A Regularization Approach to Combining Keywords and Training Data in Technology-Assisted Review

Eugene Yang
IR Lab, Georgetown University
Washington, DC, USA
eugene@ir.cs.georgetown.edu

David D. Lewis
Cyxtera Technologies
Dallas, TX, USA
icail2019paper@davelewis.com

Ophir Frieder
IR Lab, Georgetown University
Washington, DC, USA
ophir@ir.cs.georgetown.edu

ABSTRACT

Manual keyword queries and supervised learning (technology-assisted review) have been viewed as conflicting approaches to high recall retrieval tasks (such as civil discovery and sunshine law requests) in the law. We propose a synthesis that uses a keyword list as a regularizer when learning a logistic regression model from labeled examples. Balancing keywords against training data requires knowing how the regularization penalty should scale with training set size. We show, however, that advice on scaling from theory is contradictory, software defaults are inconsistent, and standard practice (validation-based tuning) is impractical in many high-recall retrieval settings. Through experiments on simulated e-discovery data sets, we show that the penalization scheme suggested by a Bayesian interpretation is substantially safer than alternatives from stochastic optimization and computational learning theory. Combining keywords and training data provides better effectiveness on our datasets than using either alone, showing that both approaches bring value.

CCS CONCEPTS

• **Information systems → Clustering and classification**; • **Computing methodologies → Regularization**; *Supervised learning by classification.*

KEYWORDS

technology-assisted review, Bayesian priors, informative priors, text categorization, regularization, logistic regression, keywords

ACM Reference Format:
Eugene Yang, David D. Lewis, and Ophir Frieder. 2019. A Regularization Approach to Combining Keywords and Training Data in Technology-Assisted Review. In *Seventeenth International Conference on Artificial Intelligence and Law (ICAIL '19), June 17–21, 2019, Montreal, QC, Canada.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3322640.3326713

1 INTRODUCTION

A range of legal tasks require finding most or all relevant documents within a large collection of mostly irrelevant material. These tasks include electronic discovery (e-discovery) in civil litigation, internal and law enforcement investigations, information governance, antitrust reviews, government responses to sunshine law requests, and patent search. Similar tasks outside the law include systematic review in medicine [32], annotation of data sets for analytics, and archival research. In computer science, these tasks are referred to as *high-recall retrieval* (HRR) [25, 34], *high-recall information retrieval* (HRIR) [1, 10], *finite population annotation* (FPA) [2], or simply *annotation.* We use the term HRR in this paper.

Technological support for HRR initially focused, and still largely relies, on manual construction of keyword queries [3, 20]. Users attempt to anticipate good search terms and combine them in a query. While the full power of Boolean logic is typically available, in practice user queries are often just disjunctions of words and phrases, sometimes quite small.

An increasingly popular alternative to keywords is the use of supervised machine learning, sometimes referred to in HRR as technology-assisted review (TAR) [1]. Users label example documents for relevance, and a supervised learning algorithm produces a predictive model that can be used to rank or classify documents. When combined with active learning [4] for selecting training data, supervised learning can vastly reduce the time to find the bulk of relevant documents [5, 32].

In e-discovery, however, keyword search and supervised learning are often viewed as in conflict. Heated debates on the appropriateness of the technologies are common, and have spilled into court cases. Blair & Maron's 1985 classic study on full-text search [3] is a common and widely misinterpreted tool in these fights [22].

In truth, both keywords and labeled documents are useful forms of knowledge, and should both be exploited in HRR tasks. This is not a new idea: relevance feedback in information retrieval has long combined user queries and training data in an iterative active learning loop [23]. What is to some extent new is a focus on high recall rather than a few top-ranked documents, and the desirability of applying modern discriminative learning algorithms rather than text retrieval heuristics.

We propose regularized logistic regression [7, 9] as an appropriate tool to combine these two sources of knowledge. Logistic regression is a widely used and effective approach in TAR applications [33]. Regularization in logistic regression is typically used to keep model coefficients close to 0 to avoid overfitting. We propose instead to keep coefficients close to values suggested by a keyword query, while also responding to training data. The result is a method that can be used with keywords only, training data only, or both to achieve maximal effectiveness.

A major question when using regularization is how strong the regularization penalty should be. This question is acute in our

ICAIL '19, June 17–21, 2019, Montreal, QC, Canada
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6754-7/19/06…$15.00
https://doi.org/10.1145/3322640.3326713

application, where regularization must balance a good but imperfect query against training sets of unpredictable quality and small but increasing size. Unfortunately, as we discuss below, different theoretical frameworks provide conflicting advice on how regularization should vary with training set size, and these conflicting recommendations have been implemented in widely used open source machine learning software. Regularizing toward nonzero values also introduces new algorithmic issues.

We present a systematic study on regularization, including regularization toward keyword queries, for the HRR problem. We review the three major theoretical frameworks for regularization, and hypothesize that the Bayesian framework provides the most useful guidance for HRR. We describe our new implementation for fitting logistic regression models in the Bayesian framework, including presenting the previously unpublished proximal updates for L1 penalties with non-zero modes. We then present an experimental study of Bayesian logistic regression on two widely used HRR test sets. We show our Bayesian MAP approach provides good effectiveness in all three HRR scenarios: query only, training data only, and both. The results also support our hypothesis that a constant regularization penalty (the Bayesian MAP approach) is safer and more effective than alternatives.

2 REGULARIZATION THEORY

Regularization—the penalization of solutions that deviate from prior expectations in some sense—is a key technique for avoiding fitting to accidental properties of training data in machine learning [26]. While a variety of regularization techniques have been developed, the most widely used is adding a penalty on coefficient magnitude (or, more generally, distance from a specified value) to the training loss, and finding model coefficients that minimize the sum of loss and penalty.

So-called L2 penalties, which are proportional to the squares of the coefficients (actually the square of the L2 norm) are the most commonly used. L1 penalties, which are proportional to the absolute values of the coefficients, are also widely used.

Both approaches have advantages beyond improving generalization. As discussed below, adding an L2 penalty to a loss function can aid convergence of fitting. Adding an L1 penalty complicates optimization, but leads to solutions that are sparse (most coefficients are 0). This provides models that are both more efficient to use and easier to interpret [9]. Elastic net regularization is a weighted combination of L1 and L2 penalties with the desirable properties of both [36].

The desirable properties of these penalties have led to their reinvention and analysis in several fields. Here we discuss motivations from three theoretical perspectives: Bayesian statistics, stochastic optimization, and computational learning theory.

2.1 Bayesian MAP Estimation

Bayesian statistics provides the most direct motivation for penalizing coefficient magnitude. Assume a conditional probability model $y = f(\mathbf{x}; \mathbf{w})$ parameterized by a d-dimensional vector of unknown real-valued coefficients $\mathbf{w}$. Suppose the analyst's prior over the coefficients is a product of independent, zero mean (and thus zero mode), equal variance Gaussians. Also suppose the analyst observes a data set $D = (\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)$ where each y value was generated by applying $f(\mathbf{x}; \mathbf{w})$ independently to the corresponding $\mathbf{x}$.

Then by Bayes Rule, the analyst's belief about $\mathbf{w}$ after seeing the data set should take the form of this posterior probability distribution:

$$p(\mathbf{w}|D) = \frac{p(D|\mathbf{w})p(\mathbf{w})}{p(D)} \tag{1}$$

$$= \frac{\left(\prod_{i=1}^{n} p(y_i|\mathbf{w}; \mathbf{x_i})\right)\left(\prod_{j=1}^{d} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{w_j^2}{2\sigma^2}}\right)}{p(D)} \tag{2}$$

where we abuse the notation and write $p(D|\mathbf{w})$ for the conditional probability of seeing the set of y values in D given the corresponding $\mathbf{x}$ values in D, and write $p(D)$ for the corresponding unconditional value.

