arXiv:1705.00740v1 [stat.ML] 1 May 2017

Regularizing Model Complexity and Label Structure for Multi-Label Text Classification

Bingyu Wang
Northeastern University
Boston, MA 02115
rainicy@ccs.neu.edu

Cheng Li
Northeastern University
Boston, MA 02115
chengli@ccs.neu.edu

Virgil Pavlu
Northeastern University
Boston, MA 02115
vip@ccs.neu.edu

Javed Aslam
Northeastern University
Boston, MA 02115
jaa@ccs.neu.edu

ABSTRACT

Multi-label text classification is a popular machine learning task where each document is assigned with multiple relevant labels. This task is challenging due to high dimensional features and correlated labels. Multi-label text classifiers need to be carefully regularized to prevent the severe over-fitting in the high dimensional space, and also need to take into account label dependencies in order to make accurate predictions under uncertainty. Most of the existing multi-label classifier designs focus on incorporating label dependencies into model training phase and optimize a strict set-accuracy measure; in practice a partial reward like F-measure makes more sense for set labels such as tags, ontologies, medical codes, movie genres, etc.

We demonstrate significant and practical improvement by carefully regularizing the model complexity during training phase, and also regularizing the label search space during prediction phase. Specifically, we regularize the classifier training using *Elastic-net* (L1+L2) penalty for reducing model complexity/size, and employ *early stopping* to prevent overfitting. At prediction time, we apply *support inference* to restrict the search space to label sets encountered in the training set, and F-optimizer *GFM* to make optimal predictions for the F1 metric. We show that although support inference only provides density estimations on existing label combinations, when combined with GFM predictor, the algorithm can output unseen label combinations.

Taken collectively, our experiments show state of the art results on many benchmark datasets. Beyond performance and practical contributions, we make some interesting observations. Contrary to the prior belief, which deems support inference as purely an approximate inference procedure, we show that support inference acts as a strong regularizer on the label prediction structure. It allows the classifier to take into account label dependencies during prediction even if the classifiers had not modeled any label dependencies during training.

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 bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
KDD'17, Halifax, Nova Scotia - Canada
© 2017 Copyright held by the owner/author(s). 978-x-xxxx-xxxx-x/YY/MM…$15.00
DOI: 10.1145/nnnnnnn.nnnnnnn

KEYWORDS

Multi-label Classification, Regularization, F Measure

ACM Reference format:
Bingyu Wang, Cheng Li, Virgil Pavlu, and Javed Aslam. 2017. Regularizing Model Complexity and Label Structure for Multi-Label Text Classification. In *Proceedings of KDD'17, Halifax, Nova Scotia - Canada, August 13 - 17, 2017,* 9 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Classifying a document into one of the predefined categories has been well studied, and many multi-class classifiers can be applied to solve this problem. In practice, however, text documents are often naturally associated with more than one label, and it is desirable to find all the relevant labels for the document. For example, in a medical note, a patient may present multiple illnesses or undergo a procedure with multiple billing codes; in news categorization, an article can be associated with multiple topics. This problem can be formulated as a multi-label classification task, which seeks to predict a *subset* of possible labels for a data object.

Formally, in a multi-label classification problem, we are given a set of label candidates $\mathcal{L} = \{1, 2, ..., L\}$. Every data point $\mathbf{x} \in \mathbb{R}^D$ matches a subset of labels $\mathbf{y} \subseteq \mathcal{L}$, which is often also written in the form of a binary label vector $\mathbf{y} \in \{0, 1\}^L$, with each bit y_l indicating the presence or absence of the corresponding label. The goal of learning is to build a classier $h : \mathbb{R}^D \mapsto \{0, 1\}^L$ which maps an instance to a subset of labels. The label subset $\mathbf{y}$ can be of arbitrary size (written as $|\mathbf{y}| = ||\mathbf{y}||_1$). Multi-label generalizes binary and multi-class: when the size is restricted to be 1, the problem is called multi-class; if the total number of label candidates L is 2, the problem is binary classification.

1.1 Challenges in Multi-label Text Classification

Multi-label text classification is a challenging task for at least two reasons: 1) the labels exhibit complex dependency structures, 2) the features are high dimensional and sparse. Text labels labels such as `election` and `politics` are *dependent* variables in general, so an independent prediction (also called Binary Relevance) is unlikely to work well [21]. Labels can be *many*, so learning approaches dealing explicitly with each of the exponential number of label subsets (e.g. PowerSet method [21]) are infeasible; even when feasible,

they suffer from scarce data and limited label subsets observed during training. In recent years, there have been an growing interest in developing multi-label methods that are capable of modeling label dependencies while at the same time avoid the exponential computational complexity. Examples include probabilistic classifier chains [20], conditional random fields [7] , and conditional Bernoulli mixtures [12].

The second challenge in multi-label text classification is that the commonly used bag-of-words feature representation is high dimensional and sparse. For example, the WISE dataset has 301,561 unigram features and 203 labels. If ngram features are further included, the feature size would grow dramatically (by at least a factor of 10). Model training without careful regularization can lead to severe over-fitting. This is especially a problem for high capacity classifiers which aim to model complex label dependencies, such as conditional Bernoulli mixtures [12] which can get very high training accuracy without improving generalization ability. The large number of features, together with the large number of labels, also make the produced classifiers large in size, as measured by the storage space they occupy. Even for the most simplistic binary relevance method, which allocates one logistic regression for each label, the model size grows linearly with the product of the number of labels and number of features. For more sophisticated methods, the model sizes will grow even faster. The training regularization steps (elastic-net, early stopping) are addressing these issues directly; we used well established ideas for both of these; but we provide an efficient and public implementation tailored to specific multi-label algorithms, analyze the specific effects, and draw novel conclusions.

