Intelligent Fusion of Structural and Citation-Based Evidence for Text Classification

Baoping Zhang, Marcos André Gonçalves, Weiguo Fan, Yuxin Chen, Edward A. Fox
Virginia Tech
Dept. Of Computer Science
Blacksburg, VA, USA
{bzhang, mgoncalv, wfan, yuchen, fox}@vt.edu

Pável Calado, Marco Cristo
Federal University of Minas Gerais
Dept. of Computer Science
Belo Horizonte, MG, Brazil
{pavel, marco}@dcc.ufmg.br

ABSTRACT

This paper investigates how citation-based information and structural content (e.g., title, abstract) can be combined to improve classification of text documents into predefined categories. We evaluate different measures of similarity, five derived from the citation structure of the collection, and three measures derived from the structural content, and determine how they can be fused to improve classification effectiveness. To discover the best fusion framework, we apply Genetic Programming (GP) techniques. Our empirical experiments using documents from the ACM digital library and the ACM classification scheme show that we can discover similarity functions that work better than any evidence in isolation and whose combined performance through a simple majority voting is comparable to that of Support Vector Machine classifiers.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; I.5.3 [**Pattern Recognition**]: Applications—*Text processing*

General Terms

Algorithms, Measurement, Performance, Experimentation

Keywords

Classification, document similarity, citation analysis, Genetic Programming

1. INTRODUCTION

In the last few years, automated classification of text into predefined categories has attracted considerable interest, due to the increasing volume of documents in digital form and the ensuing need to organize them. However, traditional content-based classifiers are known to perform poorly when documents are noisy and contain little text [3, 21].

Particularly, digital library (DL) collections offer a number of opportunities and challenges for classification. The complex internal structure of documents and metadata records in DLs provides additional information that can be used in the classification task. On the other hand, many DLs which are created by aggregation of other sub-collections/catalogs, suffer from problems of quality of information. One such problem is incompleteness (e.g., missing information). This makes it very hard to classify documents using traditional content-based classifiers like SVM, kNN or Naive Bayes. Another quality problem is imprecision. For example, citation-based information is often obtained with OCR, a process which produces a significant number of errors. In this work we try to overcome these problems by applying automatically discovered fusion techniques of the available evidence to the classification problem. Particularly, we investigate an inductive learning method - Genetic Programming (GP) - for the discovery of better fused similarity functions to be used in the classifiers and explore how this combination can be used to improve classification effectiveness.

Experiments were performed on the ACM Digital Library using the ACM classification scheme. Three different content-based similarity measures applied to the abstract and title fields were used in the combination: bag-of-words, Cosine, and Okapi. Five different citation-based similarity measures also were used: bibliographic coupling, co-citation, Amsler, and Companion (authority and hub). The new similarity functions, discovered through GP, where applied to kNN classifiers showed a significant improvement in macro-F1 over the best similarity functions in isolation. Furthermore, the performance of a simple majority voting of the kNN classifiers with the GP functions produced performance comparable to that of content-based SVM classifiers using the same training and test data.

This paper is organized as follows. In Section 2 we introduce background on Genetic Programming and present our new similarity function discovery framework using both the structural content and citation information. Section 3 describes how the GP framework and the discovered sim-

CIKM'04, November 8–13, 2004, Washington D.C., USA.

ilarity functions are applied to the classification problem. We conduct two sets of experiments to evaluate this framework and summarize the experimental findings in Section 4. Section 5 discusses the related works to this study and Section 6 concludes the paper and points out future research directions.

2. BACKGROUND

2.1 Genetic Programming

Genetic Programming (GP), an extension of Genetic Algorithms (GAs), is an inductive learning technique designed following the principles of biological inheritance and evolution [26]. In GP, a population is a set of solution formulas which are compositions of the functions and terminals. Each potential solution is called an individual in a population. An individual in GP systems is typically represented using a tree structure as shown in Figure 1.

GP works by iteratively applying genetic transformations, such as reproduction, crossover, and mutation, to a population of individuals to create more diverse and better performing individuals in subsequent generations.

In order to apply GP to the problem of classification, several required key components of a GP system need to be defined. Table 1 lists these essential components along with their descriptions.

