Personalized Classifiers: Evolving a Classifier from a Large Reference Knowledge Graph

Ramakrishna B Bairi
IITB-Monash Research Academy
IIT Bombay
Mumbai, India
bairi@cse.iitb.ac.in

Ganesh Ramakrishnan
Department of CSE
IIT Bombay
Mumbai, India
ganesh@cse.iitb.ac.in

Vikas Sindhwani
Mathematical Sciences
IBM Research
NY 10598, USA
vsindhw@us.ibm.com

ABSTRACT

Identifying the right choice of categories for organizing and representing a large digital library of documents is a challenging task. A completely automated approach to category creation from the underlying collection could be prone to noise. On the other hand, an absolutely manual approach to the creation of categories could be cumbersome and expensive. Through this work, we propose an intermediate solution, in which, a global, collaboratively-developed Knowledge Graph of categories can be adapted to a local document categorization problem effectively. We model our classification problem as that of inferring structured labels in an Associative Markov Network meta-model over SVMs, where the label space is derived from a large global category graph. We propose a joint Active Learning model over the label and the document spaces in order to incorporate active labeling feedback from the users to train the model parameters.

Keywords

Large scale text classification, Text categorization, Topic identification, Multi-label classification, Personalization, Active Learning

1. INTRODUCTION

With the growth of digital data in the form of news, blogs, web pages, scientific articles, books, images, sound, video, social networks and so on, the need for effective categorization systems to organize, search and extract information becomes self-evident.

An important aspect in building a categorization system is the choice of categories. Categories that are very generic, such as News, Entertainment, Technical, Politics, Sports, and the like may not be useful. Thousands of articles could accumulate under each such category and searching for the required piece of information could still be a challenge. On the other hand, fine-grained category creation needs domain experts and is a laborious task.

IDEAS'14 July 07-09 2014, Porrto, Portugal

http://dx.doi.org/10.1145/2628194.2628237.

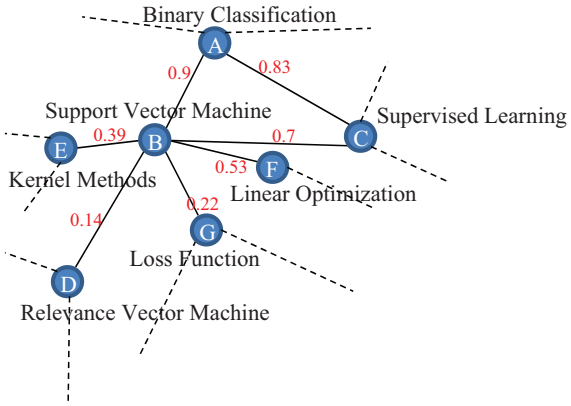


Figure 1: A part of the Knowledge Graph

Adopting predefined categories from an existing classification system (such as Reuters text classification dataset) may not be always suitable. Such a strategy could lead to (i) under or over specific categories (ii) failure to capture user intention (iii) failure to evolve with time.

In this paper we present our attempts to address these practical issues in designing a document categorization system. We call our system EVO. We assume as input to our system, a global Knowledge Graph (KnG) whose nodes are all possible categories, and edges are relationship between the categories. Each category is accompanied by some description of that category. Every relationship edge is also associated with a score between 0.0 to 1.0 indicating the strength of the relationship. This score can be generated using document similarity measurement techniques (such as Jaccard, Cosine, Kernels or semantic similarity methods). Such a knowledge graph can be built collaboratively. For experimental purposes we treat Wikipedia as a knowledge graph. Wikipedia's 4M articles cover the terminology of nearly any document collection [10], which could make it a good candidate for KnG. A part of KnG is shown in Figure 1. For brevity, only one relationship edge is shown between the categories. Next, we need sound techniques for adopting the categories in this KnG to our local collection of documents. In this paper, we propose a technique to solve this problem by learning a model to project the documents into a localized subset of the categories in KnG; this is done by capturing various signals from the documents, exploiting the

knowledge in KnG and using the feedback from a human oracle.

2. FORMAL PROBLEM STATEMENT AND SOLUTION PROPOSAL

We assume that a knowledge graph exists with *all* possible categories (that can cover the terminology of nearly any document collection; for example, Wikipedia) $C = \{C_i\}_{i=1}^{i=f}$ as nodes, and the relationship between them as edges. The categories are associated with some description of that category and edges are associated with a score reflecting the strength of the relationship between two categories. We further assume that an organization receives documents in batches $D_1, D_2, ...$where each batch D_j is received at j^{th} time period (say, j^{th} week/month and the like.) The organization needs to adopt (subset) the categories in KnG to logically build an organization-specific category catalog $C^{org} \subseteq C$ and at the same time, evolve some models to classify all $d_i \in D_j$ into C^{org}. More specifically, we assume the following goals:

1. Learning a personalised model for the association of the categories in KnG to a document collection through active learning and feature design
2. Building an evolving multi-label categorization system to categorize documents into C^{org}.

The eventual goal is to accurately identify suitable categories $\{C_{i_1}, ...C_{i_T}\}$ for every input document $d_i \in D_j \ \forall i, j$. If one could learn an SVM classier for every category in the KnG, identifying all suitable categories for a document would entail determining which classifiers label the document as positive. However, learning such classifiers upfront is prohibitively expensive because, the KnG is usually very large (for example, Wikipedia has four million titles) making it impractical to learn a classifier (SVM) for every category in KnG using limited training data. Hence, it is a challenging task to develop a classification system which can identify a subset of the millions of categories that suit an organization. We attempt to solve this problem from a new perspective of active learning and knowledge propagation techniques, which we explain next. Figure 3 illustrates the overall process of evolving a personalized classifier.

It has been observed that a document that is tagged with a category is expected to contain features such as keywords/phrases that are indicative of that category [8, 6]. For example, the text shown in Figure 2, contains several words/phrases that are indicative of some of the category titles in the KnG (Wikipedia, in our examples.) Techniques such as [9, 6, 8] can be used for spotting such keywords/phrases. We refer to such categories as *candidate categories*. "Keywords Spotter" component in Figure 3 detects candidate categories. However, some of these categories could be either (a) misleading or (b) not relevant in determining the "interesting categories." As an illustration of (a), consider, in Figure 2, the category "Jikes RVM" (which is picked up due to the spotted keyword RVM,) which means Java JVM—not relevant to the document. Thus, the word "RVM" is misleading as a feature. On the other hand, while the category "Cancer" is relevant to the document, the user may want to restrict the choice of categories to the computer science domain, and may therefore, not be interested in categories like "Cancer," thus making a case for (b). Our goal is to develop a personalized categorization system that has the capacity to evolve and learn how to accurately identify only relevant categories. This can be achieved by incrementally learning a classier for each class, based on user feedback. We expect the classifier training to result in feature weights such that the effect of misleading and irrelevant features described above is minimized.