1.2 F1 Metric and Optimal F Predictions

The most widely used evaluation measure for multi-label text classification is the F1 metric [22], which assigns partial credit to "almost correct" answers and handles label imbalance well. Let $\mathbf{x}$ be an instance (document) with ground truth label vector $\mathbf{y}$, and $\mathbf{y}'$ be a prediction label vector made. **F1-metric** is defined as

$$F(\mathbf{y}, \mathbf{y}') = \frac{2\sum_{l=1}^{L} y_l y'_l}{\sum_{l=1}^{L} y_l + \sum_{l=1}^{L} y'_l}, \tag{1}$$

which can also be written as the harmonic mean between precision and recall:

$$Precision(\mathbf{y}, \mathbf{y}') = \frac{\sum_{l=1}^{L} y_l y'_l}{\sum_{l=1}^{L} y'_l} \qquad Recall(\mathbf{y}, \mathbf{y}') = \frac{\sum_{l=1}^{L} y_l y'_l}{\sum_{l=1}^{L} y_l} \tag{2}$$

The reported F1 on a dataset is the average of per-instance F1 values. There exist a few methods which explicitly take into account the F1 metric during training [6, 16, 17], but most of the popular methods that provide a joint estimation in the form of $p(\mathbf{y}|\mathbf{x})$ are trained by standard maximum likelihood estimation without considering F1 metric as objective. For such methods, it is still possible to use an F1 optimal prediction strategy post-training, that is, output $\mathbf{y}^*$ which maximizes the expected F1:

$$\mathbf{y}^* = \arg\max_{\mathbf{y}'} \mathbf{E}_{\mathbf{y}\sim p(\mathbf{y}|\mathbf{x})}[F(\mathbf{y}, \mathbf{y}')] = \arg\max_{\mathbf{y}'} \sum_{\mathbf{y}} p(\mathbf{y}|\mathbf{x}) \cdot F(\mathbf{y}, \mathbf{y}')$$

The General F-measure Maximizer(**GFM**) algorithm [22] is an efficient algorithm that finds the F1 optimal prediction for a given instance based on some probability estimations. The GFM algorithm does not work directly with a joint estimation $p(\mathbf{y}|\mathbf{x})$, but rather, some L^2 marginal distributions (which will be defined more precisely in Section 3.2). The paper [22] proposed two ways of getting these L^2 marginals (or probabilities per testing instance): (1) a new classifier that directly estimates these marginals from data; (2) use a probabilistic joint classifier/estimator $p(\mathbf{y}|\mathbf{x})$ and sampling to generate the required L^2 probabilities.

We find that option (1) is a very difficult, if not unsolvable in practice, problem — although it is indeed appealing as a theoretical exercise. We instead develop an efficient solution based on (2) but make a critical change: we first train a joint estimator $p(\mathbf{y}|\mathbf{x})$, and then derive the required marginals using an approximation called **support inference**. These marginals are then fed into GFM to produce the F1 optimal prediction. We show that the proposed solution is reasonably simple to implement, and very effective. In the experimental section we analyze the support inference procedure as a strong regularizer on the label structures: the final optimal-F prediction takes into account label dependencies even if the joint estimation had not modeled any label dependencies during training, and even if the optimization objective for the joint was not related to F.

In summary, we are using several previous ideas to put together a good regularization for both features/training and labels/testing. Perhaps the biggest points we make are the new explanations of what works and why. Specifically we make the following contributions:

- regularize multi-label text classifier training use Elastic-net penalty (L1 + L2) and early stopping, in order to avoid over-fitting in the high dimensional space, as well as to reduce the model size.
- regularize multi-label prediction by combining support inference and GFM predictor. Support inference allows GFM to work conveniently with well developed probabilistic classifiers, and GFM outputs possibly unseen label combinations, addressing the limitation of support inference.
- show that the most elementary multi-label method — binary classifier for each label independently — can be made almost as accurate as the most sophisticated methods, by employing the support inference and the GFM predictor during prediction time (post-training).
- reveal that the effectiveness of the pair-wise CRF model is likely not due to its ability to model pair-wise label dependencies, but mostly due to the support inference used in its implementation.
- apply the proposed regularization techniques to several classifiers and achieve consistent improvements and state-of-the-art performance on many multi-label text datasets.
- code publicly available at https://github.com/cheng-li/pyramid.

2 MULTI-LABEL CLASSIFIERS

We briefly review several probabilistic multi-label classifiers that we shall use in the study. For each method, we describe its probabilistic formulation $p(\mathbf{y}|\mathbf{x})$ and its MAP inference method, which gives $\arg\max_{\mathbf{y}} p(\mathbf{y}|\mathbf{x})$. We emphasize that MAP inference, although commonly used as the default choice, only provides the optimal prediction for the exact-set accuracy measure, but not for the F1 measure [4, 22]. In later sections, we will compare MAP prediction with the F1 optimal prediction. All these classifiers, except CRF, employ logistic regression as base learners, following the common practice in text classification.

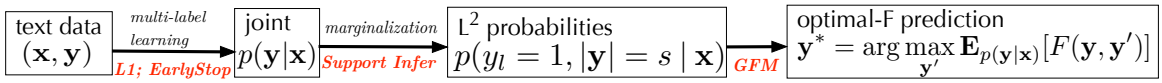

Figure 1: Our proposed pipeline with 4 regularization steps : L1 and EarlyStop (training); GFM and Support Inference (prediction). All learners also use L2 regularization.

Binary Relevance (BR) [21] assumes that all labels are independent and thus the joint over all labels is a product of marginals

$$p(\mathbf{y}|\mathbf{x}) = \prod_{\ell=1}^{L} p(y_l|\mathbf{x})$$

This independence assumption simplifies both training and prediction. During training, L binary classifiers (logistic regressions) are trained, one for each label. The MAP inference simply predicts each label independently, based on the marginals.

Probabilistic Classifier Chain (PCC) [20] decomposes the joint density estimator into a product of conditionals using the chain rule:

$$p(\mathbf{y}|\mathbf{x}) = p(y_1|\mathbf{x})p(y_2|\mathbf{x}, y_1)\cdots p(y_L|\mathbf{x}, y_1, .., y_{L-1})$$

During training, one logistic regression is trained for each label based on both features and all previous labels; it models label dependency, but the pre-set order of labels in the decomposition is critical. The exact MAP inference is generally intractable. Beam search is often used as an approximate MAP inference procedure [10].