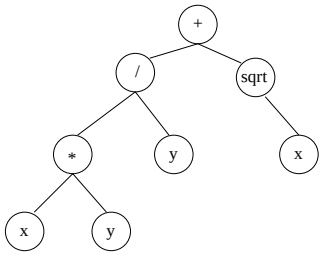

Figure 1: A sample tree representation.

We set up the configurations of the GP system used for similarity function discovery as shown in Table 2.

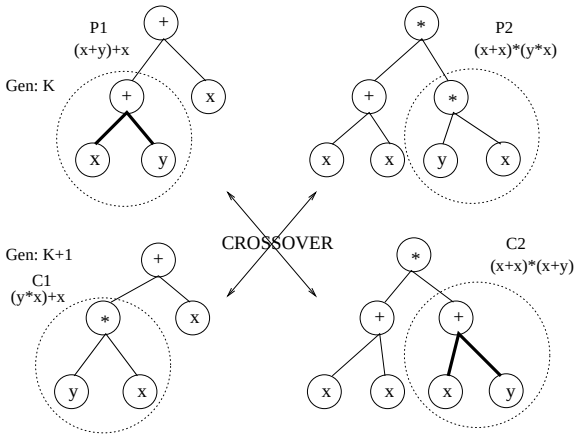

Figure 2: A graphical illustration of the crossover operation.

Algorithm 1 (below) details our fitness evaluation function which GP intends to optimize within a particular class.

Components	Meaning
Terminals	Leaf nodes in the tree structure. i.e. x, y as in Figure 1.
Functions	Non-leaf nodes used to combine the leaf nodes. Commonly numerical operations: +, -, *, /, log.
Fitness Function	The objective function GP aims to optimize.
Reproduction	A genetic operator that copies the individuals with the best fitness values directly into the population of the next generation without going through the crossover operation.
Crossover	A genetic operator that exchanges subtrees from two parents to form two new children. Its aim is to improve the diversity as well as the genetic fitness of the population. This process is shown in Figure 2.
Mutation	A genetic operator that replaces a selected individual's subtree whose root is a picked mutation point with a randomly generated subtree.

Table 1: Essential GP Components.

Terminals	We use features discussed in Section 2.2 as terminals.
Functions	+, *, /, sqrt
Fitness Function	Algorithm 1 (see below)
Genetic Operators	Reproduction, Crossover, Mutation

Table 2: Modeling setup for classification function discovery by GP. Refer to Table 1 for explanations of the various components.

```
Let Lp, Lr be empty lists
For each document D in class C
     Let Lp = Lp union (the set of |C| documents most
     similar to D)
     Let Lr = Lr union (the set of |C| documents most
     similar to D and also not already in Lr)
end for
Let P = (no. of documents in Lp that are of class C)/|Lp|
Let R = (no. of documents in Lr that are of class C)/|C|
F = 2PR/(P+R)
```

A good similarity function, i.e., a similarity function with a high fitness value, is one that, when applied to a document d_i of class C, ranks many documents from class C as similar to d_i. The higher the value of F, the better the function. It is worth to notice that the choice of fitness function can have a huge impact in the final classification performance [9]. Experiments with different fitness functions are currently being performed.

2.2 Used Terminals

We combined features regarding content-based structural information and features regarding citation-based information together to serve as the terminals in our GP system.

2.2.1 Structural similarity measures

To determine the similarity between two documents we used three different similarity measures applied to the content of abstract and title of documents separately: Bag-of-Words, Cosine, and Okapi [35]. This gave us six similarity measures, represented as document × document matrices: AbstractBagOfWords, AbstractCosine, AbstractOkapi, TitleBagOfWords, TitleCosine, and TitleOkapi. More specifically, the documents are represented as vectors in the Vector Space Model [37]. Suppose we have a collection with N distinct index terms t_j. A document d_i can be represented as follows: $d_i = (w_{i1}, w_{i2}, ..., w_{iN})$, where w_{ij} represents the weight assigned to term t_j in document d_i. For the bag-of-words measure, the similarity between two documents d_1 and d_2 can be calculated as the following:

$$bag-of-words(d_1, d_2) = \frac{|\{d_1\} \cap \{d_2\}|}{|d_1|} \quad (1)$$

where $\{d_i\}$ corresponds to the set of terms occuring in document d_i. For the Cosine measure, the similarity between two documents can be calculated as the following [38]:

$$cosine(d_1, d_2) = \frac{\sum_{i=1}^{t} w_{1i} * w_{2i}}{\sqrt{\sum_{i=1}^{t} w_{1i}^2 * \sum_{i=1}^{t} w_{2i}^2}} \quad (2)$$

