob et al., 2009; Jenatton et al., 2011).

4 Structured Regularizers for Text

Past work applying the group lasso to NLP problems has considered four ways of defining the groups. Eisenstein et al. (2011) defined groups of coefficients corresponding to the same independent variable applied to different (continuous) output variables in multi-output regression. Martins et al. (2011) defined groups based on feature templates used in chunking and parsing tasks. Nelakanti et al. (2013) defined groups based on n-gram histories for language modeling. In each of these cases, the groups were defined based on information from feature *types* alone; given the features to be used, the groups were known.

Here we build on a fourth approach that exploits structure in the data.[4] Yogatama and Smith (2014) introduced the *sentence regularizer*, which uses patterns of word cooccurrence in the training data to define groups. We review this method, then apply the idea to three more linguistically informed structure in text data.

4.1 Sentence Regularizer

The sentence regularizer exploits sentence boundaries in each training document. The idea is to define a group $g_{d,s}$ for every sentence s in every training document d. The group contains coefficients for words that occur in its sentence. This means that a word is a member of one group for every distinct (training) sentence it occurs in, and that the regularizer is based on word tokens, not types as in the approach of Martins et al. (2011) and Nelakanti et al. (2013). The regularizer is:

$$\Omega_{sen}(\mathbf{w}) = \textstyle\sum_{d=1}^{D} \sum_{s=1}^{S_d} \lambda_{d,s} \|\mathbf{w}_{d,s}\|_2,$$

where S_d is the number of sentences in document d. This regularizer results in tens of thousands to millions of heavily *overlapping* groups, since a standard corpus typically contains thousands to millions of sentences and many words that appear in more than one sentence.

[1] A Bayesian interpretation of regularization is as a prior on the weight vector $\mathbf{w}$; in many cases Ω can be understood as a log-prior representing beliefs about the model held before exposure to data. For lasso regression, the prior is a zero-mean Laplace distribution, whereas for ridge regression the prior is a zero-mean Gaussian distribution. For non-overlapping group lasso, the prior is a two-level hierarchical Bayes model (Figueiredo, 2002). The Bayesian interpretation of overlapping group lasso is not yet well understood.

[2] The lasso leads to strongly sparse solutions, in which many elements of the estimated $\mathbf{w}$ are actually zero. This is an attractive property for efficiency and (perhaps) interpretability. The ridge encourages weights to go toward zero, but usually not all the way *to* zero; for this reason its solutions are known as "weakly" sparse.

[3] Other structured regularizers include the fused lasso (Tibshirani et al., 2005) and the elitist lasso (Kowalski and Torresani, 2009).

[4] This provides a compelling reason *not* to view such methods in a Bayesian framework: if the regularizer is informed by the data, then it does not truly correspond to a prior.

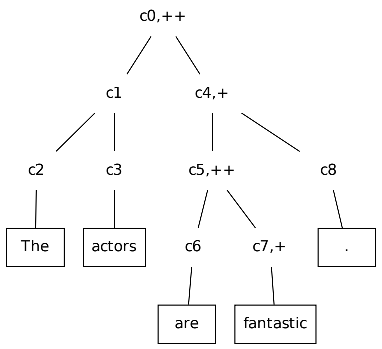

Figure 1: An example of a parse tree from the Stanford sentiment treebank, which annotates sentiment at the level of every constituent (indicated here by $+$ and $++$; no marking indicates neutral sentiment). The sentence is The actors are fantastic. Our regularizer constructs nine groups for this sentence, corresponding to $c_0, c_1, \ldots, c_8$. g_{c_0} consists of 5 weights—$\langle w_{the}, w_{actors}, w_{are}, w_{fantastic}, w_{.}\rangle$, exactly the same as the group in the sentence regularizer—g_{c_1} consists of 2 words, g_{c_4} of 3 words, etc. Notice that c_2, c_3, c_6, c_7, and c_8 each consist of only 1 word. The Stanford sentiment treebank has an annotation of sentiments at the constituent level. As in this example, most constituents are annotated as neutral.

If the norm of $\mathbf{w}_{g_{d,s}}$ is driven to zero, then the learner has deemed the corresponding sentence irrelevant to the prediction. It is important to point out that, while the regularizer prefers to zero out the weights for all words in irrelevant sentences, it also prefers *not* to zero out weights for words in relevant sentences. Since the groups overlap and may work against each other, the regularizer may not be able to drive many weights to zero on its own. Yogatama and Smith (2014) used a linear combination of the sentence regularizer and the lasso (a kind of *sparse group lasso*; Friedman et al., 2010) to also encourage weights of irrelevant word types to go to zero.[5]

4.2 Parse Tree Regularizer

Sentence boundaries are a rather superficial kind of linguistic structure; syntactic parse trees provide more fine-grained information. We introduce a new regularizer, the parse tree regularizer, in which groups are defined for every constituent in every parse of a training data sentence.

Figure 1 illustrates the group structures derived from an example sentence from the Stanford sentiment treebank (Socher et al., 2013). This regularizer captures the idea that *phrases* might be selected as relevant or (in most cases) irrelevant to a task, and is expected to be especially useful in sentence-level prediction tasks.

