Effectiveness Results for Popular e-Discovery Algorithms

Eugene Yang, David Grossman, Ophir Frieder
Georgetown University
Washington, DC, USA
{eugene,grossman,ophir}@ir.cs.georgetown.edu

Roman Yurchak
Independent Consultant
rth.yurchak@gmail.com

ABSTRACT

E-Discovery applications rely upon binary text categorization to determine relevance of documents to a particular case. Although many such categorization algorithms exist, at present, vendors often deploy tools that typically include only one text categorization approach. Unlike previous studies that vary many evaluation parameters simultaneously, fail to include common current algorithms, weights, or features, or use small document collections which are no longer meaningful, we systematically evaluate binary text categorization algorithms using modern benchmark e-Discovery queries (topics) on a benchmark e-Discovery data set. We demonstrate the wide variance of performance obtained using the different parameter combinations, motivating this evaluation.

Specifically, we compare five text categorization algorithms, three term weighting techniques and two feature types on a large standard dataset and evaluate the results of this test suite (30 variations) using metrics of greatest interest to the e-Discovery community. Our findings systematically demonstrate that an e-Discovery project is better served by a suite of algorithms rather than a single one, since performance varies greatly depending on the topic, and no approach is uniformly superior across the range of conditions and topics. To that end, we developed an open source project called FreeDiscovery that provides e-Discovery projects with simplified access to a suite of algorithms.

KEYWORDS

e-Discovery, information retrieval, text classification

ACM Reference format:
Eugene Yang, David Grossman, Ophir Frieder and Roman Yurchak. 2017. Effectiveness Results for Popular e-Discovery Algorithms. In *Proceedings of International Conference on Artificial Intelligence and Law, London, UK, June 2017 (ICAIL 2017),* 4 pages.
DOI: xx.xxx/xxxx

1 INTRODUCTION

The ten billion dollar field of e-Discovery is growing at a rate of nearly 10% annually (IDC 2016). Legal cases that previously required a manual discovery phase in which Boolean keyword searches were laboriously reviewed are now replaced with machine learning algorithms that obviate the need for an exhaustive review of numerous documents. Binary categorization algorithms exist for this problem, but at present, no direct comparison of key algorithms implemented on a meaningful e-Discovery dataset is available. Several categorization studies were conducted, but they fail to include several popular algorithms, and unfortunately, were not evaluated on a large dataset that is relevant to e-Discovery. We used a 723,537 document subset of the Enron e-mails that has ground truth developed for the TREC Legal Track. This subset was recently used in a study on improving e-Discovery effectiveness (Cormack and Grossman 2015).

ICAIL 2017, London, UK

DOI: xx.xxx/xxxx

2 E-DISCOVERY STUDY CONFIGURATION

We focus on the binary text categorization problem as it is crucial to e-Discovery. We note that in addition to simply categorizing documents, *ranking* is also crucial to e-Discovery applications. However, that evaluation is identified but left for further investigation. An overview of ranking algorithms may be found in (Grossman and Frieder 2004; Manning et al. 2008).

Specifically, we test the following algorithm:

(1) Identify seed documents (usually from Boolean searches). Identify some relevant and non-relevant exemplars that can be used as input to a Binary Categorization algorithm. The seed documents are used for training.
(2) Use these exemplars for an initial categorization run of the collection and rank the collection in relevance to the seed documents.
(3) Review the top X documents from the categorization run in Step 2; add the results of this review to the set of exemplars found in Step 1.
(4) Categorize the collection using the current set of exemplars.
(5) Repeat Step 3 and 4 until a sufficient number of relevant documents are identified. Note that the right stopping condition is still an open question, but outside the scope of this short paper.

This basic approach of training a classifier based on relevance sampling was introduced in (Lewis and Gale 1994). This is based on relevance feedback which was first described in (Rocchio 1971).

3 ALGORITHMS

The algorithms selected include some of those described in a survey by (Aggarwal and Zhai 2012). Our choice of algorithms is driven by those most commonly used in text categorization and those widely used in the e-Discovery community. Logistic Regression is used by

a large e-Discovery vendor (Losey et al. 2015). SVM's are frequently cited as a source of document categorization effectiveness and have been documented as performing well on the e-mail spam detection problem (Drucker et al. 1999). Recently, Deep Neural Networks have been touted as having superior effectiveness to other algorithms (Lai et al. 2015). Finally, Gradient Boosting (Friedman 2001) was used by all of the top ten teams in the KDD Cup (Cao et al. 2015). Hence, our selection of algorithms is based on recent publications as well as known usage in the e-Discovery world. Additionally, the algorithms are all available in the open source community, so results from this study are easily repeatable.

