Hierarchical Bayesian Inference and Recursive Regularization for Large-scale Classification

Siddharth Gopal, Carnegie Mellon University
Yiming Yang, Carnegie Mellon University

In this paper, we address open challenges in large-scale classification, focusing on how to effectively leverage the dependency structures (hierarchical or graphical) among class labels, and how to make the inference scalable in jointly optimizing all model parameters. We propose two main approaches, namely the hierarchical Bayesian inference framework and the recursive regularization scheme. The key idea in both approaches is to reinforce the similarity among parameter across the nodes in a hierarchy or network, based on the proximity and connectivity of the nodes. For scalability, we develop hierarchical variational inference algorithms and fast dual co-ordinate descent training procedures with parallelization. In our experiments for classification problems with hundreds of thousands of classes, millions of training instances with terabytes of parameters, the proposed methods show consistent and statistically significant improvements over other competing approaches, and the best results on multiple benchmark datasets for large-scale classification.

Categories and Subject Descriptors: 1.5.2 [Pattern Recognition]: Classifier Design and Evaluation
General Terms: Design, Algorithms, Experimentation
Additional Key Words and Phrases: Large-scale Optimization, Hierarchical Classification, Bayesian methods

ACM Reference Format:
DOI: http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

With the advent of big-data, there is a growing need to provide structured and organized views of the data for effective search, browsing and data mining. The large taxonomies of Yahoo! for classifying all the web-pages on the web, the International patent classification hierarchy for organizing patents, the extensive product hierarchy of Amazon \(^1\) and Google \(^2\), the Wikipedia graph that connects related Wikipedia categories with each other for easy browsing, and the Medical Subject Heading hierarchy for indexing millions of articles in PubMed, are excellent examples of human generated structures to organize data in important applications.

\(^1\)http://docs.aws.amazon.com/AWSECommerceService/latest/DG/FindingItemsUsingBrowseNodes.html
\(^2\)https://support.google.com/merchants/answer/160081?hl=en

This work is an extension of our previous works \([15]\) and \([14]\)
Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or permissions@acm.org.
© 2010 ACM 1539-9087/2010/03-ART39 $15.00
DOI: http://dx.doi.org/10.1145/0000000.0000000

Such structures present several challenges for large-scale classification, w.r.t. how to leverage the dependencies among classes for more accurate modeling and robust prediction. For example, consider classifying a webpage to the Yahoo! Directory hierarchy, a webpage that belongs to category 'Disease' is also likely to be a member of category 'Health' but unlikely to be a member of 'Music'. One-against-rest classifiers, while most popular due to their simplicity, cannot model such dependencies because classifiers are trained independently and hence lack the necessary expressive power. To address such a limitation we must go beyond the scope of independent training of classifiers.

Another challenge is the data-sparsity issue for majority of class-labels, which is often associated with extremely skewed category distributions in very large taxonomies \cite{24,36,1}. For example, 76% of the class-labels in the Yahoo! Directory have less than 5 positive instances \cite{24} and 63% of class-labels in the English portion of Wikipedia have less than 10 positive instances. Joint learning of dependent models would be a better alternative if we could effectively ‘borrow’ the positive training examples from related classes based on the proximity of classes in the provided hierarchies or graphs. Proposing multiple strategies for such structure-based learning and global optimization of classification models is a major contribution in this paper.

Lastly, the sheer sizes of very large hierarchies and graphs present a significant computational challenge for classification algorithms. For example, the Wikipedia dataset\textsuperscript{3} consists of approximately 2.4 million Wikipedia articles spanning across 325,000 classes (with graphical dependencies) and a vocabulary size of 1.6 million words. This amounts to approximately 520 billion parameters (325,000 x 1.6 million) which is roughly 2 Terabytes of storage. This is computationally challenging even for training one-against-rest classifier per class and ignoring the dependencies among classification models. Joint learning of all the parameters and taking the dependencies among classification models into account is even more challenging. Proposing scalable algorithms for such large-scale structure-based inference, with state-of-the-art performance on benchmark datasets for large-scale classification is another major contribution in this paper.

Specifically, the major contributions of our work include;

1. **Hierarchical Bayesian Logistic Regression (HBLR):** A Bayesian framework for multivariate logistic regression where the prior distribution for the parameters at a node is a Gaussian centered at the parameters of the parent node. Such recursively propagated prior encourages the parameters of closely located nodes in the hierarchy to be similar to each other. The strength of the Gaussian prior, and hence the amount of information sharing between nodes, is controlled by its covariance parameter, which is also learned from the data. We study multiple ways of modelling the covariance structures and examine how the model flexibility affects the classification performance on various data.

2. **Recursive Regularization (RR):** In this framework we use the dependencies (hierarchical or graphical) among classes and sub-classes to define a joint objective for regularization of model parameters; the model parameters of the siblings nodes who share the same parent are regularized towards the common parent node. Depending on the choice of loss function, this framework is applicable to both large-margin classifiers (Support Vector Machines) as well as probabilistic classifiers (logistic regression).

\textsuperscript{3}http://lshtc.iit.demokritos.gr/
(3) **Scalable Inference:** For our HBLR models, we develop the first hierarchical variational inference method that is orders of magnitude faster than typical MCMC approaches. For RR, we develop fast training procedures based on local dual coordinate descent as well as variants of LBFGS [23] based optimization schemes. In both cases, we localize the dependencies among model parameters to enable efficient parallelization of the training tasks and sub-tasks. We show for the first time that global hierarchical optimization (with our proposed methods) can be efficiently computed for the largest datasets such as Wikipedia with 325,000 classes and 2 Million training instances in a matter of 37 hours.

(4) **Large-scale evaluation:** We present a thorough evaluation of our proposed methods in comparison with 7 representative baseline methods on 10 benchmark datasets, including the large Wikipedia data set with 478,021 classes. Our methods significantly outperformed the baseline methods on multiple datasets, including the largest one.

### 2. RELATED WORK

There has been more than a decade of work in hierarchical classification (HC). The most popular in the early stage are the ‘pachinko-machine models’ [11], [36], [24], [20] where the classification task is decomposed into sub-tasks recursively, and each node of the hierarchy has an independently trained classifier. The hierarchy is only used to partition the training data and not used any further in the training. The simplicity makes these methods easy to scale, but also makes them limited in effectively using the hierarchical dependencies.

Several approaches have been proposed for making better use of the hierarchical structure. In [1], a cascading strategy is employed to add the output of lower-level classifiers as additional features for higher-level classifiers. In [8], a Bayesian aggregation on the results of the individual binary classifiers was proposed. In [34], a data-driven pruning strategy is proposed for reducing the size of the original hierarchy. Some improvements over the results of the pachinko-machine models have been reported; however, these approaches are heuristic by nature.

The more principled methods include the large-margin models by [31], [5], [27], [9], [7] where the discriminant functions take the contributions from all nodes along the path to the root, and the model parameters are jointly learned to minimize a global loss over the hierarchy. Similar ideas have been explored in [38] where orthogonality conditions are imposed between the parent and children classifiers, in [28] with Bayesian multinomial logistic models, and in [26] using Naive Bayes classifiers with hierarchical shrinkage. Although there are a few works that address the scalability of these methods with large number of training instances [17], [32], there is no work that focuses on scaling with large number of classes. In fact, empirical improvements of most of these methods over simpler approaches have been shown only on small datasets with typically with hundreds (or less) of class-labels. The primary difficulty for most of these methods in scaling is due to the high-degree of inter-dependencies among model parameters and the parameters for all the classes cannot be held in memory at the same time.

### 3. HIERARCHICAL BAYESIAN LOGISTIC REGRESSION (HBLR)

HBLR uses a Bayesian framework for leveraging the hierarchical class structure. The Bayesian framework is a natural fit for this problem as it can seamlessly capture the
idea that the models at the lower levels of the hierarchy are specializations of models at ancestor nodes.

Define a hierarchy as a set of nodes \( \mathcal{N} = \{1, 2, \ldots\} \) with the parent relationship \( \pi : \mathcal{N} \rightarrow \mathcal{N} \) where \( \pi(n) \) is the parent of node \( n \in \mathcal{N} \). Let \( D = \{(x_i, t_i)\}_{i=1}^{N} \) denote the training data where \( x_i \in \mathbb{R}^d \) is an instance, \( t_i \in T \) is a label, where \( T \subset \mathcal{N} \) is the set of leaf nodes in the hierarchy labeled from 1 to \( |T| \). We assume that each instance is assigned to one of the leaf nodes in the hierarchy. If there are any instances assigned to an internal node, spawn a leaf-node under it and re-assign all the instances from the internal node to this new leaf node. Let \( C_n \) be the set of all children of node \( n \).

