Regularizing Text Categorization with Clusters of Words

Konstantinos Skianis **François Rousseau** **Michalis Vazirgiannis**

LIX, École Polytechnique, France
kskianis@lix.polytechnique.fr

Abstract

Regularization is a critical step in supervised learning to not only address overfitting, but also to take into account any prior knowledge we may have on the features and their dependence. In this paper, we explore state-of-the-art structured regularizers and we propose novel ones based on clusters of words from LSI topics, word2vec embeddings and graph-of-words document representation. We show that our proposed regularizers are faster than the state-of-the-art ones and still improve text classification accuracy. Code and data are available online[1].

1 Introduction

Harnessing the full potential in text data has always been a key task for the NLP and ML communities. The properties hidden under the inherent high dimensionality of text are of major importance in tasks such as text categorization and opinion mining.

Although simple models like bag-of-words manage to perform well, the problem of overfitting still remains. Regularization as proven in Chen and Rosenfeld (2000) is of paramount importance in Natural Language Processing and more specifically language modeling, structured prediction, and classification. In this paper we build upon the work of Yogatama and Smith (2014b) who introduce prior knowledge of data as a regularization term. One of the most popular structured regularizers, the group lasso (Yuan and Lin, 2006), was proposed to avoid large L2 norms for groups of weights.

In this paper, we propose novel linguistic structured regularizers that capitalize on the clusters learned from texts using the word2vec and graph-of-words document representation, which can be seen as group lasso variants. The extensive experiments we conducted demonstrate these regularizers can boost standard bag-of-words models on most cases tested in the task of text categorization, by imposing additional unused information as bias.

2 Background & Notation

We place ourselves in the scenario where we consider a prediction problem, in our case text categorization, as a loss minimization problem, i. e. we define a loss function $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\theta}, y)$ that quantifies the loss between the prediction $h_{\boldsymbol{\theta},b}(\boldsymbol{x})$ of a classifier parametrized by a vector of feature weights $\boldsymbol{\theta}$ and a bias b, and the true class label $y \in \boldsymbol{\mathcal{Y}}$ associated with the example $\boldsymbol{x} \in \boldsymbol{\mathcal{X}}$. Given a training set of N data points $\{(\boldsymbol{x}^i, y^i)\}_{i=1\ldots N}$, we want to find the optimal set of feature weights $\boldsymbol{\theta}^*$ such that:

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \underbrace{\sum_{i=1}^{N} \mathcal{L}(\boldsymbol{x}^i, \boldsymbol{\theta}, y^i)}_{\textbf{empirical risk}} \qquad (1)$$

In the case of logistic regression with binary predictions ($\boldsymbol{\mathcal{Y}} = \{-1, +1\}$), $h_{\boldsymbol{\theta},b}(\boldsymbol{x}) = \boldsymbol{\theta}^\top \boldsymbol{x} + b$ and $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\theta}, y) = e^{-y h_{\boldsymbol{\theta},b}(\boldsymbol{x})}$ (log loss).

2.1 Regularization

Only minimizing the empirical risk can lead to overfitting, that is, the model no longer learns the underlying pattern we are trying to capture but fits the

[1] https://goo.gl/mKqvro

noise contained in the training data and thus results in poorer generalization (e. g., lower performances on the test set). For instance, along with some feature space transformations to obtain non-linear decision boundaries in the original feature space, one could imagine a decision boundary that follows every quirk of the training data. Additionally, if two hypothesis lead to similar low empirical risks, one should select the "simpler" model for better generalization power, simplicity assessed using some measure of model complexity.

Loss+Penalty Regularization takes the form of additional constraints to the minimization problem, i. e. a budget on the feature weights, which are often relaxed into a penalty term $\Omega(\boldsymbol{\theta})$ controlled via a Lagrange multiplier λ. We refer to the book of Boyd and Vandenberghe (2004) for the theory behind convex optimization. Therefore, the overall expected risk (Vapnik, 1991) is the weighted sum of two components: the empirical risk and a regularization penalty term, expression referred to as "Loss+Penalty" by Hastie et al. (2009). Given a training set of N data points $\{(\boldsymbol{x}^i, y^i)\}_{i=1\ldots N}$, we now want to find the optimal set of feature weights $\boldsymbol{\theta}^*$ such that:

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \underbrace{\underbrace{\sum_{i=1}^{N} \mathcal{L}(\boldsymbol{x}^i, \boldsymbol{\theta}, y^i)}_{\textbf{empirical risk}} \quad + \quad \underbrace{\lambda\Omega(\boldsymbol{\theta})}_{\textbf{penalty term}}}_{\textbf{expected risk}} \tag{2}$$

L$_1$ and L2 regularization The two most used penalty terms are known as L$_1$ regularization, a. k. a. *lasso* (Tibshirani, 1996), and L2 regularization, a. k. a. *ridge* (Hoerl and Kennard, 1970) as they correspond to penalizing the model with respectively the L$_1$ and L2 norm of the feature weight vector $\boldsymbol{\theta}$:

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \sum_{i=1}^{N} \mathcal{L}(\boldsymbol{x}^i, \boldsymbol{\theta}, y^i) + \lambda \sum_{j=1}^{p} |\boldsymbol{\theta}_j| \tag{3}$$