The parse-tree regularizer (omitting the group coefficients and λ) for one sentence with the parse tree shown in Figure 1 is:

$$\Omega_{tree}(\mathbf{w}) =$$

$$\sqrt{|w_{the}|^2 + |w_{actors}|^2 + |w_{are}|^2 + |w_{fantastic}|^2 + |w_{.}|^2}$$
$$+ \sqrt{|w_{are}|^2 + |w_{fantastic}|^2 + |w^2_{.}|}$$
$$+ \sqrt{|w_{the}|^2 + |w_{actors}|^2} + \sqrt{|w_{are}|^2 + |w_{fantastic}|^2}$$
$$+ |w_{the}| + |w_{actors}| + |w_{are}| + |w_{fantastic}| + |w_{.}|$$

The groups have a tree structure, in that assigning zero values to the weights in a group corresponding to a higher-level constituent implies the same for those constituents that are dominated by it. This resembles the tree-guided group lasso in Kim and Xing (2008), although the leaf nodes in their tree represent tasks in multi-task regression.

Of course, in a corpus there are many parse trees (one per sentence, so the number of parse trees is the number of sentences). The parse-tree regularizer is:

$$\Omega_{tree}(\mathbf{w}) = \sum_{d=1}^{D} \sum_{s=1}^{S_d} \sum_{c=1}^{C_{d,s}} \lambda_{d,s,c} \|\mathbf{w}_{d,s,c}\|_2,$$

where $\lambda_{d,s,c} = \lambda_{glas} \times \sqrt{size(g_{d,s,c})}$, d ranges over (training) documents and c ranges over constituents in the parse of sentence s in document d. Similar to the sentence regularizer, the parse-tree regularizer operates on word tokens. Note that, since each word token is itself a constituent, the parse tree regularizer includes terms just like the lasso naturally, penalizing the absolute value of each word's weight in isolation. For the lasso-like penalty on each word, instead of defining the group weights to be $1 \times$ the number of tokens for each word type, we tune one group weight for all word types on a development data. As a result, besides λ_{glas}, we have an additional hyperparameter, denoted by λ_{las}.

To gain an intuition for this regularizer, consider the case where we apply the penalty only for a single tree (sentence), which for ease of exposition is assumed not to use the same word more than once (i.e., $\|\mathbf{x}\|_\infty = 1$). Because it instantiates the tree-structured group lasso, the regularizer will require bigger constituents to be "included" (i.e., their words given nonzero weight) before smaller constituents can be included. The result is that some words may not be included. Of course, in some sentences, some words will occur more than once, and the parse tree regularizer instantiates groups for constituents in every sentence in the training corpus, and these groups may work against each other. The parse tree regularizer should therefore

[5] Formally, this is equivalent to including one additional group for each word type.

be understood as encouraging group behavior of syntactically grouped words, or sharing of information by syntactic neighbors.

In sentence level prediction tasks, such as sentence-level sentiment analysis, it is known that most constituents (especially those that correspond to shorter phrases) in a parse tree are uninformative (neutral sentiment). This was verified by Socher et al. (2013) when annotating phrases in a sentence for building the Stanford sentiment treebank. Our regularizer incorporates our prior expectation that most constituents should have no effect on prediction.

4.3 LDA Regularizer

Another type of structure to consider is topics. For example, if we want to predict whether a paper will be cited or not (Yogatama et al., 2011), the model can perform better if it knows beforehand the collections of words that represent certain themes (e.g., in ACL papers, these might include machine translation, parsing, etc.). As a result, the model can focus on which topics will increase the probability of getting citations, and penalize weights for words in the same topic together, instead of treating each word separately.

We do this by inferring topics in the training corpus by estimating the latent Dirichlet allocation (LDA) model (Blei et al., 2003)). Note that LDA is an unsupervised method, so we can infer topical structures from *any* collection of documents that are considered related to the target corpus (e.g., training documents, text from the web, etc.). This contrasts with typical semi-supervised learning methods for text categorization that combine unlabeled and labeled data within a generative model, such as multinomial naïve Bayes, via expectation-maximization (Nigam et al., 2000) or semi-supervised frequency estimation (Su et al., 2011). Our method does not use unlabeled data to obtain more training documents or estimate the joint distributions of words better, but it allows the use of unlabeled data to induce topics. We leave comparison with other semi-supervised methods for future work.

There are many ways to associate inferred topics with group structure. In our experiments, we choose the R most probable words given a topic and create a group for them.[6] The LDA regularizer can be written as:

$$\Omega_{lda}(\mathbf{w}) = \textstyle\sum_{k=1}^{K} \lambda_k \|\mathbf{w}_k\|_2,$$

where k ranges over the K topics. Similar to our earlier notations, $\mathbf{w}_k$ corresponds to the subvector of $\mathbf{w}$ such that the corresponding features are present in topic k. Note that in this case we can also have overlapping groups, since words can appear in the top R of many topics.

$k = 1$	$k = 2$	$k = 3$	$k = 4$
soccer	injury	physics	monday
striker	knee	gravity	tuesday
midfielder	ligament	moon	april
goal	shoulder	sun	june
defender	cruciate	relativity	sunday

Table 1: A toy example of $K = 4$ topics. The top $R = 5$ words in each topics are displayed. The LDA regularizer will construct four groups from these topics. The first group is $\langle w_{soccer}, w_{striker}, w_{midfielder}, w_{goal}, w_{defender}\rangle$, the second group is $\langle w_{injury}, w_{knee}, w_{ligament}, w_{shoulder}, w_{cruciate}\rangle$, etc. In this example, there are no words occurring in the top R of more than one topic, but that need not be the case in general.