For the Okapi measure, the similarity between two documents can be calculated as the following: $Okapi(d_1, d_2) =$

$$\sum_{t \in d_1 \cap d_2} \frac{3 + tf_{d_2}}{0.5 + 1.5 * \frac{len_{d2}}{len_{avg}} + tf_{d_2}} * \log \frac{N - df + 0.5}{df + 0.5} * tf_{d_1} \quad (3)$$

Here, tf is the term frequency in a document and df is the document frequency of the term in the whole collection. N is the number of documents in the whole collection, len is the length of a document, and len_{avg} is the average length of all documents in the collection.

From Eqs. (1), (2), and (3), we can see that the cosine similarity matrix is symmetric while the bag-of-words and okapi similarity matrices are not.

2.2.2 Citation-based similarity measures

To determine the similarity of subject between two documents we used five different similarity measures derived from link structure: co-citation, bibliographic coupling, Amsler, and Companion (authority and hub).

Co-citation was first proposed by Small [39], as a similarity measure between scientific papers. Two papers are co-cited if a third paper has citations to both of them. This reflects the assumption that the author of a scientific paper will cite only papers related to his own work. To further refine this idea, let d be a document and let P_d be the set of documents that cite d, called the *parents* of d. The co-citation similarity between two documents d_1 and d_2 is defined as:

$$cocitation(d_1, d_2) = \frac{P_{d_1} \cap P_{d_2}}{|P_{d_1} \cup P_{d_2}|} \quad (4)$$

Eq. (4) tells us that, the more parents d_1 and d_2 have in common, the more related they are. This value is normalized by the total set of parents, so that the co-citation similarity varies between 0 and 1.

Also with the goal of determining the similarity between papers, Kessler [24] introduced the measure of bibliographic coupling. Two documents share one unit of bibliographic coupling if both cite a same paper. The idea is based on the notion that authors who work on the same subject tend to cite the same papers. More formally, let d be a document. We define C_d as the set of documents that d cites, also called the *children* of d. Bibliographic coupling between two documents d_1 and d_2 is defined as:

$$bibcoupling(d_1, d_2) = \frac{C_{d_1} \cap C_{d_2}}{|C_{d_1} \cup C_{d_2}|} \quad (5)$$

Thus, according to Eq. (5), the more children in common document d_1 has with document d_2, the more related they are. This value is normalized by the total set of children, to fit between 0 and 1.

In an attempt to take the most advantage of the information available in citations between papers, Amsler [1] proposed a measure of similarity that combines both co-citation and bibliographic coupling. According to Amsler, two papers A and B are related if (1) A and B are cited by the same paper, (2) A and B cite the same paper, or (3) A cites a third paper C that cites B. Thus, let d be a document, let P_d be the set of parents of d, and let C_d be the set of children of d. The Amsler similarity between two documents d_1 and d_2 is defined as:

$$amsler(d_1, d_2) = \frac{(P_{d_1} \cup C_{d_1}) \cap (P_{d_2} \cup C_{d_2})}{|(P_{d_1} \cup C_{d_1}) \cup (P_{d_2} \cup C_{d_2})|} \quad (6)$$

Eq. (6) tell us that, the more links (either parents or children) d_1 and d_2 have in common, the more they are related.

Finally, taking a different approach, Dean and Henzinger proposed the Companion algorithm [7] for Web pages. Given a Web page d, the algorithm finds a set of pages related to d by examining its links. Companion is able to return a degree of how related the topic of each page in this set is to the topic of page d. This degree can be used as a similarity measure between d and other pages. We use a similar approach where web pages correspond to documents and links to citations.