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \sum_{i=1}^{N} \mathcal{L}(\boldsymbol{x}^i, \boldsymbol{\theta}, y^i) + \lambda \sum_{j=1}^{p} {\boldsymbol{\theta}_j}^2 \tag{4}$$

Prior on the feature weights L$_1$ (resp. L2) regularization can be interpreted as adding a Laplacian (resp. Gaussian) prior on the feature weight vector. Indeed, given the training set, we want to find the most likely hypothesis $h^* \in \boldsymbol{\mathcal{H}}$, i. e. the one with *maximum a posteriori* probability:

$$\begin{aligned}
h^* &= \underset{h\in\boldsymbol{\mathcal{H}}}{\operatorname{argmax}} \left(\mathbb{P}(h|\{(\boldsymbol{x}^i, y^i)\}_{i=1\ldots N})\right) \\
&= \underset{h\in\boldsymbol{\mathcal{H}}}{\operatorname{argmax}} \left(\frac{\mathbb{P}(\{y^i\}_i|\{\boldsymbol{x}^i\}_i, h)\,\mathbb{P}(h|\{\boldsymbol{x}^i\}_i)}{\mathbb{P}(\{y^i\}_i|\{\boldsymbol{x}^i\}_i)}\right) \\
&= \underset{h\in\boldsymbol{\mathcal{H}}}{\operatorname{argmax}} \left(\mathbb{P}(\{y^i\}_i|\{\boldsymbol{x}^i\}_i, h)\,\mathbb{P}(h|\{\boldsymbol{x}^i\}_i)\right) \\
&= \underset{h\in\boldsymbol{\mathcal{H}}}{\operatorname{argmax}} \left(\mathbb{P}(\{y^i\}_i|\{\boldsymbol{x}^i\}_i, h)\,\mathbb{P}(h)\right) \qquad (5) \\
&= \underset{h\in\boldsymbol{\mathcal{H}}}{\operatorname{argmax}} \left(\prod_{i=1}^{N} \left(\mathbb{P}(y^i|\boldsymbol{x}^i, h)\right)\mathbb{P}(h)\right) \qquad (6) \\
&= \underset{h\in\boldsymbol{\mathcal{H}}}{\operatorname{argmax}} \left(\sum_{i=1}^{N} \left(\log\mathbb{P}(y^i|\boldsymbol{x}^i, h)\right) + \log\mathbb{P}(h)\right) \\
&= \underset{h\in\boldsymbol{\mathcal{H}}}{\operatorname{argmin}} \left(\underbrace{\sum_{i=1}^{N} \left(-\log\mathbb{P}(y^i|\boldsymbol{x}^i, h)\right)}_{\textbf{empirical risk}} \underbrace{-\log\mathbb{P}(h)}_{\textbf{penalty term}}\right)
\end{aligned}$$

For the derivation, we assumed that the hypothesis h does not depend on the examples alone (Eq. 5) and that the N training labeled examples are drawn from an i.i.d. sample (Eq. 6). In that last form, we see that the loss function can be interpreted as a negative log-likelihood and the regularization penalty term as a negative log-prior over the hypothesis. Therefore, if we assume a multivariate Gaussian prior on the feature weight vector of mean vector $\mathbf{0}$ and covariance matrix $\Sigma = \sigma^2 I$ (i. e. independent features of same prior standard deviation σ), we do obtain the L2 regularization:

$$\mathbb{P}(h) = \frac{1}{\sqrt{(2\pi)^p|\Sigma|}} e^{-\frac{1}{2}\boldsymbol{\theta}^\top\Sigma^{-1}\boldsymbol{\theta}} \tag{7}$$

$$\Rightarrow -\log\mathbb{P}(h) = \frac{1}{2\sigma^2}\boldsymbol{\theta}^\top \boldsymbol{I}\boldsymbol{\theta} + \frac{p}{2}\log(2\pi\sigma)$$

$$\overset{\text{argmax}}{=} \lambda\|\boldsymbol{\theta}\|_2{}^2, \quad \lambda = \frac{1}{2\sigma^2} \tag{8}$$

And similarly, if we assume a multivariate Laplacian prior on the feature weight vector (i. e. $\boldsymbol{\theta}_i \sim Laplace(0, \frac{1}{\lambda})$), we obtain L$_1$-regularization. In practice, in both cases, the priors basically mean that we expect weights around 0 on average. The main difference between L$_1$ and L2 regularization is that the Laplacian prior will result in explicitly setting some feature weights to 0 (feature sparsity) while the Gaussian prior will only result in reducing their values (shrinkage).

2.2 Structured regularization

In L_1 and L2 regularizations, features are considered as independent, which makes sense without any additional prior knowledge. However, similar features have similar weights in the case of linear classifiers – equal weights for redundant features in the extreme case – and therefore, if we have some prior knowledge on the relationships between features, we should include that information for better generalization, i. e. include it in the regularization penalty term. Depending on how the similarity between features is encoded, e. g., through sets, trees (Kim and Xing, 2010; Liu and Ye, 2010; Mairal et al., 2010) or graphs (Jenatton et al., 2010), the penalization term varies but in any case, we take into account the structure between features, hence the "structured regularization" terminology. It should not be confused with "structured prediction" where this time the outcome is a structured object as opposed to a scalar (e. g., a class label) classically.

