Creating a Classifier for a Focused Web Crawler

Nathan Moeller

December 16, 2015

1 Abstract

With the increasing size of the web, it can be hard to find high quality content with traditional search engines. Vertical search engines have emerged as an easier way to find quality content within a specific domain. Creating vertical search engines is a difficult problem because a web crawler must decide what relevant pages to index. This paper focuses on how to create a classifier to use with a web crawler. The method used in this paper was adopted from *A machine learning approach to web page filtering using content and structure analysis* [5]. My method uses web page content and web page structure to determine if a web page belongs to the patent law domain. A neural network was used to train and classify web pages. My successful results suggest this classifier can be used in combination with a web crawler to find high quality patent law web pages.

2 Introduction

The world wide web currently has around 4.7 billion indexed pages and is growing every day [3]. To find meaningful content most people use search engines such as Google or Bing. These engines work well for finding general content that is accessed a lot, however they often

fail when trying to find specific information within a narrow scope. Vertical search engines have emerged as a solution to this problem. A vertical search engine is an engine that searches for pages within a specific scope or topic. Some examples include Yelp (reviews), Indeed (jobs), or Kayak (travel). These engines are similar to Google or Bing however they only index content relevant to their domain. Determining relevant content is difficult on the web, and is the focus of this paper. The method used in this paper is based on the method described in *A machine learning approach to web page filtering using content and structure analysis* [5], however the domain has been changed from medical to patent law. This was done to see if the results could be successfully replicated. The paper is structured into the following sections. Section 3 reviews *A machine learning approach to web page filtering using content and structure analysis* as well as other related work. Section 4 describes my approach. Section 5 shows my results and section 6 is a discussion of the results and the differences from the original method. Finally I conclude in section 7.

3 Related Work

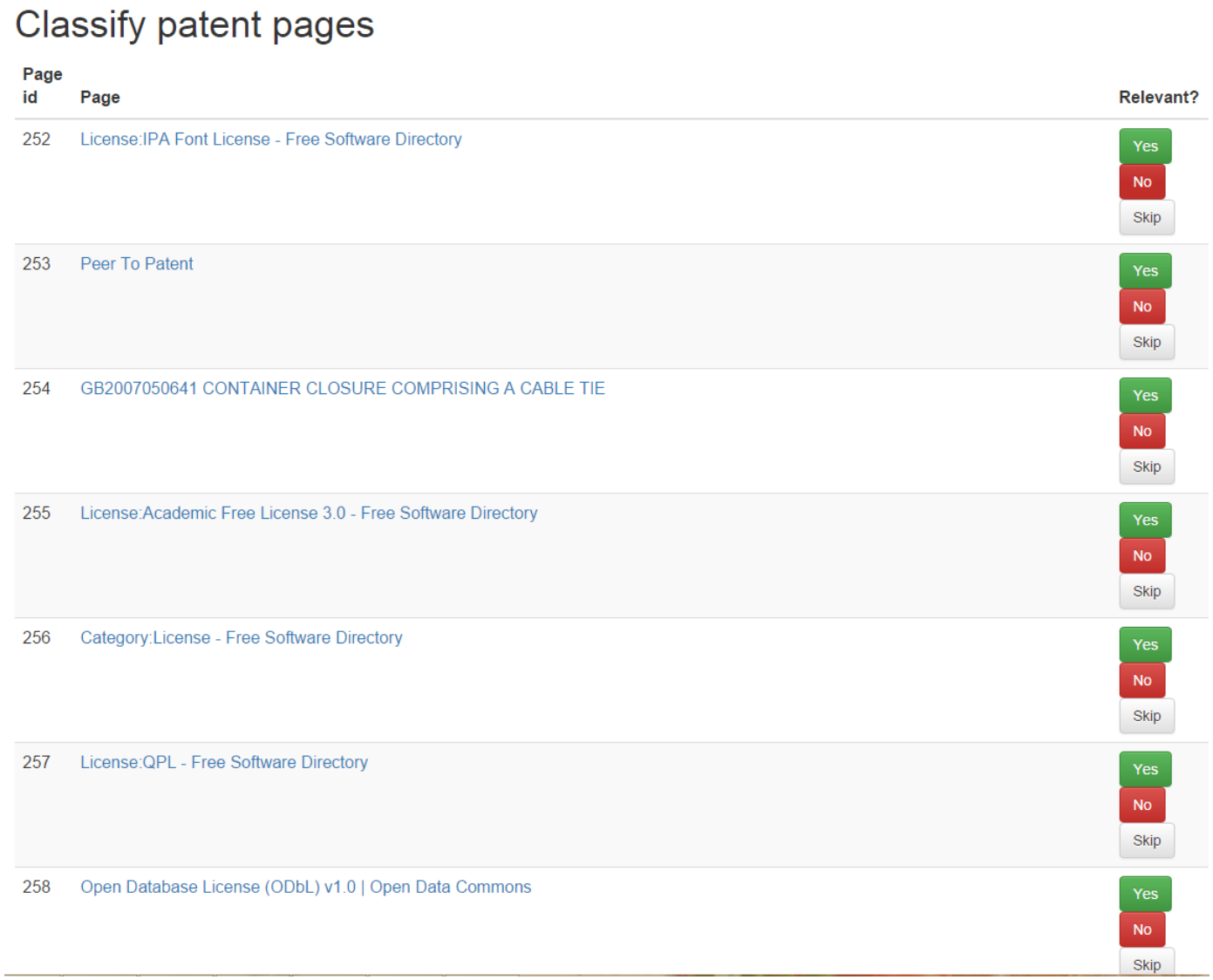
Machine learning is a common method to classify web pages. Because of this, there is a lot of literature focused on the different feature selection and classification methods. In, *A machine learning approach to web page filtering using content and structure analysis* [5] the authors created features from web page content and structure. For page content, they acquired a dictionary of medical terms and counted the number of words in the title and body of each web page. More terms on the page indicated a higher chance of it being a relevant page. For web structure, the authors looked at the term frequencies of neighboring pages. The intuition being relevant web pages are also close to other relevant web pages. The authors also included more features, which I will outline in Section 4 since my approach is based on this paper. They used a neural network and support vector machine for classification.