For each node \( n \in \mathcal{N} \), we associate a parameter vector \( w_n \), which has a Gaussian prior. We set the mean of the prior to the parameter of the parent node, \( w_{\pi(n)} \). In what follows, we consider three alternate ways to model the covariance matrix which we call M1, M2 and M3 variants of HBLR. In the M1 variant all the siblings share the same spherical covariance matrix. Formally, the generative model for M1 is

\[
\begin{align*}
M1 & \quad w_{root} \sim \mathcal{N}(w_0, \Sigma_0), \quad \alpha_{root} \sim \Gamma(a_0, b_0) \\
& \quad w_n | w_{\pi(n)}, \Sigma_{\pi(n)} \sim \mathcal{N}(w_{\pi(n)}, \Sigma_{\pi(n)}) \quad \forall n, \quad \alpha_n \sim \Gamma(a_n, b_n) \quad \forall n \notin T \\
& \quad t | x, W \sim \text{Categorical}(p_1(x), p_2(x), \ldots, p_{|T|}(x)) \quad \forall (x, t) \in D \\
& \quad p_t(x) = \exp(w_t^T x)/\sum_{t' \in T} \exp(w_{t'}^T x) 
\end{align*}
\]

The parameters of the root node are drawn using user specified parameters \( w_0, \Sigma_0, a_0, b_0 \). Each non-leaf node \( n \notin T \) has its own \( \alpha_n \) drawn from a Gamma with the shape and inverse-scale parameters specified by \( a_n, b_n \). Each \( w_n \) is drawn from the Normal with mean \( w_{\pi(n)} \) and covariance matrix \( \Sigma_{\pi(n)} = \alpha_{\pi(n)}^{-1} I \). The class-labels are drawn from a Multinomial whose parameters are a soft-max transformation of the \( w_n \)'s from the leaf nodes. This model leverages the class hierarchy information by encouraging the parameters of closely related nodes (parents, children and siblings) to be more similar to each other than those of distant ones in the hierarchy. Moreover, by using different inverse variance parameters \( \alpha_n \) for each node, the model has the flexibility to adapt the degree of similarity between the parameters (i.e. parent and children nodes) on a per family basis. For instance it can learn that sibling nodes which are higher in the hierarchy (e.g. mammals and birds) are generally less similar compared to sibling nodes lower in the hierarchy (e.g. chimps and orangutans).

Although this model is equivalent to the corrMNL proposed in [28], the hierarchical logistic regression formulation is different from corrMNL and has a distinct advantage that the parameters can be decoupled. As we shall see in Section 3.1, this enables the use of scalable and parallelizable variational inference algorithms which make our approach 750x faster than [28].

We can further extend M1 by allowing the diagonal elements of the covariance matrix \( \Sigma_{\pi(n)} \) to be feature-specific instead of uniform. In our previous example with sub-topics mammals and birds, we may want \( w_{\text{mammals}}, w_{\text{birds}} \) to be commonly close to their parent in some dimensions (e.g., in some common features like 'eyes', 'breath' and 'claw') but not in other dimensions (e.g., in 'bird' specific features like 'feathers' or 'beak'). We accommodate this by replacing prior \( \alpha_n \) using \( \alpha_n^{(i)} \) for every feature \( i \). This form of setting the prior is referred to as Automatic Relevance Determination (ARD) and forms the basis of several works such as Sparse Bayesian Learning [29], Relevance
Note that the only difference between M1 single node. M3 aims to cope up with such differences. where there is not enough training data and an entire subtree of topics is collapsed as an 'outlier' topic. Such mismatches are very typical in hierarchies; especially in cases of mammals' parent node. For example, consider topic mammals and its two sub-topics whales and carnivores; the sub-topic whales is very distinct from a typical mammal and is more of an 'outlier' topic. Such mismatches are very typical in hierarchies; especially in cases where there is not enough training data and an entire subtree of topics is collapsed as a single node. M3 aims to cope up with such differences.

Yet another extension of the M1 model would be to allow each node to have its own covariance matrix for the Gaussian prior over \( w_n \), not shared with its siblings. This enables the model to learn how much the individual children nodes differ from the parent node. For example, consider topic mammals and its two sub-topics whales and carnivores; the sub-topic whales is very distinct from a typical mammal and is more of an 'outlier' topic. Such mismatches are very typical in hierarchies; especially in cases where there is not enough training data and an entire subtree of topics is collapsed as a single node. M3 aims to cope up with such differences.

\[ \text{M3} \quad w_n \mid w_{\pi(n)}, \Sigma_n \sim \mathcal{N}(w_{\pi(n)}, \Sigma_n) \quad \forall n \]

\[ \alpha_n \sim \Gamma(a_n, b_n) \quad \forall n \notin T \]

Note that the only difference between M3 and M1 is that M3 uses \( \Sigma_n = \alpha_n^{-1} I \) instead of \( \Sigma_{\pi(n)} \) in the prior for \( w_n \). In our experiments we found that M3 consistently outperformed the other variants suggesting that such effects are important to model in hierarchies. Although it would be natural to extend M3 by placing ARD priors instead of \( \alpha_n \), we do not expect to see better performance due to the difficulty in learning a large number of parameters. Preliminary experiments confirmed our suspicions so we did not explore this direction further.

3.1. Variational Inference

We outline the inference method for M2, but the procedure can be easily extended for M1 and M3. The primary inference procedure in Bayesian methods is to calculate the posterior distribution of the parameters. Specifically, the posterior distribution of parameters in model M2 is given by

\[ p(W, \alpha | D) \propto p(t | X, W, \alpha) p(W, \alpha) \]

\[ \propto \prod_{(x,t) \in D} \sum_{T \in T} \exp(w_T^T x) \prod_{n \in N \setminus T} \prod_{i=1}^{d} p(\alpha_n(i) | a_n(i), b_n(i)) \prod_{n \in N} p(w_n | w_{\pi(n)}, \Sigma_{\pi(n)}) \]

Since closed-form solution for \( p(W, \alpha | D) \) is not possible due to the non-conjugacy between the logistic likelihood and the Gaussian prior, we resort to variational EM algorithm that approximates this posterior. More specifically, we seek such a distribution \( q \) which has a simple factored form and is closest in KL divergence to the true posterior \( p(W, \alpha | D) \). We use independent Gaussian \( q(w_n) \) and Gamma \( q(\alpha_n) \) distributions for \( w_n \) and \( \alpha_n \) per node as the factored representation:

\[ q(W, \alpha) = \prod_{n \in N \setminus T} q(\alpha_n) \prod_{n \in N} q(w_n) \propto \prod_{n \in N \setminus T} \Gamma(\tau_n(i), \nu_n(i)) \prod_{n \in N} \mathcal{N}(\cdot | \mu_n, \Psi_n) \]

Our goal is to estimate the parameters of this new factored posterior \( q(W, \alpha) \) such that it maximizes the following variational lower bound (VLB) of the likelihood of the

labels,

\[
\log P(t|X) \geq \int q(W, \alpha) \log p(W, \alpha, t|X) dW d\alpha - \int q(W, \alpha) \log q(W, \alpha) dW d\alpha
\]

\[
\Rightarrow \quad VLB = E_q[\log p(W, \alpha, t|X)] + H(q)
= E_q[\log p(t|W, \alpha, X)] + E_q[\log p(W, \alpha)] + H(q)
\tag{3}
\]

In order to tackle the non-conjugacy inside \( p(t|X, W, \alpha) \) in (2), we use a suitable lower-bound to the soft-max normalization constant proposed by [4], for any \( \beta \in \mathcal{R} \), \( \xi_k \in [0, \infty) \)

\[
\log \left( \sum_k e^{g_k} \right) \leq \beta + \sum_k \left[ \frac{g_k - \beta - \xi_k}{2} + \lambda(\xi_k)((g_k - \beta)^2 - \xi_k^2) + \log(1 + e^{\xi_k}) \right]
\]

where \( \lambda(\xi) = \frac{1}{2\xi} \left( \frac{1}{1+e^{-\xi}} - \frac{1}{2} \right) \) \tag{4}

where \( \beta, \xi_k \) are variational parameters which we can optimize to get the tightest possible bound. For every training example \( x \) we introduce variational parameters \( \beta_n \) and \( \xi_{xn} \forall n \in T \). In the E-step, the local variational parameters are fixed, and the posterior of \( \alpha \) is computed by optimizing the VLB w.r.t the posterior parameters. The parameters are updated as,

\[
\Psi_n^{-1} = I(n \in T) \sum_{(x,t) \in D} 2\lambda(\xi_{xn})xx^\top + \text{diag}(\frac{T_{\pi(n)}}{v_{\pi(n)}}) + |C_n| \text{diag}(\frac{T_n}{v_n})
\tag{5}
\]

\[
\mu_n = \Psi_n \left( I(n \in T) \sum_{(x,t) \in D} (I(t = n) - \frac{1}{2} + 2\lambda(\xi_{xn})\beta_n)x + \text{diag}(\frac{T_{\pi(n)}}{v_{\pi(n)}})\mu_{\pi(n)} + \text{diag}(\frac{T_n}{v_n}) \sum_{c \in C_n} \mu_c \right)
\]

\[
v_n^{(i)} = b_n^{(i)} + \sum_{c \in C_n} \Psi_n^{(i)} + \Psi_c^{(i)} + (\mu_n^{(i)} - \mu_c^{(i)})^2 \quad \text{and} \quad \tau_n^{(i)} = a_n^{(i)} + \frac{|C_n|}{2}
\tag{6}
\]

In the M-step, we keep the parameters of the posterior distribution fixed and optimize the variational parameters \( \xi_{xn}, \beta_n \) to maximize the VLB.