Pair-wise Conditional Random Field (CRF) [7] defines potential functions for each feature-label pair and each label-label pair and builds a log-linear model.

$$\begin{aligned} p(\mathbf{y}|\mathbf{x}) = &\frac{1}{Z(\mathbf{x})}\exp\{\sum_{l=1}^{L}\sum_{d=1}^{D} w_{ld}x_d\mathbb{1}[y_l = 1] \\ &+\sum_{l=1}^{L}\sum_{m=1}^{L}(w_{lm1}\mathbb{1}[y_l = 0, y_m = 0] \\ &+w_{lm2}\mathbb{1}[y_l = 0, y_m = 1] \\ &+w_{lm3}\mathbb{1}[y_l = 1, y_m = 0] \\ &+w_{lm4}\mathbb{1}[y_l = 1, y_m = 1])\} \end{aligned}$$

where $Z(\mathbf{x})$ is the normalization constant, and all w are learned weights. All weights are trained jointly using maximum likelihood estimation. Both computing the partition function $Z(\mathbf{x})$ and the exact MAP inference are intractable, as an exponential number of label combinations are involved. [7] suggests to use support inference to solve the intractability issues: the idea is to restrict $\mathbf{y}$ values only to label combinations observed in the training set when computing scores and probabilities. While the authors propose support inference only as an approximation, we think it is actually the main reason CRF works well: support inference adds strong regularization and dependency effects; the praised CRF design of pairwise dependencies has a smaller contribution on performance (can even be skipped) once the support inference is applied, at least on these datasets, as we shall show in our experiments.

Conditional Bernoulli Mixtures (CBM) [12] represent the joint as a mixture of K components, each with independent label classifiers

$$p(\mathbf{y}|\mathbf{x}) = \sum_{k=1}^{K}\pi(z = k|\mathbf{x})\prod_{\ell=1}^{L} b(y_\ell|\mathbf{x}, z = k)$$

The multi-class classifier (a multi-nomial logistic regression) π decides the mixing coefficient for each mixture component. Inside each component, the joint is factorized into marginals, estimated by L binary classifiers b (binary logistic regressions). Both the multi-class classifier and the binary classifiers are trained jointly by EM algorithm. The exact MAP can be computed efficiently using dynamic programming.

3 REGULARIZATION FOR TEXT DATA

We apply two independent regularizations: during training we regularize the basic learner for feature selection and model size, using the "Elastic-net" penalty and an "Early Stopping" criteria.

During prediction, we apply the "GFM" predictor to obtain an optimal instance-F prediction, and also the "Support Inference" which limits the prediction and some calculations to the label combinations observed in training set. These two work well together in that the "Support Inference" forces label dependencies in the model (even for independent classifiers), while the "GFM" beside optimizing F1, it allows for new label combinations to be predicted.

3.1 Training Regularization: Model Complexity

L1 Regularization. We take for granted that all base learners are L2 regularized (not debated in this paper). L2 regularization spreads weights across all correlated features; this makes the model more robust during testing. However, L2 does not perform any feature selection. High capacity multi-label classifiers trained on high dimensional text features without careful feature selection can lead to severe over-fitting. One remedy for the high dimensionality is to apply the L1 regularization (LASSO) or the combinations of L1 and L2, which is also called Elastic-net [5]. L1 regularization shrinks some feature weights to zero and thus can completely eliminate irrelevant features from the model, thus performing powerful label-driven feature selection. However, L1 alone can pick one out of many highly correlated features, possibly hurting the generalization — so L2 is necessary. Elastic-net regularization, with the form $\lambda\{\alpha||w||_1 + (1-\alpha)||w||_2^2\}$, combines both L1 and L2 regularization and can get the best of both worlds, when properly tuned. Both the overall strength λ and the relative ratio α can be tuned on a validation set.

For all multi-label methods except CRF, we implement elasic-net penalty to their linear base classifiers (i.e. binary and multi-nominal logistic regressions), using the coordinate descent training algorithm described in [5, 23]. The elastic-net regularized multinomial Logistic

Regression with C classes (which includes binary logistic regression as a special case) is trained by minimizing the loss

$$\frac{1}{N}\sum_{i=1}^{N}\log p_{g_i}(\mathbf{x}_i) - \lambda\sum_{c=1}^{C}[(1-\alpha)||w_c||_2^2 + \alpha||w_c||_1],$$

where

$$p_c(x) = p(g=c|\mathbf{x}) = \frac{e^{w_{0c}+\mathbf{x}^T w_c}}{\sum_{c=1}^{C} e^{w_{0c}+\mathbf{x}^T w_c}}.$$

Regularizing CRF with elastic-net penalty is considerably more complicated [11], and not considered in this paper (only L2 regularization is employed for CRF).

Early Stopping. It is often the case that even equipped with L1 and L2 regularization, the classifier can still over-fit the training data if the training algorithm is left to run until full convergence. Early stopping seeks to find the best termination point by monitoring the validation performance. Early stopping can be seen as a more general and adaptive regularizer [8]. So we also apply early stopping to all multi-label methods.

3.2 Prediction Regularization: Label Structure

The regularization techniques just described can potentially provide a good probabilistic joint model $p(\mathbf{y}|\mathbf{x})$. Then the next question is how to use this joint to make an accurate prediction for a given test instance $\mathbf{x}$. By "accurate", we mean a prediction that gives high F1 score when compared with the ground truth. Of course, the ground truth is unknown during prediction time, so we use Bayes optimal prediction to find the $\mathbf{y}^*$ that gives the highest expected F1 score:

$$\mathbf{y}^* = \arg\max_{\mathbf{y}'} \mathbf{E}_{\mathbf{y}\sim p(\mathbf{y}|\mathbf{x})}[F(\mathbf{y},\mathbf{y}')] = \arg\max_{\mathbf{y}'}\sum_{\mathbf{y}} p(\mathbf{y}|\mathbf{x})\cdot F(\mathbf{y},\mathbf{y}')$$

GFM: optimal prediction for F measure. The General F-Measure Maximizer (**GFM**) [22] algorithm (see Algorithm 1) is an exact and efficient algorithm for computing $\mathbf{y}^*$.

