Volume 3, Issue 12, December 2013 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Document-type Taxonomy using Computerized Attribute Engineering

Sharvan Kumar Garg*
Research Scholar, Magadh University
India

R.P. Yadav
Retired Professor & Head, Magadh University
India

Abstract: Document-type classifier processes outcomes from the underlying search engine and separates the desired focused outcomes from less precise ones. These classifiers are commonly built in an ad hoc manner, requiring a high level of expertise combined with a significant development effort. In this paper, we look at reducing the cost required to build web eaters (or in general RRM type applications) by using standard taxonomy algorithms, rather than ad hoc approaches, to solve the doc-type taxonomy problem. Our primary goal was to investigate whether standard readymade text classifiers can be used for classifying documents by their type, while achieving high accuracy. The second goal was to develop a method for document type taxonomy that requires relatively little training effort. Another important goal was that a document type classifier for a new document type can be developed by someone with relatively little expertise. We concluded that the presented attribute engineering techniques as a basis for a document type classifier can be used and integrated in RRM type applications, as well as for a broader variety of different information retrieval tasks that require document type taxonomy.

Keywords- Doc-Type Taxonomy, Recurring-Refining-Metasearch(RRM), Attribute Engineering, Purchasing Manual Explorer(PME), Taxonomy Algorithms

I. INTRODUCTION

Focused web search applications layered lying on general-purpose search engines – have proven to be an effective way to improve precision for focused searches without requiring users to formulate complicated queries. Examples from the literature include homepage, purchasing-manual, and movie-review explorers. Web eaters are typically built around a *document-type classifier* that processes outcomes from the underlying search engine and separates the desired focused outcomes from less precise ones. These classifiers are commonly built in an *ad hoc* manner, requiring a high level of expertise combined with a significant development effort.

In this paper, we look at reducing the cost required to build web eaters (or in general RRM type applications) by using standard taxonomy algorithms, rather than *ad hoc approaches*, to solve the doc-type taxonomy problem. This problem is central to our overall research program and has broader applications as well. It has received some attention in the literature [1], [2]. Previous work in this area has focused on *ad hoc* techniques, which, while effective, cannot easily be developed by experienced web developers or consultants. We therefore need a general doc-type taxonomy algorithm that can achieve good performance (greater than 90% accuracy) given only a small amount of labeled training data.

We have found two aspects of RRM contribute substantially to the challenges of building such a doc-type taxonomy algorithm:

Training prejudice In our metaphorical thinking, a document has two *signals*: A typically stronger signal indicating the topic and a weaker signal indicating the doc-type. To help isolate this weaker signal, it would be nice if the training data had a few examples from a large number of topics.

Input prejudice Referring to Figure 2.1, in RRM, the classifier is fed from the output of a query that was shaped to generated documents of interest. This means that the classifier is not separating the desired class of documents from a random selection of the Web but rather is separating it from *near misses*. This makes it more difficult to achieve our target of over 90% accuracy.

The main contribution of this paper is to evaluate whether standard *readymade* taxonomy algorithms and packages that are available on the Web are well suited for doc-type taxonomy.

II. RELATED WORK

This section focuses on work related to document type and genre taxonomy, research related to text taxonomy and attributes selection, and describes how it relates to our work.

Although we are aware of a number of alternative techniques for attribute selection, we did not have time to explore their impact on performance in the way we did with different classifiers. We suspect the outcome would be largely the same: While the difference between algorithms might be large enough to make this a significant engineering decision, there is no fundamental issue to be explored here.

[3] established the earliest archetype execution of *Feature Model Analyzer*, which is a structure for the computerized investigation of attribute models. They established the logic depictions presently employed in the framework and uncovered some of the most important blueprint and execution particulars.

[4] states that computerized investigation of Feature Models is an continuing research field, which comprises much more procedures that it could be primarily anticipated. Computerizing the investigation means endows to each procedure with a prescribed semantics.

[5] building feature model from an accessible system needs significant endeavor where the main dispute is the selection of a parent for each attribute. They offered a ranking heuristic for recognizing parent contenders and also computerized procedures for recognizing compulsory attributes, attribute groups and implies/eliminates limits.

[6] introduces classification algorithm that needs the subsequent parameters as input: 1) upper limit of clusters wanted; 2) estimated proportion of loose documents wanted; 3) judgment on whether or not loose documents should be intermingled into the closest cluster at the last part of the classification; 4) quantity of overlap wanted.

[7] regarded merchandises requirement, normally structured in tabular data. They projected a semi-computerized procedure, language and means maintained, to mine unpredictability of the family of merchandises. They constructed a computerized system to create a feature model supported on the amalgamation of a set of merchandises description. The amalgamated feature model signifies applicable mixtures of features based on the set of merchandises and comes with a clear hierarchy and appropriate classification of feature clusters.

