Text Classification with Evolving Label-sets

Shantanu Godbole
IIT Bombay
shantanu@it.iitb.ac.in

Ganesh Ramakrishnan
IBM India Research Lab
ganramkr@in.ibm.com

Sunita Sarawagi
IIT Bombay
sunita@it.iitb.ac.in

Abstract

We introduce the evolving label-set problem encountered in building real-world text classification systems. This problem arises when a text classification system trained on a label-set encounters documents of unseen classes at deployment time. We design a Class-Detector module that monitors unlabeled data, detects new classes, and suggests them to the administrator for inclusion in the label-set.

We propose abstractions *that group together tokens under human understandable concepts and provide a mechanism of assigning importance to unseen terms. We present generative algorithms leveraging the notion of* support *of documents in a model for (1) selecting documents of proposed new classes, and (2) automatically triggering detection of new classes. Experiments on three real world taxonomies show that our methods select new class documents with high precision, and trigger emergence of new classes with low false-positive and false-negative rates.*

1. Introduction

One important challenge in building text classification systems is that the constitution of unlabeled data changes over time. Often, new classes are introduced and need to be detected and folded into the system. We call this the **evolving label-set** problem. For example, consider a classification problem with n classes, where the classes are documents about certain countries (*India*, *US*, *UK*, ...). Over a period of time, a new country's documents (say *Australia*) are introduced into the system. The evolving label-set problem is to detect such (one or more) new classes, propose a cohesive set of documents for training the new classes, get user validation about those fitting in with the label-set, and fold these new classes into the classification system. Such problems occur especially when a nascent classification system is built from scratch, the entire set of labels is not known beforehand, and the user's understanding of the label-set evolves over time.

Such phenomenon is common place in web directory systems that manually classify ever-changing web-pages. For example, a directory of scientific disciplines would need to add "bio-informatics" as it emerged as a new discipline, or add an industry type "cell-phones" when they started becoming popular.

Our main contributions in this paper are (1) design of generative algorithms for identifying new classes, and (2) introduction of *abstractions* to capture importance of unseen terms. Abstractions provide intuitive representation of documents clearly revealing classification criteria to the user. Experiments with real-life taxonomies show our methods achieving 60–90% precision for suggesting unlabeled documents comprising a new class and low error rates in automatically detecting new classes of about 15%.

2. Problem Setting

We envision a scenario where a separate module for new class detection continuously monitors unlabeled (new) documents of a text classification system. When this Class-Detector module gathers enough evidence of an emerging new class, it sends a trigger to the administrating user. Alternately, the user could periodically query the module during maintenance. The Class-Detector presents to the user, a ranked list of documents that could comprise a new class. The user can thereafter choose to add a new class to the system if it fits in with the initial label-set. Our methods for tackling the evolving label-set problem *do not interfere* with the working of the main classifier, which can be any high-performance well-tuned learner like SVMs [7], that could be different from learners that best detect new classes.

The two main technical components of the Class-Detector are (1) the **document selector** that picks a ranked list of documents comprising a possible new class, and (2) the **new class trigger** that decides if there is enough consistent divergence in the unlabeled set to define a new class. We expect to find a new class when there are a significant number of unlabeled documents that do not fit the existing class structure and which are themselves coherent enough to be grouped into a class. Converting this intuition into a robust procedure poses two main challenges. Firstly,

separating out documents forming a new class from mis-classified documents on the basis of being "misfits" in the existing model, is challenging, particularly in the presence of multi-labeled documents. If selected documents for proposed new classes contain several mis-classifications, the user may get confused about the nature and scope of proposed classes. Secondly, documents of new classes likely contain terms not seen during training. Even unlabeled documents contain new terms, thus eliminating the possibility of depending on term frequency to detect new classes.

Usually, the importance of terms in supervised learning is established explicitly using statistical metrics like information gain. Metrics depending on labeled data are not applicable here. We depend on indirect methods to establish term importance via a notion of term **abstractions** that assigns importance to a family of terms together. Abstractions indicate various properties of features based on usage in documents. Examples are Named-Entity (NE) tags, part-of-speech (POS) tags, formatting features in HTML, and match with external dictionaries or keyword ontologies. *E.g.*, a classification system based on countries would perform well by looking only at the location NE-tags (or location dictionaries) of documents. Abstractions help a user interpret the criteria used for defining new classes.