In applications, both the difficulty of computing $p(D)$ and the demands of efficient prediction often make it impractical to use the full posterior distribution $p(\mathbf{w}|D)$. It is therefore common to instead seek a single coefficient vector $\mathbf{w}^*$ that is given maximum probability by the posterior distribution: a MAP (maximum a posteriori) estimate. Depending on the form of the posterior, there may or may not be a single unique MAP estimate.

Taking the logarithm of the posterior likelihood, negating, and dropping constants shows that $\mathbf{w}^*$ is found by maximizing a regularized loss function:

$$\mathbf{w}^* = \underset{\mathbf{w}}{\text{argmax}} \left\{ \left(-\sum_{i=1}^{n} \ln p(y_i|\mathbf{w}; \mathbf{x_i}) \right) + \lambda \sum_{j=1}^{d} w_j^2 \right\} \tag{3}$$

where λ is the regularization strength, and the Gaussian prior leads to an L2 penalty. An L1 penalty can be derived in a similar way from a Laplace prior [9].

Since the loss term (the negated log-likelihood) grows with training set size, the effect of the regularization penalty decreases as the training set grows. This is a consequence of Bayes Rule, which puts decreasing weight on the prior beliefs as more data is observed.

2.2 Stochastic Optimization

Stochastic optimization refers to finding optima of functions whose arguments include random variables. A canonical example is solving the minimization:

$$\mathbf{w}^* = \min_{\mathbf{w}} \{ f(\mathbf{w}) := \mathbb{E}[G(\mathbf{w}, \xi)] \} \tag{4}$$

where ξ is a random vector, and $G()$ is a function with both deterministic and random arguments.

Often ξ's distribution makes it impractical to directly encode or optimize $f(\mathbf{w})$. However, if a random sample of n values of ξ are available, one can instead solve the corresponding sample average approximation (SAA) problem [28]:

$$\mathbf{w}^* = \min_{\mathbf{w}} \left\{ \hat{f}_n(\mathbf{w}) := \frac{1}{n} \sum_{i=1}^{n} G(\mathbf{w}, \xi_i) \right\} \tag{5}$$

where one instead optimizes the function $\hat{f}_n()$ whose value is the sample average of n realizations of G.

In contrast to $f()$, the function $\hat{f}_n()$ can often be written down in its entirety, and thus could be optimized by general purpose optimization methods. However, the fact that $\hat{f}_n()$ is the sum of a large number of similar terms allows a specialized approach: *stochastic gradient* algorithms [15, 26]. These optimization algorithms process $\hat{f}_n()$ one term at a time, updating an estimate of $\mathbf{w}^*$ as each term is processed.

How similar the solution of Equation 5 is likely to be to the desired value (the solution to Equation 4) depends on the sample size n and on the properties of $G()$. A desirable case is when G has the form $G(\mathbf{w}, \xi) = F(\mathbf{w}, \xi) + \lambda R(\mathbf{w})$, where $R(\mathbf{w})$ is a deterministic regularization function. This gives the SAA problem:

$$\mathbf{w}^* = \min_{\mathbf{w}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left(F(\mathbf{w}; \xi_i) + \lambda R(\mathbf{w}) \right) \right\} \tag{6}$$

$$= \min_{\mathbf{w}} \left\{ \left(\sum_{i=1}^{n} F(\mathbf{w}; \xi_i) \right) + n\lambda R(\mathbf{w}) \right\} \tag{7}$$

Many powerful results are known for the convergence of algorithms for solving Equation 6 when $F(\mathbf{w})$ and/or $R(\mathbf{w})$ have desirable properties, such as strong convexity [26]. A particularly desirable case is Tikhonov regularization, where $R(\mathbf{w})$ is an L2 penalty [30].

If $F(x)$ is the loss function for a supervised learning problem, the SAA framework can be applied with the training set playing the role of the random sample. Equation 6 then expresses a penalized average loss over the training set, while the equivalent Equation 7 expresses a penalized total loss over the training set, analogous to Equation 3. The connection has made stochastic optimization theory a standard tool in proving convergence results for learning algorithms, particularly variations on stochastic gradient.

Note, however, that Equation 6 assumes that the regularization penalty is on *every term of the SAA function*. Applying this approach to batch mode supervised learning gives one SAA term per training example, and thus a regularization penalty that increases linearly with training set size (Equation 7). Thus while both a Gaussian Bayesian prior and Tikhonov regularization give L2 penalties, they suggest very different scaling behavior with training set size.

2.3 Computational Learning Theory

Computational learning theory provides several alternative perspectives on coefficient size penalties, including structural risk minimization, PAC-Bayesian analysis, and algorithmic stability [26]. All capture the fact that stronger penalties limit the hypothesis space (and thus the maximum effectiveness) available to the learning algorithm, but increase the probability that the effectiveness of the fitted coefficient vector on test data will be similar to its effectiveness on the training data.

Learning theory analyses with different goals leads to different conclusions about the ideal scaling of penalty strength with coefficient size, usually falling between the $O(1)$ scaling of the Bayesian analysis and the $O(n)$ scaling of the stochastic optimization analysis. For example, Shalev-Shwartz and Ben-David present a bound on the effectiveness of learning with L2 regularization that assumes $O(\sqrt{n})$ scaling of penalty strength with training set size ([26], Corollary 13.9).

2.4 Reconciling the Perspectives

The three perspectives have differing assumptions, so the different scalings they suggest are not inherently in conflict. The Bayesian analysis assumes the model being fit exactly describes how the data was generated. The Bayesian MAP approach finds the coefficient vector most likely to be "correct" under this strong assumption, but provides no guarantees on effectiveness when this assumption is wrong.

In contrast, many learning theory analyses (including the $O(\sqrt{n})$ rate presented above) are *agnostic*. They indicate what predictive effectiveness we can hope for regardless of how well the data corresponds to the model being fit. Unsurprisingly, such analyses are more conservative about how fast data should overwhelm regularization.

Finally, the focus of stochastic optimization is neither correctness nor predictive accuracy, but the convergence of optimization algorithms. Classic stochastic optimization analyses apply to arbitrary length sequences of random function realizations. These analyses require certain properties (e.g., degree of strong convexity) to hold regardless of number of realizations. This makes these analyses applicable even to online learning situations [11], at the cost of assuming that regularization scales in a fashion that keeps the necessary properties constant.

For a fixed training set, there is no tension between the three perspectives. One can pick λ based on desirable properties in any one of the three frameworks, and that value of λ will have a sensible interpretation in the other two. In fact, λ is often treated as a black box hyperparameter to be chosen using data held out (by validation or cross-validation) from a fixed, large, representative training set.

3 REGULARIZATION IN HRR

In TAR and other HRR applications, however, training sets are neither fixed, large, nor representative. Initial training examples, when any are available, are documents that are found by search or are opportunistically available. Training sets typically start out tiny and grow over time by a mixture of user exploration and iterative active learning [4]. The resulting training sets are often of modest size and over-sample important portions of the collection, particularly the relevant documents for the topic of interest.

