Large Scale Book Annotation with Social Tags

Sharon Givon and **Victor Lavrenko**
School of Informatics
Edinburgh University
Edinburgh, UK
{S.Givon@sms.ed.ac.uk} {v.lavrenko@gmail.com}

Abstract

We describe work on large scale automatic annotation of full texts of books with social tags. Our task consisted of assigning tags to the full texts of works of fiction and evaluating them against tags assigned by humans. We compared Boosting and Relevance Models (RM) methods to explore how they differ primarily in terms scalability and also annotation quality. We extended beyond the set of 50 tags used in earlier work to sets ranging up to 10,000 tags. We show how a RM based algorithm scales significantly better than a Boosting based algorithm when dealing with large sets of tags.

1. Introduction

Full texts of books have recently become available on-line in several forms. Access to books can be limited such as when searching the content of books on websites like Amazon.com or Google's Book Search, or unlimited with publicly available corpora such as Project Gutenberg[1]. These texts, along with other types of meta-data like social tags, reviews and ratings can be extremely useful for improving tasks such as search and browse or generating automatic book recommendations. Nevertheless, the availability of meta-data often relies on the website's users and therefore suffers from problems varying from noisiness to data sparseness or no data at all. The last two create the new item "ramp-up" problem (Konstan *et al.*, 1998), when a new item is encountered that does not have sufficient meta-data and thus cannot be easily recommended. This is where book texts can come into play as they do not depend on website users.

Book texts are substantially long, and in order to use them for more sophisticated tasks such as automatic recommendations, they should first be transformed into more efficient representation. A good form of representation is social tags. Tags are relatively easy to access, mostly keyword-like, and they convey indicative information about the book. Givon & Wilson (2008) used a Boosting based classification tool to show that tags can be successfully predicted from the full texts of books. We make use of a larger and improved corpus of books and tags and we attempt to annotate the books using much larger label sets. We investigated several sizes of label sets, from 10 tags to 10,000. Our aim in doing this was twofold. Firstly, we wished to evaluate how the different methods would scale as the size of the data set was increased. Secondly we wished to generate a larger set of meta-data to enable more accurate profiling of users and books and to improve tasks such as the automatic recommendation of books.

2. Related Work

In the music domain, Eck, Bertin-Mahieux, & Lamere (2007) looked at predicting social tags using audio features and supervised learning. They employed user-defined tags from Last.fm[2] to predict music attributes from acoustic features. Their preliminary results showed that a supervised learning approach to auto-tagging has merit. Their results suggest that auto-tagging helps solve the ramp-up problem seen in social-tag-based music recommenders. Givon & Wilson (2008) looked at auto-tagging full works of fiction using an off-the-shelf Boosting based classifier. They used BoosTexter (Schapire & Singer, 2000), a general purpose machine-learning program, and explored several sets of features, based on parts of speech and named entities. They reported that their best multi-class classifier yielded a 0.71 accuracy score. In this paper we extend and improve their methodology. We compare BoosTexter to a RM based classifier as well as extend the label set from its original size of 50 tags used by Givon & Wilson (2008) to both smaller and much larger label sets. In addition we describe a better evaluation method which uses more accurate measures based on recall and precision curves.

3. Data and Pre-processing

Social tags were collected from LibraryThing[3] and book texts mainly from Project Gutenberg. Our final set consisted of 150 books, with an addition of 30 books to the experiments described in Givon & Wilson (2008). The selection was limited to books written in English, and that are associated with the fiction/literature domains. The set was split

[1] http://www.gutenberg.org

[2] http://www.last.fm/

[3] http://www.librarything.com/

into a training and test set, consisting of 100 and 50 books respectively. The test set served as an unseen group of books for final evaluation. A development set of 100 books was used for parameter selection for the RM based algorithm.