However, the GFM algorithm does not work directly with a joint estimation $p(\mathbf{y}|\mathbf{x})$, but rather, some marginal distributions of the form

$$p(y_l = 1, |\mathbf{y}| = s \mid \mathbf{x}), \quad \forall l, s \in \{1, ..., L\},$$

where $|\mathbf{y}|$ stands for the number of relevant labels in $\mathbf{y}$. This formula can be read as, for example, "the probability of the given document having $s = 5$ relevant labels and `election` is one of them". There are (no more than) L^2 probabilities in this form, per instance.

The LSF baseline. It may appear that if the end goal is to produce these L^2 probabilities as input to the GFM algorithm, it should be more straight foreword to estimate these L^2 probabilities directly rather than going through a joint estimation $p(\mathbf{y}|\mathbf{x})$ first, which by itself is quite challenging (the joint evolves 2^L probabilities in general). In fact, it is conceptually not hard to derive an algorithm (we name it **LSF**) to estimate the L^2 marginals directly, as suggested in [22]. We can estimate each $p(y_l = 1, |\mathbf{y}| = s \mid \mathbf{x})$ using a binary logistic regression. Another option is to estimate $p(y_l = 1|\mathbf{x})$ with a binary logistic regression, and then $p(|\mathbf{y}| = s \mid \mathbf{x}, y_l = 1)$ with another multinomial logistic regression and then multiply their probabilities. However, in practice, we observe that this direct estimation approach does not perform well. Our speculation is that directly predicting the number of relevant labels $|\mathbf{y}|$ by a classifier is a very hard and unnatural task.

Algorithm 1 General F-Measure Maximizer : given L^2 probabilities of the form $r_{ls} = p(y_l = 1, |\mathbf{y}| = s \mid \mathbf{x}); \forall l, s \in \{1, ..., L\}$ finds the $h(\mathbf{x})$ in $O(L^3)$ time.

1: **Input:** $p(\mathbf{y} = \mathbf{0}|\mathbf{x})$ and L by L Matrix P with elements $p_{ls} = p(y_\ell = 1, |\mathbf{y}| = s \mid \mathbf{x})$
2: Define L by L Matrix W with elements $w_{sk} = \frac{2}{s+k}$
3: Compute L by L Matrix $\Delta = PW$
4: **for** $s = 1, 2, ..., L$ **do**
5: The best prediction $\mathbf{y}^s$ with s labels is given by including the s labels l with the highest Δ_{ls}
6: Compute the expected F measure $\mathbf{E}[F(\mathbf{y}, \mathbf{y}^s)] = \sum_{\ell=1}^{L} y_l^s \Delta_{\ell s}$
7: The expected F measure for the empty prediction $y^0 = \mathbf{0}$ is $\mathbf{E}[F(\mathbf{y}, \mathbf{y}^0)] = p(\mathbf{y} = \mathbf{0}|\mathbf{x})$.
8: **Output:** optimal $\mathbf{y}^* = \text{argmax}_{0\le s\le L}\, \mathbf{E}[F(\mathbf{y}, \mathbf{y}^s)]$

Each category here ("matching s labels") does not have a fixed meaning, like `sport` or `economy` do in typical topical classification. As "matching 2 labels" can mean many different things, for example (`sport`, `basketball`) but also (`business`, `industry`), it is hard to establish a relation between "matching 2 labels" and feature representation. This is especially so for linear models like logistic regression, where probabilities are monotonic functions of scores, which in turn are monotonic functions of features.

Support Inference. Estimating the joint $p(\mathbf{y}|\mathbf{x})$ is a more natural formulation and is what most of the methods do. Now we discuss how to make GFM work with the existing joint estimator. First, it is generally impossible to compute the required L^2 marginal probabilities analytically from the joint estimation. [22] proposed to sample from the joint and then compute the required probabilities based on the samples. Sampling works best when it takes advantage fo the classifier structuure: For BR each label can be sampled independently; in PCC one can sample labels one by one, using ancestral sampling; in CBM one can first sample a component, and from the component sample each label independently. However sampling can be very inefficient for large L and in particular low-confidence ("flat") joint that allows probability mass to spread over many label combinations.

Support Inference not only allows us to infer the required marginals from the joint, but also provides some additional regularization effect on the label structures. We say a combination $\mathbf{y}$ is a support combination if it appears at least once in the training set. For each support combination $\mathbf{y}$, we compute $p(\mathbf{y}|\mathbf{x})$ using the trained classifier. We then re-normalize all support combination probabilities to make them sum up to 1 (this is only for conceptual demonstration purpose; in practice, this step can be skipped as the constant factor does not affect the final GFM prediction). By doing so, we have put all the probability mass on only support combinations. Using these limited number (usually no more than a few thousands, see Table 1) of support combinations and their associated probabilities, we can then easily compute the required marginals.

To illustrate how support inference could possibly help regularizing label structures and modeling label dependencies, let us consider a toy example. Suppose we have a multi-class problem with 3 labels (which can be seen as a special multi-label task). Then the only valid combinations are the singleton sets $\{1\}$, $\{2\}$ and $\{3\}$. Binary

	MEDICAL	BIBTEX	IMDB	OHSUMED	ENRON	RCV1	TMC	WISE	WIPO
domain	medical	bkmark	genre	medical	email	news	reports	articles	patent
labels	45	159	27	23	53	101	22	203	188
label sets	90	2,058	2,122	1,042	653	494	1,172	3,536	155
features	679	1,836	27,228	16,344	21,190	47,236	49,060	301,561	74,435
instances	978	7,395	34,157	13,929	1,702	6,000	28,596	64,857	1,710
cardinality	1.25	2.40	2.52	1.66	3.38	3.23	2.16	1.45	4.00
inst/label	27	112	2537	1007	108	188	2,805	463	36

Table 1: Datasets characteristics; cardinality is the average number of labels of the instances and instances/label denotes the average number of training instances of the labels. The IMDB dataset is put together by us from different sources, the other ones are publicly available.

relevance may assign substantial probabilities to the invalid set $\{1, 2\}$ or the empty set $\emptyset$. Pairwise CRF can learn that no two labels should co-occur and thus can largely avoid $\{1, 2\}$, but it cannot learn that exactly one out of three labels has to be picked, since this rule involves higher order dependencies; as a result, pairwise CRF still considers the empty label set $\mathbf{y} = \emptyset$ as a valid candidate. But when support inference is employed, both binary relevance and CRF will assign zero posterior probabilities to these invalid empty sets. Thus support inference can be seen as an infinite strong prior which regularizes the posterior to only consider observed label combinations. In practice, this prediction-time regularization often allows a classifier to capture label dependencies that the classifier itself did not capture during training.