A small, unrepresentative training set makes regularization crucial for generalization. However, it also makes data-driven approaches to tuning regularization strength impossible (consider a training set of size 1) or ineffective. Theoretical guidance on regularization strength, and on how regularization strength should vary with training set size, is critical.

Unfortunately, growing training sets are exactly where the theoretical analyses are in conflict. The Bayesian perspective yields an $O(1)$ penalization on total training loss, that is:

$$\mathbf{w}^* = \underset{\mathbf{w}}{\operatorname{argmax}} \left\{ \left(\sum_{i=1}^{n} f(\mathbf{w}; \mathbf{x_i}, y_i) \right) + \lambda R(\mathbf{w}) \right\} \tag{8}$$

where $f(\mathbf{w}; \mathbf{x_i}, y_i)$ is the loss on training example i. This perspective is commonly presented in the algorithmic literature on machine learning [27, 35].

Stochastic optimization theory, on the other hand, suggests an $O(n)$ penalty on total training loss:

$$\mathbf{w}^* = \operatorname*{argmax}_{\mathbf{w}} \left\{ \left(\sum_{i=1}^{n} f(\mathbf{w}; \mathbf{x_i}, y_i) \right) + n\lambda R(\mathbf{w}) \right\}. \tag{9}$$

This formulation too is widely used in the algorithmic literature on machine learning [14, 21, 35]. As discussed above, computational learning theory suggests a range of penalization strengths, but is less widely cited by developers of practical algorithms.

3.1 Regularization in Open Source Software

This conflict between theoretical perspectives plays out in software as well, in the form of inconsistencies among software options, confusing documentation, and inflexible penalty schemes. We present three typical examples.

1. *LIBLINEAR* is a widely used training package for linear models, including logistic regression [8]. Its *-c* parameter specifies the reciprocal of the regularization penalty on the entire training set loss, i.e., the Bayesian perspective. Section N.5. of the LIBLINEAR documentation provides extensive discussion of using cross-validation to set *-c* without mentioning whether the penalty is at the instance or training set level. Reading the source code or the mathematically dense algorithmic discussion is necessary to determine that penalties are at the training set level.

2. *Scikit-learn* is a widely-used suite of Python implementations of machine learning algorithms. The parameter `C` of module `sklearn.linear_model.LogisticRegression` (default value 1.0) is documented as:

> Inverse of regularization strength; must be a positive float. Like in support vector machines, smaller values specify stronger regularization.

The parameter `alpha` of module `sklearn.linear_model.SGDClassifier`, which can also be used to train logistic regression models is documented as:

> Constant that multiplies the regularization term. Defaults to 0.0001 Also used to compute learning_rate when set to 'optimal'.

In neither case does the documentation make clear whether penalization is an the instance or training set level. Examining the source code shows in fact that penalization is at the instance level for `SGDClassifier` and at the training set level for `LogisticRegression`. The parameters name `C` and `alpha` do align with common conventions in the stochastic optimization and stochastic gradient research literature respectively. Users not familiar with both bodies of research literature are, however, likely to misuse one or the other parameter.

3. *Vowpal Wabbit* [13] is highly scalable training software for a variety of supervised and unsupervised models, including logistic regression. It supports both batch fitting algorithms (conjugate gradient and BFGS) and online fitting algorithms (many configurable variants on stochastic gradient descent). The *L1* and *L2* options allow separate specification of penalty strength for L1 and L2 regularization. The documentation is admirably clear on whether instance or training set penalization is used with each algorithm [13]. Which approach is used depends on which training algorithm is selected, a difference that may well be unappreciated by users focused on applications rather than algorithmics.

Once a user decodes the penalization scheme for a particular piece of software, either scaling scheme can be implemented by appropriate use of the respective penalty argument. However, it seems likely that ambiguity on this point, and an over-reliance on tuning by validation and cross-validation, has discouraged thinking about how penalization *should* scale with training set size.

Beyond the question of scaling, few or no widely used learning packages provide direct support for different penalty strengths for different features and, most importantly for HRR, penalization toward nonzero modes. A notable exception is the work by Madigan and collaborators in Bayesian statistics that resulted in the BBR, BMR and BXR C++ implementations [9], and the Cyclops package in R [29]. All use an in-memory, cyclic coordinate descent optimization algorithm. We are aware of no Python implementation, nor one that takes a stochastic gradient fitting approach that provides a path toward an external memory / online implementation. This motivated our own implementation.

3.2 An Implementation for HRR

Our implementation is based on the `SGDClassifier` from scikit-learn [19] (described above). We added an option to load a file specifying an individual prior mode for each feature, as well as both an L1 and L2 regularization penalty for each feature. The implementation is open-sourced and publicly available on GitHub[1]. **(EUGENE: BE SURE TO FIX THIS.)**

Stochastic gradient fitting algorithms process one training example at a time, typically taking multiple passes over the training data. For each example an update to coefficients is made based on the first derivative of the loss function with respect to each coefficient. The loss function and gradients with L2 penalization are:

$$L_\sigma(\mathbf{w}) = L(\mathbf{w}) + \sigma \left\lVert \mathbf{w} - \mathbf{b} \right\rVert_2^2 \tag{10}$$

$$\nabla L_\sigma(\mathbf{w}) = \nabla L(\mathbf{w}) + 2\sigma(\mathbf{w} - \mathbf{b}) \tag{11}$$

where $L(\mathbf{w})$ be the logistic loss function of the weight vector $\mathbf{w}$ and $\mathbf{b}$ is the vector of modes. The gradient update is straightforward, differing only by an addition or subtraction of $2\sigma\mathbf{b}$ from the typical update for zero-mode regularized logistic regression.

For L1 regularization, stochastic gradient algorithms face the problem of maintaining coefficient vector sparsity in the face of noisy per-example updates. `SGDClassifier` addresses this by using a truncated gradient update [31] that is not easily adapted to nonzero modes. We instead adapted the more recently popular approach of *proximal updates* [27]. This replace SGD's gradient-based update with two updates: a gradient-based step on the smooth components of the gradient, followed by a projection step based on the discontinuous component of the gradient (e.g., L1 penalty).

The proximal update for a zero-mode L1 penalty is widely known. The proximal update for the nonzero mode L1 penalty does not appear to have been previously published, so we present it below. Let the penalized loss be:

$$L_\sigma(\mathbf{w}) = L(\mathbf{w}) + \sigma \left\lVert \mathbf{w} - \mathbf{b} \right\rVert_1 \tag{12}$$

[1]Link will be provided after blind review.

and let $\Omega(\mathbf{w}) = \sigma \|\mathbf{w} - \mathbf{b}\|_1$. By Equation 2.2 and 6.3 in Parikh and Boyd [18], the proximal operator for coordinate i is,

$$(\mathbf{prox}_\Omega(\mathbf{w}))_i = \mathbf{prox}_{\sigma\|\cdot\|_1}(w_i - b_i) + b_i \tag{13}$$

$$= \begin{cases} w_i - \sigma, & w_i - b_i \geq \sigma \\ b_i, & \|w_i - b_i\|_1 \leq \sigma \\ w_i + \sigma, & w_i - b_i \leq -\sigma \end{cases} \tag{14}$$

For each training example, we first update the coefficient vector based on the gradient of the logistic loss. We then check whether each w_i falls within an interval of width σ around b_i. If so, w_i is replaced with b_i. Otherwise w_i has σ added or subtracted. The approach is easily generalized to different penalty strengths for each coefficient, and to the elastic net.