Group lasso Bakin (1999) and later Yuan and Lin (2006) proposed an extension of L_1 regularization to encourage groups of features to either go to zero (as a group) or not (as a group), introducing *group* sparsity in the model. To do so, they proposed to regularize with the $L_{1,2}$ norm of the feature weight vector:

$$\Omega(\boldsymbol{\theta}) = \lambda \sum_{g} \lambda_g \|\boldsymbol{\theta}_g\|_2 \tag{9}$$

where $\boldsymbol{\theta}_g$ is the subset of feature weights restricted to group g. Note that the groups can be overlapping (Jacob et al., 2009; Schmidt and Murphy, 2010; Jenatton et al., 2011; Yuan et al., 2011) even though it makes the optimization harder.

2.3 Learning

In our case we use a logistic regression loss function in order to integrate our regularization terms easily.

$$\mathcal{L}(x, \boldsymbol{\theta}, y) = \log(1 + \exp(-y\boldsymbol{\theta}^T x)) \tag{10}$$

It is obvious that the framework can be extended to other loss functions (e. g., hinge loss).

For the case of structured regularizers, there exist a plethora of optimization methods such group lasso. Since our tasks involves overlapping groups, we select the method of Yogatama and Smith (2014b). Their method uses the alternating directions method of multipliers (Hestenes, 1969; Powell, 1969).

Algorithm 1 ADMM for overlapping group-lasso

```
Require: augmented Lagrangian variable $\rho$, regularization strengths $\lambda_{glas}$ and $\lambda_{las}$
while update in weights not small do
    $\boldsymbol{\theta} = \operatorname*{argmin}_{\boldsymbol{\theta}} \Omega_{las}(\boldsymbol{\theta}) + \mathcal{L}(\boldsymbol{\theta}) + \frac{\rho}{2}\sum_{i=1}^{V} N_i(\boldsymbol{\theta}_i - \mu_i)^2$
    for $g = 1$ to $G$ do
        $\mathrm{v}_g = \mathrm{prox}_{\Omega_{glas}, \frac{\lambda_g}{\rho}}(z_g)$
    end for
    $u = u + \rho(\mathrm{v} - M\boldsymbol{\theta})$
end while
```

Now given the lasso penalty for each feature and the group lasso regularizer, the problem becomes:

$$\min_{\boldsymbol{\theta}, \mathrm{v}} \Omega_{las}(\boldsymbol{\theta}) + \Omega_{glas}(\mathrm{v}) + \sum_{d=1}^{D} \mathcal{L}(x_d, \boldsymbol{\theta}, y_d) \tag{11}$$

so that $\mathbf{v} = M\boldsymbol{\theta}$, where $\mathbf{v}$ is a copy-vector of $\boldsymbol{\theta}$. The copy-vector $\mathbf{v}$ is needed because the group-lasso regularizer contains overlaps between the used groups. M is an indicator matrix of size $L \times V$, where L is the sum of the total sizes of all groups, and its ones show the link between the actual weights $\boldsymbol{\theta}$ and their copies $\mathbf{v}$. Following Yogatama and Smith (2014b) a constrained optimization problem is formed, that can be transformed to an augmented Lagrangian problem:

$$\begin{aligned} \Omega_{las}(\boldsymbol{\theta}) + \Omega_{glas}(\mathrm{v}) + \mathcal{L}(\boldsymbol{\theta}) + \mathrm{u}^\top(\mathrm{v} - M\boldsymbol{\theta}) \\ + \frac{\rho}{2}\|\mathrm{v} - M\boldsymbol{\theta}\|_2^2 \end{aligned} \tag{12}$$

Essentially, the problem becomes the iterative update of $\boldsymbol{\theta}$, v and u:

$$\min_{\boldsymbol{\theta}} \Omega_{las}(\boldsymbol{\theta}) + \mathcal{L}(\boldsymbol{\theta}) + \mathrm{u}^\top M\boldsymbol{\theta} + \frac{\rho}{2}\|\mathrm{v} - M\boldsymbol{\theta}\|_2^2 \tag{13}$$

$$\min_{\mathrm{v}} \Omega_{glas}(\mathrm{v}) + \mathrm{u}^\top \mathrm{v} + \frac{\rho}{2}\|\mathrm{v} - M\boldsymbol{\theta}\|_2^2 \tag{14}$$

$$\mathrm{u} = \mathrm{u} + \rho(\mathrm{v} - M\boldsymbol{\theta}) \tag{15}$$

Convergence Yogatama and Smith (2014b) proved that ADMM for sparse overlapping group lasso converges. It is also shown that a good approximate solution is reached in a few tens of iterations. Our experiments confirm this as well.

3 Structured Regularization in NLP

In recent efforts there are results to identify useful structures in text that can be used to enhance the effectiveness of the text categorization in a NLP context. Since the main regularization approach we are going to use are variants of the group lasso, we are interested on prior knowledge in terms of groups/clusters that can be found in the training text data. These groups could capture either semantic, or syntactic structures that affiliate words to communities. In our work, we study both semantic and syntactic properties of text data, and incorporate them in structured regularizer. The grouping of terms is produced by either LSI or clustering in the word2vec or graph-of-words space.