Admittedly, support inference has the limitation in that no new label combination will be considered *during marginalization*. The datasets used in this study have only few new label combinations in the test set; but even if the data would have such new label combinations, this support-inference limitation is largely mitigated by GFM: GFM can output a $\mathbf{y}^*$ that maximizes the expected F1 even when $\mathbf{y}^*$ is not in the support and thus $p(\mathbf{y}^*|\mathbf{x}) = 0$. Using the example above and assuming support inference gives $p(\{1\}|\mathbf{x}) = 0.5$, $p(\{2\}|\mathbf{x}) = 0.4$ and $p(\{3\}|\mathbf{x}) = 0.1$ and $p(\mathbf{y}|\mathbf{x}) = 0$ for all other $\mathbf{y}$. In this case, the F1 optimal prediction is $\mathbf{y}^* = \{1, 2\}$, with expected F1 $0.5 \times \frac{2}{3} + 0.4 \times \frac{2}{3} + 0.1 \times 0 = 0.6$. As a comparison, the joint mode $\mathbf{y} = \{1\}$ (which maximizes the set accuracy) has an expected F1 of only $0.5 \times 1 + 0.4 \times 0 + 0.1 \times 0 = 0.5$. Thus support inference provides a regularized probability estimation by only assigning probability mass to observed combinations, and GFM takes this regularized probability estimation as the input and outputs F optimal prediction, which could potentially be an unobserved label combination. We observe this strategy to work remarkably well for many classifiers and datasets.

4 EXPERIMENTAL RESULTS & ANALYSIS

In this section, we empirically verify the effectiveness of the proposed regularization techniques L1+L2, early stopping, support inference and GFM for 5 multi-label methods on 9 multi-label text datasets.

4.1 Datasets and Experiment Setup

The 9 multi-label text datasets used are shown in Table 1. We adopt the given train/test split whenever it is provided; otherwise we use a random 20% of the data as the test set. We further split 20% data from the training data as the validation set. Hyper parameter tuning for all algorithms is done on the validation set and F1 metric on the test set is reported. For methods involving random initializations or sampling, reported results are averaged over 3 runs. When we apply L1 regularization, we use L1 penalty together with the basic L2 penalty in the elastic-net form $\lambda\{\alpha||w||_1 + (1-\alpha)||w||_2^2\}$, and we tune the overall strength λ and the L1 ratio α. When L1 penalty is not included, we only keep L2 penalty by setting $\alpha = 0$ and we only tune λ. For early stopping, we tune the optimal number of training iterations on the validation set. When early stopping is not applied, the training is left running until full convergence. Support inference and GFM do not contain any tunable hyper parameters. When GFM is not used, we perform MAP inference instead. The number of mixture components in CBM is fixed as 20.

4.2 L1 Regularization and Early Stopping

First we analyze the regularization effects of L1 penalty and early stopping during training. The experiment results are summarized in in Table 2. The "No REG" column does not use any of the proposed training or prediction regularization techniques (Elastic-net, early stopping, support inference, GFM). "No REG" follows the convention that uses only L2 penalty to regularize logistic regression learners, trains each model until full convergence, and perform MAP inference during prediction. This column serves as a baseline. All other columns have the prediction strategy fixed as support inference + GFM, and only vary the training regularizations. The letters "L, E, S, G" in the table stand for L1, early stopping, support inference and GFM, respectively, and "All4" stands for "L+E+S+G", i.e., using all four techniques. We did not implement L1 regularization for CRF, and thus do not include CRF in this table. Comparing the column "L+S+G" with the column "S+G", we can see the difference due to L1. The results show that how L1 influences the results depends more on the datasets, and less on the classifiers. The performance of all classifiers generally improves on MEDICAL, BIBTEX, OHSUMED, RCV1, WISE, and WIPO, but not on IMDB, ENRON and TMC.

One can imagine that each dataset has some intrinsic properties such as the number of relevant documents per label, the diversity of the topics, the total number of documents and the total number of features, that dictate how many features have to be used in order to explain the given labels/topics well and how many model parameters

Data	Model	No REG	S+G	L+S+G	E+S+G	All4 REG
MEDICAL	BR	68.8	80.5	81.1	80.3	**81.1**
	LSF	-	81.3	81.6	81.4	**82.1**
	PCC	73.6	78.3	80.6	78.3	**80.8**
	CBM	79.2	80.1	**82.6***	81.3	**82.6***
BIBTEX	BR	37.8	44.5	**45.5**	45.4	45.4
	LSF	-	42.5	43.3	43.2	**43.9**
	PCC	37.6	45.3	45.8	45.3	**47.3**
	CBM	44.0	45.9	47.3	46.5	**49.5***
IMDB	BR	59.4	61.8	**61.4**	63.1	**61.4**
	LSF	-	59.9	**60.0**	59.4	59.8
	PCC	59.6	**63.9**	62.8	**63.9**	62.8
	CBM	61.6	65.1	65.0	**65.4***	65.2
OHSUMED	BR	60.2	67.9	68.8	68.3	**69.1**
	LSF	-	64.2	64.5	64.2	**65.0**
	PCC	62.5	70.1	69.2	70.1	**70.4**
	CBM	68.7	70.3	71.1	70.1	**71.7***
ENRON	BR	57.9	**61.1**	**61.1**	60.4	**61.1**
	LSF	-	**58.0**	55.2	57.8	57.6
	PCC	60.1	61.0	**61.5**	61.0	**61.5**
	CBM	58.2	59.8	60.9	61.1	**62.5***
RCV1	BR	72.1	73.7	**75.1**	73.8	**75.1**
	LSF	-	73.4	73.5	73.4	**73.6**
	PCC	71.0	73.6	**74.1**	73.6	**74.1**
	CBM	76.6	77.3	78.9	77.7	**79.2***
TMC	BR	61.6	**63.2**	63.1	62.4	63.1
	LSF	-	58.9	55.9	**60.0**	57.6
	PCC	61.9	**63.4**	63.2	**63.4**	63.2
	CBM	59.8	61.7	62.0	**63.8***	**63.8***
WISE	BR	68.0	77.3	**79.6**	78.6	79.3
	LSF	-	75.9	76.5	76.3	**76.7**
	PCC	70.7	76.0	77.2	76.1	**78.0**
	CBM	77.9	78.6	**80.4***	78.7	80.3
WIPO	BR	63.4	71.2	73.1	71.9	**74.0**
	LSF	-	65.0	71.0	65.2	**71.1**
	PCC	68.8	71.5	71.2	70.7	**72.3**
	CBM	63.0	70.8	73.9	71.8	**74.3***