Our implementation supports both $O(1)$ and $O(n)$ scaling. The version of the software used in this study assumes the data set fits in memory, but as an SGD algorithm could be adapted to the online setting. As is common for in-memory SGD implementations, an iteration processes each example in the training set once, with a different random order used on each iteration. The convergence test is based on the *tolerance*, i.e., minimum of the absolute difference of the average example loss between two consecutive epochs. We set the tolerance to be 0.001, i.e., terminate SGD when the absolute difference of the average example loss between two consecutive epochs is smaller than 0.001, in our study. **(DDL TO EUGENE: HERE IT SAYS 0.001 BUT BELOW IT SAYS 10^-4.)**

4 EXPERIMENTAL METHODS

With an implementation in hand, our goal was to understand how to use regularization to effectively strike a balance between keywords and training data in HRR. Our core hypothesis was that Bayesian penalization, i.e., a regularization penalty that is constant rather than growing with training set size, would be the most effective approach.

The experimental challenge we faced were the many strong conflating factors that needed to be controlled for: the variable difficulty of classification problems, the variable effectiveness of their keyword queries, and the variable quality of training sets even for a given classification problem and training set size. Our design controlled for all these factors in order to tease out the impact of penalty strength.

4.1 Modeling

We fit logistic regression models by optimizing the training set MAP estimate under either a zero mode or keyword-based prior. Fitting for each run continued until an optimization tolerance of 10^{-4} was reached, or until $10,000$ iterations were run. Hitting the iteration limit typically occurred only for very low values of penalization.

We tested priors with two types of modes: all zeros, or all zeroes except for words that occur in the keyword query for that topic. The keyword-based nonzero mode when used was $1 + \log(qtf)$, where *qtf* is the number of occurrences of the term in the keyword query. We refer to this as either the nonzero mode case or the QTF mode case. In almost all cases this value was simply 1.0, since few query words occurred more than once.

For each mode type we studied both L1 and L2 penalization for distance from mode, and a range of strengths for that penalty.

4.2 Datasets

We used two datasets popular in e-discovery research.

4.2.1 Jeb Bush Collection. The Jeb Bush data consists of electronic mail communications involving the governor of the US state of Florida [24]. Several versions of the data have been distributed. We obtained our copy from Gordon Cormack, co-organizer of the TREC 2015 and 2016 Total Recall tracks. This version consisted of 290,099 files, each with one message. We removed exact duplicates based on the MD5 hash of file, resulting in 274,124 unique documents.

The TREC 2015 Total Recall Track defined 10 binary topics on the Jeb Bush data and distributed short titles and class labels for each [24]. The TREC 2016 Total Recall Track defined 34 more topics with titles and ternary ("non-relevant" vs. "relevant" vs. "important") class labels [12]. For the 2016 data we treated both "relevant" and "important" as positive labels, and "non-relevant" as the negative label. This gave 44 binary classification problems. To ensure enough positive examples for accurate estimation of effectiveness, however, we limited ourselves to the 33 topics with at least 160 positive documents. We used the topic title to simulate keywords selected by an expert user. The length of titles range from one word (e.g. "Space") to five (e.g. "Lost Foster Child Rilya Wilson").

4.2.2 RCV1-v2. We know of no other email datasets besides Jeb Bush with a comparable number of topics and thoroughness of labeling. We thus supplemented our experiments with tests on the RCV1-v2, a widely used text categorization test collection [17].

RCV1-v2 consists of 804,414 newswire stories exhaustively categorized by professional coders with respect to 658 categories. We used as our experimental topics the 82 categories (which include categories from the Reuters "Topics", "Regions", and "Industries" subgroups) that had at least 10,000 positive documents. Each RCV1-v2 category has a Reuters Business Briefing (RBB) description of between one and seventeen English words. We used these as our expert keyword queries.

4.3 Text Representation

Many text representation strategies in information retrieval implicitly exert regularization effects [16]. These include stopword removal, collection weighting, stemming, thesauri, clustering, and latent space representations (LSI, word2vec, etc.). Since our goal is to understand penalization-based regularization, we omit all these. We simply downcased text, replaced punctuation with whitespace, and separated text into tokens at whitespace boundaries. Each unique type in a document was treated as a feature whose value was $1 + \log(tf)$, where *tf* was the number of occurrences of tokens for that type in the document.

4.4 Training and Test Data

Given our interest in fundamental properties of regularization, we adopted a conventional training / test split of the collection even though this is less natural for HRR studies. In particular, each test collection was split randomly into a 40% proportion used as a source for (variable and much smaller) training sets, and a 60% portion used in all cases as the test set.

Training set size can of course not be varied without varying training set composition. As is usual in learning curve experiments we use a nested design where smaller training sets are contained within larger ones. In particular, we use nested training sets of size ranging from 2 to 128 by powers of two.

Initial training sets in HRR tasks are typically created from opportunistically available documents, or those found by keyword searches. They are then grown using one of many active learning algorithms, which tend to enrich the proportion of positive examples compared to the dataset as a whole. Since our focus is not on the details of active learning, we took the expedient of simply making training sets contain equal numbers of randomly selected positive and negative examples.

In addition, variability in effectiveness between training sets is extremely high for small training sets. We addressed this by randomly selecting 20 different training sets (replicates) of the maximum size (128 documents) for each topic. Each maximum size training set consisted of 64 positives and 64 negatives drawn by simple random sampling from the 40% training pool. The nested smaller training sets within each replicate were then produced by randomly sampling separately from the 64 positives and 64 negatives, so that all training sets had equal numbers of positive and negative examples.

For RCV1-v2, sampling of the twenty replicas was without replacement, making each of the twenty training sets independent. For the Jeb Bush collection the 40% pool was not large enough to support this, so sampling was *with* replacement. This means there is some overlap (mostly in positive examples) among replicas. To allow easy replication, all random sampling was done based on a lexicographic ordering of the MD5 hash of the unique document ID. This meant also that we held constant the choice of negative examples to the extent possible.

4.5 Measuring Effectiveness

Both ranked retrieval and classification effectiveness measures have been used for evaluating HRR tasks depending on whether assisted retrieval or culling workflows are the focus of interest. We chose the ranked retrieval measure *R-precision* (proportion of relevant documents above a cutoff equal to the number of testset relevant documents). This is an easily interpretable measure which ranges from 0 to 1, and can take on values of both 0 (assuming the percentage of relevant documents is below 50%) and 1 for some ranking.

We computed testset R-precision for each run, and averaged it across the 20 replicates of a given training set size for a penalty level and topic. For dataset level measures, we then averaged this value across all topics for a given training set size and penalty level.

4.6 Experiment Design

Our hypothesis was that the $O(1)$ penalty scaling suggested by Bayesian statistics was more appropriate than the $O(n)$ scaling suggested by stochastic optimization. We therefore varied our penalty strengths on the same power-of-two scale used for varying training set sizes.

We present average R-precision values for runs using heatmaps with penalty strength on the x-axis and training set size on the y-axis, both using a logarithmic scale. We could vary penalty strength over a much wider range (from 2^{-24} to 2^{16}) than training set size, however, so scales are not identical for the two axes. In all cases, however, $O(n)$ scaling corresponds to diagonals through the heat map from lower left to upper right, while $O(1)$ scaling corresponds to vertical heat map columns.

We stress that the heatmaps are colored *row-by-row*: the highest average R-precision in each row has the lightest color, and the lowest having the darkest color. This reflects the fact that we can always choose our regularization penalty, but at any given moment have only a certain amount of training data. A good penalty for a particular training set size is one that yields a light colored cell (high R-precision). A good penalty *scaling scheme* is one that provides a path through light-colored cells in all rows as training set size increases.