These algorithms fall into several classic categories of machine learning algorithms. SVM (Support Vector Machine) identifies a separating hyperplane between two classes in a multidimensional space (Cortes and Vapnik 1995). Logistic Regression (LR) fits a sigmoidal conditional probability model by maximizing a penalized posterior likelihood function (Ng and Jordan 2001). Multi-layer perceptron (MLP) classifiers have recently been described in (Joulin et al. 2016) and exhibit good results. Other recent work has also shown promise with recurrent neural networks (RCNN) (Lai et al. 2015) or LSTM-GRNN (Tang et al. 2015). Gradient Boosting algorithms produce a weighted combination of a large number of models, in our case small decision trees (Friedman 2001).

Our implementation of Logistic Regression, Linear SVM, MLP, and LSI in the next section come from the scikit-learn library (Pedregosa et al. 2011), and the implementation of gradient boosting trees from the XGBoost library (Chen and Guestrin 2016). We added a layer of abstraction with a web services tier called FreeDiscovery[1] to make it easier to reproduce our results.

3.1 Text Representation: LSI

LSI (Latent Semantic Indexing) is the core technology used by Content Analyst (recently acquired by kCura), a market leader in the e-Discovery industry. LSI is essentially a text representation, not a supervised learning algorithm. Any ranking algorithm can be used with LSI for binary categorization (e.g.; just assume documents higher than a particular cutoff are relevant). Note that the choice of the cutoff is crucial to the effectiveness of categorization with LSI.

In the LSI implementation tested, categorization is determined by the cosine similarity between the query document and the nearest relevant exemplar in the LSI space. It is implemented over the term-document matrix. The reduced matrix is then queried to provide a ranked result set. For our tests we used $k = 300$ dimensions.

3.2 Feature Types: BoW or N-grams

The choice of features with e-Discovery is complex. Should you take features only from the title of the e-mail or also the content? How many distinct terms should be used? Should phrases be used? The choice of features can dramatically impact effectiveness. We focus on the most practical for e-Discovery.

We tested two types of text representation. The first, Bag of Words (BoW), simply identifies space-delimited tokens in the text. The second identifies overlapping four character n-grams. Character n-grams attempt to capture some of the essence of phrase processing as words are spanned. For example, character n-grams of size four for *New York* are *New_*, *ew_Y*, *w_Yo*, *_Yor*, *York*. Character n-grams of other sizes could have been tested, but recent publications suggest n-grams of size four are often useful for the binary categorization problem (Cormack and Grossman 2014). Hashing was used during feature extraction to limit dictionary size (Weinberger et al. 2009). We use a feature hash table of size 100,001 for our features. Different hash sizes were tested in (Ganchev and Dredze 2008) and over 100,000 is shown to be a reasonable feature size.

3.3 Weights: Binary or Log-TF or Log-TF-IDF

We only use: (1) binary weights shown as best weight for spam detection (Drucker et al. 1999) and (2) the $1 + log(TF)$ sublinear weight referenced in (Dumais 1990) as performing well. Term frequency (TF) refers to the term frequency of the term (or in our case the term or overlapping n-gram) in a document. Our discovery suite likewise includes the product of the sublinear TF and the IDF (inverse document frequency) but this is a collection level statistic that is often not easy to update in real-world applications. However, for completeness, we test this particular weight. Normalization for the length of the document is not included.

3.4 Document Collection

We used the 723,537 subset of the EDRM collection used in TREC 2011 (Grossman et al. 2011) [2].

3.5 Topics

The TREC Legal track designed several different queries or topics for which relevance assessments were obtained. It is noted that (Vinjumur et al. 2014; Webber 2011) demonstrated a variance in the quality of assessments derived for the TREC topics used. To address this concern, our evaluation strictly used the same four TREC topics as (Cormack and Grossman 2014) (topic numbers 201, 202, 203, and 207). Mitigating evaluation bias in topic (query) selection.