Table 2: performance (F1 on test) w/ and w/out Training Regularization L1, Early Stop. L=L1; E=Early Stopping; S=Support Inference; G=GFM prediction. CRF excluded as not L1 regularized during training. bold=best in row, *=best in dataset

can be reliably estimated based on the given dataset size, and these factors in turn influence how much performance improvement L1 feature selection can bring in. The heap maps in Figure 2 shows how varying the overall regularization strength and the L1 ratio affects the test performance. One can see that the optimal amount of L1 penalty varies on different datasets. Overall introducing some L1 penalty leads to the performance improvement on 6 out of 9 datasets.

Apart from improving test performance, L1 also shrinks the model sizes massively. The model size is measured by the disk space the model occupies. Since the model weights are sparse when L1 is added, only non-zero weights are saved. Table 3 compares the sizes

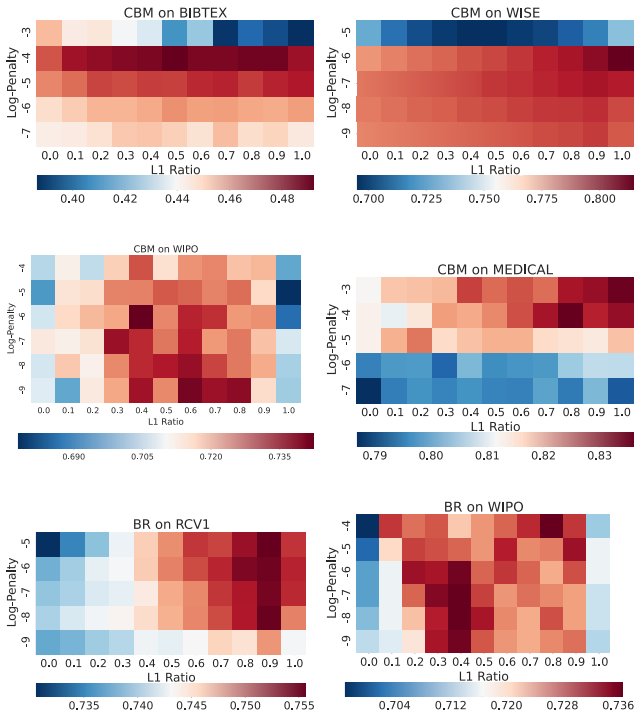

Figure 2: Test F1 versus the penalty strength and the L1 ratio. The penalty strength is on the log scale.

of models trained with only L2 penalty versus models trained with both L1 and L2 penalty. Generally, adding L1 shrinks models to no more than 10% of its original sizes. On some datasets, such as ENRON, RCV1, WISE and WIPO, the shrunk CBM models are only about 1% of its original sizes. Interesting, if we look at the total number of features selected by the classifiers (union of features used in all base learners), the reduction in feature size is not as dramatic as the reduction in model size. This is in direct contrast with binary classification, where these two reductions mostly agree. By looking into the trained multi-label classifiers, we notice that although many features are relevant for some labels and are thus included in the classifiers, for each individual label, only a few features actually have non-zero weights. Thus although the union of all relevant features for all labels can be large, each label predictor can be a small model that includes a few features, and the entire multi-label classifier is therefore quite compact.

To understand the impact of early stopping, we compare the "S+G" column with the "E+S+G" column. We observe that unlike L1, which is more dataset dependent, early stopping is more algorithm dependent. Early stopping helps most for CBM, and less for other methods. CBM has higher capacities and is more likely to over-fit compared with other methods, and thus benefits more from early stopping. Comparing the last 4 columns of the table, it is clear that typically the best performance is achieved when both L1 and early stopping are employed.

model	BR		LSF		CBM	
	Features selected	Model size	Features selected	Model size	Features selected	Model size
MEDICAL	60.08	14.89	75.40	12.62	81.15	6.67
BIBTEX	100.00	26.08	99.89	11.25	100.00	4.44
IMDB	66.41	20.5	100.00	7.06	99.54	9.57
OHSUMED	52.62	33.67	61.81	5.09	68.42	5.54
ENRON	31.81	7.5	26.24	2.41	50.24	1.54
RCV1	69.99	12.29	79.41	7.98	77.29	1.52
TMC	37.87	13.84	40.54	2.82	78.22	2.84
WISE	14.15	1.12	20.18	0.48	24.41	0.33
WIPO	41.64	2.07	38.75	2.02	76.57	1.54

Table 3: Model Size and Feature Used after adding L1 in CBM, LSF, BR (as percentages of the original sizes)

4.3 GFM prediction and Support Inference

We now compare the proposed support inference + GFM prediction strategy with other prediction strategies. Experiment results are summarized in Table 4. LSF is designed to always work with GFM and the concept of support inference do not apply to LSF, so LSF is not included in this table. As before, the "No REG" column does not use any of the proposed training time or prediction time regularizations and serves as a baseline. All other columns have L1 and early stopping regularizations applied during model training and only vary the prediction methods. The "L+E" column uses MAP inference as described in Section 2. The "All4" column uses the proposed support inference + GFM prediction strategy. Comparing the "L+E" column with the "All4" column it is clear that support inference + GFM prediction performs better than MAP prediction for all methods on all datasets.