5 RESULTS & ANALYSIS

Our Bayesian logistic regression approach is applicable to all three settings faced by HRR users: keywords only, labeled training examples only, and both keywords and labeled training examples. We discuss results for each.

5.1 Keywords Only

A straightforward but important benefit of the Bayesian approach is that the nonzero components of a keyword-based prior act like a statistical ranked retrieval query. When there is no training data, the posterior mode is the same as the prior mode. Since both simple ranked retrieval and logistic regression scoring are based on dot products, a prior mode which is a good ranked retrieval query provides us nontrivial effectiveness even with no training data.

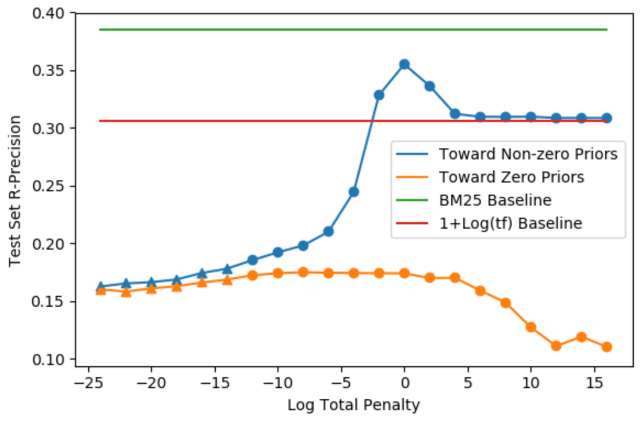
Figure 1 show the average R-precision across classes for the Jeb Bush and RCV1 data sets respectively. The lower red line in each graph corresponds to using our QTF prior modes as a query. For comparison we also include (upper green line) the results from running the textual query using the BM25 option of the state-of-the-art Elasticsearch search engine.

Using the QTF mode as a query achieves better effectiveness than a logistic regression model trained on four examples using a zero-mode prior. The Elasticsearch BM25 query is even better, raising the intriguing possibility of using task IDF weighting in Bayesian logistic regression (see Future Work).

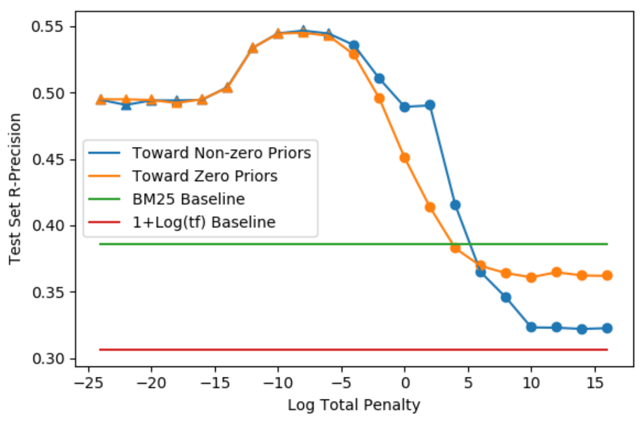
By the time 128 training examples are available, however, both zero-mode and keyword-based priors have much greater effectiveness than even state of the art statistical retrieval using the keyword query, emphasizing why supervised learning is increasingly dominant in HRR tasks.

5.2 Labeled Examples Only

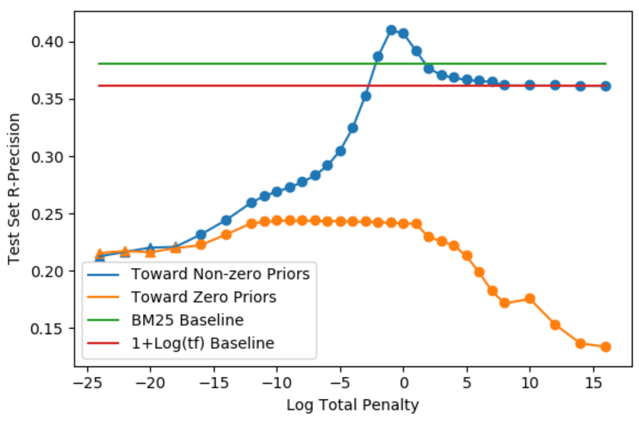
A widely used approach to HRR is to train a regularized logistic regression model on labeled data. The lines for Zero priors in Figure 1 show average R-precision values for this approach for training sets of size 4 and 128, with a range of penalty strengths for L2 regularization. We see the classic humped pattern where intermediate penalty strengths provide maximal effectiveness. The optimal range



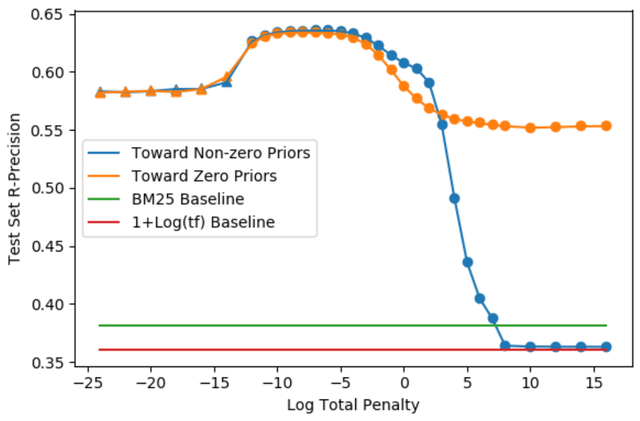
(a) Jeb Bush Collection / Training Set Size = 4



(b) Jeb Bush Collection / Training Set Size = 128



(c) RCV1-v2 Collection / Training Set Size = 4



(d) RCV1-v2 Collection / Training Set Size = 128

Figure 1: Line Charts of training set size 4 and 128. Red and green lines are the query retrieval baselines using log QTF and BM25 model accordingly. The circle dots indicate the statistically significant with 99% confidence between regularizing toward non-zero and zero priors. The triangle dots indicate not statistically significant between the two.

of strengths with 4 training examples is roughly from 2^{-12} to 2^5 for both test collections. With 128 examples, the range is roughly 2^{-10} to 2^{-5}. This is consistent with $O(1)$ scaling of penalties being an effective heuristic, and argues against $O(n)$ scaling.

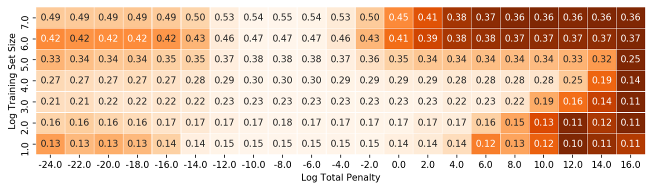
The Figure 2 heatmaps make this point more directly. An effective penalization policy is one that stays in light colored region of these graphs as training set size increases (bottom to top). $O(1)$ policies (vertical columns) provide near optimal effectiveness at all training set sizes for a range of initial penalty strengths. In contrast, $O(n)$ policies are diagonals from lower left to upper right, and risk entering (or never leaving) the zone of low effectiveness. Indeed, these results suggest a decreasing rate of penalization might be even safer, since there appears to be little downside of underpenalization in these averaged graphs.