Nave Bayes vs. Decision Trees vs. Neural Networks in the Classification of Training Web Pages [6] is a similar paper that compared three different classification algorithms. This paper took a different approach by only looking at page content. Instead of term frequency, the authors used a bag of words model which means every word on the page is a feature. A one indicates the presence of the word and a zero indicates the absence of the word. The authors limited the number of features for a given sample to the top 100 words on the page. From there they created a Naive Bayes classifier, decision tree, and neural network and compared them. They found that their Naive Bayes classifier performed the best achieving 95.2 % accuracy, 99.37 % precision, and 95.23 % recall. Their decision tree also performed extremely well. The neural network ended up being too complex to learn because it requires learning a weight for every possible word, which was 5217 words. As words get added, the neural network would have to be extended each time. Because of this limitation, the authors decided not to include it in their analysis.

Lastly, *Focused crawling: a new approach to topic-specific Web resource discovery* [4] described an algorithm for creating a custom personalized web crawler. The focus of this paper wasn't specifically web page classification, but rather efficient web crawling. They developed a system where a user supplied initial documents, and a training set was built by finding similar pages close to the supplied pages. The user then manually classified each web page. Then, the authors also used a bag of words model to create features and a Naive Bayes classifier for the model. Finding relevant pages was only half of the topic for this paper. The other half consisted of creating a "distiller", which is defined as identifying a page that has many links to possibly relevant pages. My approach did not include a distillation step, however the classification techniques used in each paper described above served as motivation to create my own classifier.

4 Approach

As stated before my approach is based on the method described in *A machine learning approach to web page filtering using content and structure analysis* [5], however I changed the topic domain from medical to patent law. This was done to see if results could be successfully replicated. To do this, I first obtained a dictionary of patent law terms. The dictionary was obtained from wikipedia's "Glossary of Patent Law" page [1]. There were 156 terms and each was individually reviewed to make sure it was in fact a patent law term. I then wrote a web crawler using the python library Scrapy [2]. The web crawler began at 5 root pages known to be highly relevant and only followed links that contained a term from the dictionary in the anchor text. I imposed this limitation to find possibly relevant web pages for the training set. Without this restriction, the web crawler finds mostly irrelevant web pages. Around 2000 total urls were collected, along with their referring urls. The referring url where the crawler came from. This is known as the "parent" url. After the web pages were collected, I manually classified 527 web pages as relevant, or irrelevant. I created a web interface to do this efficiently which can be seen in the figure below.



After classifying, I had 527 web pages classified as relevant or irrelevant. I then calculated the features for each web page. The feature selection came from basic web intuition. First, more patent law terms in the title and body of the pages indicates a relevant document. Second, a relevant web page is likely to have neighboring pages that are also relevant. Lastly, a web page linked to by many other web pages is likely to have quality content. This intuition led to 12 features, described below.

1. Words(title) - Number of dictionary terms in the title

2. Words(body) - Number of dictionary terms in the body
3. Page Rank - Page rank of current page
4. Outlinks - Number of links on current page
5. InLinks - Number of links pointing to current page
6. Anchor links - Number of links with a term from dictionary in anchor text on the current page
7. WordsParent(title) - Number of dictionary terms in title of parent page
8. WordsParent(body) - Number of dictionary terms in body of parent page
9. AvgWordsSibling(title) - Average number of dictionary terms in title of sibling pages
10. AvgWordsSibling(body) - Average number of dictionary terms in body of sibling pages
11. AvgWordsChildren(title) - Average number of dictionary terms in title of children pages
12. AvgWordsChildren(body) - Average number of dictionary terms in body of children pages

Every feature except InLinks was calculated for each web page. InLinks was excluded because of the inconsistency of the API. This resulted in 11 features for each web page. Once the data was calculated I began to implement a neural network in MATLAB. A neural network was chosen for the model because it was the best performing model in *A machine learning approach to web page filtering using content and structure analysis* [5]. My neural network had 11 input nodes, 16 hidden nodes, and one output node. Each input node corresponded to 1 feature, and the output node indicated if the sample was relevant to the patent law domain. The results of my neural network are described in the next section.