Next we break down the support inference + GFM prediction combination and test how each of them works separately. The "L+E+S" column only uses support inference but not GFM prediction. We use support inference to restrict the label combinations to those observed in the training set, and among them, we pick the one with the highest probability. So this is basically MAP inference restricted on support combinations. Comparing the "L+E" column with the "L+E+S" column, we see that adding the support inference alone consistently improves the test performance (except for CRF, for which "L+E" is the same as "L+E+S", as described in Section 2). The improvement is most substantial on BR, which did not estimate any label dependencies during training, and is relatively small on CRF, PCC and CBM, which already estimated some label dependencies during training. This observation matches our analysis in Section 3.2 that support inference acts as a regularizer on the label structures – it allows the classifier to take into account label dependencies during prediction even if the classifiers had not modeled any label dependencies during training.

It is known that if CRF as described in Section 2 only contains label-feature interaction but not label-pair interaction, and if the partition function is computed exactly by summing overall all label combinations (conceptually), then the resulting model is mathematically equivalent to BR. In this case, theoretically there is no need to compute the partition function approximately using support combinations. So the authors in [7] only use support inference when CRF contains label pair interactions and thus the model does not factorizes and one has to somehow compute the partition function. Support inference there was deemed purely as an approximate inference procedure. Here our results show, perhaps surprisingly, that

Data	Model	No REG	L+E	L+E+S	L+E+G	All4 REG
MEDICAL	BR	68.8	75.4	**81.5**	76.3	81.1
	CRF	-	77.8	77.8	-	**79.4**
	PCC	73.6	79.0	80.1	80.6	**80.8**
	CBM	79.2	82.2	82.3	79.6	**82.6***
BIBTEX	BR	37.8	39.8	44.4	40.2	**45.4**
	CRF	-	46.5	46.5	-	**49.4**
	PCC	37.4	39.5	45.0	40.1	**47.3**
	CBM	44.0	45.3	46.9	40.4	**49.5***
IMDB	BR	59.4	59.6	59.7	61.0	**61.4**
	CRF	-	63.0	63.0	-	**66.6***
	PCC	59.6	60.1	60.2	61.5	**62.8**
	CBM	61.6	62.2	62.2	64.8	**65.2**
OHSUMED	BR	60.2	63.6	68.0	64.3	**69.1**
	CRF	-	66.4	66.4	-	**69.6**
	PCC	62.5	64.7	68.4	65.8	**70.4**
	CBM	68.7	69.5	70.3	65.4	**71.7***
ENRON	BR	57.9	59.8	59.3	60.4	**61.1**
	CRF	-	60.2	60.2	-	**64.1***
	PCC	60.1	60.5	60.3	61.2	**61.5**
	CBM	58.2	58.5	60.9	59.2	**62.5**
RCV1	BR	72.1	73.8	74.6	74.9	**75.1**
	CRF	-	74.4	74.4	-	**75.8**
	PCC	71.0	72.7	72.8	**74.3**	74.1
	CBM	76.6	77.3	78.5	77.9	**79.2***
TMC	BR	61.6	61.6	62.6	62.2	**63.1**
	CRF	-	62.0	62.0	-	**64.7***
	PCC	61.9	62.0	62.4	62.7	**63.2**
	CBM	59.8	60.4	62.7	60.7	**63.8**
WISE	BR	68.0	72.8	79.0	73.0	**79.3**
	CRF	-	77.7	77.7	-	**79.0**
	PCC	70.7	74.6	76.7	77.1	**78.0**
	CBM	77.9	79.8	79.8	73.6	**80.3***
WIPO	BR	63.4	69.5	73.2	70.0	**74.0**
	CRF	-	70.3	70.3	-	**72.2**
	PCC	68.8	70.2	70.4	70.6	**72.3**
	CBM	63.0	69.6	72.5	70.3	**74.3***

Note: '-': Not available; ElasticNet for CRF is L2 regularization only.

Table 4: performance (instance-F1 values on test) w/ and w/out Prediction Regularization: Support Inference and GFM. L=L1; E=Early Stopping; S=Support Inference; G=GFM prediction. LSF excluded since it must use GFM and cannot use Support Inference. bold=best in row, *=best in dataset

support inference in fact helps on BR – in other words, even if we could compute the CRF partition function exactly, it is still beneficial to compute it approximately, using support inference, which helps to capture label dependencies. Then it is natural to raise a question regarding pair-wise CRF, which is one of the state-of-the-art multi-label classifiers: does its good performance come from its pair-wise label interaction terms, or from this support inference? To test this idea, we removed the label pair terms in CRF. Table 5 shows that CRF without label pair terms do almost equally well as the one with label pair terms. This analysis is not to criticize the design of pair-wise CRF (in fact, our study has drawn lot of inspiration from the paper [7]), but rather, to shed lights on the effectiveness of CRF, and to bring to researchers' attention of support inference as a simple yet powerful regularizer, besides its original role as an approximation.

The "L+E+G" column only uses GFM predictor but not support inference. For each method, we sample 1000 times based on the estimated joint and use samples to compute the marginals probabilities required by the GFM predictor, as described in Section 3.2 (The CRF numbers are missing because there is no straight forward way of sampling from CRF). Comparing column "L+E+G" with column "L+E", we see that GFM alone always gives some improvement for BR and PCC, but is less effective for CBM. However, CBM clearly benefits from GFM when GFM is used in conjunction with support inference. Comparing all columns, we can conclude that support inference + GFM is the most effective prediction strategy, which consistently boosts the performance of all classifiers on all datasets.

4.4 Other Related Work

It is shown in [19] that the GFM algorithm can also work with non-probabilistic learners. For example, we could train a linear regression with the target $\mathbb{1}(y_l = 1, |\mathbf{y}| = s \mid \mathbf{x})$, and the resulting regression scores can be fed into GFM. Besides the GFM method which works in the most general setting, there exist a few other F1 predictors specifically designed for BR [2, 9, 15, 18]. There are also a few methods that take into account F1 metric during training. [16] reduces F metric maximization to cost-sensitive classification, and [6] studies F metric maximization with conditionally independent label subsets [6]. [17] provides an up-to-date review on different F metric maximization methods.

There are also several neural network based multi-label classification methods [1, 3, 13, 14]. We ran the code associated with Structured Prediction Energy Networks (SPEN) [1] as it is without further adding the regularizations techniques discussed here, and the results are listed in Table 6 for reference. Granted that we did not spend as much time tuning the network parameters as we spent on the other methods, SPEN's performance looks less competitive, possibly due to over-fitting in high dimensional data with its high model capacity.