To gain an intuition for this regularizer, consider the toy example in Table 1. the case where we have $K = 4$ topics and we select $R = 5$ top words from each topic. Supposed that we want to classify whether an article is a sports article or a science article. The regularizer might encourage the weights for the fourth topics' words toward zero, since they are less useful for the task. Additionally, the regularizer will penalize words in each of the other three groups collectively. Therefore, if (for example) *ligament* is deemed a useful feature for classifying an article to be about sports, then the other words in that topic will have a smaller effective penalty for getting nonzero weights—even weights of the opposite sign as $w_{ligament}$. It is important to distinguish this from unstructured regularizers such as the lasso, which penalize each word's weight on its own without regard for related word types.

Unlike the parse tree regularizer, the LDA regularizer is not tree structured. Since the lasso-like penalty does not occur naturally in a non tree-structured regularizer, we add an additional lasso penalty for each word type (with hyperparameter λ_{las}) to also encourage weights of irrelevant words to go to zero. Our LDA regularizer is an instance of sparse group lasso (Friedman et al., 2010).

[6] Another possibility is to group the smallest set of words whose total probability given a topic amounts to P (e.g., 0.99). mass of a topic. Preliminary experiments found this not to work well.

4.4 Brown Cluster Regularizer

Brown clustering is a commonly used unsupervised method for grouping words into a hierarchy of clusters (Brown et al., 1992). Because it uses local information, it tends to discover words with similar syntactic behavior, though semantic groupings are often evident, especially at the more fine-grained end of the hierarchy.

We incorporate Brown clusters into a regularizer in a similar way to the topical word groups inferred using LDA in §4.3, but here we make use of the hierarchy. Specifically, we construct tree-structured groups, one per cluster (i.e., one per node in the hierarchy). The Brown cluster regularizer is:

$$\Omega_{brown}(\mathbf{w}) = \textstyle\sum_{v=1}^{N} \lambda_v \|\mathbf{w}_v\|_2,$$

where v ranges over the N nodes in the Brown cluster tree. As a tree structured regularizer, this regularizer enforces constraints that a node v's group is given nonzero weights only if those nodes that dominate v (i.e., are on a path from v to the root) have their groups selected.

Consider a similar toy example to the LDA regularizer (sports vs. science) and the hierarchical clustering of words in Figure 2. In this case, the Brown cluster regularizer will create 17 groups, one for every node in the clustering tree. The regularizer for this tree (omitting the group coefficients and λ) is:

$$\begin{aligned}\Omega_{brown}(\mathbf{w}) = \textstyle\sum_{i=0}^{7} \|\mathbf{w}_{v_i}\|_2 + |w_{goal}| + |w_{striker}| \\ + |w_{midfielder}| + |w_{knee}| + |w_{injury}| \\ + |w_{gravity}| + |w_{moon}| + |w_{sun}|\end{aligned}$$

The regularizer penalizes words in a cluster together, exploiting discovered syntactic relatedness. Additionally, the regularizer can zero out weights of words corresponding to any of the internal nodes, such as v_7 if the words `monday` and `sunday` are deemed irrelevant to prediction.

Note that the regularizer already includes terms like the lasso naturally. Similar to the parse tree regularizer, for the lasso-like penalty on each word, we tune one group weight for all word types on a development data with a hyperparameter λ_{las}.

A key difference between the Brown cluster regularizer and the parse tree regularizer is that there is only one tree for Brown cluster regularizer, whereas the parse tree regularizer can have millions (one per sentence in the training data). The LDA and Brown cluster regularizers offer ways to incorporate unlabeled data, if we believe that the unlabeled data can help us infer better topics or clusters. Note that the processes of learning topics or clusters, or parsing training data sentences, are a separate stage that precedes learning our predictive model.

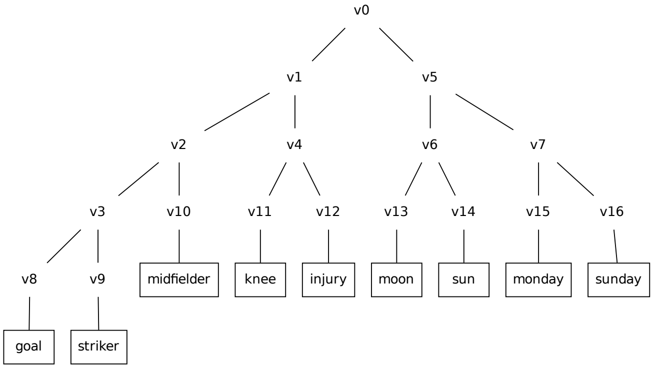

Figure 2: An illustrative example of Brown clusters for $N = 9$. The Brown cluster regularizer constructs 17 groups, one per node in for this tree, $v_0, v_1, \ldots, v_{16}$. v_0 contains 8 words, v_1 contains 5, etc. Note that the leaves, $v_8, v_9, \ldots, v_{16}$, each contain one word.