Another benefit of having a candidate categories identification phase is that, it allows us to evolve C^{org} with more categories when the documents with new categories are seen by our system. The spotter can recognize these new categories which can become part of C^{org} eventually. By this process, we overcome the problem of under-specified categories that prevails in the classification systems with predefined categories. However, in the process, we may result in over-specified categories, if we do not control the addition of new categories to C^{org}. We observed that, simple heuristics such as generating a histogram of categories with the number of documents classified under them and then pruning the categories that have very few or very high number of documents can work reasonably well in practice. In addition, our user feedback mechanism, which we explain later in the paper, will also help in limiting the number of categories in C^{org}. More sophisticated approaches to address under or over specified categories using category hierarchies from KnG, which we are exploring currently, will form the part of our future work.

We also observe that categories that get assigned to a document either exhibit semantic relations such as "associations,"[1] "descriptions overlap," and the like or tend to be frequently assigned together (that is, tend to co-occur) in a particular instance of the classification exercise. For example, with the Reuters RCV1-v2 dataset, we observe that all pairs of categories that co-occur even once in the training set, co-occur multiple times in test set. In other instances of classified data such as DMOZ or the Yahoo! Directory, we make an additional observation that co-occurring categories exhibit semantic relations such as "association." For example, the category "Linear Classifier" is related to categories such as "Kernel Methods in Classifiers," "Machine Learning," and the like, and are observed to co-occur as labels for a document on "Classifiers." Another illustration: categories "Support Vector Machines" and "Kernel Methods" exhibit a lot of overlap in their textual descriptions. To sum up, we identify two types of informative features to identify relevant categories for each document: (i) a feature that is a function of the document and a category, such as the category-specific classifier scoring function evaluated on a document and (ii) a feature that is a function of two categories, such as their co-occurrence frequency or textual overlap between their descriptions. We find Associative Markov Network (AMN) [15], a very natural way of modeling these two types of features. Next, we provide a more detailed description of our modeling of this problem as an Associative Markov Network.

For every input document d, we construct a Markov Network (MN) from the candidate categories, such that, each node represents a candidate category $C_i \in C$ and edges represent the association between the categories. Modeling inter-category relations through edges serves two important purposes in our approach: i) When a new organization starts categorizing documents, the classifier models are ini-

[1] http://marciazeng.slis.kent.edu/Z3919/44association.htm

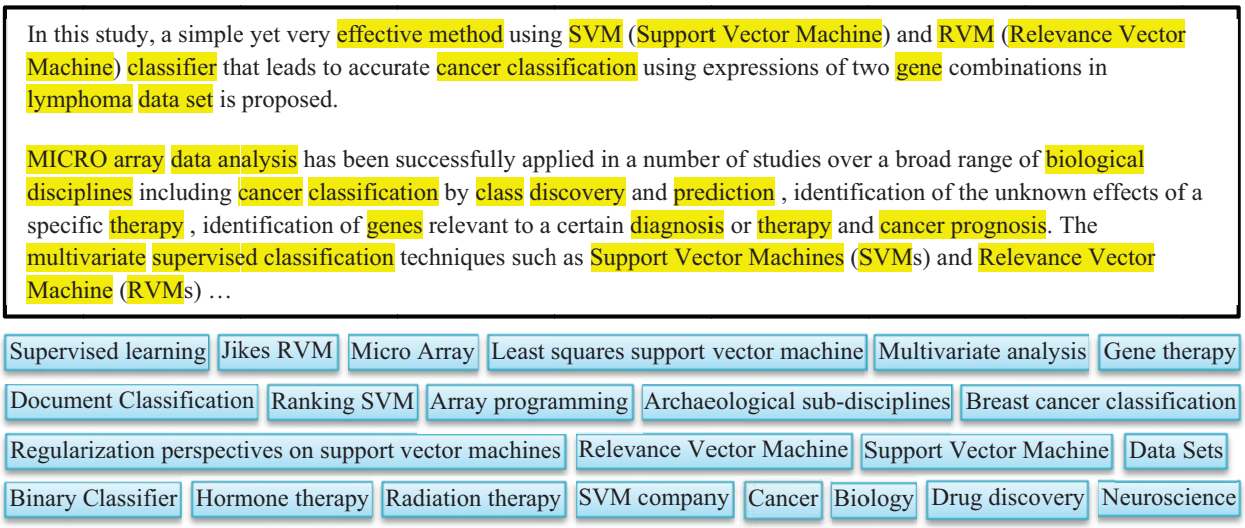


Figure 2: Document with detected keywords (in yellow) and sample candidate categories (in blue)

tially not tuned. The only information available to the categorization system are the category descriptions. It is not practical to assume that perfect descriptions will be available for every category. In such cases, the relationship between the categories can help propagate descriptions across categories via their neighbors. ii) As part of learning the model parameters, the system solicits user feedback on some of the suggested categories for a document. Based on the feedback, the category-specific model (SVM) is updated. The category relationship helps in propagating the learning to the neighbors. This reduces the number of feedbacks needed to learn the model parameters. We will illustrate both these advantages in our experimental section.

Our aim is to learn to assign a binary label (0/1) for every category node C_i in the above MN. Label 1 indicates that the category C_i is valid for the document d and 0 indicates invalid. The collective assignment of labels for all the nodes in the Markov network produces relevant categories for the document d. As we see later in the paper, optimal assignment of these labels can be achieved through MAP inference using Integer Linear Programming.

The "Amn + SVM classifier" component in Figure 3 performs the AMN inference using the learned model parameters and user feedback (along with user defined constrains, explained later in this paper.)

The "Active Learner" component in Figure 3 solicits user feedback (which also includes constraints) and updates model parameters, which is explained later in this paper.

3. LEARNING PERSONALIZED CLASSIFIER

3.1 Building AMN model from categories

For a given document d, we create an MN $G = (N, E)$, whose nodes N are the candidate categories from the KnG and edges E are the association between them, as present in KnG.

In an AMN, only node and edge potentials are considered. For an AMN with a set of nodes N and edges E, the conditional probability of label assignment to nodes is given by

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z}\prod \varphi\left(\mathbf{x}_i, y_i\right)\prod \psi\left(\mathbf{x}_{ij}, y_i, y_j\right) \quad (1)$$

We use notation $\mathbf{x}_i$ to denote a set of node features for the candidate category node C_i and $\mathbf{x}_{ij}$ to denote the set of edge features for the edge connecting C_i and C_j. y_i and y_j are the binary labels for nodes C_i and C_j.

The node features in AMN determine the relevance of a category to the input document d and the edge features capture the strength of the various associations between the categories. Note, here the node features $\mathbf{x}_i$ are computed by considering the node description and the input document text. Hence the above distribution is for a given document d.