III. COMPUTERIZED ATTRIBUTE ENGENEERING

When we were studying the literature on document type taxonomy, we noticed that many classifiers were built based on manual regulations, or *unplanned* elucidations. These extremely customized classifiers typically performed well for the particular task for which they were designed. However, these solutions are difficult to develop. Therefore, we turned our attention to standard text-book taxonomy algorithms and asked: Can these algorithms achieve at least 90% taxonomy accuracy? And can they do so with relatively little training data?

Standard taxonomy algorithms can indeed achieve these goals, but not without careful engineering of the attribute space. Further, given our requirement that RRM's be buildable by advanced web developers or consultants, this attribute engineering needs to be done in a computerized fashion. We have developed three such computerized means of attribute engineering:

Attribute selection Traditionally, attribute selection is used to boost accuracy by eliminating noise introduced by inappropriate attributes. In our perspective, attribute selection has the added benefit of handling training prejudice, eliminating topic signals that may confuse the classifier.

Attribute augmentation For certain document types, text attributes alone will not produce the desired accuracy. We therefore need to introduce additional type of attributes that we can derive in an automatic way and which represent good discriminators.

Attribute co-occurrence The co-occurrence of certain attributes within a document is not statistically random.

Phrases are the most valuable of these co-occurrences. For example, in purchasing manuals there are phrases like *purchasing manual* or *product manual* that happen to appear closely together. Therefore another important form of attribute augmentation is finding these high-information co-occurrences.

IV. ALTERING THE DOC-TYPE CLASSIFIER

Although the design goal is to have a system that requires relatively little hand tuning and expertise – our baseline accuracy of over 90% on average should be sufficient for most purposes – it will probably almost always be the case that when working with a new document type there will be manual tuning efforts that could potentially improve the overall taxonomy accuracy even further. We therefore want to provide relatively easy knobs and methods so that this tuning is in line with our goal—such that an advanced web developer or consultant should be able to perform that tuning relatively easy if needed.

V. EXPERIMENTS AND OUTCOMES

This section explains the experiments we performed with our doc-type classifier in detail.

For these experiments we didn't fine-tune our classifier, since we were interested in obtaining a high base-line accuracy, which should require no manual intervention.

1. Building Training Sets

In our experiments we focused on three common and popular document types:

- Purchasing Manuals
- Personal Homepages
- News articles

2. Metrics

In each trial the classifiers would output its taxonomy accuracy. The taxonomy accuracy is defined to be the ratio between all correctly predicted documents and the total number of documents in the test set. We multiply this ratio by 100 to obtain the accuracy as a percentage. More particularly:

$$accuracy = \frac{c}{n} \times 100$$

In this equation c represents the number of correctly predicted documents, and n the total number of documents in the test set. To get reliable data for the average taxonomy accuracy, we would then re-run the trials and average out the accuracy over a full amount of 2,000 experiments.

3. Methodology

For the purpose of our experiments we used a standard *readymade* taxonomy package *Mallet[8]*: A Machine Learning for Language Toolkit, that applied different regular text taxonomy algorithms. From this package we used primarily the *Naive Bayes* classifier, plus one supported on an *utmost entropy* representation [9]. We picked a Naive Bayes classifier since it is popular, easy to implement, and efficient in usage. The latter aspect is especially important if we are considering integrating a taxonomy solution within an *RRM* type application. The next classifier, the maximum entropy, was used to see whether the improvements we would see over the Naive Bayes were also similar for different sort of classifiers.

In our experiments we would always run both classifiers on the preprocessed (detagged) data set. The tool would pick randomly 50% from the labeled data to be the training set. The other remaining 50% would then represent our test set in the first experiment we ran both classifiers on each document type on the pre-processed data to obtain baseline accuracy.

We can see that the maximum entropy classifier performs better on average compared to Naïve Bayes: On average for all document type Naive Bayes performs roughly at 75.30% accuracy, while maximum entropy performs on average of 83.56% (which is somewhat below our desired target of 90%). In the next experiment we then applied attribute selection (FDI) to the set of all attributes in the document collection. While keeping only the top 10% of the attributes we ran both classifiers again and noted down the expected performance improvements.

On average Naive Bayes was now executed at 85.43%, whereas utmost entropy was executing at 90.08%. On both classifiers we could see a considerable enhancement. Utmost entropy was even executing above the 90% range.

We now applied our attribute augmentation technique combined with attribute selection as described in the previous section. Again, we measured the accuracy of both classifiers when only the top 10% of all attributes were used. On average over all document types Naive Bayes performed now at 93.37 % accuracy, and maximum entropy was up at 91.48%. We have attained our objective that both classifiers are now performing at over 90% accuracy on average.