Figure 1 highlights importance of choosing correct abstractions for understanding the *Industries* taxonomy from RCV1 [9]. The full vocabulary makes it hard to judge and understand what the label-set is about, whereas, looking only at organization names makes it clear that the label-set is about industry types. New classes discovered in unlabeled data based on the full vocabulary here may not help the user judge the nature or constitution of the new class. The correct set of abstractions (organization names here) helps in understanding and identifying new industry types.

Full vocab
bank,issue,warrant,fee oil,crude,million,refinery compuserve,service,subscribe,cost
Organization names
Comm. Bank of Aus, Commerzbank, Central Bank, Fleet Fin Group Gulf Oil, Chinese Petro Corp, Esso Aus Ltd, Natural Gas Corp Europe Online, Compuserve, First Data Corp, AOL

Figure 1. Indicative features for a label-set

3. Class-Detector methods

In this section, we propose generative methods based on *support* for both parts of the Class-Detector modules, selecting documents and automatically triggering new classes. Generative models for text like LDA [3], Aspect [6] and BayesANIL [11] model the process of generation of documents and document features (*e.g.* words) from classes. Given a corpus of documents $d \in \mathcal{D}$ and some of the documents labeled with classes $c \in \mathcal{C}$ and $|\mathcal{C}| = n$, BayesANIL provides an estimate of the joint distribution $Pr(c, d)$ of training documents and classes using a generalization of Expectation Maximization (EM). $Pr(c, d)$ can be interpreted as a measure of *support* for membership of d in c. The marginal probability $Pr(d) = \sum_{c \in \mathcal{C}} Pr(c, d)$ is a measure of how well d fits into the existing label-set $\mathcal{C}$. We choose BayesANIL since experiments [11] show that $Pr(c, d)$ reflects the notion of support in classification contexts in the presence of noisy, approximate and incomplete labeling while also folding in evidence from unlabeled documents. In general, we could use any generative model that (1) provides such a notion of support via joint $Pr(c, d)$ and (2) folds in feature evidence from unlabeled data.

A simple method of selecting documents belonging to a new class is to select documents with the lowest $\Pr(d)$ values (method called *SortPrD*). Another method for suggesting new class documents is to seed an $(n+1)^{th}$ class with documents having the lowest $Pr(d)$ values, re-train the generative model for $(n+1)$ classes, and select unlabeled documents with the highest $Pr(c_{n+1}, d)$. We call this method *PrDNewClass*. It is likely that documents selected by both these methods will include documents of existing classes that were mis-classified either due to noise or for being multi-labeled. Next, we propose an algorithm that avoids this limitation.

The *GenSupp* algorithm: We project all training and unlabeled documents in an n-dimensional *support space* where the components for d along the n dimensions are its $\Pr(c, d)$ values. For the training documents, the $\Pr(c, d)$ values could be pre-computed during the training phase and stored. We then use a hierarchical clustering (HAC) algorithm to group similar documents with Ward's method for combining clusters, using average KL-distance between $\Pr(c, d)$ vectors as distance between the documents.

We grow the dendrogram till we have a large number (say $5n$) of small clusters. Since $Pr(d)$ scores give the probability of generating the document from the model, we expect the lowest $Pr(d)$ values to be assigned to the new class or noisy, multi-labeled documents. We found this to be empirically true. To get tight sub-clusters from these candidates, we chose clusters which had the lowest average value of $Pr(d)$ of its constituents. This algorithm is called *GenSupp* for Generative model method based on Support. We require each candidate cluster to have a minimum number of unlabeled documents (say 5) to guard against outliers and to be able to define a new class. We also require clusters to be *pure* that is the fraction of training documents is at most $p\%$ (we chose 20%); this ensures that the new class lies in an area of support space where there are no (or few) training documents in the vicinity.

Automatically triggering new classes: We now consider the problem of detecting whether or not selected documents indeed comprise a new class. In general, given a classifier, there will exist several unlabeled documents that are not classified into any of the existing classes, or that have very low probability of being generated by the learned model (low $Pr(d)$). Only a small subset of these will be due to the introduction of a new class. We approach the problem as follows. Let $T = \{T_1, T_2, \ldots, T_n\}$ be the training documents for the original n-classes. We keep aside a set $V = \{V_1, V_2, \ldots, V_n\}$ of documents for measurement. Another set $U = \{U_1, U_2, \ldots, U_n\}$ is treated as unlabeled documents on which we invoke *GenSupp* to cluster T and U together. We choose one of the resultant clusters that predominantly has unlabeled documents of some U_i (corresponding T_i already exists in the label-set). We now add a fake class T_{n+1} which contains cohesive unlabeled documents from this cluster. We re-train this $n+1$ class document collection using BayesANIL and find the value of two measures $MV = \sum_{d \in V} Pr(c_{n+1}, d)$, and $MT = \sum_{d \in T_{n+1}} Pr(c_{n+1}, d)$. We repeat this for each class in the label-set, each time choosing a *GenSupp* cluster containing unlabeled documents of that class, and introduce its fake replica.