Z denotes the partition function given by $Z = \sum_{\mathbf{y}'}\prod \varphi\left(\mathbf{x}_i, y_i'\right)\prod \psi\left(\mathbf{x}_{ij}, y_i', y_j'\right)$.

A simple way to define the potentials φ and ψ is the log-linear model. In this model, a weight vector is introduced for each class label $k = 1..K$. The node potential φ is then defined as $log\varphi\left(\mathbf{x}_i, y_i\right) = \mathbf{w}_n^k \cdot \mathbf{x}_i$ where $k = y_i$. Accordingly, the edge potentials are defined as $log\psi\left(\mathbf{x}_{ij}, y_i, y_j\right) = \mathbf{w}_e^{k,l} \cdot \mathbf{x}_{ij}$ where $k = y_i$ and $l = y_j$. Note that there are different weight vectors $\mathbf{w}_n^k \in \mathbb{R}^{d_n}$ and $\mathbf{w}_e^{k,l} \in \mathbb{R}^{d_e}$ for the nodes and edges.

Using the indicator variables y_i^k we can express the potentials as: $log\varphi\left(\mathbf{x}_i, y_i\right) = \sum_{k=1}^{K}\left(\mathbf{w}_n^k \cdot \mathbf{x}_i\right) y_i^k$ and $log\psi\left(\mathbf{x}_{i,j}, y_i, y_j\right) = \sum_{k=1}^{K}\left(\mathbf{w}_e^{k,l} \cdot \mathbf{x}_{ij}\right) y_i^k y_j^l$; where y_i^k is an indicator variable which is 1 if node C_i has label k and 0, otherwise.

To bring in the notion of association, we introduce the constraints $\mathbf{w}_e^{k,l} = 0$ for $k \neq l$ and $\mathbf{w}_e^{k,k} \geq 0$. This results in $\psi\left(\mathbf{x}_{ij}, k, l\right) = 1$ for $k \neq l$ and $\psi\left(\mathbf{x}_{ij}, k, k\right) \geq 1$. The idea here is that edges between nodes with different labels should be penalized over edges between equally labeled nodes.

Learning feature weight vectors is based on Max Margin training, which is of the form

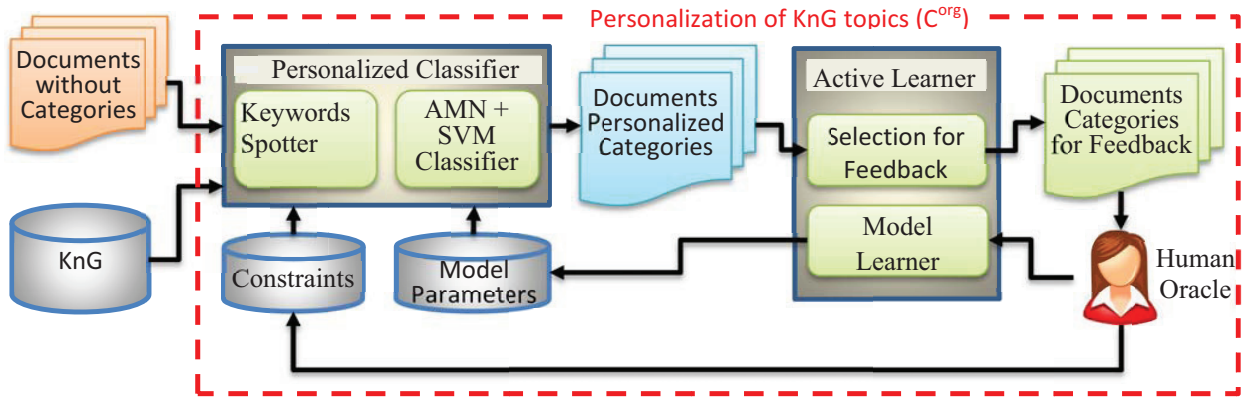


Figure 3: Architecture of KnG category Personalization

$$\underset{\mathbf{w},c}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{w}\|^2 + c\xi$$
$$s.t.\ \mathbf{w}\mathbf{X}\hat{\mathbf{y}} + \xi \geq \max_{\mathbf{y}} \mathbf{w}\mathbf{X}\mathbf{y} + (|N| - \mathbf{y}.\hat{\mathbf{y}}_\mathbf{n})\,;\ \ w_e \geq 0$$

Using compact representation, we define the node and edge feature weight vectors $\mathbf{w}_n = \left(\mathbf{w}_n^1, ..., \mathbf{w}_n^K\right)$ and $\mathbf{w}_e = \left(\mathbf{w}_e^{1,1}, ..., \mathbf{w}_e^{K,K}\right)$, and let $\mathbf{w} = (\mathbf{w}_n, \mathbf{w}_e)$ be the vector of all the weights. Also, we define the node and edge labels vectors, $\mathbf{y}_n = (..., y_i^1, ..., y_i^K, ...)^{\mathrm{T}}$ and $\mathbf{y}_e = (..., y_{ij}^{1,1}, ..., y_{ij}^{K,K}, ...)^{\mathrm{T}}$, where $y_{ij}^{k,l} = y_i^k y_j^l$, and the vector of all labels $\mathbf{y} = (\mathbf{y}_n, \mathbf{y}_e)$. The matrix $\mathbf{X}$ contains the node feature vectors $\mathbf{x}_i$ and edge feature vectors $\mathbf{x}_{ij}$ repeated multiple times (for each label k or label pair k, l respectively), and padded with zeros appropriately. $\hat{\mathbf{y}}$ is the vector of true label assignments given by the training instance. $|N|$ is the number of nodes in the graph G.

We request the reader to refer to[15] for details of solving this optimization.

3.2 Inferring categories for a document

The problem of inference is to select a subset of nodes (that is, categories) from G that have the highest probability of being relevant to the input document. To model this selection, we attach a binary label $\{0, 1\}$ to a node. A node C_i with label 1 is considered to be a valid category for the input document and invalid if its label is 0.

Correctly determining the categories for the input document is equivalent to solving the MAP optimization problem in (2).

$$\max_{\mathbf{y}} \sum_{i=1}^{N}\sum_{k=0}^{1}\left(w_n^k \cdot x_i\right) y_i^k + \sum_{(ij)\in E}\sum_{k=0}^{1}\left(w_e^k \cdot x_{ij}\right) y_{ij}^k \quad (2)$$
$$s.t.\quad y_i^k \geq 0,\ \ \forall i, k \in \{0,1\}\,;$$
$$\textstyle\sum_{k=0}^{1} y_i^k = 1,\ \ \forall i$$
$$y_{ij}^k \leq y_i^k,\ \ y_{ij}^k \leq y_j^k,\ \ \forall ij \in E, k \in \{0,1\}$$
$$y_i^0 \ = 1\ \ \forall i \text{ with Hard Constraints}$$

The variables y_{ij}^k represent the labels of two nodes connected by an edge. The inequality conditions on the fourth line are a linearization of the constraint $y_{ij}^k = y_i^k \wedge y_j^k$; We explain Hard Constraints in section 3.5.