3.1 Statistical regularizers

In this section, we present *statistical* regularizers, i. e. with groups of words based on co-occurrences, as opposed to *syntactic* ones (Mitra et al., 1997).

Network of features Sandler et al. (2009) introduced regularized learning with networks of features. They define a graph G whose edges are nonnegative with larger weights indicating greater similarity. Conversely, a weight of zero means that two features are not believed a priori to be similar. Previous work (Ando and Zhang, 2005; Raina et al., 2006; Krupka and Tishby, 2007) shows such similarities can be inferred from prior domain knowledge and statistics computed on unlabeled data.

The weights of G are mapped in a matrix P, where $P_{ij} \geq 0$ gives the weight of the directed edge from vertex i to vertex j. The out-degree of each vertex is constrained to sum to one, $\sum_j P_{ij} = 1$, so that no feature “dominates" the graph.

$$\Omega_{network}(\boldsymbol{\theta}) = \lambda_{net} \sum \boldsymbol{\theta}_k^\top M \boldsymbol{\theta}_k \qquad (16)$$

where $M = \alpha(I-P)^\top(I-P)+\beta I$. The matrix M is symmetric positive definite, and therefore it possesses a Bayesian interpretation in which the weight vector $\boldsymbol{\theta}$, is *a priori* normally distributed with mean zero and covariance matrix $2M^{-1}$. However, preliminary results show poorer performance compared to structured regularizers in larger datasets.

Sentence regularizer Yogatama and Smith (2014b) proposed to define groups as the sentences in the training dataset. The main idea is to define a group $d_{d,s}$ for every sentence s in every training document d so that each group holds weights for occurring words in its sentence. Thus a word can be a member of one group for every distinct (training) sentence it occurs in. The regularizer is:

$$\Omega_{sen}(\boldsymbol{\theta}) = \sum_{d=1}^{D} \sum_{s=1}^{S_d} \lambda_{d,s} \|\boldsymbol{\theta}_{d,s}\|_2 \qquad (17)$$

where S_d is the number of sentences in document d.

Since modern text datasets typically contain thousands of sentences and many words appear in more than one sentence, the sentence regularizer could potentially lead to thousands heavily overlapping groups. As stated in the work of Yogatama and Smith (2014b), a rather important fact is that the regularizer will force all the weights of a sentence, if it is recognized as irrelevant. Respectively, it will keep all the weights of a relevant sentence, even though the group contains unimportant words. Fortunately, the problem can be resolved by adding a lasso regularization (Friedman et al., 2010).

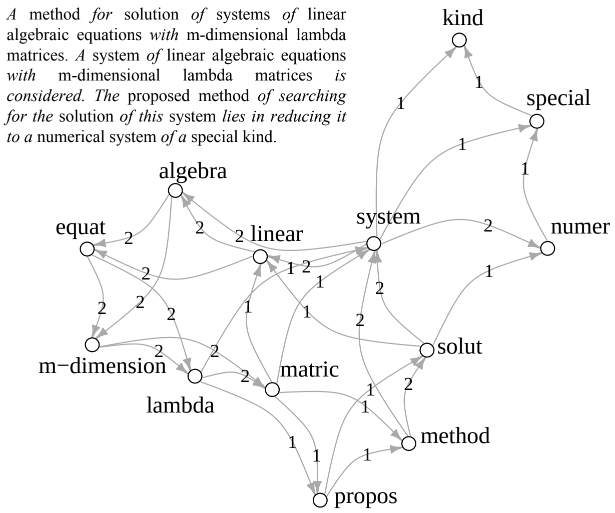
3.2 Semantic regularizers

In this section, we present *semantic* regularizers that define groups based on how semantically close words are.

LDA regularizer Yogatama and Smith (2014a) considered topics as another type of structure. It is obvious that textual data can contain a huge number of topics and especially topics that overlap each other. Again the main idea is to penalize weights for words that co-occur in the same topic, instead of treating the weight of each word separately.

Having a training corpus, topics can be easily extracted with the help of the latent Dirichlet allocation (LDA) model (Blei et al., 2003). In our experiments, we form a group by extracting the n most probable words in a topic. We note that the extracted topics can vary depending the text preprocessing methods we apply on the data.

LSI regularizer Latent Semantic Indexing (LSI) can also be used in order to identify topics or groups and thus discover correlation between terms (Deerwester et al., 1990). LSI uses singular value decomposition (SVD) on the document-term matrix to



A method *for* solution *of* systems *of* linear algebraic equations *with* m-dimensional lambda matrices. *A* system *of* linear algebraic equations *with* m-dimensional lambda matrices *is considered. The* proposed method *of searching for the* solution *of this* system *lies in reducing it to a* numerical system *of a* special kind.

Figure 1: A Graph-of-words example.

identify latent variables that link co-occurring terms with documents. The main basis behind LSI is that words being used in the same contexts (i. e. the documents) tend to have similar meanings. We used LSI as a baseline and compare it with other standard baselines as well as other proposed structured regularizers. In our work we keep the top 10 words which contribute the most in a topic.