3.6 Experimental Design

The categorization process works by identifying relevant and non-relevant documents. In the previous published work on the EDRM dataset (Cormack and Grossman 2015), a seed set of 1,000 documents is identified, and the relevant and non-relevant documents are determined based on known ground truth. We used the same training documents as used in (Cormack and Grossman 2015). More recently, the process was initiated with as little as one document but added documents at around 200 documents per training round (Cormack and Grossman 2015). Hence, our results may be thought of as having run five rounds of training with a 200 document batch size.

We only present the evaluation of the combination of five algorithms (LSI, SVM, Logistic Regression, Gradient Boosting, and MLP), two feature types (bag of words and character 4-grams), three weights (binary, log-tf, and log-tf-idf) and two different text representation (with and without LSI). In addition, we tested many other

[1]The authors would like to acknowledge funding provided by ONE Discovery **http://www.onediscovery.com/** to support some of the development of the open source Free Discovery platform.

[2]`http://cormack.uwaterloo.ca/tar-toolkit/`

combinations of features and weights that due to space constraints, we do not include. We limit the discussion to only these algorithms, features and weights as they are the ones that exhibited the largest differences. They are also commonly found in the literature and practical for real-world e-Discovery applications.

4 RESULTS

We initially present results for a bag of words model study using the same four TREC topics as used by (Cormack and Grossman 2014) (topic numbers 201, 202, 203, and 207) for Logistic Regression, SVM, XGBoost, Multi-layered Perceptron, and Nearest Neighbor are given in Table 1. We methodically measured effectiveness for each algorithm while holding the feature weight and the text representation constant. The results include a separate row for each of the feature weights. Changes in the feature weight significantly impact effectiveness. For example, for Logisitic Regression, for Topic 201, binary weights result in a score of 71 while sublinear-IDF weights yield a score of 92. The boldface numerals indicate the best result for a given algorithm. Results using n-grams are omitted due to space constraints; results using n-grams for every configuration are approximately 20 percent lower than using the bag of words model.

The impact of LSI representation, shown in Table 2, depends on the type of categorization algorithm used. For linear models, such as Logistic Regression and SVM, as well as for XGBoost that tries to capture the separation of the data, LSI transformation generally yields a worse result. For models that are good at capturing local structures such as Multi-layered Perceptron and 1-Nearest Neighbor, LSI potentially reduces the search space and noise, generating a stabler model. In terms of efficiency, linear models without LSI transformation require less training time but still provide excellent outcomes.

Table 1: Recall (%) at 20% of Documents Reviewed
Feature Type: Bag of Words (BoW)

Algorithm	F. Weighting	201	202	203	207
Logistic Regression	Binary	71	86	73	81
	TF	68	89	59	65
	TF-IDF	**92**	**96**	**90**	**90**
Linear SVM	Binary	76	89	82	78
	TF	80	94	92	83
	TF-IDF	**95**	**97**	**98**	**92**
XGBoost	Binary	**95**	94	78	85
	TF	91	96	82	87
	TF-IDF	93	**96**	**87**	**85**
MLP Classifier	Binary	61	80	50	82
	TF	**87**	**92**	**85**	85
	TF-IDF	74	87	65	**86**
NN-1	Binary	74	73	33	55
	TF	55	74	53	57
	TF-IDF	**89**	**92**	**92**	**84**

4.1 Metrics

The effectiveness of the categorization was measured with the following metrics:

- The Legal Track considered the recall (e.g.; the percentage of relevant documents found) at a 20% cutoff. This provides insight into effectiveness of categorization and ranking.
- AUC: The area under the Receiver Operating Characteristic (ROC) Curve, which represents recall as a function of the false positive rate.
- Mean Average Precision (MAP): the area under the precision-recall curve.
- F1 score is the harmonic mean of the precision and recall scores. We note that computing a classic F-measure requires that a cutoff be applied.

The first three scores are classical ranking metrics, while the F-measures are often used for categorization tasks but lack insight into the ranking of the output.

Table 3 provides results for each algorithm using a variety of metrics: ROC-AUC, Average Precision (AP), F1 and our previously discussed Recall@20% retrieved. Results are shown with a range that indicates the lowest and highest value across the four topics. F1 is computed based on the class identified for supervised learning algorithms.

- As suggested by (Dumais 1990) logarithmic scaling with TF-IDF weights showed good effectiveness. For this dataset, it also was the best term weight for Logistic Regression and Linear SVM.