The above MAP inference produces the optimum assignment of labels y_i^k that maximizes the probability function in Equation 1. It can be shown that the Equation 2 produces integer solution when unique solution exists. When $y_i^1 = 1$, we attach the label 1 to the node C_i, and when $y_i^0 = 1$,we attach the label 0 to the node C_i. (Note, both y_i^0 and y_i^1 cannot be 0 or 1 simultaneously, due the second constraint.)

3.3 Personalization of KnG

Personalization is the process of learning to categorize with categories that are of interest to an organization. We achieve this by soliciting feedback from a human oracle on the system-suggested categories and using it to retrain the system parameters. The feedback is solicited as "correct", "incorrect" or "never again" for the categories assigned to a document by the system. In the next few sections, we describe how this feedback is used to train our model for personalization.

3.4 Personalization using per-class SVM

We divide the node features $\mathbf{x}_i$ into two types : i) Global node features $\mathbf{x}_i^s$ and ii) Local node features SVM_i^0 and SVM_i^1. The node feature vector becomes $\mathbf{x}_i = \left[\mathbf{x}_i^s; SVM_i^0; SVM_i^1\right]$.

Global features: These features aid in capturing the structural similarity of a node to the input document through a combination of different kernels such as Bag of Words kernels, N-gram kernels, Relational kernels, among others. The values of global features do not change over time. An example of global feature could be, cosine similarity between the bag of words representations of a document and the description associated with a node in the KnG.

Local features: These features aid in the personalization of KnG. Essentially, we learn an SVM model for a category based on our active learning and user feedback. We employ the decision function of the classifier as a node feature in the AMN. That is, $Svm_{C_i}(d) = \mathbf{w}_{C_i}^T\mathbf{d} + b_{C_i}$, where $\mathbf{w}_{C_i}$ and b_{C_i} are the SVM parameters learned for the category C_i. The output of the SVM decision function is positive if C_i is relevant for the document d and negative, if not relevant. We also treat the output of deci-

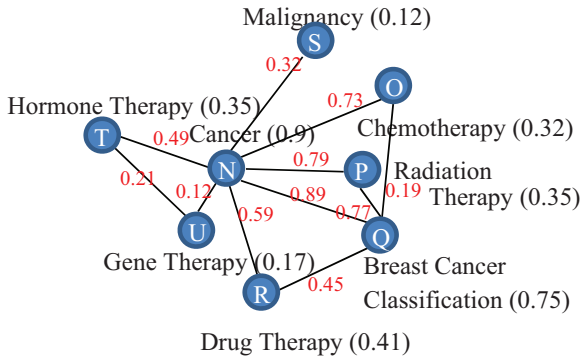


Figure 5: Constraint Propagation: By applying a "never again" constraint on node N, the label of Node N is forced to 0. This forces labels of strongly associated neighbors (O,P,Q,R) to 0. This is due to the AMN MAP inference, which attains maximum value when the labels of these neighbors (with high edge potentials) are assigned label 0.

sion function to be 0 if the SVM model is not available for the category C_i. We introduce two features in the node feature vector $\mathbf{x}_i$, *viz*, SVM_i^1 and SVM_i^0 whose feature values are computed as $SVM_i^1 = \max\left(Svm_{C_i}\left(d\right), 0\right)$ and $SVM_i^0 = \min\left(Svm_{C_i}\left(d\right), 0\right)$.

Due to the associative property of AMN, the SVM parameters learned for a node can also influence the label assignments of its neighbors. In other words, if there is a strong edge potential between categories C_i and C_j, the SVM score propagates from C_i to C_j. This helps in correct detection of the label of node C_j even though there may not be a trained SVM classifier available for node C_j. The example in Figure 4 illustrates the knowledge propagation between highly associated (that is, with high edge potential) nodes. This is precisely what we aim to model using an AMN.

3.5 Personalization from category Constraints

In the process of personalizing the KnG, users can indicate (via feedback) that a category C_i suggested by the system should never reappear in future categorization, because the organization is not interested in that category. For example, an organization working in the core area of Computer Science may not be interested in a detailed categorization of cancers, even though there may be some documents on classification algorithms for different types of cancers. The system remembers this feedback as a *hard constraint*. By *hard constraint* for a category C_i, we mean the inference that is subject to a constraint set that includes $y_i^0 = 1$, as in Equation 2. If categories C_i and C_j are *related*, we would expect the effect of this constraint to propogate from C_i to C_j and encourage y_j^0 also to become 1. As shown in the example in Figure 5, if the user suppresses the category *Cancer* by introducing a hard constraint, the AMN inference will try to suppress *related* categories as well. This is precisely what we aim to model using an AMN.

3.6 Personalization from Active Learning

In the process of providing feedback for a document d, the user needs to mark every category suggested (by the system) for every document, as "correct" or "incorrect". This can produce a lot of cognitive load on the users. To reduce this cognitive load and to achieve a better learning rate, we adopt the Active Learning strategy, where we seek feedback from the user on select categories for select documents. We incorporate information from this feedback for retraining the AMN and SVM model parameters. In a binary classification problem where positive instances are separated from negative instances by a hyperplane, one of effective Active Learning techniques is to choose instances closer to the hyperplane and seek their actual labels (feedback) from the user [16]. This technique is known as uncertainty sampling. However, in our case, the active learning problem has two dimensions: (i) selecting "good documents" for feedback and (ii) selecting "good categories" for feedback. As per uncertainty sampling techniques, "good documents" are the ones with the most uncertain categories. Similarly, "good categories" are the ones with the most uncertain assignment to a document. We propose a novel approach to simultaneous active learning in the document and category spaces next.

3.6.1 *Joint identification of categories and documents for feedback*

For any document d, based on [16], we say that "good" (most uncertain) categories for seeking feedback are those which are closer to the hyperplane separating the categories with label 0 (negative categories) from the categories with label 1 (positive categories) for the document d. To materialize this, we need the notion of hyperplane for separating the nodes (categories) with label 0 from the nodes with label 1 in the Markov Network of categories for a document d. The following claim introduces the notion of a margin separator that separates positive and negative categories for a document.