The regularizer for both LDA and LSI is:

$$\Omega_{LDA,LSI}(\boldsymbol{\theta}) = \sum_{k=1}^{K} \lambda \|\boldsymbol{\theta}_k\|_2 \tag{18}$$

where K is the number of topics.

3.3 Graphical regularizers

In this section we present our proposed regularizers based on graph-of-words and word2vec. Essentially the word2vec space can be seen as a large graph where nodes represent terms and edges similarities between them.

Graph-of-words regularizer Following the idea of the network of features, we introduce a simpler and faster technique to identify relationships between features. We create a big collection graph from the training documents, where the nodes correspond to terms and edges correspond to co-occurrence of terms in a sliding window. We present a toy example of a graph-of-words in Figure 1.

A critical advantage of graph-of-words is that it easily encodes term dependency and term order (via edge direction). The strength of the dependence between two words can also be captured by assigning a weight to the edge that links them.

Graph-of-words was originally an idea of Mihalcea and Tarau (2004) and Erkan and Radev (2004) who applied it to the tasks of unsupervised keyword extraction and extractive single document summarization. Rousseau and Vazirgiannis (2013) and Malliaros and Skianis (2015) showed it performs well in the tasks of information retrieval and text categorization. Notably, the former effort ranked nodes based on a modified version of the PageRank algorithm.

Community detection on graph-of-words Our goal is to identify groups or communities of words. Having constructed the collection-level graph-of-words, we can now apply community detection algorithms (Fortunato, 2010).

In our case we use the Louvain method, a community detection algorithm for non-overlapping groups described in the work of Blondel et al. (2008). Essentially it is a fast modularity maximization approach, which iteratively optimizes local communities until we reach optimal global modularity given some perturbations to the current community state. The regularizer becomes:

$$\Omega_{gow}(\boldsymbol{\theta}) = \sum_{c=1}^{C} \lambda \|\boldsymbol{\theta}_c\|_2 \tag{19}$$

where c ranges over the C communities. Thus $\boldsymbol{\theta}_c$ corresponds to the sub-vector of $\boldsymbol{\theta}$ such that the corresponding features are present in the community c. Note that in this case we do not have overlapping groups, since we use a non-overlapping version of the algorithm.

As we observe that the collection-level graph-of-words does not create well separated communities of terms, overlapping community detection algorithms, like the work of Xie et al. (2013) fail to identify “good" groups and do not offer better results.

Word2vec regularizer Mikolov et al. (2013) proposed the word2vec method for learning continuous vector representations of words from large text datasets. Word2vec manages to capture the actual meaning of words and map them to a multi-dimensional vector space, giving the possibility of applying vector operations on them. We introduce another novel regularizer method, by applying unsupervised clustering algorithms on the word2vec space.

Clustering on word2vec Word2vec contains millions of words represented as vectors. Since word2vec succeeds in capturing semantic similarity between words, semantically related words tend to group together and create large clusters that can be interpreted as “topics".

In order to extract these groups, we use a fast clustering algorithm such as K-Means (Macqueen, 1967) and especially Minibatch K-means. The regularizer is:

$$\Omega_{word2vec}(\boldsymbol{\theta}) = \sum_{k=1}^{K} \lambda \|\boldsymbol{\theta}_k\|_2 \tag{20}$$

where K is the number of clusters we extracted from the word2vec space.

Clustering these semantic vectors is a very interesting area to study and could be a research topic by itself. The actual clustering output could vary as we change the number of clusters we are trying to identify. In this paper we do not focus on optimizing the clustering process.

4 Experiments

We evaluated our structured regularizers on several well-known datasets for the text categorization task. Table 1 summarizes statistics about the ten datasets we used in our experiments.

4.1 Datasets

Topic categorization. From the 20 Newsgroups[2] dataset, we examine four binary classification tasks. We end up with binary classification problems, where we classify 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. We use the 20NG dataset from Python.

Sentiment analysis. The sentiment analysis datasets we examined include movie reviews (Pang and Lee, 2004; Zaidan and Eisner, 2008)[3], floor speeches by U.S. Congressmen deciding “yea"/“nay" votes on the bill under discussion (Thomas et al., 2006)[3] and product reviews from Amazon (Blitzer et al., 2007)[4].

	dataset	train	dev	test	# words	# sents
20NG	science	949	238	790	25787	16411
	sports	957	240	796	21938	14997
	religion	863	216	717	18822	18853
	comp.	934	234	777	16282	10772
Sentiment	vote	1175	257	860	19813	43563
	movie	1600	200	200	43800	49433
	books	1440	360	200	21545	13806
	dvd	1440	360	200	21086	13794
	electr.	1440	360	200	10961	10227
	kitch.	1440	360	200	9248	8998

Table 1: Descriptive statistics of the datasets

4.2 Experimental setup

As features we use unigram frequency concatenated with an additional unregularized bias term. We reproduce standard regularizers like lasso, ridge, elastic and state-of-the-art structured regularizers like sentence, LDA as baselines and compare them with our proposed methods.

For LSI, LDA and word2vec we use the gensim package (Řehůřek and Sojka, 2010) in Python. For the learning part we used Matlab and specifically code by Schmidt et al. (2007).

We split the training set in a stratified manner to retain the percentage of classes. We use 80% of the data for training and 20% for validation.