\[
\xi_{xn}^2 = x^\top \text{diag}(\frac{T_n}{v_n})x + (\beta_n - \mu_n^\top x)^2 \quad \beta_n = \frac{\frac{1}{2}(|T| - 1) + \sum_{n \in T} \lambda(\xi_{xn})\mu_n^\top x}{\sum_{n \in T} \lambda(\xi_{xn})}
\]

3.2. Partial MAP Inference

In most applications, the requirement for a matrix inversion of \( \Psi_n \) in step (5) could be demanding. In such scenarios, we split the inference into two stages, first calculating the posterior of \( w_n \) (for the leaf nodes) using MAP solution, and second calculating the posterior of \( \alpha_n \). In the first stage, we find the MAP estimate \( w_n^{\text{map}} \) (for \( n \in T \)) and then use laplace approximation to approximate the posterior using a separate Normal distribution for each dimension, thereby leading to a diagonal covariance matrix. Note
that due to the laplace approximation, $w_n^{map}$ and the posterior mean $\mu_n$ coincide.

\[
w_n^{map} = \arg \max_w \sum_{n \in T} -\frac{1}{2}(w_n - \mu_{x(n)})^\top \text{diag} \left( \frac{\tau_{n}(n)}{v_{x(n)}} \right) (w_n - \mu_{x(n)}) + \log p(t|W, X, \alpha) \tag{7}
\]

\[
(\Psi_n^{(i)})^{-1} = \sum_{(x,t) \in D} x^{(i)} p_{xn}(1 - p_{xn}) x^{(i)} + \text{diag} \left( \frac{\tau_{n}(n)}{v_{x(n)}} \right)
\]

where $p_{xn}$ is the probability that training instance $x$ is labeled as $n$. The $\arg \max$ in (7) can be computed for all $\mu_n$ at the same time using optimization techniques like LBFGS [23]. For the second stage, parameters $\tau_n$ and $\nu_n$ are updated using (6). Full MAP inference is also possible by performing an alternating maximization between $w_n, \alpha_n$ but we do not recommend it as there is no gain in scalability compared to partial MAP Inference and it loses the posterior distribution of $\alpha_n$.

### 3.3. Parallelization

For large hierarchies, it might be impractical to learn the parameters of all classes, or even store them in memory, on a single machine. We therefore, devise a parallel memory-efficient implementation scheme for our partial MAP Inference. There are 4 sets of parameters that are updated - $\{\mu_n, \Psi_n, \tau_n, \nu_n\}$. The $\Psi_n, \tau_n, \nu_n$ can be updated in parallel for each node using (5), (6).

For $\mu$, the optimization step in (7) is not easy to parallelize since the $w$’s are coupled together inside the soft-max function. To make it parallelizable we replace the soft-max function in (1) with multiple binary logistic functions (one for each terminal node), which removes the coupling of parameters inside the log-normalization constant. The optimization can now be done in parallel by making the following observations - firstly note that the optimization problem in (7) is concave maximization, therefore any order of updating the variables reaches the same unique maximum. Secondly, note that the interactions between the $w_n$’s are only through the parent and child nodes. By fixing the parameters of the parent and children, the parameter $w_n$ of a node can be optimized independently of the rest of the hierarchy. One simple way to parallelize is to traverse the hierarchy level by level, optimize the parameters at each level in parallel, and iterate until convergence. A better way that achieves a larger degree of parallelization is to iteratively optimize the odd and even levels - if we fix the parameters at the odd levels, the parameters of parents and the children of all nodes at even levels are fixed, and the $w_n$’s at all even levels can be optimized in parallel. The same goes for optimizing the odd level parameters. To aid convergence we interleave the $\mu, \Psi$ updates with the $\tau, \nu$ updates and warm-start with the previous value of $\mu_n$. In practice, for the larger hierarchies we observed speedups linear in the number of processors. Note that the convergence follows from viewing this procedure as block co-ordinate ascent on a concave differentiable function [25].

### 3.4. Setting prior parameters

The $w_0, \Sigma_0$ represent the overall mean and covariance structure for the $w_n$. We set $w_0 = 0$ and $\Sigma_0 = I$ because of their minimal effect on the rest of the parameters. The $a_n(i), b_n(i)$ are variance components such that $\frac{a_n(i)}{a_n}$ represents the expected variance of the $w_n(i)$. Typically, choosing these parameters is difficult before seeing the data. The traditional way to overcome this is to learn $\{a_n, b_n\}$ from the data using Empirical Bayes. Unfortunately, in our proposed model, one cannot do this as each $\{a_n, b_n\}$ is
associated with a single $\alpha_n$. Generally, we need more than one sample value to learn the prior parameters effectively [6].

We therefore resort to a data dependent way of setting these parameters by using an approximation to the observed Fisher Information matrix. We first derive on a simpler model and then extend it to a hierarchy. Consider the following binary logistic model with unknown $w$ and let the Fisher Information matrix be $I$ and observed Fisher Information $\hat{I}$. Given some training examples $D_{\text{binary}}$

\[
Y \mid x \sim \text{Bernoulli} \left( \frac{\exp(w^T x)}{1 + \exp(w^T x)} \right) \\
I = E \left[ p(x)(1 - p(x))xx^T \right] \\
\hat{I} = \sum_{(x,t) \in D_{\text{binary}}} \hat{p}(x)(1 - \hat{p}(x))xx^T
\]

It is well known that $I^{-1}$ is the asymptotic covariance of the MLE estimator of $w$, so reasonable guess for the covariance of a Gaussian prior over $w$ could be the observed $I^{-1}$ from a dataset $D_{\text{binary}}$. The problem with $I^{-1}$ is that we do not have a good estimate $\hat{p}(x)$ for a given $x$ as we have exactly one sample for a given $x$ i.e each instance $x$ is labeled exactly once with certainty, therefore $\hat{p}(x)(1 - \hat{p}(x))$ will always be zero. Therefore we approximate $\hat{p}(x)$ as the sample prior probability independent of $x$, i.e. $\bar{p}(x) = \bar{p} = \sum_{(x,t) \in D_{\text{binary}}} \frac{1}{|D|}$. Now, the prior on the covariance of $w$ can be set such that the expected covariance is $\hat{I}^{-1}$.

To extend this to hierarchies, we need to handle multiple classes, which can be done by estimating $I(n)^{-1}$ for each $n \in T$, as well handle multiple levels, which can be done by recursively setting $a_n, b_n$ as follows,

\[
(a_n^{(i)}, b_n^{(i)}) = \begin{cases} 
(\sum_{c \in C_n} a_c^{(i)}, \sum_{c \in C_n} b_c^{(i)}) & \text{if } n \notin T \\
(1, \hat{I}(n)^{-1}((i,i))) & \text{if } n \in T
\end{cases}
\]

where $\hat{I}(n)$ is the observed Fisher Information matrix for class label $n$. This way of setting the priors is similar to the method proposed in [19], the key differences are in approximating $\bar{p}(x)(1 - \bar{p}(x))$ from the data rather using $\bar{p}(x) = \frac{1}{2}$, extension to handle multiple classes as well as hierarchies.

We also tried other popular strategies such as setting improper gamma priors $\Gamma(\epsilon, \epsilon)$ $\epsilon \rightarrow 0$ widely used in many ARD works (which is equivalent to using type-2 ML for the $\alpha$’s if one uses variational methods [2]) and Empirical Bayes using a single $a$ and $b$ (as well as other Empirical Bayes variants). Neither of worked well, the former being to be too sensitive to the value of $\epsilon$ which is in agreement with the observations made by [12] and the latter constraining the model by using a single $a$ and $b$.

4. RECURSIVE REGULARIZATION

In general, not all dependencies are given in the form of hierarchies (for e.g. the dependencies among Wikipedia categories are given in the given in the form of a graph). To our knowledge, there has been no work that addresses the problem of classification with graphical dependencies between class-labels. The Hierarchical Bayesian approach developed before cannot be generalized to graphs as there is no notion of parent or child. By resorting to a non-Bayesian, Risk minimization framework, we develop the first discriminative methods that can leverage both hierarchical and graphical dependencies between the class-labels. The key idea is to incorporate the class-label dependencies
into the regularization structure of the parameters. Depending on the choice of the loss function the framework can support both probabilistic classifiers such as Logistic regression (RR-LR) as well as large-margin methods (RR-SVM).