CLAIM 1. *There exists a feature space and a hyperplane in the feature space that separates AMN nodes with label* 1 *from the nodes that have label* 0 *and that passes through the origin.*

PROOF. Consider a node C_i that is labeled 1 after MAP inference. Let $Nbr_0\left(C_i\right)$ be the neighbors of C_i which are labeled 0 and $Nbr_1\left(C_i\right)$ be the neighbors that are labeled 1. We have,
$\mathbf{w}^1.\mathbf{x}_i + \mathbf{w}^{11}.\sum_{j \in Nbr_1(C_i)} \mathbf{x}_{ij} \geq \mathbf{w}^0\mathbf{x}_i + \mathbf{w}^{00}.\sum_{j \in Nbr_1(C_i)} \mathbf{x}_{ij}$.
Using simple algebraic manipulations, we can re-write this expression as $\overline{\mathbf{w}}.\overline{\mathbf{x_i}} \geq 0$, where $\overline{\mathbf{w}} = \overline{\mathbf{w}^1} - \overline{\mathbf{w}^0}$, $\overline{\mathbf{w}^1} = \left[\mathbf{w}^1; \mathbf{0}; \mathbf{w}^{11}\right]$, $\overline{\mathbf{w}^0} = \left[\mathbf{w}^0, \mathbf{w}^{00}, \mathbf{0}\right]$ and
$\overline{\mathbf{x}_i} = \left[\mathbf{x}_i, \sum_{j \in Nbr_0(C_i)} \mathbf{x}_{ij}, \sum_{j \in Nbr_1(C_i)} \mathbf{x}_{ij}\right]$. $\mathbf{0}$ is a vector of zeros. The expression $\overline{w}.\overline{\mathbf{x}_i} \geq 0$ represents the half space separated by the hyperplane specified by $\overline{w}$, which passes through origin in the feature space of $\overline{\mathbf{x}_i}$. Similarly we can show that the node $\mathbf{x}_i$ labeled 0, resides in the half space $\overline{w}.\overline{\mathbf{x}_i} \leq 0$. □

Based on the notion of the hyperplane defined in Claim 1, we can now choose L categories that are closest to the hyperplane. These are the most uncertain categories of document d for which we seek feedback. Now, we need to select few "good" (most uncertain) documents from the batch D_i for feedback.

The association between documents and categories can be represented as a bipartite graph, with documents on one side and categories on the other side, as shown in Figure 6. Each document is connected to its L most uncertain categories.

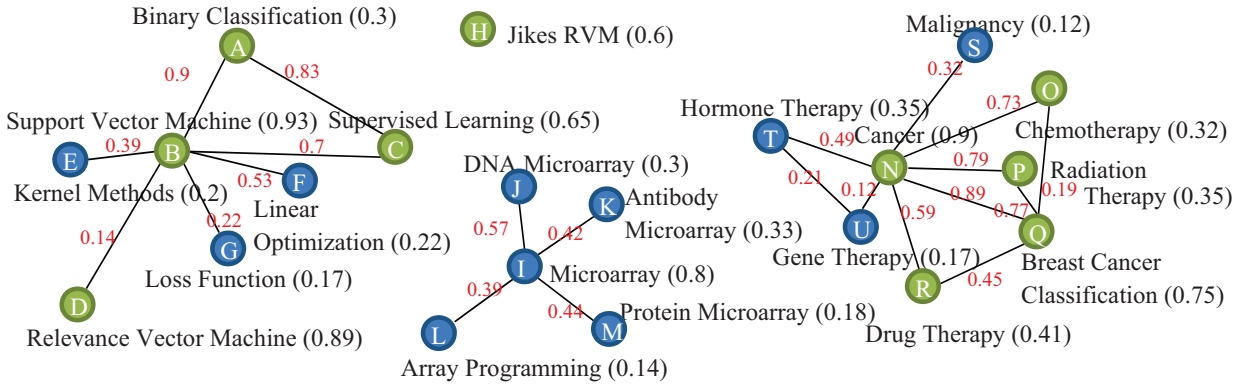


Figure 4: Knowledge Propagation: 1. Nodes B and C with label 1 force the strongly associated neighbor node A to assume label 1. We say that, knowledge from node B and C propagates to node A. 2. Though node I seems to be valid for the document (with high node potential), given the context, it is not. Strongly associated neighbors of I, that is, nodes J,K,L,M which have low node potentials force the node I to attain label 0. Here again we say that, knowledge flows from J,K,L,M to I.

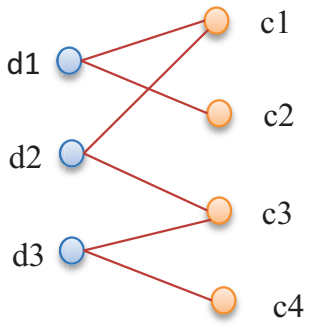


Figure 6: Document Category bipartite graph

Note that a category can be associated with more than one document.

Our approach is to select a subset of documents that results in the maximum coverage of the most uncertain categories. Specifically, we solve the following optimization problem to identify the uncertain documents.

$$\underset{\mathbf{y},\mathbf{z}}{\text{argmax}} \quad \sum a_i y_i + \sum b_j z_j \tag{3}$$

$$\text{s.t.} \quad \sum z_j = P \tag{4}$$

$$\sum z_j \geq y_i \quad \forall i \text{ connected to } j \tag{5}$$

$$0 \leq z_j \leq 1 \tag{6}$$

$$0 \leq y_i \leq 1 \tag{7}$$

$$\forall i \in I \text{ and } j \in J$$

where, I is the set of indices of categories and J is the set of indices of documents.

a_i is the gain associated with selecting the i^{th} category C_i. We choose this to be the maximum uncertainty score of C_i. The uncertainty score of C_i is the *margin distance* [16] from the margin (hyperplane) introduced in Claim 1.

b_j is the gain associated with selecting the j^{th} document d_j. We choose this to be the uncertainty score of $d_j = f(C_{J_1}, ..., C_{J_L})$; for some function f of L categories connected to d_j in the bipartite graph. For example, a simple version of f can be the one that chooses the score of the most uncertain category connected to d_j.

$z_j \in \{0,1\}$ and $y_i \in \{0,1\}$ will be the integer solution at optimality. P is the number of documents for which the user is willing to give feedback. The constraint labeled 4 enforces the selection of P number of documents for the feedback.

Constraint labeled 5 ensures that a category is chosen for feedback ($y_i = 1$) if there exists at least one document associated with that category which is also chosen for feedback ($z_j = 1$).

Feedback is sought from the user for the documents with $z_j = 1$. Note that for each document (with $z_j = 1$), feedback is sought only for those categories that are identified as the most uncertain for that document ($y_i = 1$).