To find a set of documents related to a document d, the Companion algorithm has two main steps. In step 1, we build the set $\mathcal{V}$, the vicinity of d, that contains the parents of d, the children of the parents of d, the children of d, and the parents of the children of d. This is the set of documents related to d. In step 2 we compute the degree to which the documents in $\mathcal{V}$ are related to d. To do this, we consider the documents in $\mathcal{V}$ and the citations among them as a graph. This graph is then processed by the HITS algorithm [25], which returns a degree of *authority* and *hubness* for each document in $\mathcal{V}$. Intuitively, a good authority is a document with important information on a given subject. A good hub is a document that cites many good authorities. Companion uses the degree of authority as a measure of similarity between d and each document in $\mathcal{V}$. For a more detailed description of the Companion and HITS algorithms, the user is referred to [7] and [25], respectively.

3. THE FRAMEWORK FOR CLASSIFICATION

All of the similarity measures discussed in section 2.2.1 and 2.2.2 were represented as document × document matrices and served as the terminals in the GP system. With

the above settings, the overall classification framework is as follows:

1. For each class, generate an initial population of random "similarity trees"
2. For each class, perform the following sub-steps on training documents for N_{gen} generations
   (a) Calculate the fitness of each similarity tree
   (b) Record the top N_{top} similarity trees
   (c) Create new population by:
      i. Reproduction
      ii. Crossover
      iii. Mutation
3. Apply the "best similarity tree" of each class (i.e., the first tree of the last generation) on a set of testing documents to a kNN algorithm (see below)
4. Combine the output of each classifier through a simple majority voting

Steps 1 and 2 concern the training process within GP which intends to discover better similarity functions for each class. However, the discovered functions can only be used to calculate the similarity between any two documents. In order to evaluate the performance of those functions in the classification task, we used a strategy based on a nearest neighbor classifier. This classifier assigns a category label to a test document, based on the categories attributed to the k most similar documents in the training set. The most widely used such algorithm was introduced by Yang [42] and is referred to, in this work, as *kNN*. The *kNN* algorithm was chosen since it is simple and makes a direct use of similarity information.

In the *kNN* algorithm, to a given test document d is assigned a relevance score $s_{c_i,d}$ associating d to each candidate category c_i. This score is defined as:

$$s_{c_i,d} = \sum_{d' \in \mathcal{N}_k(d)} similarity(d, d') f(c_i, d') \qquad (7)$$

where $\mathcal{N}_k(d)$ are the k nearest neighbors (the most similar documents) of d in the training set and $f(c_i, d')$ is a function that returns 1 if document d' belongs to category c_i and 0 otherwise. In Step 3 of our framework the generic similarity function of kNN is substituted by the functions discovered by GP for each class.

In multi-classification problems with n classes, we effectively end up with n *kNN* classifiers using the described framework. In order to produce a final classification result, we combine the output of all n classifiers using a simple *majority voting* scheme, whereby the class of a document d_i is decided by the most common class assigned by all the n classifiers. In case of ties, we assign d_i to the larger class. Besides its simplicity we chose to use majority voting in our framework (Step 4) to: 1) help alleviate the common problem of overfitting found in GP training [11] and; 2) help boost performance by allowing *kNN* classifiers to apply different similarity functions which explore and optimize the characteristics of each particular class in different ways. A reasonable alternative here would be to generate only one "global" similarity function instead of n "local" ones. However, discovery of such a globally unique similarity function, besides the potential of suffering overfitting, was too demanding in terms of training time and necessary computational resources while the applied "per class" training allowed easy distribution of the training task. Nonetheless, we are working on parallel strategies to allow "global vs. local" experiments.

4. EXPERIMENTS

To test the hypotheses that GP is able to adapt itself to find the best similarity functions we run two sets of experiments following the framework of the previous section. For these experiments, we used only the first level of the ACM classification scheme (11 categories, A to K)(`http://www.acm.org/class/1998/`) and a subset of the ACM collection with 30K metadata records corresponding to those classified under only one category in the first level. The ACM digital library suffers from most of the problems we mentioned before. For example, only 42% of the records have abstracts, which makes it very hard to classify them using traditional content-based classifiers. For these records, the only available textual content is title. But titles contain normally only 5 to 10 words. Citation information was created with OCR and had a significant number of errors. A very imprecise process of matching between the citation text and the documents, using adaptations of techniques described in [17, 27, 28], had to be performed. This introduced noise and incompleteness in the citation-based similarity matrices computed with measures such as co-citation or bibliographic coupling. The other impact factors were the large searching space and skewed distributions of some categories.

Stratified random sampling (cf. section 4.1) was used to create training and test collections. The combination of these experiments should provide us with insights about the capability of the GP-based discovery framework.