	MEDICAL	BIBTEX	OHSUMED	ENRON	WIPO
SPEN	70.9	38.7	60.8	59.4	64.0
CBM	82.6	49.5	71.7	62.5	74.3

Table 6: Test performance (F1) of Structured Prediction Energy Networks compared versus CBM

5 CONCLUSION

In this paper our main goal is to make effective F-measure optimal predictions on multi-label text data. In doing so we show that most multi-label classification algorithms can be used if they produce a joint estimator $p(\mathbf{y}|\mathbf{x})$, to which we apply Support Inference ad GFM predictions, which work well together. In fact we show that on most datasets, classifiers both simple (BR) and complex (CRF) rely more on these two regularization ideas, rather than the their internal label dependency models (if any).

Working with text data, we also apply well established training regularization ideas: L1, L2, Early Stopping. We analyze the effects of these and notice significant reduction in model size (otherwise seriously big) and achieve state of the art performance on benchmark datasets. We make our code available at https://github.com/cheng-li/pyramid.

REFERENCES

[1] David Belanger and Andrew McCallum. 2016. Structured prediction energy networks. In *Proceedings of the International Conference on Machine Learning*.

[2] Kian Ming Adam Chai. 2005. Expectation of F-measures: tractable exact computation and some empirical observations of its properties. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 593–594.

[3] Moustapha Cissé, COM Maruan Al-Shedivat, and Samy Bengio. 2016. ADIOS: Architectures deep in output space. In *Proceedings of the 33rd International Conference on Machine Learning, ICML*.

[4] Krzysztof Dembczyński, Willem Waegeman, Weiwei Cheng, and Eyke Hüllermeier. 2012. On label dependence and loss minimization in multi-label classification. *Machine Learning* 88, 1-2 (2012), 5–45.

[5] Jerome Friedman, Trevor Hastie, and Rob Tibshirani. 2010. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software* 33, 1 (2010), 1.

[6] Maxime Gasse and Alex Aussem. 2016. F-Measure Maximization in Multi-Label Classification with Conditionally Independent Label Subsets. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 619–631.

[7] Nadia Ghamrawi and Andrew McCallum. 2005. Collective multi-label classification. In *Proceedings of the 14th ACM international conference on Information and knowledge management*. ACM, 195–200.

[8] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep learning*. MIT Press.

		MEDICAL	BIBTEX	IMDB	OHSUMED	ENRON	RCV1	TMC	WISE	WIPO
CRF w/o pair label depend	w/o GFM	79.2	46.9	61.3	66.1	59.8	73.8	62.6	78.2	70.7
	w/ GFM	80.5	49.4	66.1	69.8	64.4	75.8	64.5	79.4	71.8
CRF w/ pair label depend	w/o GFM	77.8	46.5	63.0	66.4	60.2	74.4	62.0	77.7	70.3
	w/ GFM	79.4	49.4	66.6	69.6	64.1	75.8	64.7	79.0	72.2

Table 5: Instance F1 compared bewteen CRF w/ and w/o pairwise dependencies

[9] Martin Jansche. 2007. A maximum expected utility framework for binary sequence labeling. In *Annual Meeting-Association For Computational Linguistics*, Vol. 45. 736.

[10] Abhishek Kumar, Shankar Vembu, Aditya Krishna Menon, and Charles Elkan. 2013. Beam search algorithms for multilabel learning. *Machine learning* 92, 1 (2013), 65–89.

[11] Thomas Lavergne, Olivier Cappé, and François Yvon. 2010. Practical very large scale CRFs. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 504–513.

[12] Cheng Li, Bingyu Wang, Virgil Pavlu, and Javed A. Aslam. 2016. Conditional Bernoulli Mixtures for Multi-label Classification. In *Proceedings of the 33rd International Conference on Machine Learning*. 2482–2491.

[13] Volodymyr Mnih, Hugo Larochelle, and Geoffrey E Hinton. 2012. Conditional restricted boltzmann machines for structured output prediction. *arXiv preprint arXiv:1202.3748* (2012).

[14] Jinseok Nam, Jungi Kim, Eneldo Loza Mencía, Iryna Gurevych, and Johannes Fürnkranz. 2014. Large-scale multi-label text classification—revisiting neural networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 437–452.

[15] Ye Nan, Kian Ming Chai, Wee Sun Lee, and Hai Leong Chieu. 2012. Optimizing f-measure: A tale of two approaches. *arXiv preprint arXiv:1206.4625* (2012).

[16] Shameem Puthiya Parambath, Nicolas Usunier, and Yves Grandvalet. 2014. Optimizing F-measures by cost-sensitive classification. In *Advances in Neural Information Processing Systems*. 2123–2131.

[17] Ignazio Pillai, Giorgio Fumera, and Fabio Roli. 2017. Designing multi-label classifiers that maximize F measures: State of the art. *Pattern Recognition* 61 (2017), 394–404.

[18] José RamóN Quevedo, Oscar Luaces, and Antonio Bahamonde. 2012. Multilabel classifiers with a probabilistic thresholding strategy. *Pattern Recognition* 45, 2 (2012), 876–883.

[19] Harish Guruprasad Ramaswamy. 2015. *Design and Analysis of Consistent Algorithms for Multiclass Learning Problems*. Ph.D. Dissertation. Indian Institute of Science Bangalore.

[20] Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. 2011. Classifier chains for multi-label classification. *Machine learning* 85, 3 (2011), 333–359.

[21] Grigorios Tsoumakas and Ioannis Katakis. 2007. Multi-label classification: An overview. *Int J Data Warehousing and Mining* 2007 (2007), 1–13.

[22] Willem Waegeman, Krzysztof Dembczyński, Arkadiusz Jachnik, Weiwei Cheng, and Eyke Hüllermeier. 2014. On the bayes-optimality of F-measure maximizers. *The Journal of Machine Learning Research* 15, 1 (2014), 3333–3388.

[23] Guo-Xun Yuan, Chia-Hua Ho, and Chih-Jen Lin. 2012. An improved glmnet for l1-regularized logistic regression. *Journal of Machine Learning Research* 13, Jun (2012), 1999–2030.