5 Learning

There are many optimization methods for learning models with structured regularizers, particulary group lasso (Jacob et al., 2009; Jenatton et al., 2011; Chen et al., 2011; Qin and Goldfarb, 2012; Yuan et al., 2013). We choose the optimization method of Yogatama and Smith (2014) since it handles millions of overlapping groups effectively. The method is based on the alternating directions method of multipliers (ADMM; Hestenes, 1969; Powell, 1969). We review it here in brief, for completeness, and show how it can be applied to tree-structured regularizers (such as the parse tree and Brown cluster regularizers in §4) in particular.

Our learning problem is, generically:

$$\min_{\mathbf{w}} \Omega(\mathbf{w}) + \textstyle\sum_{d=1}^{D} \mathcal{L}(\mathbf{x}_d, \mathbf{w}, y_d).$$

Separating the lasso-like penalty for each word type from our group regularizers, we can rewrite this problem as:

$$\min_{\mathbf{w},\mathbf{v}} \Omega_{las}(\mathbf{w}) + \Omega_{glas}(\mathbf{v}) + \textstyle\sum_{d=1}^{D} \mathcal{L}(\mathbf{x}_d, \mathbf{w}, y_d) \quad \text{s.t. } \mathbf{v} = \mathbf{M}\mathbf{w}$$

where $\mathbf{v}$ consists of copies of the elements of $\mathbf{w}$. Notice that we work directly on $\mathbf{w}$ instead of the copies for the lasso-like penalty, since it does not have overlaps and has its own hyperparameters λ_{las}. For the remaining groups with size greater than one, we create copies $\mathbf{v}$ of size

$L = \sum_{g=1}^{G} \text{size}(g)$. $\mathbf{M} \in \{0,1\}^{L \times V}$ is a matrix whose 1s link elements of $\mathbf{w}$ to their copies.[7] We now have a constrained optimization problem, from which we can create an augmented Lagrangian problem; let $\mathbf{u}$ be the Lagrange variables:

$$\begin{aligned} &\Omega_{las}(\mathbf{w}) + \Omega_{glas}(\mathbf{v}) + \mathcal{L}(\mathbf{w}) \\ &\qquad + \mathbf{u}^\top(\mathbf{v} - \mathbf{M}\mathbf{w}) + \frac{\rho}{2}\|\mathbf{v} - \mathbf{M}\mathbf{w}\|_2^2 \end{aligned}$$

ADMM proceeds by iteratively updating each of $\mathbf{w}$, $\mathbf{v}$, and $\mathbf{u}$, amounting to the following subproblems:

$$\min_{\mathbf{w}} \Omega_{las}(\mathbf{w}) + \mathcal{L}(\mathbf{w}) - \mathbf{u}^\top \mathbf{M}\mathbf{w} + \frac{\rho}{2}\|\mathbf{v} - \mathbf{M}\mathbf{w}\|_2^2 \quad (1)$$

$$\min_{\mathbf{v}} \Omega_{glas}(\mathbf{v}) + \mathbf{u}^\top \mathbf{v} + \frac{\rho}{2}\|\mathbf{v} - \mathbf{M}\mathbf{w}\|_2^2 \quad (2)$$

$$\mathbf{u} = \mathbf{u} + \rho(\mathbf{v} - \mathbf{M}\mathbf{w}) \quad (3)$$

Yogatama and Smith (2014) show that Eq. 1 can be rewritten in a form quite similar to ℓ_2-regularized loss minimization.[8]

Eq. 2 is the proximal operator of $\frac{1}{\rho}\Omega_{glas}$ applied to $\mathbf{M}\mathbf{w} - \frac{\mathbf{u}}{\rho}$. As such, it depends on the form of $\mathbf{M}$. Note that when applied to the collection of "copies" of the parameters, $\mathbf{v}$, Ω_{glas} no longer has overlapping groups. Defined $\mathbf{M}_g$ as the rows of $\mathbf{M}$ corresponding to weight copies assigned to group g. Let $\mathbf{z}_g \triangleq \mathbf{M}_g\mathbf{w} - \frac{\mathbf{u}_g}{\rho}$. Denote $\lambda_g = \lambda_{glas}\sqrt{\text{size}(g)}$. The problem can be solved by applying the proximal operator used in non-overlapping group lasso to each subvector:

$$\begin{aligned} \mathbf{v}_g &= \text{prox}_{\Omega_{glas}, \frac{\lambda_g}{\rho}}(\mathbf{z}_g) \\ &= \begin{cases} \mathbf{0} & \text{if } \|\mathbf{z}_\mathbf{g}\|_2 \leq \frac{\lambda_g}{\rho} \\ \frac{\|\mathbf{z}_g\|_2 - \frac{\lambda_g}{\rho}}{\|\mathbf{z}_g\|_2}\mathbf{z}_g & \text{otherwise.} \end{cases} \end{aligned}$$

For a tree structured regularizer, we can get speedups by working from the root node towards the leaf nodes when applying the proximal operator in the second step. If g is a node in a tree which is driven to zero, all of its children h that has $\lambda_h \leq \lambda_g$ will also be driven to zero.