In practice, however, a user is typically interested in effectiveness on a particular topic, not averaged effectiveness over many topics. We were concerned that averaging R-precision over topics with very different frequencies and effectiveness levels might have hidden important variation among topics. In Figure 3 we therefore break out the individual zero-mode L2 results for 40 RCV1 topics, selected to exhibit all major patterns we found on our 82 RCV1 topics. Topics are sorted from least frequent (upper left) to most frequent (bottom right). The topic name and logarithm base two of the topic frequency are shown for each graph. As before, the x- and y-axes are the base two logarithms of the penalty strength and training set size respectively.

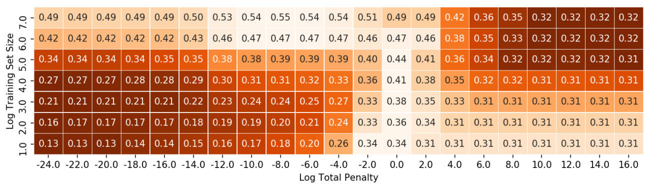
At this level of detail, it is clear that both underpenalization and overpenalization can be dangers, but for different topics. An $O(1)$ scaling, however, is safe in essentially all cases. We found similar behavior for the other RCV1 topics, for L1 regularization instead of L2, and for the Jeb Bush collection instead of RCV1.

5.3 Keywords Plus Labeled Data

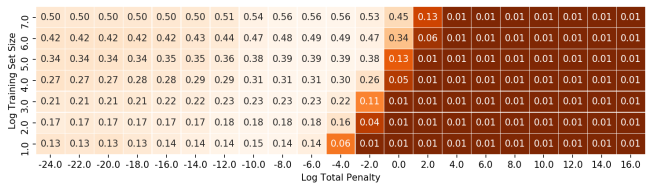
Figure 1 show that a keyword-based (non-zero) prior more than doubles the maximum effectiveness of the usual zero mode prior when only 4 training documents are available. Circles are shown when the improvement for using the non-zero prior is statistically significant at the 99% confidence level under the assumption that topics are independently sampled from some population. While such tests



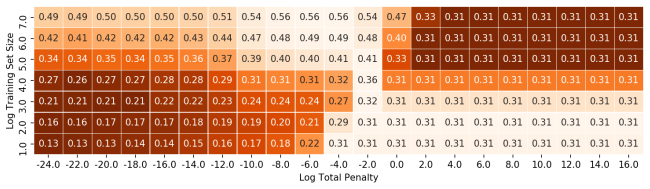
(a) Jeb Bush Collection / L2 Regularization with zero mode



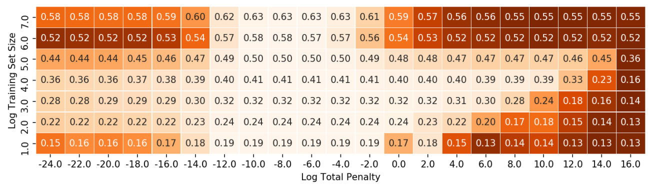
(b) Jeb Bush Collection / L2 Regularization with non-zero mode



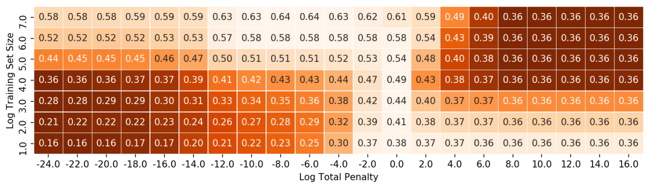
(c) Jeb Bush Collection / L1 Regularization with zero mode



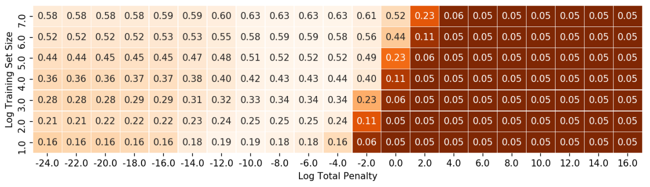
(d) Jeb Bush Collection / L1 Regularization with none-zero mode



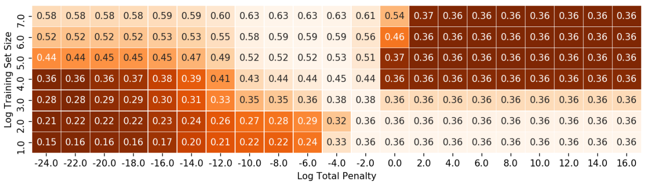
(e) RCV1-v2 Collection / L2 Regularization with zero mode



(f) RCV1-v2 Collection / L2 Regularization with non-zero mode



(g) RCV1-v2 Collection / L1 Regularization with zero mode



(h) RCV1-v2 Collection / L1 Regularization with none-zero mode

Figure 2: R-Precision heatmaps using L1/L2 regularization with zero and non-zero mode on Jeb Bush and RCV1-v2 collection. The value of each cell are averaging over 33 topics for Jeb Bush heatmaps(a-d) and over 82 topics for RCV1-v2 heatmaps(e-h). Cells are colored row-by-row, where the darkest cells are having lowest values and vice versa.

are common in information retrieval research, they should be considered only suggestive: topics are *not* statistically independent of each other [6].

With very weak penalties (left side of graph) we see the same result as for a very weak zero-mode prior: poor effectiveness based on overfitting to the small training set. With very strong penalties (right side of graph) we get simply the effectiveness of the QTF keyword query. The fact that maximum effectiveness is reached at an intermediate level shows that both keywords and training data are being leveraged. That maximum effectiveness is roughly double than of the best zero-mode regularized model.

For a training set of 128 documents, on the other hand, there is little difference between optimal effectiveness with and without keywords, though the keyword query gives some additional margin of error against a poor choice of penalty. This is unsurprising, since a large amount of training allows tuning of coefficient weights to the behavior of the terms in the particular data set, not just taking advantage of the general relatedness of terms. Comparing the numerical R-precision values in Figure 2 tells the same story, showing that a keyword-based prior provides substantial effectiveness improvements until training sets reach roughly 16 to 32 documents in size. This improvement, however, comes with an increase in sensitivity to penalty strength, with a narrower window for optimal effectiveness. In contrast to the zero-mode case, both averaged heatmaps and per-topic heatmaps suggest that a less than $O(1)$ penalization may be desirable to compensate for the narrow window.

The topic heatmaps for non-zero priors show more topic-to-topic variation than was true for zero-mode priors. This is unsurprising, since not just topic richness and effectiveness, but also keyword quality, affect results. Worrisomely, for many topics there is a training set size where optimal penalty strength is very constrained, i.e. the range of good penalty values is particularly narrow. This corresponds to a transition from a state where underregularization (ignoring the keywords) is the greatest risk to a state where overregularization (ignoring the training data) is the greatest risk.

The transition point is heavily influenced by the quality or the usefulness of the keywords. For topics such as `regions_MEX`, `industries_HKONG` and `regions_FRA`, one or two keywords are extremely suggestive of relevance, so a large number of training examples are necessary to do better than the keywords. In contrast, for broad topics such as `topics_C41`, `topics_GSPO` and `topics_CCAT`, the keywords have little value, and training data is superior almost immediately.

6 FUTURE WORK

An obvious next step in our work is to verify our results carry over from balanced training sets to training sets produced by active learning algorithms, as is typical in HRR systems. These include both relevance feedback and variants of uncertainty sampling.

The strong performance of the Elasticsearch BM25 baseline raises the obviously possibility of basing a prior on that technique. This would replace Dayanik, et al's use of a corpus of prior knowledge instances for IDF weighting [**?**] with the IDF statistics from the task corpus itself. An interesting question then is whether IDF weights are better incorporated into the prior mode, or into the penalty strengths.