To formulate the regularization framework, we resort to the Structural Risk Minimization framework which prescribes choosing a prediction function to minimize a combination of the empirical risk on the training dataset and a regularization term to penalize the complexity of the function. In the context of hierarchies, the prediction function is parameterized by \( \mathbf{W} = \{w_n : n \in \mathcal{N}\} \) and the empirical risk is defined to be the loss incurred by the instances at the leaf-nodes of the hierarchy using a loss function \( L \). The parameters are estimated by minimizing,

\[
\arg\min_{\mathbf{W}} \lambda(\mathbf{W}) + C \sum_{n \in \mathcal{T}} \sum_{i=1}^{N} L(y_{in}, x_i, w_n) \quad \text{where} \quad y_{in} = 2I(t_i = n) - 1
\]

(9)

where \( \lambda(\mathbf{W}) \) denotes the regularization term and \( C \) is a parameter that controls how much to fit to the training data. Note that unlike HBLR, each instance can belong to one or more leaf nodes in the hierarchy. We propose to use the hierarchy in the learning process by incorporating a recursive structure into the regularization term for \( \mathbf{W} \). More specifically, the regularization term is

\[
\lambda(\mathbf{W}) = \arg\min_{\mathbf{W}} \sum_{n \in \mathcal{N}} \frac{1}{2}\|w_n - w_{\pi(n)}\|^2
\]

This recursive form of regularization enforces the parameters of the node to be similar to the parameters of its parent under euclidean norm. Intuitively, it models the hierarchical dependencies in the sense that it encourages parameters which are nearby in the hierarchy to be similar to each other. This helps classes to leverage information from nearby classes while estimating model parameters and helps share statistical strength across the hierarchy. In the case of graphs, the parameters of each node in the graph is regularized towards each of its neighbours (instead of its parent and children). Specifically, given a graph with a set of edges \( E \subseteq \{(i,j) : i, j \in \mathcal{N}\} \), the regularization term is given by

\[
\lambda(\mathbf{W}) = \sum_{(i,j) \in E} \frac{1}{2}\|w_i - w_j\|^2
\]

By defining \( L \) to be the hinge-loss, we get a large-margin framework (RR-SVM) and by defining \( L \) to be the logistic loss, we get the probabilistic framework (RR-LR). The key advantage of RR-(SVM,LR) over other hierarchical models such as [31], [5], [38] is that there are no constraints that maximizes the margin between correct and incorrect predictions. This keeps the dependencies between the parameters minimal and in turn enables us to develop a parallel-iterative method to optimize the objective thereby scaling to very large problems.

\[\text{In our previous work [14],[16], the methods were denoted by HR-(SVM,LR). In this paper, we will use the notation RR-(SVM,LR) for convenience.}\]
4.1. Training RR models

4.1.1. RR-SVM. The primary optimization problem in RR-SVM is to estimate the parameter $w$,

$$
\min_w \sum_{n \in N} \frac{1}{2} ||w_n - w_{\pi(n)}||^2 + C \sum_{n \in T} \sum_{i=1}^N (1 - y_{in} w_n^T x_i) \tag{10}
$$

We resorted to an iterative approach where we update the parameters associated with each node $n$ iteratively by fixing the rest of the parameters. To tackle the non-differentiability in some of the updates (i.e. the updates at the leaf-nodes), we converted these sub-problems into their dual form which is differentiable and optimized it using co-ordinate descent. For each non-leaf node $n \notin T$, differentiating eq (10) w.r.t $w_n$ yields a closed-form update for $w_n$ which is given by,

$$
w_n = \frac{1}{|C_n| + 1} \left( \frac{w_{\pi(n)}}{|C_n|} + \sum_{c \in C_n} w_c \right) \tag{11}
$$

For each leaf node $n \in T$, the objective cannot be differentiated due to a discontinuous Hinge-loss function. Isolating the terms that depend on $w_n$ and introducing slack variables $\xi_{in}$, the primal objective of the subproblem for $w_n$ is given by,

$$
\min_{w_n} \frac{1}{2} ||w_n - w_{\pi(n)}||^2 + C \sum_{i=1}^N \xi_{in}
$$

subject to

$$
\xi_{in} \geq 0 \quad \forall i = 1..N
$$

$$
\xi_{in} \geq 1 - y_{in} w_n^T x_i \quad \forall i = 1..N
$$

The dual of the above subproblem by introducing appropriate dual variables $\alpha_i$, $i = 1..N$ is

$$
\min_{\alpha} \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_{in} y_{jn} x_i^T x_j - \sum_{i=1}^N \alpha_i (1 - y_{in} w_{\pi(n)}^T x_i) \tag{12}
$$

$$
0 \leq \alpha_i \leq C
$$

To solve this subproblem, one can easily use any second order methods such as interior-point methods etc. The downside of such solvers is that it takes a long time even for a single iteration and requires the entire kernel matrix of size $O(N^2)$ to be stored in memory. Typical large-scale HC problems have at least hundreds of thousands of instances and the memory required to store the kernel matrix is in the order of 100 GB for each class, thereby rendering it impractical. Instead we propose a co-ordinate descent approach which has minimal memory requirements and converges quickly even for large problems. Our work is based on the dual co-ordinate descent developed in [18].

The core idea in co-ordinate descent is to iteratively update each dual variable. In the objective function eq (12), the update for each dual variable has a simple closed form solution. To derive the update for the $i^{th}$ dual variable $\alpha_i$ given by $\alpha_i + d$, we solve the following one-variable problem,

$$
\min_d \frac{1}{2} d^T (x_i^T x_i) + d \left( \sum_{i=1}^N \alpha_i y_{in} x_i \right)^T x_i - d (1 - y_{in} w_{\pi(n)}^T x_i)
$$

subject to

$$
0 \leq \alpha_i + d \leq C
$$
ALGORITHM 1: Optimization of RR-SVM and RR-LR

Input: \( D, C, \pi, T, N \)
Result: weight vectors \( W^* \)
While Not Converged
For each \( n \in N \)
If \( n \notin T \)
Update \( w_n \) of the non-leaf node using equation (11)
Else
If solving RR-SVM
1. Solve the dual optimization problem (12)
2. Update the primal parameters \( w_n \) using equation (13)
Else If solving RR-LR
1. Solve optimization problem (16) using LBFGS
End
End
End

Basically, we substituted \( \alpha_i \) by \( \alpha_i + d \) in eq (12) and discarded all the terms that do not depend on \( d \). This one-variable problem can be solved in closed form by considering the gradient of the objective and appropriately updating \( \alpha_i \) to obey the constraints. The gradient \( G \) for the above objective and the corresponding update for \( \alpha_i \) is given by,

\[
G = d^T x_i - 1 + y_{in} w_{\pi(n)}^T x_i
\]

\[
\alpha_i^{new} = \min \left( \max \left( \alpha_i^{old} - \frac{G}{x_i^T x_i}, 0 \right), C \right)
\]

where \( d' = \sum_{i=1}^{N} \alpha_i y_{in} x_i \) is an auxiliary variable maintained and updated throughout the optimization of the subproblem. The time complexity for each \( \alpha_i \) update is \( O(\#nnz in x_i) \) - the number of non-zero dimensions in \( x_i \) and the memory requirement for solving the entire subproblem is \( O(N) \) - far more efficient than that \( O(N^2) \) compared to the second order methods. Finally, after solving the dual subproblem, the update for \( w_n \) in the primal form can be derived from the K.K.T conditions for the subproblem,

\[
w_n = w_{\pi(n)} + \sum_{i=1}^{N} \alpha_i y_{in} x_i
\]  

(13)

Note that the convergence of the above optimization method can be derived by viewing the procedure as a block co-ordinate descent scheme on a convex function where the blocks corresponds to parameters at each node of the hierarchy [25], [30].

For graphs, we employ a similar block co-ordinate descent algorithm by iteratively optimizing each parameter \( w_n \). The optimization subproblem at node \( n \) is given by,

\[
\min_{w_n} \frac{1}{2} \sum_{j:(n,j) \in E} ||w_n - w_j||^2 + C \sum_{i=1}^{N} (1 - y_{in} w_n^T x_i)_+
\]

(14)

In order to derive the dual of the above subproblem, we first re-write it by expanding the regularization term and discarding all terms that do not depend on \( w_n \),

\[
\min_{w_n} \frac{1}{2} S_n ||w_n||^2 - S_n w_n^T m + C \sum_{i=1}^{N} (1 - y_{in} w_n^T x_i)_+
\]
where \( S_n \) denotes the number of neighbors for node \( n \), and \( m \) denotes the mean of neighbors \( m = \frac{1}{S_n} \sum_{j:(n,j) \in E} w_j \). We now add a constant term \( \frac{1}{2} S_n \|m\|^2 \) (constant w.r.t this optimization subproblem) and divide by constant \( S_n \) and rewrite the optimization problem as,

\[
\min_{w_n} \frac{1}{2} \|w_n - m\|^2 + \frac{C}{S_n} \sum_{i=1}^N (1 - y_{in} w_n^T x_i) +
\]

Now, we solve (15) using the same co-ordinate descent technique as described earlier and recover the optimal \( w_n \).

4.1.2. RR-LR. We follow a similar iterative strategy for optimizing RR-LR, i.e. we update the parameter \( w_n \) of each node \( n \) iteratively by fixing the rest of the parameters. Unlike RR-SVM, the objective function in RR-LR is convex and differentiable, therefore we can directly use quasi newton methods such as LBFGS for each inner optimization.