The overall amount of training data needed was moderate (only 100 positive and 100 negative labels), which fulfills our goal of requiring relatively little training effort. However, given the small size of the benchmark the outcomes may not be statistically significant. Since we used many diverse topics in each benchmark set (for purchasing manuals and News articles we had 50 topics, and for homepages even 100), and also looked at three different document types (and observed the same trend), we are optimistic that the outcomes should generalize well to different document types. We are also planning to increase our benchmark size.

From these experiments we conclude that attribute augmentation along with attribute selection on average improved the taxonomy accuracy considerably. In cases where attribute selection merely was not able to boost outcomes in the 90% region, attribute augmentation helped here to achieve our goal. We therefore achieved our accuracy goal of at least 90% over all document types we tested. This compares favorably with the 91% accuracy we achieved using our PME on the same benchmark data for purchasing manuals: We were able to obtain a better taxonomy accuracy as PME while using standard taxonomy algorithms, therefore confirming that indeed our doc-type taxonomy technique meets (or in this case beats) the taxonomy accuracy of an *ad-hoc* solution.

VI. CONCLUSIONS

Our primary goal was to investigate whether standard *readymade* text classifiers can be used for classifying documents by their type, while achieving high accuracy. The conducted experiments with the three different document types confirm that indeed this is feasible. However, achieving the desired high accuracy was only possible with a combination of attribute selection and attribute augmentation techniques that we introduced earlier: We then achieved accuracy above our desired threshold goal of 90% with different text classifiers. It was interesting to see how both techniques complemented each other when looking at different document types. For example, for *purchasing manuals* attribute selection was less important, but attribute augmentation made the difference. While for *News articles* attribute selection alone provide very good outcomes. In the case of *homepages* both attribute selection and attribute augmentation together helped to achieve the desired accuracy. The advantage of this outcome is that we can leverage all the research and work related to text taxonomy and attribute selection, without having to rely on hand-crafted rules and ad-hoc taxonomy implementations when developing document classifiers for new document types.

The second goal was to develop a method for document type taxonomy that requires relatively little training effort. In our experiments we only used a set of 100 positively labeled and 100 negatively labeled documents. That was sufficient to achieve the desired accuracy. Having more training data would probably be helpful and further increase accuracy.

Another important goal was that a document type classifier for a new document type can be developed by someone with relatively little expertise. It can be seen that we would be able to develop tooling that would allow an advanced web developer or consultant to train a classifier by presenting a list of samples. A modified version of a web eater could be used to build such a tool. Once the training phase has been completed the document type classifier would be ready for usage.

We conclude that the presented attribute engineering techniques as a basis for a document type classifier can be used and integrated in *RRM* type applications, as well as for a broader variety of different information retrieval tasks that require document type taxonomy.

REFERENCES

[1] Hu, J., Kashi, R., & Wilfong, G.T. (1999). Document classification using layout analysis. In *DEXA Workshop*, pages 556–560, 1999.

[2] Matsuda, K., & Fukushima, T. (1999). Task-oriented world wide web retrieval by document type classification. In *Proceedings of the eighth international conference on Information and knowledge management*, pages 109–113. ACM Press.

[3] Benavides, D., Segura, S., Trinidad, P., & Cortés, A. R. (2007). FAMA: Tooling a Framework for the Automated Analysis of Feature Models. *VaMoS*, *2007*, 01.

[4] Benavides, D., Cortés, A. R., Trinidad, P., & Segura, S. (2006, October). A Survey on the Automated Analyses of Feature Models. In *JISBD* (pp. 367-376).

[5] She, S., Lotufo, R., Berger, T., Wasowski, A., & Czarnecki, K. (2011, May). Reverse engineering feature models. In *Software Engineering (ICSE), 2011 33rd International Conference on* (pp. 461-470). IEEE.

[6] Dattola, R. T. (2013). A fast algorithm for automatic classification. *Information Technology and Libraries*, *2*(1), 31-48.

[7] Acher, M., Cleve, A., Perrouin, G., Heymans, P., Vanbeneden, C., Collet, P., & Lahire, P. (2012, January). On extracting feature models from product descriptions. In *Proceedings of the Sixth International Workshop on Variability Modeling of Software-Intensive Systems* (pp. 45-54). ACM.

[8] Mallet. http://www.cs.umass.edu/ mccallum/mallet/

[9] Lafferty, J., Nigam, K. & McCallum, A. (1999). Using maximum entropy for text classification. In *IJCAI-99 Workshop on Machine Learning for Information Filtering*, pages 61–67.