3.1 Tagging Data

Using LibraryThing's API we collected all available ISBNs for the same work. This set was used to get the aggregated tagging information for that work. The information for each tag included the tag name, a group of aliases (if they exist) and the tag count (the number of times the tag was assigned to that work). We cleaned and filtered the tags associated with each book to obtain the sets that we ultimately use in our experiments. Initial cleaning consisted mainly of stemming, removing duplications based on synonym sets (obtained from LibraryThing), and removal of non-alphanumeric, irrelevant or general tags. Multi-word tags were eliminated in cases where they could be unified with their subset corresponding single-word tags. For example, multi-word tags such as 'child fantasy', or 'drug addiction' would be omitted where occurrences of each word in isolation appears frequently enough in the tag set of the book.

3.2 Tag Selection Per Book

To generate the global label-sets we picked the top-n most frequent tags across all the books in the corpus, where n is the size of the label set. Each book in the set can be assigned up to n relevant tags. Givon & Wilson (2008) used the top 10 most frequent tags out of the possible 50 tags to represent the book. This may exclude some relevant tags and can also include irrelevant ones. The number of selected tags should vary according to the global tag-set and the selection should be based on the tag probability of occurrence under the assumption of randomness. In order to identify the tags that are likely to be random we looked at their hypergeometric distribution - a discrete probability distribution that describes the number of successes in a sequence of n draws from a finite population without replacement. We tested a number of threshold values and picked 0.05 as the randomness probability upper bound. If a tag yields a value that is greater than 0.05, it is not added to the tag set of the book.

3.3 Book Text Data

Each book was processed using a pipeline of NLP tools (Curran & Clark, 2003a,b) which produced an output that consisted of the tokenised text and information such as sentence splits, part of speech tags, and named entities (we discuss those in detail in Section 4.2)

4. Methodology

We defined our task as tagging books automatically such that the tags assigned are as close as possible to those that were assigned by humans. We approach the problem of automatic tagging as one of supervised text classification and explore the hypothesis that a RM based machine learning method can be used to predict tags based on the frequency, uniqueness and parts of speech of certain word groups in the book, as well as named entities. We chose RM as a method for deriving social tag probabilities since this method proved to perform well given the problem of automatically annotating images with keywords, which is relatively close to our task. We compared the results to those yielded by BoosTexter, a boosting based machine learning meta-algorithm for performing supervised learning (Schapire & Singer, 2000), using the same methodology reported by Givon & Wilson (2008).

4.1 Relevance Models

Relevance models (RM) are specifically designed to capture dependencies in *bags of labels*. The approach was originally developed for alleviating the problem of vocabulary mismatch in Information Retrieval (Lavrenko & Croft, 2001), and has since been applied to problems ranging from automatic tagging of images (Jeon, Lavrenko, & Manmatha, 2003) to recovering missing values in database records (Yi, Allan, & Lavrenko, 2006). We will use the following formulation of relevance models to predict a set of *social tags* based on the full text of a book.

Let T represent a training set of books, for which the social tags have already been assigned. Each book B in the training set is represented as a set of user-assigned social tags, together with a set of words which represent the actual content of the book. Relevance models operate by estimating a joint probability distribution $P(t_1 \ldots t_n, w_1 \ldots w_m)$, which stipulates how likely we are to see a set of social tags $t_1 \ldots t_n$ assigned to a book containing words $w_1 \ldots w_m$. The core assumption behind RM is that the individual words are *exchangeable*, which means that any re-ordering of $w_1 \ldots w_m$ is as probable as the original sequence. Similarly, the tags $t_1 \ldots t_n$ are assumed to be exchangeable. However, we do not permit exchanging of words and tags, so observing "drama" as a user-assigned tag is a different kind of event than observing "drama" in the full text of the book. Under the assumption of exchangeability, the joint probability of observing the words together with the tags can be expressed in the following form: [4]

$$P(t_1...t_n, w_1...w_m) = \sum_{B \in T} P(B) \prod_{i=1}^{n} P_B(t_i) \prod_{j=1}^{m} P_B(w_j) \quad (1)$$