Table 2: Impact of LSI
Feature Type: Bag of Words (BoW) with TF-IDF weighting

Algorithm	LSI	201	202	203	207
Logistic Regression	True	83	96	**96**	82
	False	**92**	**96**	90	**90**
Linear SVM	True	80	94	92	80
	False	**95**	**97**	**98**	**92**
XGBoost	True	68	91	86	78
	False	**93**	**96**	**87**	**85**
MLP Classifier	True	58	**90**	**73**	75
	False	**74**	87	65	**86**
NN-1	True	87	**96**	**96**	**93**
	False	**89**	92	92	84

Table 3: Summary of Categorization Results (Bag Of Words, sublinear TF-IDF).

Algorithm	AUC	MAP	Recall @20%
Logistic Regression	63-83	50-63	89-96
Linear SVM	59-87	57-84	92-97
XG Boost	59-83	55-82	84-96
MLP Classifier	59-85	39-82	65-87
LSI + NN-1	69-89	23-56	42-81

- The XGBoost classifier appears to be relatively insensitive to the feature weights.
- The MLP classifier is based on a neural-network architecture without dropout so it may be subject to over-fitting (Srivastava et al. 2014).

Overall, BoW outperformed n-grams; for efficiency, this is advantageous as n-grams reduce the sparsity of term-document matrix and are more computationally expensive. We note that we have not tested more complex phrase processing approaches. Also, TF-IDF slightly outperforms TF, but not by enough to justify it for many large-scale applications. The presence of IDF requires updating term weights whenever the document collection is modified. Nevertheless, simpler linear models such as Logistic Regression and Linear SVM were more effective than more complex and possibly sub-optimally tuned XGboost and MLPClassifier models. This could be due to over-fitting, particularly considering the relatively small training set size (0.14% of the full dataset). We note that in this case the feature engineering (weighting, normalization) has a comparable or larger impact on the effectiveness than the choice of the algorithm.

5 SUMMARY AND DIRECTIONS FOR FUTURE WORK

This is the first published study germane to e-Discovery in which feature selection and term weights have been held constant for the widely used classification algorithms. This has obvious impact for the e-Discovery community, and we hope that it will encourage researchers who develop new algorithms to carefully test the impact of term weights and feature selection on any proposed new algorithm.

Clearly, more datasets and topics should be tested in the future. In addition to improving the validity of our evaluation, further testing on additional datasets and topics is warranted as TREC evaluations are known to be of limited value after the actual TREC forum. Such a caveat is noted as TREC scoring is based on the pooling of systems under evaluation, and not all algorithms that we evaluated were equally represented by TREC systems in the generation of results used as ground truth for our study.

Also, more analysis is needed to understand what makes one of the algorithms significantly outperform another. Our focus here is to obtain some preliminary results and to highlight the need for methodical analysis. This may appear obvious, but many papers simply compare a result to the most recent published result and obfuscate details surrounding feature engineering (term weights and feature types). Fairly small and seemingly trivial changes in term weighting clearly impact effectiveness of supervised binary categorization algorithms. The framework we used to test these algorithms within an environment of carefully controlled term weights and term features is called FreeDiscovery. We believe this open source framework, built on top of the well-tested open source scikit-learn library (Pedregosa et al. 2011), may be useful to methodically measure the effectiveness of categorization algorithms. An initial version of this framework is freely available so that all of the results in this paper may be reproduced[3].

[3]The software is available at: https://github.com/FreeDiscovery.

Finally, it is well understood that e-Discovery is far more than the initial categorization round. This paper serves as a starting point and clearly more work needs to be done to learn about the impact of different algorithms, feature weights, and feature types in the presence of multiple rounds of training using iterative methodologies such as CAL (Continuous Active Learning) (Cormack and Grossman 2015).

REFERENCES

C. Aggarwal and C. Zhai. 2012. *Mining Text Data.* Springer.

L. Cao, C. Zhang, T. Joachims, G. Webb, D. D. Margineantu, and G. Williams. 2015. Proceedings of the 21th ACM SIGKDD. *International Conference on Knowledge Discovery and Data Mining* (2015).

T. Chen and C. Guestrin. 2016. XGBoost : Reliable Large-scale Tree Boosting System. *arXiv* (2016), 1–6. DOI : http://dx.doi.org/10.1145/2939672.2939785 arXiv:1603.02754

G. F. Cormack and M. F. Grossman. 2014. Evaluation of machine-learning protocols for technology-assisted review in electronic discovery. *Proceedings of the SIGIR Conference 2014* (2014), 153–162. DOI : http://dx.doi.org/10.1145/2600428.2609601.