All the hyperparameters are tuned on the development dataset, using accuracy as the evaluation criterion. For lasso and ridge regularization, we choose λ from $\{10^{-2}, 10^{-1}, 1, 10, 10^{2}\}$. 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 LDA, LSI, sentence, graph-of-words (GoW), word2vec regularizers, we perform grid search on the same set of values as ridge and lasso experiments for the ρ, λ_{glas}, λ_{las} parameters. In the case we get the same accuracy on the development data, the model with the highest sparsity is selected. For

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

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

[4] http://www.cs.jhu.edu/~mdredze/datasets/sentiment/

	dataset	no reg.	lasso	ridge	elastic	group lasso				
						LDA	LSI	sentence	GoW	word2vec
20NG	science	0.946	0.916	0.954	0.954	**0.968**	**0.968***	0.942	0.967*	**0.968***
	sports	0.908	0.907	0.925	0.920	0.959	0.964*	**0.966**	0.959*	0.946*
	religion	0.894	0.876	0.895	0.890	0.918	0.907*	**0.934**	0.911*	0.916*
	computer	0.846	0.843	0.869	0.856	0.891	0.885*	0.904	0.885*	**0.911***
Sentiment	vote	0.606	0.643	0.616	0.622	**0.658**	0.653	0.656	0.640	0.651
	movie	0.865	0.860	0.870	0.875	**0.900**	0.895	0.895	0.895	0.890
	books	0.750	0.770	0.760	0.780	0.790	0.795	0.785	0.790	**0.800**
	dvd	0.765	0.735	0.770	0.760	0.800	**0.805***	0.785	0.795*	0.795*
	electr.	0.790	0.800	0.800	**0.825**	0.800	0.815	0.805	0.820	0.815
	kitch.	0.760	0.800	0.775	0.800	0.845	**0.860***	0.855	0.840	0.855*

Table 2: Accuracy results on the test sets. Bold font marks the best performance for a dataset. * indicates statistical significance of improvement over lasso at $p < 0.05$ using micro sign test for one of our models LSI, GoW and word2vec (underlined).

	dataset	no reg.	lasso	ridge	elastic	group lasso				
						LDA	LSI	sentence	GoW	word2vec
20NG	science	100	1	100	63	19	20	86	19	21
	sports	100	1	100	5	60	11	6.4	55	44
	religion	100	1	100	3	94	31	99	10	85
	computer	100	2	100	7	40	35	77	38	18
Sentiment	vote	100	1	100	8	15	16	13	97	13
	movie	100	1	100	59	72	81	55	90	62
	books	100	3	100	14	41	74	72	90	99
	dvd	100	2	100	28	64	8	8	58	64
	electr.	100	4	100	6	10	8	43	8	9
	kitch.	100	5	100	79	73	44	27	75	46

Table 3: Fraction (in %) of non-zero feature weights in each model for each dataset: the smaller, the more compact the model.

LDA we set the number of topics to 1000 and we keep the 10 most probable words of each topic as a group. For LSI we keep 1000 latent dimensions and we select the 10 most significant words per topic. For the clustering process on word2vec we ran Minibatch-Kmeans for max 2000 clusters. For each word belonging to a cluster, we also keep the top 5 or 10 nearest words so that we introduce overlapping groups. The intuition behind this is that words can be part of multiple “concepts" or topics, thus they can belong to many clusters.

4.3 Results

In Table 2 we report the results of our experiments on the aforementioned datasets, and we distinguish our proposed regularizers LSI, GoW, word2vec with underlining. Our results are inline and confirm that of (Yogatama and Smith, 2014a) showing the advantages of using structured regularizers in the text categorization task. The group based regularizers perform systematically better than the baseline ones.

We observe that the word2vec clustering based regularizers performs very well - achieving best performance for three out of the ten data sets while it is quite fast with regards to execution time as it appears in Table 3 (i. e. it is four to ten times faster than the sentence based one).

The LSI based regularization, proposed for the first time in this paper, performs surprisingly well as it achieves the best performance for three of the ten datasets. This is somehow interpreted by the fact that this method extracts the inherent dimensions that best represent the different semantics of the documents - as we see as well in the anecdotal

	dataset	GoW	word2vec
20NG	science	79	691
	sports	137	630
	religion	35	639
	computer	95	594

Table 4: Number of groups.

					group lasso				
	dataset	lasso	ridge	elastic	LDA	LSI	sentence	GoW	word2vec
20NG	science	10	1.6	1.6	15	11	76	12	19
	sports	12	3	3	7	20	67	5	9
	religion	12	3	7	10	4	248	6	20
	computer	7	1.4	0.8	8	6	43	5	10

Table 5: Time (in seconds) for learning with best hyperparameters.

= 0	piscataway combination jil@donuts0.uucp jamie reading/seeing chambliss left-handedness abilities lubin acad sci obesity page erythromycin bottom
≠ 0	and space the launch health for use that medical you space cancer and nasa hiv health shuttle for tobacco that cancer that research center space hiv aids are use theory keyboard data telescope available are from system information space ftp

Table 6: Examples with LSI regularizer.

= 0	village town edc fashionable trendy trendy fashionable points guard guarding crown title champion champions
≠ 0	numbness tingling dizziness fevers laryngitis bronchitis undergo undergoing undergoes undergone healed mankind humanity civilization planet nasa kunin lang tao kay kong

Table 7: Examples with word2vec regularizer.