4.1 Sampling

The collection used in our experiments has 30,022 documents. Each terminal or feature described is a similarity matrix which contains the similarity between each pair of documents. Using half or more of the whole collection as our training data, the required resources, as CPU time, and amount of memory, would be enormous. The time required to discover a proper classification framework also would be significant. To reduce the high cost of resources and at the same improve efficiency, sampling was used. A sample is a finite part of a statistical population whose properties are studied to gain information about the whole [31]. Sampling is the act, process, or technique of selecting a suitable sample, or a representative part of a population for the purpose of determining parameters or characteristics of the whole population. A random sample is a sample selected based on a known probability that each elementary unit will be chosen. For this reason, it is sometimes referred to as a probability sample. A stratified sample is one type of random sample. A stratified sample is obtained by independently selecting a separate simple random sample from each population stratum. A population can be divided into different groups based on some characteristic. In our case, the documents belonging to each category of the ACM classification scheme (first level) corresponded to different population strata. We can then randomly select from each stratum a given number of units which may be based on a proportion, like 15%, for each category. However special attention needs

to be paid to skewed categories. For example, category E only has 94 documents while the average size of the other categories is in the range of thousands. 15% of 94 only gives us 14 documents and this would be too small to serve as the sample to discover the whole category's characteristic. In this case, we might want to have larger samples. Classification statistics for the whole collection would be used to control the sampling procedure. That is, baselines based on the samples would be compared with baselines for the whole collection to ensure that the samples mirror the whole collection as well as possible. We generate two sets of training samples using stratified sample strategy. The first set used a random 15% sample for large classes and 50% for skewed classes (A, E, and J). And the second set used a random 30% sample for large classes and 50% for skewed classes (A, E, and J)[1]. The rest of the samples will be used for testing and performance comparison. All results reported in later sections are based on test data sets only.

4.2 Baselines

In order to demonstrate that the combination of different features by GP is able to provide better classification results, we need to compare it with the classification statistics of each feature in isolation (baselines). We used F1 as our comparison criteria. F1 is a combination of precision and recall. Precision is defined as the proportion of correctly classified records in the set of all records assigned to the target class. Recall is defined as the proportion of correctly classified records out of all the records having the target class. F1 is defined as 2*Precision*Recall / (Precision + Recall). It is worth to notice that F1 is an even combination of precision and recall. It reduces the risk that you can get perfect precision by always assigning zero categories or a perfect recall by always assigning every category. The result we want is to assign the correct categories and only the correct categories, maximizing precision and recall at the same time, and therefore maximizing F1. Table 3 shows the evidence that performs the best when applied to a kNN algorithm for a specific category in isolation in the test collections, among all similarity evidence, based on macro F1. Table 4 shows the average macro F1 over all categories for each similarity evidence, also in isolation.

Evidence	Macro F1 (15%)	Macro F1(30%)
Abstract_BagOfWords	17.64	19.50
Abstract_Cosine	32.59	34.29
Abstract_Okapi	32.60	33.86
Bib_Coup	31.27	34.73
Amsler	37.44	41.23
Co-citation	21.88	27.31
Comp_Authority	26.53	32.09
Comp_Hub	29.93	33.95
Title_BagOfWords	45.68	49.20
Title_Cosine	50.41	52.53
Title_Okapi	50.06	52.53

Table 4: Macro F1 on individual evidence.

From Table 3 it can be seen that title-based evidence is the most important for the majority of the classes. For those classes whose best performer was a citation-based evidence, Amsler was the best measure. From Table 4, it can be seen that the best types of evidence are the title-based ones, followed by citation-based and abstract-based evidence, respectively. This should be expected since title is the only evidence which appears in all the documents while the information provided by the citation structure is very incomplete and imprecise.

[1]In the remainder of the paper, we use 15% to refer to the first sample set and 30% to refer to the second sample set.

4.3 Experimental Set Up

We run several experiments on the two training samples using different parameters. Particularly, we noticed that a larger population size and different random seeds[2] produce better results. On the other hand, they have a huge effect on the training time. The settings for our GP system are shown in Table 5. In the next section, we only report performance of the best tree in the training sets applied to the test sets.