3.7 Inferring C^{org}

So far, we have shown how to infer a set of categories for a document d. We have indicated that these categories come from C^{org}. Essentially, $C^{org} \subseteq C$ is hidden behind our model parameters (AMN and SVM) and hard constraints, which keeps updating with every feedback. For any new document d, when we apply our inference logic, we essentially derive the categories for d from C^{org}. However, if all the members of C^{org} need to be enumerated, we need to infer all the categories of all the documents seen by the organization so far, with the current set of model parameters and hard constraints. However, in practice, we may not have to enumerate C^{org} for the functioning of our system.

Evolving C^{org} over time has two dimensions: (i) evolving C^{org} when new documents with new categories (which exist in KnG) are seen by our system, and (ii) evolving C^{org} when new categories are added to KnG. For the first case, assuming that the collaboratively built knowledge graph KnG is up-to-date with all the categories, our spotting phase identifies the features in the document corresponding to the new categories and adds them to the *candidate categories*. If

these categories get label 1 during the inference, they are considered to be part of C^{org}. For the second case, the challenge lies in updating the already classified documents with the new categories added to KnG. One strategy of handling this could be to look at the neighborhood of newly added categories in KnG, retrieve the already classified documents that have categories present in this neighborhood and reassign categories to these documents by repeating our inference algorithm. In our current work, we limit the evolution C^{org} to case (i). Handling of case (ii) will be part of our future work.

4. EXPERIMENTS AND EVALUATION

4.1 Global Knowledge Graph (KnG)

We extract Wikipedia Page/Article titles and add them to our KnG. We also construct description text for each category in KnG from the first few paragraphs (gloss) of Wikipedia's page. We introduced edges between the nodes connected via hyperlinks to capture the association in terms of text overlap, title overlap, gloss overlap and anchor text overlap.

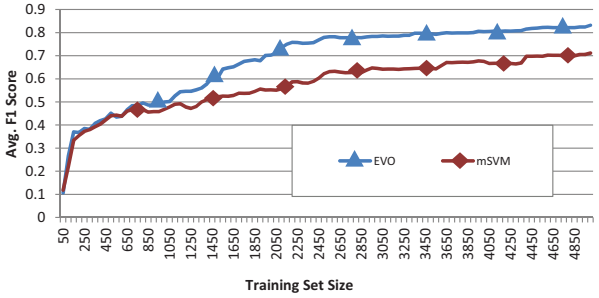
4.2 Data-sets

We report experiments on the RCV1-v2 benchmark dataset and a manually curated dataset from arXiv. Our choice of datasets was based on the existence of at least 100 class labels in the dataset. The Reuters RCV1-v2 collection consists of 642 categories and a collection of 23,149 documents in the training set and 781,265 documents in the test set. The arXiv is an archive for electronic preprints of scientific papers in various fields and can be accessed online. Using the Amazon S3 service, we downloaded 263 technical documents from arXiv under different streams of Computer Science. With the help of eight human annotators, we assigned categories to each document using the vocabulary of Wikipedia article names.

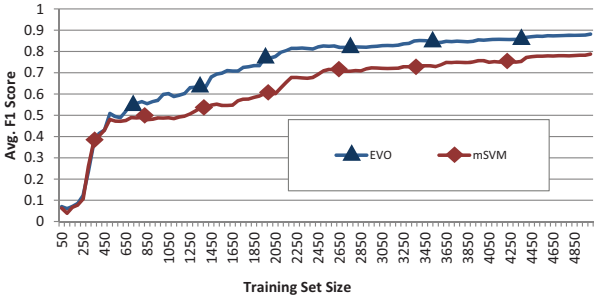
4.3 Evaluation Methodology

We report our results in two different settings: (i) Warm start (ii) Cold start. In all these experiments we refer to our system as EVO.

In the Warm start setting, we assume that the user has a fair idea of what categories she needs and has identified them *a priori*. Such a setting helps us demonstrate how, on a standard classification dataset, the Markov network helps propagate learnings from a category to other *related* categories. We performed warm start experiments on the Reuters RCV1-v2 collection. We selected 66 pairs of *related* Reuters categories, spanning 96 categories. For example, the categories MANAGEMENT and MANAGEMENT MOVES are *related*. Two categories were considered *related* if the number of training documents carrying both labels, exceeded a certain threshold. We picked 5000 training documents and 2000 test documents using this clustered sampling procedure. We further divided the training set into 100 batches of 50 documents each. We iterated through the batches and in the k^{th} iteration, we trained our model (SVMs, AMN feature weights) using training documents from all batches up to the k^{th} batch. For each iteration, we performed AMN inference on the sample of 2000 test documents. In Figure 7, we report the best average F1 score (observed when SVM parameter C=1 and 10) on the test



(a) With SVM parameter C=1



(b) With SVM parameter C=10

Figure 7: Comparison of avg (macro) F1 scores of our system (EVO) with SVM on different c values

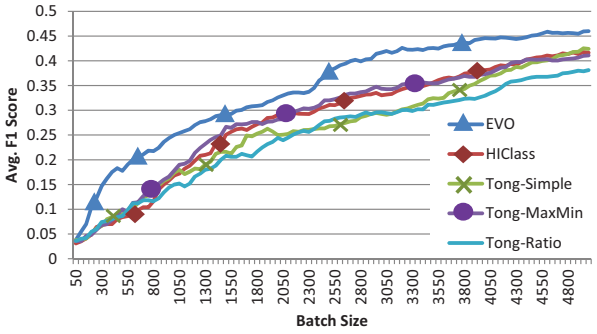
sample for each of the 100 iterations. Clearly, our technique - EVO - outperforms the multi-label SVM (mSVM).

We also compared our proposed joint active learning technique with other techniques from literature, namely, HIClass [5] and Tong [16] (adapted to multi-class classification). In Figure 8a we compare the average F1 scores. Clearly, joint learning on label space and document space achieves a better learning rate compared to an active learner only in the document space. Viewing it as number feedback needed to achieve a required level of F1 score, we observe in Figure 8b that, with this joint learning, we need significantly less feedback, resulting in lesser cognitive load on the user.

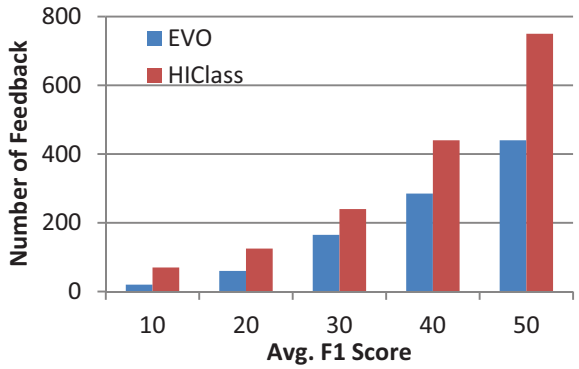
In the Cold start setting, we assume that the user does not have any predefined categories to start with. She wants to adapt the categories from KnG. We performed cold start experiments on the arXiv document collection and carried out five-fold cross-validation with each fold containing 210 training documents and 53 test documents. In each fold, we trained our model (SVMs, AMN feature weights) using the training set and evaluated Consistency, Precision and Recall on the test set. During the training phase, we also applied localization techniques in which we recorded feedback for the system suggested categories in three forms: "Correct,", "Incorrect" and "Never again."