= 0	islands inta spain galapagos canary originated anodise advertises jewelry mercedes benzes diamond trendy octave chanute lillienthal
≠ 0	vibrational broiled relieving succumb spacewalks dna nf-psychiatry itself commented usenet golded insects alternate self-consistent retrospect

Table 8: Examples with graph-of-words regularizer.

examples in Table 6, 7, 8. This method proves as well very fast as it appears in Table 5 (i.e. it is three to sixty times faster than the sentence based one).

The GoW based regularization although very fast, did not outperform the other methods (while it has a very good performance in general). It remains to be seen whether a more thorough parameter tuning and community detection algorithm selection would improve further the accuracy of the method.

In Table 3 we present the feature space sizes retained by each of the regularizers for each dataset. As expected the lasso regularizer sets the vast majority of the features' weights to zero, and thus a very sparse feature space is generated. This fact has as a consequence the significant decrease in accuracy performance. Our proposed structured regularizers managed to perform better in most of the cases, introducing more sparse models compared to the state-of-the-art regularizers.

4.4 Time complexity

Although certain types of structured regularizers improve significantly the accuracy and address the problem of overfitting, they require a notable amount of time in the learning process.

As seen in Yogatama and Smith (2014b), a considerable disadvantage is the need of search for the optimal hyperparameters: λ_{glas}, λ_{lasso} , and ρ, whereas standard baselines like lasso and ridge only have one hyperparameter and elastic net has two.

Parallel grid search can be critical for finding the optimal set of hyperparameters, since there is no dependency on each other, but again the process can be very expensive. Especially for the case of the sentence regularizer, the process can be extremely slow due to two factors. First, the high number of sentences in text data. Second, sentences consist of heavily overlapping groups, that include words reappearing in one or more sentences. On the contrary, as it appears on Table 4, the number of clusters in the clustering based regularizers is significantly smaller than that of the sentences - and definitely controlled by the designer - thus resulting in much faster computation. The update of $\mathbf{v}$ still remains time consuming for small datasets, even with parallelization.

Our proposed structured regularizers are considerably faster in reaching convergence, since they offer a smaller number of groups with less overlapping between words. For example, on the computer subset of the 20NG dataset, learning models with the best hyperparameter value(s) for lasso, ridge, and elastic net took 7, 1.4, and 0.8 seconds, respectively, on an Intel Xeon CPU E5-1607 3.00 GHz machine with 4 cores and 128GB RAM. Given the best hyperparameter values the LSI regularizer takes 6 seconds to converge, the word2vec regularizer takes 10 seconds to reach convergence, the graph-of-words takes 4 seconds while the sentence regularizer requires 43 seconds. Table 5 summarizes required learning time on 20NG datasets.

We also need to consider the time needed to extract the groups. For word2vec, Minibatch K-means requires 15 minutes to cluster the pre-trained vectors by Google. The clustering is executed only once. Getting the clusters of words that belong to the vocabulary of each dataset requires 20 minutes, but can be further optimized. Finding also the communities in the graph-of-words approach with the Louvain algorithm, is very fast and requires a few minutes depending on the size and structure of the graph.

In Tables 6, 7, 8 we show examples of our proposed regularizers-removed and -selected groups (in v) in the science subset of the 20NG dataset. Words with weights (in w) of magnitude greater than 10^{-3} are highlighted in red (sci.med) and blue (sci.space).

5 Conclusion & Future Work

This paper proposes new types of structured regularizers to improve not only the accuracy but also the efficiency of the text categorization task. We mainly focused on how to find and extract semantic and syntactic structures that lead to sparser feature spaces and therefore to faster learning times. Overall, our results demonstrate that linguistic prior knowledge in the data can be used to improve categorization performance for baseline bag-of-words models, by mining inherent structures. We only considered logistic regression because of its interpretation for L2 regularizers as Gaussian prior on the feature weights and following Sandler et al. (2009), we considered a non-diagonal covariance matrix for L2 based on word similarity before moving to group lasso as presented in the paper. We are not expecting a significant change in results with different loss functions as the proposed regularizers are not log loss specific.

Future work could involve a more thorough investigation on how to create and cluster graphs, i. e. covering weighted and/or signed cases. Finding better clusters in the word2vec space is also a critical part. This is not only restricted in finding the best number of clusters but what type of clusters we are trying to extract. Gaussian Mixture Models (McLachlan and Basford, 1988) could be applied in order to capture overlapping groups at the cost of high complexity. Furthermore, topical word embeddings (Liu et al., 2015) can be considered for regularization. This approach could enhance the regularization on topic specific datasets. Additionally, we plan on exploring alternative regularization algorithms diverging from the group-lasso method.

References

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853.

Sergey Bakin. 1999. *Adaptive regression and model selection in data mining problems*. Ph.D., The Australian National University, Canberra, Australia, May.

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

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boomboxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, ACL '07, pages 440–447. ACL.

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008.

Stephen Boyd and Lieven Vandenberghe. 2004. *Convex Optimization*. Cambridge University Press, New York, NY, USA.

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.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

Günes Erkan and Dragomir R. Radev. 2004. LexRank: graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22(1):457–479.