Population size	400[a], 300
Crossover rate	0.7
Mutation rate	0.25
Reproduction rate	0.05
Generations	30[a], 20
No. of seeds	4 (maximum)

[a]Only for 15% sample

Table 5: GP system experimental settings.

4.4 Experimental Results

We demonstrate the effectiveness of our classification framework in three ways: 1) by comparing its performance against the best baselines per class in isolation; 2) by comparing it against a majority voting of classifiers using those best baseline similarity functions; and 3) by comparing our experimental results with the results achieved through a content-based SVM classifier[3]. While the third comparison may seem inappropriate since the classifiers are trained and applied to different types of content, it does provide a good idea of the core performance of our method, clearly showing it as a valid alternative in classification tasks similar to the ones used in this paper. The SVM classifier has been extensively evaluated for text classification on reference collections, thus offering a strong baseline for comparison. A SVM classifier was first used in text classification by Joachims [22]. It works over a vector space, where the problem is to find a hyperplane with the maximal margin of separation between two classes. This hyperplane can be uniquely constructed by solving a constrained quadratic optimization problem, by means of quadratic programming techniques.

In a comparison class by class between the majority GP (Table 7) and the best evidence (Table 3) in isolation, the majority GP outperforms the best evidence in 10 out of 11 classes in both samples (only the performance for class A is worse).

When comparing the majority GP against the majority using the best evidence (Table 6) it is clear that the former presents better performance: we obtain a gain of 13.38% in the 15% sample and 13.35% in the 30% sample.

[2]Random seed impacts population initialization, which will accordingly affect the final learning results.

[3]For content we used a concatenation of title + abstract.

Class	Macro F1/ class(15%)	Best Evidence (15%)	Macro F1/ class(30%)	Best Evidence (30%)
A	40.00	Title_BagOfWords	43.56	Title_BagOfWords
B	63.89	Amsler	70.58	Amsler
C	60.32	Title_Okapi	63.01	Title_Cosine
D	67.97	Title_Okapi	69.03	Title_Okapi
E	20.69	Title_Cosine	15.38	Title_Cosine
F	45.83	Amsler	53.15	Amsler
G	63.37	Title_Okapi	66.27	Title_Okapi
H	65.58	Title_Okapi	69.27	Title_Okapi
I	58.90	Title_Okapi	61.84	Title_Okapi
J	22.63	Title_Cosine	18.58	Title_Cosine
K	66.42	Title_Cosine	68.38	Title_Cosine

Table 3: Best baseline for each category.

Class	Majority Best Evidence (15%)	Majority Best Evidence (30%)
A	32.61	39.77
B	63.35	68.33
C	62.22	64.27
D	68.28	69.32
E	17.54	12.00
F	41.83	47.36
G	64.90	68.22
H	67.94	71.43
I	59.80	63.05
J	20.10	16.39
K	67.09	69.45
Avg F1	51.42	53.60

Table 6: Majority Voting using the best evidence per class in isolation.

Class	Majority GP (15%)	Majority GP (30%)
A	32.97	41.42
B	75.43	78.19
C	66.96	69.81
D	75.22	76.44
E	26.32	20.25
F	55.76	61.04
G	70.41	74.30
H	74.68	78.15
I	69.03	72.48
J	24.46	24.15
K	70.01	72.67
Avg. Macro F1	58.30	60.81

Table 7: Macro F1 for the combined majority GP with the 15% and 30% samples.

Finally, it can be seen from Table 8 that the performance of SVM is slightly worse than that of the majority GP, which suggests that we have a comparable classification method.

5. RELATED WORK

In the World Wide Web environment, several works have successfully used link information to improve classification performance. Different information about links, such as anchor text describing the links, text from the paragraphs surrounding the links, and terms extracted from linked documents, has been used to classify documents. For example, Furnkranz et al. [16], Glover et al. [18] and Sun et al. [41] show that anchor text and the paragraphs and headlines that surround the links helped improve the classification result. Similarly, Yang et al. [43] claimed that the use of terms from linked documents works better when neighboring documents are all in the same class.

Other researchers applied learning algorithms to handle both the text components of the Web pages and the links between them. For example, Joachims et al. [23] studied the combination of support vector machine kernel functions representing co-citation and content information. Cohn et al. [5] show that a combination of link-based and content-based probabilistic methods improved classification performance. Fisher and Everson [15] extended this work by showing that link information is useful when the document collection has a sufficiently high density in the linkage matrix and the links are of high quality.