The update for each non-leaf node is given in eq (11). For each leaf node \( n \), isolating the terms that depend on \( w_n \), the objective and the corresponding gradient \( G \) can be written as

\[
\min_{w_n} \frac{1}{2} \|w_n - w_{(n)}\|^2 + C \sum_{i=1}^M \log(1 + \exp(-y_{in} w_n^T x_i)) \tag{16}
\]

\[
G = w_n - w_{(n)} - C \sum_{i=1}^M \frac{1}{1 + \exp(y_{in} w_n^T x_i)} y_{in} x_i \tag{17}
\]

In the case of graphs, the only change is that instead of a single parent node, we subtract the difference from each neighbor,

\[
G = \sum_{j:(n,j) \in E} (w_n - w_j) - C \sum_{i=1}^M \frac{1}{1 + \exp(y_{in} w_n^T x_i)} y_{in} x_i
\]

Note that although HBLR and RR-LR seem like two very different models, RR-LR can be viewed a successively simplified version of HBLR. If we replace the multiclass logistic function in HBLR with multiple binary logistic functions, and have a single common fixed \( \alpha \) for all the nodes without any gamma prior, the HBLR model becomes identical to the RR-LR model - the only difference being the inference. Furthermore, if we estimate the parameters using the posterior mode i.e MAP estimate instead of the posterior mean, even the training procedures become identical.

These connections also highlight the fundamental theme in all our proposed HBLR, RR-SVM and RR-LR models i.e. to enforce similarity between model parameters based on the hierarchical or graphical dependencies between the class labels. In the HBLR models, we enforce it by propagating a Bayesion prior over the hierarchical structure, whereas in RR models we enforce it via regularization of parameters over the hierarchical or graphical structure.

4.2. Parallelization

On hierarchies, we can parallelize the optimization of the parameters exactly as discussed in section 3.3. For graphs however, parallelization is a little tricky. The ideal parallelization on graphs involves finding the chromatic number of the graph - which
Table I: Dataset Statistics

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Training</th>
<th>#Testing</th>
<th>#Class-Labels</th>
<th>#Leaf-labels</th>
<th>Depth</th>
<th>#Features</th>
<th>Avg #labels per instance</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLEF</td>
<td>10,000</td>
<td>1,006</td>
<td>87</td>
<td>63</td>
<td>4</td>
<td>89</td>
<td>1</td>
</tr>
<tr>
<td>NEWS20</td>
<td>11260</td>
<td>7505</td>
<td>27</td>
<td>20</td>
<td>3</td>
<td>53975</td>
<td>1</td>
</tr>
<tr>
<td>RCV1</td>
<td>23,149</td>
<td>784,446</td>
<td>137</td>
<td>101</td>
<td>6</td>
<td>48,734</td>
<td>3.18</td>
</tr>
<tr>
<td>IPC</td>
<td>46,324</td>
<td>28,926</td>
<td>552</td>
<td>451</td>
<td>4</td>
<td>541,869</td>
<td>1</td>
</tr>
<tr>
<td>LSHTC-small</td>
<td>4,463</td>
<td>1,858</td>
<td>1,563</td>
<td>1,139</td>
<td>6</td>
<td>51,033</td>
<td>1</td>
</tr>
<tr>
<td>DMOZ-2010</td>
<td>128,710</td>
<td>34,880</td>
<td>15,358</td>
<td>12,294</td>
<td>6</td>
<td>381,580</td>
<td>1</td>
</tr>
<tr>
<td>DMOZ-2012</td>
<td>383,408</td>
<td>103,435</td>
<td>13,347</td>
<td>11,947</td>
<td>6</td>
<td>348,548</td>
<td>1</td>
</tr>
<tr>
<td>DMOZ-2011</td>
<td>394,756</td>
<td>104,263</td>
<td>35,448</td>
<td>27,875</td>
<td>6</td>
<td>594,158</td>
<td>1.03</td>
</tr>
<tr>
<td>SWIKI-2011</td>
<td>456,886</td>
<td>81,262</td>
<td>50,312</td>
<td>36,504</td>
<td>11</td>
<td>346,299</td>
<td>1.85</td>
</tr>
<tr>
<td>LWIKI</td>
<td>2,365,436</td>
<td>452,167</td>
<td>478,020</td>
<td>325,056</td>
<td>-</td>
<td>1,617,899</td>
<td>3.26</td>
</tr>
</tbody>
</table>

is the smallest $K$ such that each node of the graph is assigned one of $K$ different colors from 1 to $K$ and no two adjacent nodes have the same color. Once the nodes have been assigned colors, we can pick a color and parallelly optimize the parameters of the nodes which have been assigned that color and repeat. However, finding the chromatic number of the graph is a NP-hard problem, therefore, we can only resort to approximate schemes to find the chromatic number. The degree of parallelization in graphs is given by $\frac{|V|}{K}$ which is in contrast to $\frac{|V|}{2}$ for hierarchies. Note that the minimum coloring needs to be solved only to achieve the best possible parallelization; in practice any reasonable coloring achieves good parallelization.

Alternatively, one could also resort to other schemes such as performing multiple iterations of optimizing all nodes in parallel (although convergence is not guaranteed in theory). Furthermore, to aid in convergence, we also used other tricks such as warm starting with the previously found dual solution and random permutation of subproblems in co-ordinate descent method [18].

5. EXPERIMENTS

5.1. Datasets

We used several large-scale benchmark datasets whose statistics are listed in Table I. To maintain comparability with previously published evaluations, we used the conventional train-test splits wherever available.

1. CLEF [10] A hierarchical collection of medical X-ray images with EHD-diagram features. The dataset was normalized to have zero mean and unit variance to improve convergence.

2. NEWS20 A collection of newsgroup posts across 20 newsgroup categories.

3. RCV1 [21] A collection of Reuters news articles from 1996 to 1997. We used the topic-based classification as it has been most popular in evaluations.

4. IPC [33] A collection of patents organized according to the International Patent Classification Hierarchy.

http://people.csail.mit.edu/jrennie/20Newsgroups/
LSHTC-small, DMOZ-2010, DMOZ-2012 and DMOZ-2011 Multiple web-page collections released as a part of the LSHTC (Large-Scale Hierarchical Text Classification) evaluation during 2010-12. It is essentially a subset of the web pages from the Open Directory Project.

SWIKI-2011, LWIKI Two subsets (small and large, respectively) of Wikipedia pages with human-generated topic class-labels. The dependencies between the class labels in SWIKI-2011 are given as links in a directed acyclic graph while in LWIKI they are given as links in an undirected graph.

Note that RCV1, DMOZ-2011, SWIKI-2011, LWIKI are multi-label datasets, meaning that an instance may have multiple correct labels; the other datasets only have one correct label per instance. For the graph based datasets, we introduced a dummy node between any two adjacent leaf-nodes - this is to prevent leaf classes from directly getting regularized towards each other (the leaf-nodes in the case of graphs refers to nodes which have associated positive training examples).

5.2. Methods for Comparison

We include four types of methods for comparison:

- **HBLR models**: Our proposed hierarchical Bayesian model as described in section 3. For ease of presentation, we mainly report the results of model M3 using partial MAP inference (M3-map) as it seemed to perform the best in our experiments. In section 6.5, we present a conclusive comparison of the different models and inference schemes.

- **RR models**: Our proposed methods Recursive Regularization methods, i.e., RR-SVM and RR-LR.

- **Flat baselines**: These include methods that cannot leverage class-label dependencies - one-versus-rest binary Support Vector Machines (BSVM), one-versus-rest regularized Binary Logistic Regression (BLR), Multiclass Support Vector Machines (MSVM), Multiclass Logistic Regression (MLR).

- **Hierarchical baselines**: We choose 4 hierarchical methods with competitive performance:
  
  (1) **Correlated Multinomial Logit** (CorrMNL) [28]: This method uses a Bayesian form of the multinomial logit model with a prior that introduces correlations between the parameters for classes that are nearby in the hierarchy. However, the model suffers from two important limitations; Firstly, sensitive hyperparameters of the model need to be tweaked and manually set by the user. This is hard, especially if the user has no prior information. Secondly, the inference is performed using MCMC sampling and therefore cannot scale to large number of classes. Due to this limitation, we report the results of this method only on the smallest CLEF dataset.

  (2) **Hierarchical SVM** (HSVM) [31]: This method is one of the most popular large-margin discriminative method for structured output prediction. We use the specific instantiation of this framework developed for hierarchical classification. The model defines a hierarchical path dependent discriminant function that minimizes a global hierarchical loss. More specifically, the parameters of the model...
are estimated by maximizing the margin between every pair of correct and incorrect label with a penalty proportional to the hierarchical distance between the labels. The resulting optimization problem is solved by using a constraint generation framework that repeatedly adds the most violated constraint to the working set. Note that this method is not applicable on multilabel datasets.

(3) **Hierarchical Orthogonal Transfer (OT)** [38]: OT is a large-margin method that encourages the classifiers at each node of the hierarchy to be different from the classifiers at its ancestors. Specifically, the regularization term is modified to enforce orthogonality between the parameters of parent and the children. The resulting optimization problem is solved using regularized dual averaging method.

(4) **Top-down SVM (TD)** [24], [11], [20] We employed the simplest variant of top-down method using support vector machine classifier. This is a popular baseline in several previous works.