The summation is over the books B in the training set. $P(B)$ is assumed to be uniform over T, and the probabilities assigned to a particular tag t and word w are estimated as follows:

$$\begin{aligned} P_B(t) &= (1-\alpha)1_{t \in B} + \alpha \frac{\#(t,T)}{\#(T)} \\ P_B(w) &= (1-\beta)1_{w \in B} + \beta \frac{\#(w,T)}{\#(T)} \end{aligned} \quad (2)$$

Here $1_{t \in B}$ is an indicator function, it equals 1 if the tag t was assigned to the training book B by some user, and equals 0 otherwise. Similarly, $1_{w \in B}$ indicates whether word w occurred in the full text of the book B. $\#(T)$ is

[4] (Yi, Allan, & Lavrenko, 2006) provides a detailed discussion of the assumptions underlying Relevance Models and a derivation for equation 1

the number of training books, and $\#(t, T)$ represents how many of those books were tagged with t. Finally, α and β represent smoothing parameters which were tuned on a held-out portion of the training set.

Using Relevance Models for Tagging:
Let B' represent a testing book, for which we would like to predict the most likely social tags. Let $w'_1 \ldots w'_m$ represent the full text of B'. The probability that tag t should be assigned to book B' can be estimated as:

$$P(t|B') = \frac{P(t, w'_1 \ldots w'_m)}{P(w'_1 \ldots w'_m)} \qquad (3)$$

Here both numerator and denominator are computed according to equations (1,2). For a rank-based evaluation we sort all tags t in the order of decreasing probability $P(t|B')$. When a fixed-length annotation is desired, we label B' with n tags that have the highest $P(t|B')$.

4.2 The Feature Space

In order to find out which word groups in the book are most indicative and useful for the task of annotating book texts with tags, we experimented with several sets of features. The features were fed to the RM and BoosTexter in the form of a bag-of-words (BOW) but varied in composition in terms of: (1) the parts of speech (PoS) used, (2) the number of tokens and (3) the use of Named Entities. In (1) we used the output of a PoS tagger (Curran & Clark, 2003a) to investigate different sets of PoS and mainly the open class ones that consist of nouns, adjectives and verbs. In (2) we experimented with limiting the number of the selected tokens by their tf-idf score (Salton & Buckley, 1997). Lastly in (3), we tested each feature set consisting of tokens from the text with the addition of Named Entities (NEs) to the BOW.

5. Evaluation

Each algorithm outputs a weighted list of the whole tag set used in that experiment. We evaluated the results using TrecEval[5], a method used in Information Retrieval based scoring. The results are displayed in terms of (a) Mean Average Precision (MAP) and (b) Precision at rank 10. (a) is calculated after each relevant tag is retrieved for all relevant tags. Precision values are averaged together to get a single number for the performance of a query (book) and the values are averaged over all queries. (b) measures precision at a fixed low level (10) of the number of assigned tags. As a baseline, we annotated each book in the test set with the same n most frequent tags across the whole corpus, ordered by frequency.

6. Results

6.1 Scalability

The performance results of RM and BoosTexter are shown in Figure 1. These figures show performance values in seconds for all label set sizes on a logarithmic scale. It is clear to see from the charts that for RM runtime is largely invariant across BOW and label set sizes. In contrast, BoosTexter gets progressively slower as the sizes of BOW and label set increase. Moreover, BoosTexter did not complete the experiments beyond 500 tags[6] and for label sets larger than that, we extrapolated BoosTexter's runtimes using linear regression applied to the runs that completed. An average RM runtime ranges from a minimum of 0.6 seconds on a BOW consisting of 10 tokens for a label set of 10 tags, to 3.9 on a bag of all tokens in the book and 2,000 tags. BoosTexter's shortest runtime is 7 seconds on a BOW consisting of 10 tokens for a label set of 10 tags and, as estimated by linear regression, running a bag of all tokens in the book with a label set of 10,000 tags is close to 3.25 hours.