An interesting challenge is exploring ways to widen the regularization bottleneck that occurs during the transition from preferring keywords to preferring training data. One possible approach would be to replace a single logistic regression model with an ensemble of models drawn from the posterior distribution. This is computationally intensive, but approximations are possible.

Finally, if indeed a sublinear penalization scheme proves superior over a wide range of experiments, it may suggest the need for a different theoretical perspective on regularization in HRR. Properly accounting for the effectively unbounded feature set in HRR problems is one possible avenue.

7 SUMMARY

We have demonstrated that Bayesian MAP logistic regression provides a common framework for handling keywords, training data, or both in high-recall retrieval problems. We have also provided guidance on appropriately scaling the strength of regularization penalties as training set sizes increase, and provided some warnings for users of regularization in established machine learning tools. Keyword queries and supervised learning need not to be viewed as in conflict, but instead can be combined to improve on either alone.

REFERENCES

[1] Mustafa Abualsaud, Nimesh Ghelani, Haotian Zhang, Mark D. Smucker, Gordon V. Cormack, and Maura R. Grossman. 2018. A System for Efficient High-Recall Retrieval. In *SIGIR 2018l*. ACM, New York, NY, USA, 1317–1320. https://doi.org/10.1145/3209978.3210176

[2] Mossaab Bagdouri, William Webber, David D Lewis, and Douglas W Oard. 2013. Towards minimizing the annotation cost of certified text classification. In *CIKM 2013*. ACM, 989–998.

[3] David C Blair and Melvin E Maron. 1985. An evaluation of retrieval effectiveness for a full-text document-retrieval system. *Commun. ACM* 28, 3 (1985), 289–299.

[4] Gordon F. Cormack and Maura F. Grossman. 2014. Evaluation of machine-learning protocols for technology-assisted review in electronic discovery. *SIGIR 2014* (2014), 153–162. https://doi.org/10.1145/2600428.2609601.

[5] Gordon V Cormack and Maura R Grossman. 2015. Autonomy and reliability of continuous active learning for technology-assisted review. *arXiv:1504.06868* (2015).

[6] Gordon V Cormack and Thomas R Lynam. 2006. Statistical precision of information retrieval evaluation. In *SIGIR 2006*. ACM, 533–540.

[7] Aynur Dayanik, David D Lewis, David Madigan, Vladimir Menkov, and Alexander Genkin. 2006. Constructing informative prior distributions from domain knowledge in text classification. In *SIGIR 2006*. ACM, 493–500.

[8] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *JMLR* 9, Aug (2008), 1871–1874.

[9] Alexander Genkin, David D. Lewis, and David Madigan. 2007. Large-Scale Bayesian Logistic Regression for Text Categorization. *Technometrics* 49, 3 (Aug. 2007), 291–304. https://doi.org/10.1198/004017007000000245

[10] Lorraine Goeuriot, Liadh Kelly, Hanna Suominen, Aurélie Névéol, Aude Robert, Evangelos Kanoulas, Rene Spijker, Joao Palotti, and Guido Zuccon. 2017. Clef 2017 ehealth evaluation lab overview. In *CLEF 2017*. Springer, 291–303.

[11] Gene H Golub, Michael Heath, and Grace Wahba. 1979. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics* 21, 2 (1979), 215–223.

[12] Maura R Grossman, Gordon V Cormack, and Adam Roegiest. 2016. TREC 2016 Total Recall Track Overview.

[13] Lihong Li John Langford and Alexander Strehl. 2007. Vowpal wabbit open source project. *Technical Report, Yahoo!* (2007).

[14] Rie Johnson and Tong Zhang. 2013. Accelerating stochastic gradient descent using predictive variance reduction. In *NIPS 2013*. 315–323.

[15] Harold J Kushner and G George Yin. 1997. Applications to Learning, State Dependent Noise, and Queueing. In *Stochastic Approximation Algorithms and Applications*. Springer, 25–46.

[16] David Dolan Lewis. 1992. *Representation and learning in information retrieval*. Ph.D. Dissertation. University of Massachusetts at Amherst.

[17] David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. 2004. RCV1: A New Benchmark Collection for Text Categorization Research. *JMLR* 5 (2004), 361–397. https://doi.org/10.1145/122860.122861

[18] Neal Parikh, Stephen Boyd, et al. 2014. Proximal algorithms. *Foundations and Trends® in Optimization* 1, 3 (2014), 127–239.

[19] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *JMLR* 12 (2011), 2825–2830.

[20] Tamara Rader, Mala Mann, Claire Stansfield, Chris Cooper, and Margaret Sampson. 2014. Methods for documenting systematic review searches: a discussion of common issues. *Research synthesis methods* 5, 2 (2014), 98–115.

[21] Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. 2011. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *NIPS 2011*. 693–701.

[22] Daniel Regard. 2013. A Re-Examination of Blair & Maron (1985). In *DESI Workshop 2013*.

[23] Joseph John Rocchio. 1971. Relevance feedback in information retrieval. *The SMART retrieval system: experiments in automatic document processing* (1971), 313–323.

[24] Adam Roegiest and Gordon V Cormack. 2015. TREC 2015 Total Recall Track Overview. (2015).

[25] Adam Roegiest, Gordon V Cormack, Charles LA Clarke, and Maura R Grossman. 2015. Impact of surrogate assessments on high-recall retrieval. In *SIGIR 2015*. ACM, 555–564.

[26] Shai Shalev-Shwartz and Shai Ben-David. 2014. *Understanding machine learning: From theory to algorithms*. Cambridge university press.

[27] Shai Shalev-Shwartz and Tong Zhang. 2012. Proximal stochastic dual coordinate ascent. *arXiv:1211.2717* (2012).

[28] Alexander Shapiro. 2009. Statistical Inference. In *Lectures on Stochastic Programming: Modeling and Theory*, Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczynski (Eds.). SIAM, 155–252.

[29] Marc A. Suchard, Shawn E. Simpson, Ivan Zorych, Patrick Ryan, and David Madigan. 2013. Massive Parallelization of Serial Inference Algorithms for a Complex Generalized Linear Model. *ACM Trans. Model. Comput. Simul.* 23, 1, Article 10 (Jan. 2013), 17 pages. https://doi.org/10.1145/2414416.2414791

[30] Andreĭ Nikolaevich Tikhonov, A Goncharsky, VV Stepanov, and Anatoly G Yagola. 2013. *Numerical methods for the solution of ill-posed problems*. Vol. 328. Springer Science & Business Media.

[31] Yoshimasa Tsuruoka, Jun'ichi Tsujii, and Sophia Ananiadou. 2009. Stochastic gradient descent training for l1-regularized log-linear models with cumulative penalty. In *ACL-IJCNLP 2009*. Association for Computational Linguistics, 477–485.