Eq. 3 is a simple update of the dual variable $\mathbf{u}$. Algorithm 1 summarizes our learning procedure.[9]

Algorithm 1 ADMM for overlapping group lasso

```
Input: augmented Lagrangian variable $\rho$, regularization strengths $\lambda_{glas}$ and $\lambda_{las}$
while stopping criterion not met do
    $\mathbf{w} = \arg\min_{\mathbf{w}} \Omega_{las}(\mathbf{w}) + \mathcal{L}(\mathbf{w}) + \frac{\rho}{2}\sum_{i=1}^{V} N_i(w_i - \mu_i)^2$
    for $g = 1$ to $G$ do
        $\mathbf{v}_g = \text{prox}_{\Omega_{glas}, \frac{\lambda_g}{\rho}}(\mathbf{z}_g)$
    end for
    $\mathbf{u} = \mathbf{u} + \rho(\mathbf{v} - \mathbf{M}\mathbf{w})$
end while
```

6 Experiments

6.1 Datasets

We use publicly available datasets to evaluate our model described in more detail below.

Topic classification. We consider four binary categorization tasks from the 20 Newsgroups dataset.[10] Each task involves categorizing a document according to two related categories: comp.sys: ibm.pc.hardware vs. mac.hardware; rec.sport: baseball vs. hockey; sci: med vs. space; and alt.atheism vs. soc.religion.christian.

Sentiment analysis. One task in sentiment analysis is predicting the polarity of a piece of text, i.e., whether the author is favorably inclined toward a (usually known) subject of discussion or proposition (Pang and Lee, 2008). Sentiment analysis, even at the coarse level of polarity we consider here, can be confused by negation, stylistic use of irony, and other linguistic phenomena. Our sentiment analysis datasets consist of movie reviews from the Stanford sentiment treebank (Socher et al., 2013),[11] and floor speeches by U.S. Congressmen alongside "yea"/"nay" votes on the bill under discussion (Thomas et al., 2006).[12] For the Stanford sentiment treebank, we only predict binary classifications (positive or negative) and exclude neutral reviews.

Text-driven forecasting. Forecasting from text requires identifying textual correlates of a response variable revealed in the future, most of which will be weak and many of which will be spurious (Kogan et al., 2009). We consider two such problems. The first one is predicting whether a scientific paper will be cited or not within three years of its publication (Yogatama et al., 2011);

[7] For the parse tree regularizer, L is the sum, over all training-data word tokens t, of the number of constituents t belongs to. For the LDA regularizer, $L = R \times K$. For the Brown cluster regularizer, $L = V - 1$.

[8] The difference lies in that the squared ℓ_2 norm in the penalty penalizes the difference between $\mathbf{w}$ and a vector that depends on the current values of $\mathbf{u}$ and $\mathbf{v}$. This does not affect the algorithm or its convergence in any substantive way.

[9] We use relative changes in the ℓ_2 norm of the parameter vector $\mathbf{w}$ as our convergence criterion (threshold of 10^{-3}), and set the maximum number of iterations to 100. Other criteria can also be used.

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

[11] `http://nlp.stanford.edu/sentiment/`

[12] `http://www.cs.cornell.edu/~ainur/data.html`

	Dataset	D	# Dev.	# Test	V
20N	science	952	235	790	30,154
	sports	958	239	796	20,832
	relig.	870	209	717	24,528
	comp.	929	239	777	20,868
Sent.	movie	6,920	872	1,821	17,576
	vote	1,175	257	860	24,508
Fore.	science	3,207	280	539	42,702
	bill	37,850	7,341	6,571	10,001

Table 2: Descriptive statistics about the datasets.

the dataset comes from the ACL Anthology and consists of research papers from the Association for Computational Linguistics and citation data (Radev et al., 2009). The second task is predicting whether a legislative bill will be recommended by a Congressional committee (Yano et al., 2012).[13]

Table 2 summarizes statistics about the datasets used in our experiments. In total, we evaluate our method on eight binary classification tasks.

6.2 Setup

In all our experiments, we use unigram features plus an additional bias term which is not regularized. We compare our new regularizers with state-of-the-art methods for document classification: lasso, ridge, and elastic net regularization, as well as the sentence regularizer discussed in §4.1 (Yogatama and Smith, 2014).[14]

We parsed all corpora using the Berkeley parser (Petrov and Klein, 2007).[15] For the LDA regularizers, we ran LDA[16] on training documents with $K = 1,000$ and $R = 10$. For the Brown cluster regularizers, we ran Brown clustering[17] on training documents with $5,000$ clusters for the topic classification and sentiment analysis datasets, and $1,000$ for the larger text forecasting datasets (since they are bigger datasets that took more time).

6.3 Results

Table 3 shows the results of our experiments on the eight datasets. The results demonstrate the superiority of structured regularizers. One of them achieved the best result on all but one dataset.[18] It is also worth noting that in most cases all variants of the structured regularizers outperformed lasso, ridge, and elastic net. In four cases, the new regularizers in this paper outperform the sentence regularizer.

We can see that the parse tree regularizer performed the best for the movie review dataset. The task is to predict sentence-level sentiment, so each training example is a sentence. Since constituent-level annotations are available for this dataset, we only constructed groups for neutral constituents (i.e., we drive neutral constituents to zero during training). It has been shown that syntactic information is helpful for sentence-level predictions (Socher et al., 2013), so the parse tree regularizer is naturally suitable for this task.

The Brown cluster and LDA regularizers performed best for the forecasting scientific articles dataset. The task is to predict whether an article will be cited or not within three years after publication. Regularizers that exploit the knowledge of semantic relations (e.g., topical categories), such as the Brown cluster and LDA regularizers, are therefore suitable for this type of prediction.