We measured consistency [10] as $Consistency = \frac{2C}{A+B}$, where A and B are the total number of categories two systems assign (in this case, one is our system and other is a human labeler) and C is the number they have in common.

Table 1 shows the overall consistency of our system with human annotators. We also have compared the consistency with the WikipediaMiner (WM) [12] system. Note that, in



(a) F1 score comparison



(b) Feedback comparison

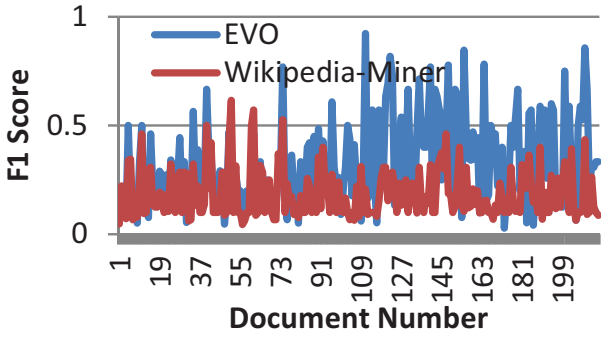
Figure 8: Comparing our Active Learning technique with others

some cases, WikipediaMiner may generate a category which is relevant to the input document; it may, however, be out of the computer science domain. We have treated such labels as incorrect because we are interested in evolving an organization-specific categorization system, which is the goal of this paper.

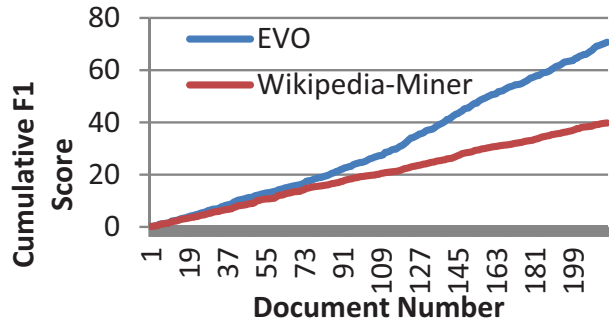
In Figure 9 we compare the F1 scores of all the documents with WikipediaMiner. In this experiment, we picked up arXiv documents one by one, generated categories and recorded user feedback for 10 categories. We computed the F1 score for each document by considering the number of categories retrieved by our system and those entered by the human annotators. As evident from Figure 9, our system performs better than WikipediaMiner due to its learning

Number of Documents	263
Categories Discovered by Human Labelers	1054
Categories Discovered by EVO	819
Common categories: EVO and Human Labelers	353
Categories Discovered by WM	1943
Common categories: WM and Human Labelers	368
Avg Consistency over all docs by EVO	**37**.69%
Avg Consistency over all docs by WM	24.56%

Table 1: Cold Start experiment results and comparison with WikipediaMiner (WM)



(a) Documentwise F1 score



(b) Cumulative F1 Score

Figure 9: F1 score comparison of EVO with WM

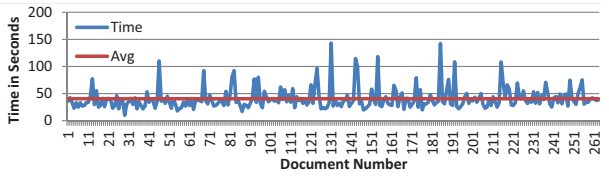


Figure 10: Time measurements for EVO

ability and propagation of learning to neighbors over AMN. (The cumulative F1 score in 9b is the sum of F1 scores of all documents)

In Figure 10 we show the time requirement of our system to discover the categories for the 263 arXiv documents. On an average, it took around 47 seconds to discover the categories for the documents of length about 10 pages, double column, similar to this document. The time needed by our inference procedure depends upon the number of spots, number of nodes and edges in the graph formed by the candidate categories. We ran our experiments on a machine with 16 GB RAM, Six-Core AMD Opteron(tm) Processor 2427.

In Figure 11, we show an example of set of categories generated by our system for the abstract of an arXiv document. Note that, the system was personalized for Computer Science domain (with few hard constraints on categories like Cancer, Gene, Peptides and the like) before generating the categories. Hence, certain categories like 'Gene Expression' are suppressed during the inference.