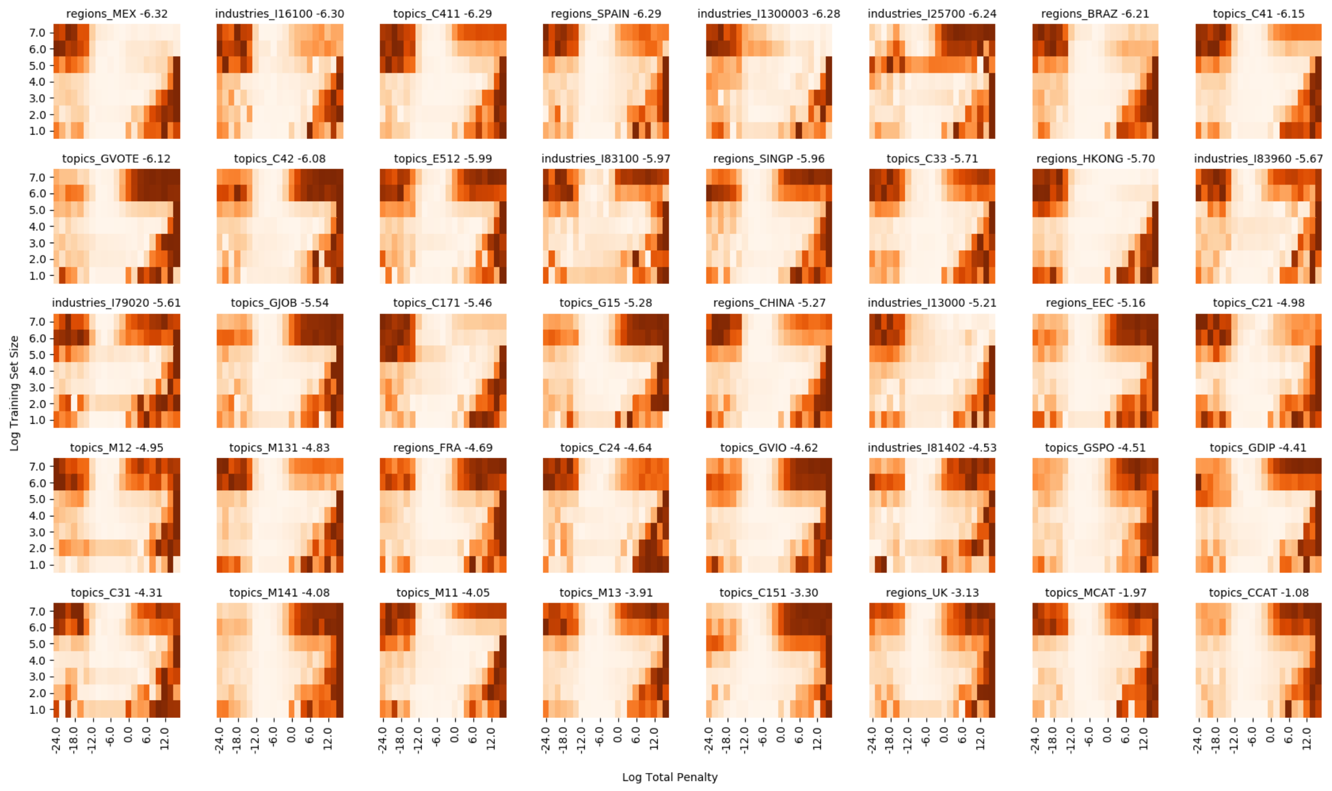
[32] Byron C Wallace, Thomas A Trikalinos, Joseph Lau, Carla Brodley, and Christopher H Schmid. 2010. Semi-automated screening of biomedical citations for systematic reviews. *BMC bioinformatics* 11, 1 (2010), 55.

[33] E. Yang, D. Grossman, O. Frieder, and R. Yurchak. 2017. Effectiveness Results for Popular e-Discovery Algorithms. *ICAIL 2017* (2017).

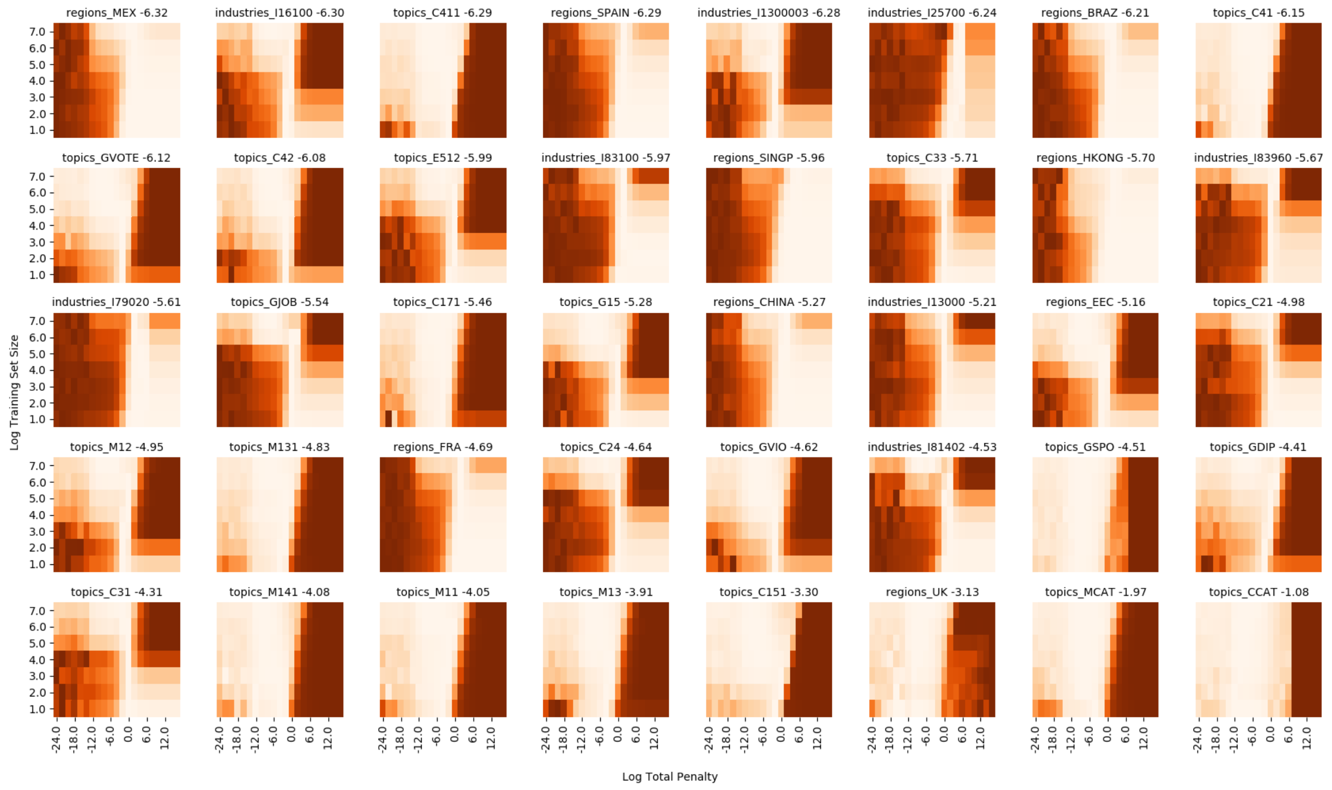
[34] Haotian Zhang, Mustafa Abualsaud, Nimesh Ghelani, Mark D Smucker, Gordon V Cormack, and Maura R Grossman. 2018. Effective user interaction for high-recall retrieval: Less is more. In *CIKM 2018*. ACM, 187–196.

[35] Tong Zhang. 2004. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *ICML 2005*. ACM, 116.

[36] Hui Zou and Trevor Hastie. 2005. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67, 2 (2005), 301–320.



(a) L2 Regularization with zero mode



(b) L2 Regularization non-zero mode

Figure 3: RCV1-v2 collection topic R-Precision heatmaps using L2 regularization with zero and non-zero mode. Topics are sampled from all 82 experimented topics to represent different patterns.