MV measures the support for the validation set V from the newly added class, and MT measures the support of the newly added class for itself. n such fake class additions gives us n prototype values which we store as MV_i and MT_i, where $i = 1 \ldots n$. These MV and MT vectors prototype the range of values these support measures take when fake classes are introduced into the label-set.

We expect that if we really detect a new class from the unlabeled data, then it's corresponding MT_{n+1} value should be higher than all previous MT_i's. Since the fake classes always had a corresponding class in T, these documents in T_{n+1} share the probability mass of $d \in T_i$ for some T_i. A real new class will take away some probability mass from all classes in T and $MT_{n+1} > MT_i \; \forall i = 1 \ldots n$. Similarly, we should get $MV_{n+1} < MV_i \; \forall i = 1 \ldots n$ for a genuine $(n+1)^{th}$ class because a genuine new class have very low support for the n-class documents $d \in V$. MT and MV heuristics are first steps in looking at the new-class detection problem. New-class detection can be converted into a general learning problem – making binary decisions about existence or absence of new classes given model support.

4. Experiments

Setup: We conducted Class-Detector experiments with the RCV1 dataset [9]. For evaluating our algorithms for the evolving label-set problem, we used the first two days' news stories - first day for training and second day as unlabeled. We considered the 20 most populous classes from the *regions*, *topics*, and *industries* taxonomies without root-level classes. This resulted in 4525, 11637, and 1571 documents respectively. While a rich variety of abstractions can be used, here we report experiments with the full vocabulary (global *G*), and location (*L*), organization (*O*), and person name (*P*) NE-tags which worked best with these datasets. We used a custom named-entity tagger [10] for finding NE-tags. Tagging was imperfect and noisy, yet our methods worked well. We used a Java implementation of BayesANIL [11] and HAC software [1] written in C.

Selecting new class documents: For each dataset, we hid a class in turn in every experiment from the training data and introduced it in unlabeled data. We checked if our algorithms could detect this hidden class and suggest a good set of its documents for inspection. Our algorithms present a ranked list of suggestions and we measured average precision (ratio of correctly suggested new-class documents to all suggestions). We used 20 suggestions for the reported experiments and results with varying number of suggestions were similar. We believe 20 is a good number for the user to judge existence of a new class fitting the existing label-set. Average precision over 20 experiments is reported for *GenSupp* ($G20$), *SortPrD* ($P20$), and *NotaSVM* ($N20$). In Figure 2, we see the precision values for all datasets with their best abstractions. The graphs reveal interesting results. First, our $G20$ method is either better or at par with $P20$ baseline method in most dataset-abstractions combinations. $N20$ is usually worse than both these methods. This illustrates that while the $\Pr(d)$ scores are valuable for detecting new classes, they by themselves do not suffice, and it is important to account for coherency of the selected documents in defining a possible new class via *GenSupp*.

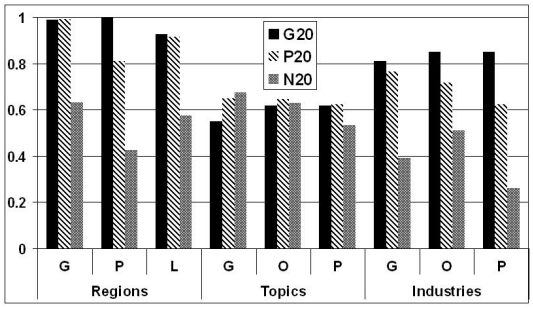

Figure 2. Precision for selecting documents

Second, we see that the precision of selecting new class documents with abstractions is as good as or much better than that obtained with the full vocabulary (*G*). Abstractions play an important role in *industries* where *O* and *P* provide higher precision than *G*. Similarly, *P* works best for *regions*, and *O* works best for *topics*.