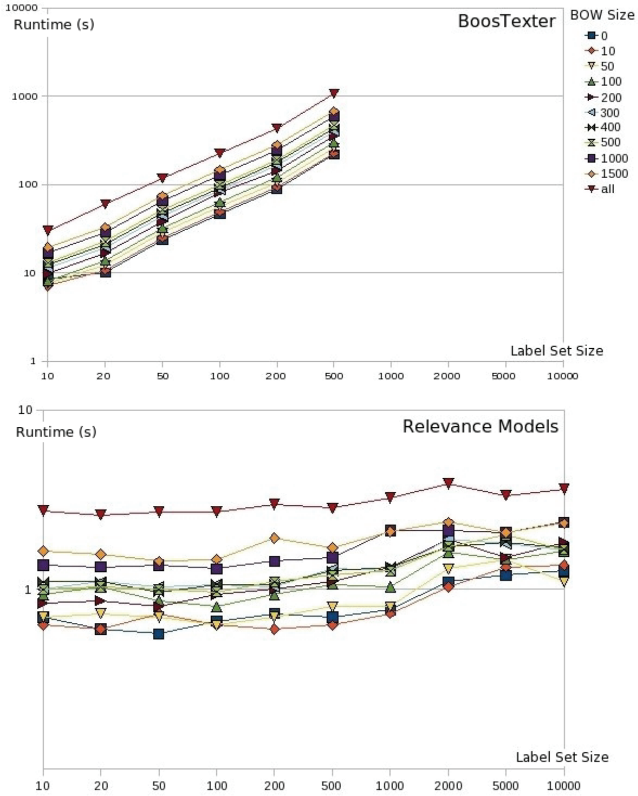

Figure 1: Experiments Runtimes

6.2 Annotation Quality

As shown in table 1, MAP drops as expected as the labelset is increased but the MAP of both tools is significantly higher than the corresponding baseline in all cases[7]. When comparing results on a 50 tags label set to experiments performed by Givon & Wilson (2008) we found that precision at 10 tags for both RM and BoosTexter increased by 2% and 3% respectively. Table 2 displays the results for RM for the larger label sets. As we hypothesised, there are minor differences between the two methods when testing small label sets. They are almost identical on 10 and 20 labels, then the

[5] http://trec.nist.gov/trec_eval/index.html

[6] During the uncompleted experiments BoosTexter hung without producing annotation results.

[7] Tested with the Wilcoxon signed-ranks test.

Table 1: Best Results for BoosTexter (BT) and Relevance Models (RM)

	Set	BoW	PoS	NEs	Ave. Prec.	Prec. at 10
RM	10	1000	ADJs	-	0.921	0.588
BT	10	300	NNs	+	0.920	0.588
BL	10				0.675	0.588
RM	20	200	ADVs	+/-	0.856	0.702
BT	20	300	NN NNS JJ VB VBN VBD	+	0.856	0.714
BL	20				0.641	0.588
RM	50	500	VB VBN VBD	+	0.768	0.769
BT	50	300	VBs	+	0.758	0.750
BL	50				0.553	0.588
RM	100	1000	ADJs	-	0.688	0.750
BT	100	all	NN NNS JJ	+	0.668	0.730
BL	100				0.488	0.588
RM	200	all	NN NNS JJ VB VBN VBD	+	0.601	0.752
BT	200	300	all	+	0.579	0.694
BL	200				0.414	0.588
RM	500	all	NNs VBs ADJs ADVs	-	0.469	0.734
BT	500	400	NNs VBs ADJs ADVs	+	0.478	0.734
BL	500				0.325	0.588

NNs: all nouns, VBs: all verbs; ADJs: all adjectives; ADVs: all adverbs; BL: baseline

gap grows and its maximum occurs on 200 labels with just over 2% difference but no overall significance. RM still beat the baseline on the larger label set sizes, although significance weakens and for 5,000 and 1,000 tags, the difference is not significant. In terms of the selected PoS and the BoW size, there is no clear trend reflected by the results. RM in almost all cases and mainly on the larger label-sets, yields best scores when fed with a large BoW (mostly all words) consisting of a large variety of PoS. BoosTexter, on the other hand, achieves better results when fed with smaller BoW (500 or less). In both cases, there is no clear preference of specific PoS groups or combinations.