Table 4 shows model sizes obtained by each of the regularizers for each dataset. While lasso prunes more aggressively, it almost always performs worse. Our structured regularizers were able to obtain a significantly smaller model (27%, 34%, 19% as large on average for parse tree, Brown, and LDA regularizers respectively) compared to the ridge model.

Topic and cluster features. Another way to incorporate LDA topics and Brown clusters into a linear model is by adding them as additional features. For the 20N datasets, we also ran lasso, ridge, and elastic net with *additional* LDA topic and Brown cluster features.[19] Note that these new baselines use more features than our model. We can also add these additional features to our model

[13] http://www.ark.cs.cmu.edu/bills

[14] Hyperparameters are tuned on a separate development dataset, using accuracy as the evaluation criterion. For lasso and ridge models, we choose λ from $\{10^{-2}, 10^{-1}, 1, 10, 10^{2}, 10^{3}\}$. For elastic net, we perform grid search on the same set of values as ridge and lasso experiments for λ_{rid} and λ_{las}. For the sentence, Brown cluster, and LDA regularizers, we perform grid search on the same set of values as ridge and lasso experiments for $\rho, \lambda_{glas}, \lambda_{las}$. For the parse tree regularizer, because there are many more groups than other regularizers, we choose λ_{glas} from $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10\}$, ρ and λ_{las} from the same set of values as ridge and lasso experiments. If there is a tie on development data we choose the model with the smallest number of nonzero weights.

[15] https://code.google.com/p/berkeleyparser/

[16] http://www.cs.princeton.edu/~blei/lda-c/

[17] https://github.com/percyliang/brown-cluster

[18] This "bill" dataset, where they offered no improvement, is the largest by far (37,850 documents), and therefore the one where regularizers should matter the least. Note that the differences are small across regularizers for this dataset.

[19] For LDA, we took the top 10 words in a topic as a feature. For Brown clusters, we add a cluster as an additional feature if its size is less than 50.

Task	Dataset	Accuracy (%)							
		m.f.c.	**lasso**	**ridge**	**elastic**	**sentence**	**parse**	**Brown**	**LDA**
20N	science	50.13	90.63	91.90	91.65	**96.20**	92.66	93.04	93.67
	sports	50.13	91.08	93.34	93.71	**95.10**	93.09	93.71	94.97
	religion	55.51	90.52	92.47	92.47	92.75	**94.98**	92.89	93.03
	computer	50.45	85.84	86.74	87.13	**90.86**	89.45	86.36	88.42
Sentiment	movie	50.08	78.03	80.45	80.40	80.72	**81.55**	80.34	78.36
	vote	58.37	73.14	72.79	72.79	**73.95**	73.72	66.86	73.14
Forecasting	science	50.28	64.00	66.79	66.23	67.71	66.42	69.02	**69.39**
	bill	87.40	88.36	87.70	**88.48**	88.11	87.98	88.20	88.27

Table 3: Classification accuracies on various datasets. “m.f.c.” is the most frequent class baseline. Boldface shows best results.

Task	Dataset	Model size (%)							
		m.f.c.	**lasso**	**ridge**	**elastic**	**sentence**	**parse**	**Brown**	**LDA**
20N	science	-	1	100	34	12	2	42	9
	sports	-	2	100	15	3	3	16	9
	religion	-	0.3	100	48	94	72	41	15
	computer	-	2	100	24	10	5	24	8
Sentiment	movie	-	10	100	54	83	87	59	12
	vote	-	2	100	44	6	2	30	4
Forecasting	science	-	31	100	43	99	9	50	90
	bill	-	7	100	7	8	37	7	7

Table 4: Model sizes (percentages of nonzero features in the resulting models) on various datasets.

Dataset	+ LDA features			LDA reg.
	lasso	**ridge**	**elastic**	
science	90.63	91.90	91.90	**93.67**
sports	91.33	93.47	93.84	**94.97**
religion	91.35	92.47	91.35	**93.03**
computer	85.20	86.87	86.35	**88.42**

Dataset	+ Brown features			Brown reg.
	lasso	**ridge**	**elastic**	
science	86.96	90.51	91.14	**93.04**
sports	82.66	88.94	85.43	**93.71**
religion	94.98	**96.93**	**96.93**	92.89
computer	55.72	**96.65**	67.57	86.36

Table 5: Classification accuracies on the 20N datasets for lasso, ridge, and elastic net models with additional LDA features (top) and Brown cluster features (bottom). The last column shows structured regularized models from Table 3.

and treat them as regular features (i.e., they do not belong to any groups and are regularized with standard regularizer such as the lasso penalty). The results in Table 5 show that for these datasets, models that incorporate this information through structured regularizers outperformed models that encode this information as additional features in 4 out 4 of cases (LDA) and 2 out of 4 cases (Brown). Sparse models with Brown clusters appear to overfit badly; recall that the clusters were learned on only the training data—clusters from a larger dataset would likely give stronger results. Of course, better performance might also be achieved by incorporating new features as well as using structured regularizers.

6.4 Examples

To gain an insight into the models, we inspect group sparsity patterns in the learned models by looking at the parameter copies $\mathbf{v}$. This lets us see which groups are considered important (i.e., “selected” vs. “removed”). For each of the proposed regularizers, we inspect the model a task in which it performed well.