We tuned the regularization parameter for all the non-Bayesian methods using cross-validation with a range of values from $10^{-3}$ to $10^3$. On the multi-label datasets, to make the baselines as competitive as possible, we used an instance-based cut-off strategy as used in [13]. This provided a better performance than using the usual cut-off of zero as well as other threshold methods like $rcut$ or $scut$ [35]. Note that HSVM, OT and HBLR (with soft-max logistic function) are inherently multiclass methods and are not applicable in multilabeled scenarios. To enable HBLR models to be applicable on multilabel datasets, we replace the soft-max function with multiple binary logistic functions and use the same instance-based cut-off strategy. The prior parameters for HBLR was set as discussed in 3.4, to enable faster convergence on the larger datasets we estimated $\hat{p}$ in eq (8) using the parameters learnt by BLR.

To scale up to the larger datasets (e.g., LWIKI, DMOZ-2011 etc), for the HBLR and RR models we used the approximate parallelization as discussed in sections 3.3, 4.2. The parallelization for the flat one-versus-rest baselines (BSVM and BLR) is straightforward, i.e., simply learn the models for all the class-labels in parallel. Among the hierarchical baselines, only TD can be easily parallelized as the class models can be trained independently. It is not known how to parallelize the rest of the methods - MSVM, HSVM and OT and hence they cannot be scaled to the larger datasets. Therefore, we only report the results of these methods on the smaller datasets where they scaled.

Our experiments on the smaller datasets - CLEF, RCV1, NEWS20, IPC and LSHTC-small were conducted on a 32 core Intel Xeon X7560 @ 2.27GHz, 32GB RAM. For the rest of the large-scale datasets we used a Hadoop with 64 worker nodes having 8 cores and 16GB RAM each. Around 300 cores were used as Mappers and 220 cores were used as reducers.

In addition, we also include the best results in benchmark evaluations for comparison, according to the numbers available on the LSHTC website.

### 5.3. Evaluation Metrics

We use the following standard evaluation metrics [37] to measure the performance of all the methods.

- **Micro-$F_1$** is a conventional metric used to evaluate classification decisions [37], [22]. Let $TP_t$, $FP_t$, $FN_t$ denote the true-positives, false-positives and false-negatives...
for the class-label \( t \in T \). The micro-averaged \( F_1 \) is
\[
P = \frac{\sum_{t \in T} TP_t}{\sum_{t \in T} TP_t + FP_t}
\]
\[
R = \frac{\sum_{t \in T} TP_t}{\sum_{t \in T} TP_t + FN_t}
\]
\[
Micro-F_1 = \frac{2PR}{P + R}
\]

— **Macro-** \( F_1 \) is also conventional metric used to evaluate classification decisions; unlike Micro-\( F_1 \) which gives equal weight to all instances in the averaging process, Macro-\( F_1 \) gives equal weight to each class-label.
\[
P_t = \frac{TP_t}{TP_t + FP_t}
\]
\[
R_t = \frac{TP_t}{TP_t + FN_t}
\]
\[
Macro-F_1 = \frac{1}{|T|} \sum_{t \in T} \frac{2PR_t}{P_t + R_t}
\]

### 6. RESULTS

We present five sets of results;

1. The first set of results compares our proposed methods with the flat baselines - BSVM, BLR, MSVM and MLR.
2. The second set of results compares our proposed methods with state-of-art hierarchical baselines - TD, HSVM and OT.
3. The third set of results analyzes the efficiency of the proposed methods against the flat and hierarchical baselines.
4. The fourth set of results compares the performance of our proposed methods with the best results on the datasets used in the benchmark evaluations conducted by LSHTC.*
5. The fifth set of results analyses the performance of the different Bayesian HBLR models as well as compares the performance against the hierarchical Bayesian baseline - corrMNL.
6. The sixth set of results analyses the performance of our approach at various levels of the hierarchy and across classes with different training set sizes.

#### 6.1. Comparison against flat baselines

Table II reports the results of our proposed models and several popular flat baselines - BSVM, MSVM, BLR and MLR. The former two methods are based on hinge-loss while the latter two methods are based on the logistic loss. On all the datasets except NEWS20, our proposed models perform better. To validate the results, we conducted pairwise significance tests between the RR and non-RR counterparts, i.e. between SVM and RR-SVM; LR and RR-LR. We used the sign test for Micro-F1 and wilcoxon rank

---

*http://lshtc.iit.demokritos.gr/lehtc2_evaluation, excluding our own own submissions to these evaluations.
Table II: Comparison against flat baselines: Macro-$F_1$ and Micro-$F_1$ on 10 datasets. Bold faced number indicates best performing method. NS denotes method is not scalable to the dataset, NA denotes the method is not applicable since the dataset is multilabeled or has graph-based dependencies. The significance-test results between RR-SVM and BSVM, RR-LR and BLR on the first five datasets are denoted † for a p-value less than 5% and †† for p-value less than 1%.

<table>
<thead>
<tr>
<th></th>
<th>BSVM</th>
<th>BLR</th>
<th>MSVM</th>
<th>MLR</th>
<th>RR-SVM</th>
<th>RR-LR</th>
<th>HBLR (M3-map)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLEF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Macro-$F_1$</td>
<td>48.59</td>
<td>53.26</td>
<td>54.33</td>
<td>54.76</td>
<td>53.92†</td>
<td>55.83†</td>
<td>59.65</td>
</tr>
<tr>
<td>Micro-$F_1$</td>
<td>77.53</td>
<td>79.92</td>
<td>80.02</td>
<td>80.52</td>
<td>80.02†</td>
<td>80.12†</td>
<td>81.41</td>
</tr>
<tr>
<td>RCV1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Macro-$F_1$</td>
<td>54.72</td>
<td>53.38</td>
<td>NA</td>
<td>NA</td>
<td>56.56†</td>
<td>55.81†</td>
<td>56.08</td>
</tr>
<tr>
<td>Micro-$F_1$</td>
<td>80.82</td>
<td>80.08</td>
<td></td>
<td></td>
<td>81.66†</td>
<td>81.23†</td>
<td>81.98</td>
</tr>
<tr>
<td>NEWS20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Macro-$F_1$</td>
<td>82.32</td>
<td>82.17</td>
<td>81.73</td>
<td>81.82</td>
<td>82.00</td>
<td>82.06</td>
<td>81.69</td>
</tr>
<tr>
<td>Micro-$F_1$</td>
<td>83.10</td>
<td>82.97</td>
<td>82.47</td>
<td>82.56</td>
<td>82.78</td>
<td>82.86</td>
<td>82.56</td>
</tr>
<tr>
<td>IPC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Macro-$F_1$</td>
<td>45.71</td>
<td>48.29</td>
<td>NS</td>
<td>NS</td>
<td>47.89†</td>
<td>49.60</td>
<td>51.06</td>
</tr>
<tr>
<td>Micro-$F_1$</td>
<td>53.12</td>
<td>55.03</td>
<td></td>
<td></td>
<td>54.26†</td>
<td>55.37†</td>
<td>56.02</td>
</tr>
<tr>
<td>LSHTC-small</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Micro-$F_1$</td>
<td>45.21</td>
<td>44.94</td>
<td>45.62</td>
<td>45.20</td>
<td>45.31</td>
<td>45.11</td>
<td>46.03</td>
</tr>
</tbody>
</table>

DMOZ-2010              |      |      |      |      |        |       |                |
| Macro-$F_1$ | 32.64 | 31.58 | NS   | NS   | 33.12  | 32.42  | 31.88          |
| Micro-$F_1$ | 45.36 | 45.40 |      |      | 46.02  | 45.84  | 45.64          |

DMOZ-2012              |      |      |      |      |        |       |                |
| Macro-$F_1$ | 31.59 | 14.18 | NS   | NS   | 33.05  | 20.04  | 27.19          |
| Micro-$F_1$ | 56.44 | 52.79 |      |      | 57.17  | 53.18  | 53.15          |

DMOZ-2011              |      |      |      |      |        |       |                |
| Macro-$F_1$ | 24.34 | 21.67 | NA   | NA   | 25.69  | 23.90  | 23.46          |
| Micro-$F_1$ | 42.88 | 41.29 |      |      | 43.73  | 42.27  | 41.35          |

SWIKI-2011              |      |      |      |      |        |       |                |
| Macro-$F_1$ | 26.57 | 19.51 | NA   | NA   | 28.72  | 24.26  | NA             |
| Micro-$F_1$ | 40.87 | 37.65 |      |      | 41.79  | 40.99  | NA             |

LWIKI                  |      |      |      |      |        |       |                |
| Macro-$F_1$ | 19.89 | 18.65 | NA   | NA   | 22.31  | 20.22  | NA             |
| Micro-$F_1$ | 37.66 | 36.96 |      |      | 38.08  | 37.67  |                |
Table III: **Comparison against hierarchical baselines**: Macro-$F_1$ and Micro-$F_1$ on 10 datasets. Bold faced number indicates best performing method. **NS** denotes method is not scalable to the dataset, **NA** denotes the method is not applicable since the dataset is multilabeled or has graph-based dependencies.