7. Conclusions & Future Directions

In this paper we showed results for large scale annotation of book texts. Previous work does not scale to the amount of useful data available on websites such as Amazon.com and does not suit tasks that involve sophisticated profiling of entities in terms of social tags. We performed experiments on larger magnitudes of label sets stretching them from 50 tags to thousands of tags. We introduced Relevance Models, a method adapted from Information Retrieval to match documents to a given query, which we used as an annotation tool and compared its results to a Boosting based algorithm. We performed our experiments on a larger corpus of book texts and introduced an improved process for cleaning tag data and a method for removing tags that are likely to occur randomly from the participating tag sets. Lastly, we introduced a more accurate evaluation method which is based on Mean Average Precision (MAP).

Our results show that RM scales substantially better than BoosTexter. RM runtimes only slightly increase when the model is applied to larger BOW and tag sets. In terms of annotation quality, the results outperformed the ones yielded by BoosTexter as reported by Givon & Wilson (2008) but there were no significant differences in prformance between BoosTexter and RM. From this we can conclude that RM is clearly a more suitable model for the task of large scale annotation of book texts with social tags.

Table 2: Relevance Models on Large Label Sets

	Set	BoW	PoS	NEs	Ave. Prec.	Prec. at 10
RM	1000	all	NN NNS VB VBN VBD JJ RB	+	0.358	0.730
BL	1000				0.269	0.588
RM	2000	all	NNs VBs ADJs ADVs	-	0.272	0.748
BL	2000				0.204	0.588
RM	5000	all	NNs VBs ADJs ADVs	-	0.152	0.710
BL	5000				0.122	0.588
RM	10000	500	NN NNs VB VBN VBD JJ RB	-	0.109	0.710
BL	10000				0.085	0.588

NNs: all nouns, VBs: all verbs; ADJs: all adjectives; ADVs: all adverbs; BL: baseline

References

Curran, J., and Clark, S. 2003a. Investigating gis and smoothing for maximum entropy taggers. In *Proceedings of the 11th Meeting of the European Chapter of the Association for Computational Linguistics*, 91–98.

Curran, J., and Clark, S. 2003b. Language independent ner using a maximum entropy tagger. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*, 164–167. Morristown, NJ, USA: Association for Computational Linguistics.

Eck, D.; Bertin-Mahieux, T.; and Lamere, P. 2007. Autotagging music using supervised machine learning. In *Proceedings of The 8th International Conference on Music Information Retrieval.*

Givon, S., and Wilson, T. 2008. An automatic classification of book texts to user-defined tags. In *Proceedings of the Second International Conference on Weblogs and Social Media(ICWSM 2008).* Seattle, WA, USA: AAAI Press.

Jeon, J.; Lavrenko, V.; and Manmatha, R. 2003. Automatic image annotation and retrieval using cross-media relevance models. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, 119–126. New York, NY, USA: ACM.

Konstan, J. A.; Riedl, J.; Borchers, A.; and Helocker, J. 1998. Recommender systems: A grouplens perspective. In *Recommender Systems: Papers from the 1998 Workshop (AAAI Technical Report WS-98-08)*, 60–64. AAAI Press.

Lavrenko, V., and Croft, W. 2001. Relevance-based language models. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in informaion retrieval*, 120–127. New York, NY, USA: ACM.

Salton, G., and Buckley, C. 1997. Term-weighting approaches in automatic text retrieval. 323–328.

Schapire, R., and Singer, Y. 2000. Boostexter: A boosting-based system for text categorization. *Machine Learning* 39(2/3):135–168.

Yi, X.; Allan, J.; and Lavrenko, V. 2006. Discovering missing values in semi-structured databases. In *Proceedings of RIAO 2007 - 8th Conference - Large-Scale Semantic Access to Content (Text, Image, Video and Sound).* electronic proceedings only.