Santo Fortunato. 2010. Community detection in graphs. *Physics reports*, 486(3):75–174.

Jerome H. Friedman, Trevor Hastie, and Robert Tibshirani. 2010. A note on the group lasso and a sparse group lasso. Technical report, Department of Statistics, Stanford University.

Trevor Hastie, Robert Tibshirani, and Jerome H. Friedman. 2009. *The elements of statistical learning*, volume 2. Springer.

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 *Proceedings of the 26th International Conference on Machine Learning*, ICML '09, pages 433–440.

Rodolphe Jenatton, Julien Mairal, Francis Bach, and Guillaume Obozinski. 2010. Proximal methods for sparse hierarchical dictionary learning. In *Proceedings of the 27th International Conference on Machine Learning*, ICML '10, pages 487–494.

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

Seyoung Kim and Eric P. Xing. 2010. Tree-guided group lasso for multi-task regression with structured sparsity. In *Proceedings of the 27th International Conference on Machine Learning*, ICML '10, pages 543–550.

Eyal Krupka and Naftali Tishby. 2007. Incorporating Prior Knowledge on Features into Learning. In *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics*, volume 2 of *AISTATS '07*, pages 227–234.

Jun Liu and Jieping Ye. 2010. Moreau-Yosida Regularization for Grouped Tree Structure Learning. In *Advances in Neural Information Processing Systems 23*, NIPS '10, pages 1459–1467.

Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. 2015. Topical word embeddings. In *Proceedings of the 29th national conference on Artificial intelligence*, pages 2418–2424.

J. Macqueen. 1967. Some methods for classification and analysis of multivariate observations. In *In 5-th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297.

Julien Mairal, Rodolphe Jenatton, Francis Bach, and Guillaume Obozinski. 2010. Network flow algorithms for structured sparsity. In *Advances in Neural Information Processing Systems 23*, NIPS '10, pages 1558–1566.

Fragkiskos D. Malliaros and Konstantinos Skianis. 2015. Graph-based term weighting for text categorization. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, pages 1473–1479.

G.J. McLachlan and K.E. Basford. 1988. *Mixture Models: Inference and Applications to Clustering*. Marcel Dekker, New York.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into texts. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, EMNLP '04, pages 404–411.

Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of Workshop at International Conference on Learning Representations*, ICLR '13.

Mandar Mitra, Chris Buckley, Amit Singhal, and Claire Cardie. 1997. An Analysis of Statistical and Syntactic Phrases. In *Proceedings of the 5th International Conference on Computer-Assisted Information Retrieval*, volume 97 of *RIAO '97*, pages 200–214.

Bo Pang and Lilian Lee. 2004. A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ACL '04, pages 271–278.

M. J. D. Powell. 1969. A method for nonlinear constraints in minimization problems. *R. Fletcher editor, Optimization*, pages 283—-298.

Rajat Raina, Andrew Y. Ng, and Daphne Koller. 2006. Constructing Informative Priors Using Transfer Learning. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 713–720.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50.

François Rousseau and Michalis Vazirgiannis. 2013. Graph-of-word and tw-idf: New approach to ad hoc ir. In *Proceedings of the 22nd ACM international conference on Information and knowledge management*, CIKM '13, pages 59–68.

Ted Sandler, John Blitzer, Partha P. Talukdar, and Lyle H. Ungar. 2009. Regularized learning with networks of features. In *Advances in Neural Information Processing Systems 22*, NIPS '09, pages 1401–1408.

Mark W. Schmidt and Kevin Murphy. 2010. Convex structure learning in log-linear models: Beyond pairwise potentials. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, AISTATS '10, pages 709–716. JMLR Workshop and Conference Proceedings.

Mark W. Schmidt, Glenn Fung, and Rómer Rosales. 2007. Fast optimization methods for L1 regularization: A comparative study and two new approaches. In *Proceedings of the 18th European Conference on Machine Learning*, ECML '07, pages 286–297.

Matt Thomas, Bo Pang, and Lillian Lee. 2006. Get out the vote: Determining support or opposition from Congressional floor-debate transcripts. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, EMNLP '06, pages 327–335.

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

Vladimir Naumovich Vapnik. 1991. Principles of Risk Minimization for Learning Theory. In *Advances in Neural Information Processing Systems 4*, NIPS '91, pages 831–838.

Jierui Xie, Stephen Kelley, and Boleslaw K. Szymanski. 2013. Overlapping community detection in networks: The state-of-the-art and comparative study. *ACM Computing Surveys*, 45(4):43:1–43:35.

Dani. Yogatama and Noah A. Smith. 2014a. Linguistic structured sparsity in text categorization. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, ACL '14, pages 786–796.

Dani Yogatama and Noah A. Smith. 2014b. Making the most of bag of words: Sentence regularization with alternating direction method of multipliers. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *ICML '14*, pages 656–664.

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

Lei Yuan, Jun Liu, and Jieping Ye. 2011. Efficient methods for overlapping group lasso. In *Advances in Neural Information Processing Systems 24*, NIPS '11, pages 352–360.

Omar Zaidan and Jason Eisner. 2008. Modeling annotators: A generative approach to learning from annotator rationales. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 31–40.