The significance test results are between RR-SVM, RR-LR and HBLR against the scalable hierarchical baseline - TD on the first five datasets (†† denotes significance at 1% level)

<table>
<thead>
<tr>
<th></th>
<th>TD</th>
<th>HSVM</th>
<th>OT</th>
<th>RR-SVM</th>
<th>RR-LR</th>
<th>HBLR (M3-map)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CLEF</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Macro-$F_1$</td>
<td>32.32</td>
<td>57.23</td>
<td>37.12</td>
<td>53.92††</td>
<td>55.83††</td>
<td><strong>59.65††</strong></td>
</tr>
<tr>
<td>Micro-$F_1$</td>
<td>70.11</td>
<td>79.72</td>
<td>73.84</td>
<td>80.02††</td>
<td>80.12††</td>
<td><strong>81.41††</strong></td>
</tr>
<tr>
<td><strong>RCV1</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Macro-$F_1$</td>
<td>34.15</td>
<td>NA</td>
<td>NA</td>
<td>56.56††</td>
<td>55.81††</td>
<td><strong>56.08††</strong></td>
</tr>
<tr>
<td>Micro-$F_1$</td>
<td>71.34</td>
<td></td>
<td></td>
<td>81.66††</td>
<td>81.23††</td>
<td><strong>81.98††</strong></td>
</tr>
<tr>
<td><strong>NEWS20</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Macro-$F_1$</td>
<td>80.86</td>
<td>80.04</td>
<td>81.20</td>
<td>82.00††</td>
<td><strong>82.06††</strong></td>
<td>81.69††</td>
</tr>
<tr>
<td>Micro-$F_1$</td>
<td>81.20</td>
<td>80.79</td>
<td>81.98††</td>
<td>82.78††</td>
<td><strong>82.86††</strong></td>
<td>82.56††</td>
</tr>
<tr>
<td><strong>IPC</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Macro-$F_1$</td>
<td>42.62</td>
<td>NS</td>
<td>NS</td>
<td>47.89††</td>
<td>49.60††</td>
<td><strong>51.06††</strong></td>
</tr>
<tr>
<td>Micro-$F_1$</td>
<td>50.34</td>
<td></td>
<td></td>
<td>54.26††</td>
<td>55.37††</td>
<td><strong>56.02††</strong></td>
</tr>
<tr>
<td><strong>LSHTC-small</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Macro-$F_1$</td>
<td>20.01</td>
<td>21.95</td>
<td>19.45</td>
<td>28.94††</td>
<td><strong>28.48††</strong></td>
<td><strong>30.81††</strong></td>
</tr>
<tr>
<td>Micro-$F_1$</td>
<td>38.48</td>
<td>39.66</td>
<td>37.12</td>
<td>45.31††</td>
<td><strong>45.11††</strong></td>
<td><strong>46.03††</strong></td>
</tr>
</tbody>
</table>

**No significance tests reported on the following datasets due to lack of test labels**

|         |             |             |             |             |             |               |
| **DMOZ-2010** |             |             |             |             |             |               |
| Macro-$F_1$ | 22.30       | NS          | NS          | **33.12**   | 32.42       | 31.88         |
| Micro-$F_1$ | 38.64       |             |             | **46.02**   | 45.84       | 45.64         |
| **DMOZ-2012** |             |             |             |             |             |               |
| Macro-$F_1$ | 30.01       | NS          | NS          | **33.05**   | 20.04       | 27.19         |
| Micro-$F_1$ | 55.14       |             |             | **57.17**   | 53.18       | 53.15         |
| **DMOZ-2011** |             |             |             |             |             |               |
| Macro-$F_1$ | 21.07       | NA          | NA          | 25.69       | 23.90       | 23.46         |
| Micro-$F_1$ | 35.91       |             |             | **43.73**   | 42.27       | 41.35         |
| **SWIKI-2011** |             |             |             |             |             |               |
| Macro-$F_1$ | 17.39       | NA          | NA          | **28.72**   | 24.26       | NA            |
| Micro-$F_1$ | 36.65       |             |             | **41.79**   | 40.99       |               |

RR-models significantly outperform the non-RR models on three out of the five tested datasets.

Comparing the HBLR and RR models, on the smaller datasets like CLEF, RCV1, NEWS20 and IPC, the HBLR model M3-map offers the best performance, while on the larger datasets like DMOZ and Wikipedia, RR-SVM gives the best performance. This observation suggests that the choice of loss function affects the performance; hinge-loss seems to work better on the larger datasets (with high dimensions and skewed class-label distributions), while logistic loss is suited to datasets with balanced class labels.
Hierarchical Bayesian Inference and Recursive Regularization for Large-scale Classification

6.2. Comparison against hierarchical baselines

Table III compares the performance of our proposed methods against three hierarchical baselines - TD, HSVM and OT. On all the datasets, the proposed methods are able to leverage the hierarchy better and outperform existing hierarchical methods. In fact, on some datasets like LSHTC-small, there is a 16% relative improvement in performance. We conducted pairwise significance tests between the most scalable hierarchical baseline TD against our proposed methods - HBLR and RR. We used the setup as described in the previous section, sign test for Micro-$F_1$ and wilcoxon rank test for Macro-$F_1$. All the results are statistically significant.

6.3. Scalability Analysis

Table IV reports the training times taken for all the methods. Among the flat baselines, the multiclass classifiers - MLR and MSVM cannot even be scaled to datasets with a moderate number of class labels. The one-versus-rest baselines on the other hand are very scalable and faster than our proposed models. BSVM is on average 1.92x faster than RR-SVM, BLR is on average 2.87x faster than RR-LR, and BLR is on average 4.89x faster than M3-map. This is not surprising - the better performance of our models comes at the cost of increased computational time. However even on the largest dataset LWIKI, RR-SVM takes about 37 hours, although slower than BSVM, the computation time certainly falls within the tractable range.

Among the hierarchical baselines, TD is the only one that can be scaled to the larger datasets, although with a low performance. HSVM and OT could only scale to the smallest datasets (CLEF and LSHTC-small) and both of them are orders of magnitude slower than our proposed methods. On the rest of the datasets neither of them could be tested successfully either due to scalability issues or modelling (inability to handle multilabel data or graph-based dependencies) issues.

Between HBLR and RR-LR there is an efficiency vs effectiveness tradeoff. Although HBLR achieves a better performance than RR-LR with a 7% and 5% improvement in Macro-$F_1$ and Micro-$F_1$, it is 1.62x slower than RR-LR. The choice of the method depends on the user’s preference between performance and scalability.

6.4. Comparison against established benchmark results

Table V compares the results of our proposed models with the well established results on the large-scale datasets released by the LSHTC community. We focus on only those datasets for which benchmark evaluations were available on the website. Table V shows that the our proposed models are able to perform better than the state-of-the-art results reported so far on most of these datasets. In fact, on four out of the five datasets, RR-SVM shows a consistent 10% relative improvement than the currently published results.

6.5. Comparison of HBLR models and CorrMNL

First, we analyse the performance our HBLR models - M1, M2 and M3. Table VI shows the results of the different models on four datasets using partial MAP inference. The performance of M3 seems to be consistently better than M1, followed by M2. Although
Table IV: **Computational efficiency**: The training time (in mins) for all the methods.

<table>
<thead>
<tr>
<th>Flat baselines</th>
<th>Hierarchical Baselines</th>
<th>Proposed methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSVM</td>
<td>BLR</td>
<td>MSVM</td>
</tr>
<tr>
<td>CLEF</td>
<td>.15</td>
<td>.24</td>
</tr>
<tr>
<td>RCV1</td>
<td>.273</td>
<td>2.89</td>
</tr>
<tr>
<td>NEWS20</td>
<td>.04</td>
<td>.11</td>
</tr>
<tr>
<td>IPC</td>
<td>3.12</td>
<td>4.17</td>
</tr>
<tr>
<td>LSHTC-small</td>
<td>.31</td>
<td>1.93</td>
</tr>
<tr>
<td>DMOZ-2010</td>
<td>5.12</td>
<td>97.24</td>
</tr>
<tr>
<td>DMOZ-2012</td>
<td>22.31</td>
<td>95.38</td>
</tr>
<tr>
<td>SWIKI-2011</td>
<td>54</td>
<td>99.46</td>
</tr>
<tr>
<td>LWIKI</td>
<td>1114.23</td>
<td>2134.46</td>
</tr>
</tbody>
</table>

Table V: **Comparison against established benchmark results** Macro-$F_1$ and Micro-$F_1$ of benchmark results established by the LSHTC\(^7\) (we excluded our own submissions to the system). Bold faced number indicates best performing method. NA denotes that the method is not applicable on that dataset due to graph based dependencies between class labels.

<table>
<thead>
<tr>
<th>LSHTC Published Results</th>
<th>RR-SVM</th>
<th>RR-LR</th>
<th>HBLR (M3-map)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMOZ-2010</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Macro-$F_1$</td>
<td>34.12</td>
<td>33.12</td>
<td>32.42</td>
</tr>
<tr>
<td>Micro-$F_1$</td>
<td>46.76</td>
<td>46.02</td>
<td>45.84</td>
</tr>
<tr>
<td>DMOZ-2012</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Macro-$F_1$</td>
<td>31.36</td>
<td>33.05</td>
<td>20.04</td>
</tr>
<tr>
<td>Micro-$F_1$</td>
<td>51.98</td>
<td>57.17</td>
<td>53.18</td>
</tr>
<tr>
<td>DMOZ-2011</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Macro-$F_1$</td>
<td>26.48</td>
<td>25.69</td>
<td>23.90</td>
</tr>
<tr>
<td>Micro-$F_1$</td>
<td>38.85</td>
<td>43.73</td>
<td>42.27</td>
</tr>
<tr>
<td>SWIKI-2011</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Macro-$F_1$</td>
<td>23.16</td>
<td>28.72</td>
<td>24.26</td>
</tr>
<tr>
<td>Micro-$F_1$</td>
<td>37.39</td>
<td>41.79</td>
<td>40.99</td>
</tr>
<tr>
<td>LWIKI</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Macro-$F_1$</td>
<td>18.68</td>
<td>22.31</td>
<td>20.22</td>
</tr>
<tr>
<td>Micro-$F_1$</td>
<td>34.67</td>
<td>38.08</td>
<td>37.67</td>
</tr>
</tbody>
</table>

M2 is more expressive than M1, the benefit of the expressiveness seems to be offset by the difficulty in learning a larger number of parameters.