5 Results

Classifiers are typically evaluated by a few different measures. Accuracy, precision, recall and F-measure. F-measure is a combination between precision and recall, so I have chosen to exclude this measure and present each separately. Each measure is defined as follows.

$$Accuracy = \frac{\text{Documents correctly classified}}{\text{Total documents classified}} \tag{1}$$

$$Precision = \frac{\text{True positives}}{\text{True positives + False positives}} \tag{2}$$

$$Recall = \frac{\text{True positives}}{\text{True positives + False negatives}} \tag{3}$$

Each of these measures is important for a quality classifier. Precision and recall are highly related measures. A low precision value indicates that the focused crawler has indexed many irrelevant web pages. A good classifier maximizes precision, however this may come at a cost to recall. A low recall value indicates that the focused crawler missed relevant information. The goal is to find the right balance between precision and recall so both metrics can be maximized. To calculate these measures a confusion matrix was generated which can be seen in the figure below.

The data was broken in 70% training, 15% validation, and 15% testing. My analysis will focus on the "All Confusion Matrix" since that captures how the classifier did as a whole. My classifier had 131 true positives, 11 false positives, and 12 false negatives. From these numbers, the above measures can be calculated and are shown in Table 1. My neural network achieved above 90% for each measure with accuracy being the highest performing at 95.6%. These results were extremely encouraging to show that patent law web pages can be successfully detected on the web.

Table 1: Neural Network Results

	Accuracy	**Precision**	**Recall**
My Neural Network	95.6%	92.25%	91.6%
Their Neural Network	89.4%	81.38%	76.19%

6 Discussion

The differences in my approach from the original paper can explain my successful results. The biggest difference was the collection of the training set. My web crawler only followed links with a term from the dictionary in the anchor text. I made this decision after inspecting the collected web pages when this rule was not enforced. The web crawler quickly navigated away from the 5 root pages to parts of the web that clearly had nothing to do with patent law. It only logged around 15 relevant patent law pages. To have a good classifier, I knew my training set needed more high quality relevant examples. The original paper did not impose this limitation which may have resulted in a training set with poor positive examples. Another difference was the size of my dictionary. In the original paper, the authors had a dictionary that contained 300,442 medical terms. My dictionary had 156 patent law terms. A large dictionary probably resulted in more false positives because a page may use many terms from the medical dictionary, but not necessarily be classified as a medical web page. This explains the original paper's lower precision. The size of my training set was also smaller than the original paper. I used 527 examples while the original paper used 1000. I do not believe the size effected the results, however the percentage of negative documents may have affected the results. My data set had 72.9% negative examples. I chose to have more negative examples because relevant patent law web pages are rare, and I wanted the

data to reflect that. The original paper did not state the percentage of positive and negative examples, however an evenly distributed training set would explain the lower accuracy and recall.

There were a few differences from the original method that did not affect the results. First, I excluded 3 features: In-Links, Hub Score, and Authority score. All of these metrics are used to assess the quality of a web page. Page rank also assess the quality of a web page, and is more applicable in the context of a web. From this logic these three features were excluded. Also, when calculating the average sibling and average children features I only looked at a subset. The original paper calculated these features for every sibling and child, however to save computation time I only calculated for 10 siblings or children. There is no evidence that this effected the results in any way.

7 Conclusion

My results suggest that a focused web crawler can be created for the patent law domain with my classifier. Combining my classifier while only following links in the dictionary will find high quality relevant web pages. To do this, the neural network classifier will have to be implemented in python or as a web service. Currently, the web crawler and classifier are two separate entities. The web crawler is implemented in Python, and the classifier is implemented in MATLAB. In the future I would like to combine these to create a functional focused web crawler. There are also more features that can be generated to improve the classifier. The structure of the page content can be inspected to find which content is most important. For instance, important information about a web page is often found in h1 or h2 tags. These type of features could be included in the future to improve performance.

References

[1] Glossary of patent law. `https://en.wikipedia.org/wiki/Glossary_of_patent_law_terms/`. Accessed: 2015-11-01.

[2] Scrapy. `http://scrapy.org/`. Accessed: 2015-11-01.

[3] The size of the world wide web. `http://www.worldwidewebsize.com/`. Accessed: 2015-12-12.

[4] Soumen Chakrabarti, Martin van den Berg, and Byron Dom. Focused crawling: a new approach to topic-specific web resource discovery. *Computer Networks*, 31(1116):1623 – 1640, 1999.

[5] Michael Chau and Hsinchun Chen. A machine learning approach to web page filtering using content and structure analysis. *Decision Support Systems*, 44(2):482 – 494, 2008.

[6] Roger G. Stone, Christopher J. Hinde, and Daniela Xhemali. Nave bayes vs. decision trees vs. neural networks in the classification of training web pages. *International Journal of Computer Science Issues*, 4(1):16 – 23, 2009.