For the parse tree regularizer, we inspect the model for the 20N:religion task. We observed that the model included most of the sentences (root node groups), but in some cases removed phrases from the parse trees, such as *ozzy osbourne* in the sentence *ozzy osbourne , ex-singer and main character of the black sabbath of good ole days past , is and always was a devout catholic .*

For the LDA regularizer, we inspect zero and nonzero groups (topics) in the forecasting scientific articles task. In this task, we observed that 642 out of 1,000 topics are driven to zero by our model. Table 6 shows examples of zero and nonzero topics for the dev.-tuned hyperparameter values. We can see that in this particular case, the model kept meaningful topics such as parsing and speech processing, and discarded general topics that are not correlated with the content of the papers (e.g., acknowledgment, document metadata, equation, etc.). Note that most weights for nonselected groups, even in $\mathbf{w}$, are near zero.

For the Brown cluster regularizer, we inspect the model from the 20N:science task. 771 out of 5,775 groups were driven to zero for the best model tuned on the development set. Examples of zero and nonzero groups are shown in Table 7. Similar to the LDA example, the groups that were driven to zero tend to contain generic words that are not relevant to the predictions. We can also see the tree structure effect in the regularizer. The group {*underwater*, *industrial*} was

$= 0$	"acknowledgment":	workshop arpa program session darpa research papers spoken technology systems
	"document metadata":	university references proceedings abstract work introduction new been research both
	"equation":	pr w h probability wi gram context z probabilities complete
	"translation":	translation target source german english length alignment hypothesis translations position
$\neq 0$	"translation":	**korean** translation english rules sentences parsing input evaluation machine verb
	"speech processing":	speaker **identification** topic recognition recognizer models acoustic test vocabulary independent
	"parsing":	parser parsing **probabilistic** prediction parse **pearl** edges chart phase theory
	"classification":	documents learning accuracy **bayes** **classification** wt document **naive** method selection

Table 6: Examples of LDA regularizer-removed and -selected groups (in $\mathbf{v}$) in the forecasting scientific articles dataset. Words with weights (in $\mathbf{w}$) of magnitude greater than 10^{-3} are highlighted in **red** (not cited) and **blue** (cited).

$= 0$	underwater industrial
	spotted hit reaped rejuvenated destroyed stretched undertake shake run
	seeing developing tingles diminishing launching finding investigating receiving maintaining
	adds engage explains builds
$\neq 0$	failure reproductive **ignition** reproduction
	cyanamid **planetary** **nikola** fertility **astronomical** geophysical # **lunar** cometary **supplying** astronautical
	magnetic atmospheric
	std underwater **hpr** wordscan exclusively aneutronic industrial peoples **obsessive**
	congenital **rare** **simple** **bowel** hereditary breast

Table 7: Examples of Brown regularizer-removed and -selected groups (in $\mathbf{v}$) in the 20N:science task. # denotes any numeral. Words with weights (in $\mathbf{w}$) of magnitude greater than 10^{-3} are highlighted in **red** (space) and **blue** (medical).

driven to zero, but not once it combined with other words such as *hpr*, *std*, *obsessive*. Note that we ran Brown clustering only on the training documents; running it on a larger collection of (unlabeled) documents relevant to the prediction task (i.e., semi-supervised learning) is worth exploring in future work.

7 Related and Future Work

Overall, our results demonstrate that linguistic structure in the data can be used to improve bag-of-words models, through structured regularization. State-of-the-art approaches to some of these problems have used additional features and representations (Yessenalina et al., 2010; Socher et al., 2013). For example, for the vote sentiment analysis datasets, latent variable models of Yessenalina et al. (2010) achieved a superior result of 77.67%. To do so, they sacrificed convexity and had to rely on side information for initialization. Our experimental focus has been on a controlled comparison between regularizers for a fixed model family (the simplest available, linear with bag-of-words features). However, the improvements offered by our regularization methods can be applied in future work to other model families with more carefully engineered features, metadata features (especially important in forecasting), latent variables, etc. In particular, note that other kinds of weights (e.g., metadata) can be penalized conventionally, or incorporated into the structured regularization where it makes sense to do so (e.g., n-grams, as in Nelakanti et al., 2013).

8 Conclusion

We introduced three data-driven, linguistically informed structured regularizers based on parse trees, topics, and hierarchical word clusters. We empirically showed that models regularized using our methods consistently outperformed standard regularizers that penalize features in isolation such as lasso, ridge, and elastic net on a range of datasets for various text prediction problems: topic classification, sentiment analysis, and forecasting.

Acknowledgments

The authors thank Brendan O'Connor for help with visualization and three anonymous reviewers for helpful feedback on an earlier draft of this paper. This research was supported in part by computing resources provided by a grant from the Pittsburgh Supercomputing Center, a Google research award, and the Intelligence Advanced Research Projects Activity via Department of Interior National Business Center contract number D12PC00347. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/NBC, or the U.S. Government.

References

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18:467–479.

Stanley F. Chen and Ronald Rosenfeld. 2000. A survey of smoothing techniques for me models. *IEEE Transactions on Speech and Audio Processing*, 8(1):37–50.