Next, we analyze the performance of the HBLR models against the MCMC based Bayesian hierarchical baseline - CorrMNL [28]. For corrMNL, we performed sampling for 2500 iterations with 1000 for burnin. We present the result of our models using full variational inference \{M1,M2,M3\}-var as well as partial MAP inference \{M1,M2,M3\}-map. With regards to scalability, partial MAP inference is orders of magnitude faster.
Table VI: Comparison of HBLR models using partial MAP inference: Macro-$F_1$ and Micro-$F_1$ the three models - M1, M2 and M3. Bold faced number indicates best performing model.

<table>
<thead>
<tr>
<th></th>
<th>CLEF</th>
<th>NEWS20</th>
<th>LSHTC-small</th>
<th>IPC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>M1-map</td>
<td>M2-map</td>
<td>M3-map</td>
<td></td>
</tr>
<tr>
<td>Macro-$F_1$</td>
<td>55.53</td>
<td>54.76</td>
<td>59.65</td>
<td></td>
</tr>
<tr>
<td>Micro-$F_1$</td>
<td>80.88</td>
<td>80.25</td>
<td>81.41</td>
<td></td>
</tr>
<tr>
<td>Macro-$F_1$</td>
<td>81.54</td>
<td>80.91</td>
<td>81.69</td>
<td></td>
</tr>
<tr>
<td>Micro-$F_1$</td>
<td>82.24</td>
<td>81.54</td>
<td>82.56</td>
<td></td>
</tr>
<tr>
<td>Macro-$F_1$</td>
<td>28.32</td>
<td>24.93</td>
<td>28.76</td>
<td></td>
</tr>
<tr>
<td>Micro-$F_1$</td>
<td>43.98</td>
<td>43.11</td>
<td>44.05</td>
<td></td>
</tr>
<tr>
<td>Macro-$F_1$</td>
<td>50.43</td>
<td>47.45</td>
<td>51.06</td>
<td></td>
</tr>
<tr>
<td>Micro-$F_1$</td>
<td>55.80</td>
<td>54.22</td>
<td>56.02</td>
<td></td>
</tr>
</tbody>
</table>

Table VII: Comparison with CorrMNL: Macro-$F_1$ and Micro-$F_1$ of the HBLR models and CorrMNL the CLEF dataset

<table>
<thead>
<tr>
<th></th>
<th>CorrMNL</th>
<th>{M1,M2,M3}-var</th>
<th>{M1,M2,M3}-map</th>
<th>{M1,M2,M3}-flat</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>M1</td>
<td>M2</td>
<td>M3</td>
<td>M1</td>
</tr>
<tr>
<td>Macro-$F_1$</td>
<td>55.59</td>
<td>56.67</td>
<td>51.23</td>
<td>59.67</td>
</tr>
<tr>
<td>Micro-$F_1$</td>
<td>81.10</td>
<td>81.21</td>
<td>79.92</td>
<td>81.61</td>
</tr>
<tr>
<td>Time (mins)</td>
<td>2279</td>
<td>79</td>
<td>81</td>
<td>80</td>
</tr>
</tbody>
</table>

than CorrMNL (750x), followed by full variational inference (20x). In terms of performance, M3-var offers the best performance followed by M3-map. MAP inference has only a small drop in performance compared to full variational inference while being 27x faster than full variational inference.

6.6. Detailed Analysis

We present detailed analysis of the results of our approach on two different fronts (a) the performance improvement at various levels of the hierarchy and (b) the performance improvement on the classes with fewer training examples. Due to the lack of test-set labels (see section 6.1), we partitioned the training data from one of the datasets (DMOZ-2012) with a 50%-50% train-test split.

Table VIII reports the Macro-averaged $F_1$ score at various levels of the hierarchy. On all the levels our proposed RR-SVM performs better than SVM. The improvement seems to be particularly high at the lower levels (4, 5 and 6) where the leaf nodes are located.

Table IX reports the Macro-averaged $F_1$ score for classes with different training set sizes. The improvement is highest in classes with moderately small number of training examples (6-50 training examples). The improvement seems to be smaller when the number of training examples is too low (1-5) or higher (> 100). This is expected because, in the former case, learning a good classifier is generally very hard, and in the
Table VIII: The Macro-$F_1$ of BSVM, RR-SVM at various levels of the hierarchy along with the number of nodes at each level. The improvement in percentage of RR-SVM over BSVM is shown in parenthesis.

<table>
<thead>
<tr>
<th>Level</th>
<th>BSVM</th>
<th>RR-SVM</th>
<th># Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100.0</td>
<td>100.0</td>
<td>(0.00%)</td>
</tr>
<tr>
<td>2</td>
<td>71.43</td>
<td>72.50</td>
<td>(1.50%)</td>
</tr>
<tr>
<td>3</td>
<td>41.62</td>
<td>42.93</td>
<td>(3.12%)</td>
</tr>
<tr>
<td>4</td>
<td>25.71</td>
<td>26.73</td>
<td>(3.93%)</td>
</tr>
<tr>
<td>5</td>
<td>26.54</td>
<td>27.27</td>
<td>(3.03%)</td>
</tr>
<tr>
<td>6</td>
<td>13.06</td>
<td>16.72</td>
<td>(28.08%)</td>
</tr>
</tbody>
</table>

Table IX: The Macro-$F_1$ of BSVM, RR-SVM across classes with different training set sizes. The number of classes in each range of training set size is also shown. The improvement in percentage of RR-SVM over BSVM is shown in parenthesis.

<table>
<thead>
<tr>
<th># Examples</th>
<th>BSVM</th>
<th>RR-SVM</th>
<th># Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-5</td>
<td>20.24</td>
<td>20.62</td>
<td>(1.86%)</td>
</tr>
<tr>
<td>6-10</td>
<td>24.84</td>
<td>26.59</td>
<td>(6.57%)</td>
</tr>
<tr>
<td>11-20</td>
<td>32.16</td>
<td>34.08</td>
<td>(5.63%)</td>
</tr>
<tr>
<td>21-50</td>
<td>39.94</td>
<td>41.23</td>
<td>(3.12%)</td>
</tr>
<tr>
<td>51-100</td>
<td>49.25</td>
<td>50.26</td>
<td>(1.99%)</td>
</tr>
<tr>
<td>&gt; 100</td>
<td>59.81</td>
<td>60.64</td>
<td>(1.38%)</td>
</tr>
</tbody>
</table>

latter case, the training examples already adequate and the hierarchy does not help further.

7. CONCLUSION

In this paper, we proposed two frameworks for large-scale classification - HBLR and RR that can leverage hierarchical and graphical dependencies between class-labels for improving classification. The proposed methods rely on enforcing similarity between the model parameters based on the proximity of the classes in the provided structure. For scalability, we developed fast hierarchical variational inference algorithms and efficient training procedures for both the frameworks. Our proposed models achieved state-of-the-art results on multiple benchmark datasets and showed a consistent improvement in performance over both flat as well as other hierarchical methods. For future work, we will explore the possibility of incrementally training the system as new training examples are provided for the rarer classes.

ACKNOWLEDGMENTS

This work is supported, in part, by the National Science Foundation (NSF) under grant IIS_1216282. We thank Alexandru Niculescu-Mizil for useful discussions about model design and parallelization, Guy Blelloch for sharing his computational resources, and the Open Cloud cluster for the Hadoop framework. The Open Cloud cluster is supported, in part, by NSF, under award CCF_1019104, and the Gordon and Betty Moore Foundation, in the eScience project.
REFERENCES


C.M. Bishop. Pattern recognition and machine learning. 8

C.M. Bishop and M.E. Tipping. Bayesian regression and classiﬁcation. 2003. 5

G. Bouchard. Efﬁcient bounds for the softmax function, applications to inference in hybrid models. 2007. 6


George Casella. Empirical bayes method - a tutorial. Technical report. 8


A. Gelman. Prior distributions for variance parameters in hierarchical models. BA. 8


Siddharth Gopal and Yiming Yang. Recursive regularization for large-scale classiﬁcation with hierarchical and graphical dependencies. In Special Interest Group in Knowledge Discovery and Data Mining (KDD), 2013. 1, 9


D. Koller and M. Sahami. Hierarchically classifying documents using very few words. 1997. 3, 15


M.E. Tipping. Sparse bayesian learning and the relevance vector machine. JMLR, 1:211–244, 2001. 4