5. PRIOR WORK

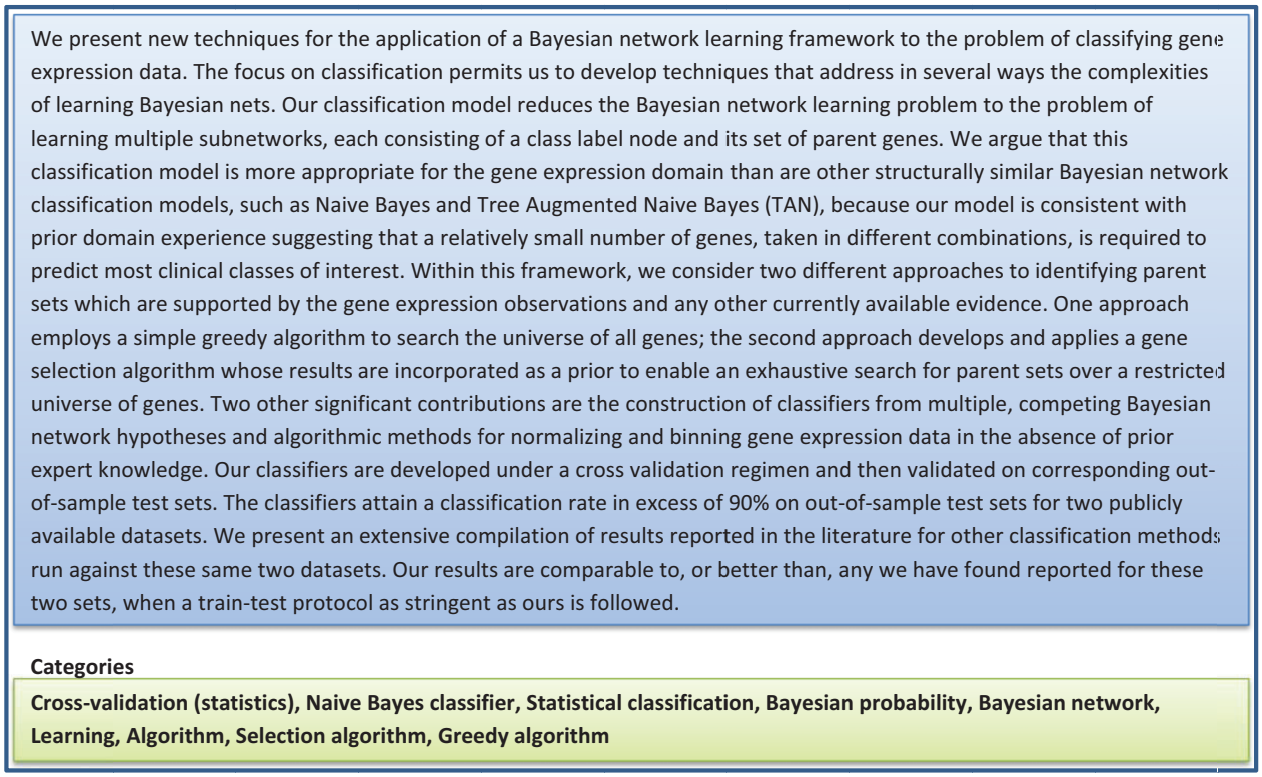


Figure 11: Abstract from an arXiv document with the categories assigned by our system personalized for Computer Science domain.

Text categorization systems with active learning capability iteratively selects a sample of the data to a label based on some selection strategies, suggesting that the data most deserves to be labeled. Thus, it can achieve a comparable performance with supervised learners while using much less labeled data. Active learning becomes very important in multi-label text classification as the human oracle needs to label all possible categories for each instance. Thus, the effort of assigning labels for multi-label data is much larger than that for the single-label data. Many algorithms have been proposed [16, 5, 11] which adopt uncertainty-based principles for active learning. Dan Roth et al. [13] present global and local margin-based techniques for active learning in the structured output spaces with multiple interdependent output variables. Aron Culotta et al. [3] present a new active learning paradigm which reduces not only the number of instances the annotator must label, but also the difficulty of annotating each instance. We propose a new technique for the joint Active Learning on label space and category space, in which a stream of documents are continuously received. Based on this feedback, we learn a model that tailors global categories to a local set of documents. In this process, we also learn curated training data.

[14] present an algorithm to build a hierarchical classification system with predefined class hierarchy. Their classification model is a variant of the Maximum Margin Markov Network framework, where the classification hierarchy is represented as a Markov tree.

Topic Modeling in an unsupervised setting has been studied in CTM[2], PAM[7], NMF[1], which identify topics as a group of prominent words. Discovering several hundred topics using these techniques turns out be practically challenging with a moderately sized system. In addition, finding a good representative and grammatically correct topic name for a group needs additional effort.

Nadia and Andrew [4] explore multi-label conditional random field (CRF) classification models that directly parameterize label co-occurrences in multi-label classification. They show that such models outperform their single label counterparts on standard text corpora. We draw inspiration from [4] and jointly make use of relations between the categories in KnG along with the category similarity to the document to learn the categories relevant to a document.

Medelyan [10] detect topics for a document using Wikipedia article names as category vocabulary. However, their system does not adapt to the user perspective. Whereas, our proposed techniques support personalized category detection.

6. CONCLUSION

We presented an approach for evolving an organization-specific multi-label document categorization system by adapting the categories in a global Knowledge Graph to a local document collection. It not only fits the documents in the digital library, but also caters to the perceptions of users in the organization. We address this by learning an organization-specific document categorization meta-model using Associative Markov Networks over SVM by blend-

ing (a) global features that exploit the structural similarities between the categories in the global category catalog and input document and (b) local features including machine learned discriminative SVM models in an AMN setup along with user defined constraints that help in localization of the global category catalog (Knowledge Graph). In the process, we also curate the training data. Currently our system works only with a flat category structure. We believe that our technique can be improved to handle a hierarchical category structure, which will form part of our future work.

7. REFERENCES

[1] S. Arora, R. Ge, and A. Moitra. Learning topic models - going beyond svd. *CoRR*, abs/1204.1956, 2012.

[2] D. Blei and J. Lafferty. Correlated Topic Models. *ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS*, 18:147, 2006.

[3] A. Culotta and A. McCallum. Reducing labeling effort for structured prediction tasks. In *Proceedings of the 20th national conference on Artificial intelligence - Volume 2*, AAAI'05, pages 746–751. AAAI Press, 2005.

[4] N. Ghamrawi and A. McCallum. Collective multi-label classification. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, CIKM '05, pages 195–200, New York, NY, USA, 2005. ACM.

[5] S. Godbole, A. Harpale, S. Sarawagi, and S. Chakrabarti. Hiclass: Hyper interactive text classification by interactive supervision of document and term labels.

[6] K. M. Hammouda, D. N. Matute, and M. S. Kamel. Corephrase: Keyphrase extraction for document clustering. In *Proceedings of the 4th International Conference on Machine Learning and Data Mining in Pattern Recognition*, MLDM'05, pages 265–274, Berlin, Heidelberg, 2005. Springer-Verlag.

[7] W. Li and A. McCallum. Pachinko allocation: DAG-structured mixture models of topic correlations. *Proceedings of the 23rd international conference on Machine learning*, pages 577–584, 2006.

[8] Z. Liu, W. Huang, Y. Zheng, and M. Sun. Automatic keyphrase extraction via topic decomposition. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 366–376, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[9] O. Medelyan and I. H. Witten. Thesaurus based automatic keyphrase indexing. In *Proceedings of the 6th ACM/IEEE-CS Joint Conference on Digital Libraries*, JCDL '06, pages 296–297, New York, NY, USA, 2006. ACM.

[10] O. Medelyan, I. H. Witten, and D. Milne. Topic indexing with wikipedia, 2008.

[11] P. Melville and V. Sindhwani. Active dual supervision: reducing the cost of annotating examples and features. In *Proceedings of the NAACL HLT 2009 Workshop on Active Learning for Natural Language Processing*, HLT '09, pages 49–57, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[12] D. Milne. An open-source toolkit for mining wikipedia. In *In Proc. New Zealand Computer Science Research Student Conf*, page 2009, 2009.

[13] D. Roth and K. Small. Margin-based active learning for structured output spaces. In *Proceedings of the 17th European conference on Machine Learning*, ECML'06, pages 413–424, Berlin, Heidelberg, 2006. Springer-Verlag.

[14] J. Rousu, C. Saunders, S. SzedmÃąk, and J. Shawe-Taylor. Kernel-based learning of hierarchical multilabel classification models. *Journal of Machine Learning Research*, 7:1601–1626, 2006.

[15] B. Taskar, V. Chatalbashev, and D. Koller. Learning associative markov networks. In *Proceedings of the twenty-first international conference on Machine learning*, ICML '04, pages 102–, New York, NY, USA, 2004. ACM.

[16] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66, 2001.