Xi Chen, Qihang Lin, Seyoung Kim, Jaime G. Carbonell, and Eric P. Xing. 2011. Smoothing proximal gradient method for general structured sparse learning. In *Proc. of UAI*.

Jacob Eisenstein, Noah A. Smith, and Eric P. Xing. 2011. Discovering sociolinguistic associations with structured sparsity. In *Proc. of ACL*.

Mario A. T. Figueiredo. 2002. Adaptive sparseness using Jeffreys' prior. In *Proc. of NIPS*.

Jerome Friedman, Trevor Hastie, and Robert Tibshiran. 2010. A note on the group lasso and a sparse group lasso. Technical report, Stanford University.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proc. of ICML*.

Magnus R. Hestenes. 1969. Multiplier and gradient methods. *Journal of Optimization Theory and Applications*, 4:303–320.

Arthur E. Hoerl and Robert W. Kennard. 1970. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67.

Laurent Jacob, Guillaume Obozinski, and Jean-Philippe Vert. 2009. Group lasso with overlap and graph lasso. In *Proc. of ICML*.

Rodolphe Jenatton, Jean-Yves Audibert, and Francis Bach. 2011. Structured variable selection with sparsity-inducing norms. *Journal of Machine Learning Research*, 12:2777–2824.

Mahesh Joshi, Dipanjan Das, Kevin Gimpel, and Noah A. Smith. 2010. Movie reviews and revenues: An experiment in text regression. In *Proc. of NAACL*.

Seyoung Kim and Eric P. Xing. 2008. Feature selection via block-regularized regression. In *Proc. of UAI*.

Shimon Kogan, Dimitry Levin, Bryan R. Routledge, Jacob S. Sagi, and Noah A. Smith. 2009. Predicting risk from financial reports with regression. In *Proc. of HLT-NAACL*.

Matthieu Kowalski and Bruno Torresani. 2009. Sparsity and persistence: mixed norms provide simple signal models with dependent coefficients. *Signal, Image and Video Processing*, 3(3):251–0264.

Xiaolei Lv, Guoan Bi, and Chunru Wan. 2011. The group lasso for stable recovery of block-sparse signal representations. *IEEE Transactions on Signal Processing*, 59(4):1371–1382.

Andre F. T. Martins, Noah A. Smith, Pedro M. Q. Aguiar, and Mario A. T. Figueiredo. 2011. Structured sparsity in structured prediction. In *Proc. of EMNLP*.

Anil Nelakanti, Cedric Archambeau, Julien Mairal, Francis Bach, and Guillaume Bouchard. 2013. Structured penalties for log-linear language models. In *Proc. of EMNLP*.

Kamal Nigam, Andrew McCallum, Sebastian Thrun, and Tom Mitchell. 2000. Text classification from labeled and unlabeled documents using em. *Machine Learning*, 39(2-3):103–134.

Bo Pang and Lilian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1–2):1–135.

Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Proc. of HLT-NAACL*.

M. J. D. Powell. 1969. A method for nonlinear constraints in minimization problems. In R. Fletcher, editor, *Optimization*, pages 283–298. Academic Press.

Zhiwei (Tony) Qin and Donald Goldfarb. 2012. Structured sparsity via alternating direction methods. *Journal of Machine Learning Research*, 13:1435–1468.

Dragomir R. Radev, Pradeep Muthukrishnan, and Vahed Qazvinian. 2009. The ACL anthology network corpus. In *Proc. of ACL Workshop on Natural Language Processing and Information Retrieval for Digital Libraries*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Chris Manning, Andrew Ng, and Chris Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proc. of EMNLP*.

Jiang Su, Jelber Sayyad-Shirabad, and Stan Matwin. 2011. Large scale text classication using semi-supervised multinomial naive Bayes. In *Proc. of ICML*.

Matt Thomas, Bo Pang, and Lilian Lee. 2006. Get out the vote: Determining support or opposition from congressional floor-debate transcripts. In *Proc. of EMNLP*.

Robert Tibshirani, Michael Saunders, Saharon Rosset, Ji Zhu, and Keith Knight. 2005. Sparsity and smoothness via the fused lasso. *Journal of Royal Statistical Society B*, 67(1):91–108.

Robert Tibshirani. 1996. Regression shrinkage and selection via the lasso. *Journal of Royal Statistical Society B*, 58(1):267–288.

Tae Yano, Noah A. Smith, and John D. Wilkerson. 2012. Textual predictors of bill survival in congressional committees. In *Proc. of NAACL*.

Ainur Yessenalina, Yisong Yue, and Claire Cardie. 2010. Multi-level structured models for document sentiment classification. In *Proc. of EMNLP*.

Dani Yogatama and Noah A. Smith. 2014. Making the most of bag of words: Sentence regularization with alternating direction method of multipliers. In *Proc. of ICML*.

Dani Yogatama, Michael Heilman, Brendan O'Connor, Chris Dyer, Bryan R. Routledge, and Noah A. Smith. 2011. Predicting a scientific community's response to an article. In *Proc. of EMNLP*.

Ming Yuan and Yi Lin. 2006. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68(1):49–67.

Lei Yuan, Jun Liu, and Jieping Ye. 2013. Efficient methods for overlapping group lasso. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(9):2104–2116.

Hui Zou and Trevor Hastie. 2005. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B*, 67:301–320.