G. F. Cormack and M. F. Grossman. 2015. Autonomy and reliability of continuous active learning for technology-assisted review. *arXiv* (2015), 19.

C. Cortes and V. Vapnik. 1995. Support-Vector Networks. *Machine Learning* 20, 3 (1995), 273–297. DOI : http://dx.doi.org/10.1023/A:1022627411411 arXiv:arXiv:1011.1669v3

H. Drucker, D. Wu, and V. Vapnik. 1999. Support vector machines for spam categorization. *IEEE Transactions on Neural Networks* 10 (1999), 1048–1054.

S. T. Dumais. 1990. Latent semantic analysis. *JASIS* 3, 11 (1990), 4356. DOI : http://dx.doi.org/10.4249/scholarpedia.4356

J. H. Friedman. 2001. Greedy Function Approximation: A Gradient Boosting Machine. *Annals of Statistics* 29, 5 (2001), 1189–1232. DOI : http://dx.doi.org/10.1017/CBO9781107415324.004 arXiv:arXiv:1011.1669v3

K. Ganchev and M. Dredze. 2008. Small statistical models by random feature mixing. In *Proceedings of the ACL08 HLT Workshop on Mobile Language Processing.*

D. A. Grossman and O. Frieder. 2004. *Information retrieval: Algorithms and heuristics.* Vol. 1. Springer Science & Business Media.

M. Grossman, G. Cormack, B. Hedin, and D. Oard. 2011. Overview of the TREC 2011 Legal Track. In *Proceedings of the Text Retrieval and Evaluation Conference.*

IDC. 2016. *Worldwide eDiscovery Services Forecast, 2015-2019.* Technical Report.

A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov. 2016. Bag of Tricks for Efficient Text Classification. *arXiv preprint* (July 2016). arXiv:1607.01759 http://arxiv.org/abs/1607.01759

S. Lai, L. Xu, K. Liu, and J. Zhao. 2015. Recurrent Convolutional Neural Networks for Text Classification. *Twenty-Ninth AAAI Conference on Artificial Intelligence* (2015), 2267–2273.

D. D. Lewis and W. A. Gale. 1994. A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval.* Springer-Verlag New York, Inc., 3–12.

R. C. Losey, J. L. P. C, J. Sullivan, and T. Reichenberger. 2015. e-Discovery Team at TREC 2015 Total Recall Track. (2015).

C. D. Manning, P. Raghavan, H. Schütze, and others. 2008. *Introduction to information retrieval.* Vol. 1. Cambridge university press Cambridge.

A. Ng and M. I. Jordan. 2001. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *NIPS* (2001), 841–848. DOI : http://dx.doi.org/10.1007/s11063-008-9088-7 arXiv:http://dx.doi.org/10.1007/s11063-008-9088-7

F. Pedregosa, O. Grisel, R. Weiss, A. Passos, and M. Brucher. 2011. Scikit-learn : Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830. arXiv:arXiv:1201.0490v2

J. J. Rocchio. 1971. Relevance feedback in information retrieval. (1971).

N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. 2014. Dropout : A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research (JMLR)* 15 (2014), 1929–1958. DOI : http://dx.doi.org/10.1214/12-AOS1000 arXiv:1102.4807

D. Tang, B. Qin, and T. Liu. 2015. Document Modeling with Gated Recurrent Neural Network for Sentiment Classification. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* September (2015), 1422–1432. http://aclweb.org/anthology/D15-1167

J. K. Vinjumur, D. Oard, and J. Paik. 2014. Assessing the Reliability and Reusability of an E-Discovery Privilege Test Collection. In *Proceedings of the SIGIR (Special Interest Group on Information Retrieval).*

W. Webber. 2011. Re-examining the Effectiveness of Manual Review. In *Proceedings of the SPIRE Conference.*

K. Weinberger, A. Dasgupta, J. Attenberg, J. Langford, and A. Smola. 2009. Feature Hashing for Large Scale Multitask Learning. *Proceedings of the 26th Annual International Conference on Machine Learning* ICML (2009), (pp. 1113–1120). DOI : http://dx.doi.org/10.1145/1553374.1553516 arXiv:0902.2206