[1] http://www.let.rug.nl/~kleiweg/clustering/clustering.html

Triggering new classes: We report experiments for detecting new classes according to the MT and MV measures. We report 20 experiments, with *regions* and *industries*, for counting errors in automatically triggering detection of new classes. To measure false negative (FN) trigger rate, we hid one class in training data, introduced it in unlabeled data, and triggered detection based on MT and MV values being beyond the fake-class prototype vectors as outlined in Section 3. We measured false positive (FP) rates by hiding one class, *not* introducing it in unlabeled data, and determined false triggering in detecting classes from prototypes. Figure 3 shows abstractions working better than *G* for triggering classes with low FN and FP rates. We see that MT and MV work well, one being better than the other depending on the dataset.

	False Negatives				False Positives			
	Reg		Ind		Reg		Ind	
	G	P	G	O	G	P	G	O
MT	8	3	14	7	5	3	8	5
MV	9	7	13	5	6	5	7	4

Figure 3. FN and FP rates out of 20 runs

5. Related Work

Our work is related to the work on Topic Detection and Tracking (TDT) [2, 1, 12] but the problem setting and approaches are different. The aim of TDT is to monitor an online feed of news stories and to detect the first occurrence of a new real world event reported in the news. This is called First Story Detection (FSD) and is followed by tracking further follow up news stories about the event. Most popular techniques at new event detection and tracking (Allan *et al.* [2, 12]) involve a single pass clustering algorithm with well-tuned novelty detection thresholds. Incoming stories are compared to prototypes of existing events and if more than a threshold away, these stories spawn new events. Some systems also explore the use of NE tags [4, 12] to define more meaningful similarities between documents. This is related our notion of abstractions, but abstractions are more general and not limited to NE tags. In summary, most work on TDT needs to rely on unsupervised clustering techniques using word-based or NE tags-based similarity. In our setting, categories are limited and known in advance. This makes it possible to project documents in a space that better captures their grouping as far as the set of classes in concerned. Also, most TDT systems cannot handle multi-labeled (multi-event) stories. Concept drift [8] in classification is a related field of work, but quite different from our setting. In concept drift, the distribution of indicative words, and pattern of an *one* class changes over time.

6. Conclusions

We have introduced the evolving label-set problem and presented generative methods for dealing with this problem in text classification systems. We introduced abstractions, helpful for the user in understanding label-sets, and use them as a basis for detecting new classes. We have also developed discriminative methods for the evolving label-set problem and observed that while accuracy of discriminative methods is higher, generative methods performed the document selection and triggering tasks better. This is due to the notion of model support for documents in a classification setting. Details, experiments, and analysis appear in the first authors PhD thesis.

In future work, we would like to integrate evolving label-set detection in working text classification systems and workbenches like HIClass [5]. We would also like to study detection of more than one class at a time, and in the presence of concept drift.

Acknowledgements: The first author is supported by the Infosys Fellowship Award from Infosys Technologies Limited, India.

References

[1] J. Allan, V. Lavrenko, and H. Jin. First story detection in TDT is hard. In *Proc. of CIKM 2000*.

[2] J. Allan, R. Papka, and V. Lavrenko. Online new event detection and tracking. In *Proc. of SIGIR '98*.

[3] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. In *Proc. of NIPS14, 2002*.

[4] E. Gabrilovich, S. Dumais, and E. Horvitz. Newsjunkie: providing personalized newsfeeds via analysis of information novelty. In *Proc. of WWW '04*, 2004.

[5] S. Godbole, A. Harpale, S. Sarawagi, and S. Chakrabarti. Document classification through interactive supervision of document and term labels. In *Proc. of ECML/PKDD '04*.

[6] T. Hofmann. Probabilistic latent semantic analysis. In *Proc. of UAI'99*.

[7] T. Joachims. Text categorization with support vector machines: learning with many relevant features. In *Proc. of ECML-98*.

[8] R. Klinkenberg and T. Joachims. Detecting concept drift with support vector machines. In *Proc. of ICML-00*.

[9] D. Lewis, Y. Yang, T. Rose, and F. Li. Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 2004.

[10] G. Ramakrishnan. *Bridging chasms in text mining through Word and Entity Associations*. PhD thesis, IIT Bombay, 2005.

[11] G. Ramakrishnan, K. P. Chitrapura, R. Krishnapuram, and P. Bhattacharya. A model for handling approximate, noisy or incomplete labeling in text classification. In *Proc. of ICML 2005*.

[12] Y. Yang, J. Zhang, J. Carbonell, and C. Jin. Topic-conditioned novelty detection. In *Proc of SIGKDD '02*.