Chakrabarti et al. [3] estimate the category of test documents by studying the known classes of neighboring training documents. Oh et al. [32] improved on this work by using a filtering process to further refine the set of linked documents to be used. Calado et. al [2] proposed a Bayesian network model to combine the output of a content-based classifier and the information provided by the document's link structure.

Both GP and GA have been applied to the information retrieval field [6, 9, 10, 12–14, 19, 20, 29, 33, 34]. GP has been applied for data classification [4, 8], but not on text classification.

6. CONCLUSION

In this paper, we considered the problem of classification in the context of a document collections where textual content is scarce and imprecise citation information exists. A framework for tackling this problem based on Genetic Programming has been proposed and tested. Our experimental results on two different sets of documents have demonstrated that the GP framework can be used to discover better similarity functions that, when applied to a *kNN* algorithm, can produce better classifiers than ones using individual ev-

Class	SVM (15%)	SVM (30%)
A	44.65	52.19
B	68.41	72.94
C	65.06	68.49
D	72.13	74.67
E	11.76	4.13
F	51.01	56.03
G	64.65	70.3
H	71.45	74.03
I	64.21	68.88
J	18.38	19.93
K	70.08	73.58
Avg. Macro F1	54.71	57.74

Table 8: Macro F1 for SVM with the 15% and 30% samples.

idence in isolation. Our experiments also showed that the framework achieved results as good as traditional content-based SVM classifiers.

Our future work includes improving scalability through parallel computation. Other sampling strategies like active sampling [30, 36, 40] will also be used to within our classification framework. We also want to test this framework in different document collections to see its viability (e.g., the Web). We also will use the GP framework to combine the current results with content-based classifiers as the SVM classifiers used in the comparisons. Besides that, we want to improve our current evidence, for example, using better methods for citation matching, by trying to fix some OCR errors and using different matching strategies. Finally, new terminals (features) representing additional evidence may be explored. For example, matrices representing relations from other data spaces like author information and patterns of authorship in certain categories can be explored.

7. ACKNOWLEDGEMENTS

This research work was funded in part by the NSF through grants DUE-0136690, DUE-0121679 and IIS-0086227 to Edward A. Fox(Baoping Zhang and Yuxin Chen). Marcos André Gonçalves is supported by CAPES, process 1702-98 and a fellowship by American Online (AOL). Pável Calado is supported by MCT/FCT scholarship grant SFRH/BD/-4662/2001. Marco Cristo is supported by Fucapi, Technology Foundation, Manaus, AM, Brazil.

8. REFERENCES

[1] R. Amsler. Application of citation-based automatic classification. Technical report, The University of Texas at Austin, Linguistics Research Center, Austin, TX, December 1972.

[2] P. Calado, M. Cristo, E. S. de Moura, N. Ziviani, B. A. Ribeiro-Neto, and M. A. Gonçalves. Combining link-based and content-based methods for Web document classification. In *Proceedings of CIKM-03, 12th ACM International Conference on Information and Knowledge Management*, pages 394–401, New Orleans, US, 2003. ACM Press, New York, US.

[3] S. Chakrabarti, B. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 307–318, Seattle, Washington, June 1998.

[4] S. M. Cheang, K. H. Lee, and K. S. Leung. Data classification using genetic parallel programming. In E. Cantú-Paz, J. A. Foster, K. Deb, D. Davis, R. Roy, U.-M. O'Reilly, H.-G. Beyer, R. Standish, G. Kendall, S. Wilson, M. Harman, J. Wegener, D. Dasgupta, M. A. Potter, A. C. Schultz, K. Dowsland, N. Jonoska, and J. Miller, editors, *Genetic and Evolutionary Computation – GECCO-2003*, volume 2724 of *LNCS*, pages 1918–1919, Chicago, 12-16 July 2003. Springer-Verlag.

[5] D. Cohn and T. Hofmann. The missing link - a probabilistic model of document content and hypertext connectivity. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 430–436. MIT Press, 2001.

[6] I. De Falco, A. Della Cioppa, and E. Tarantino. Discovering interesting classification rules with genetic programming. *Applied Soft Computing*, 1(4F):257–269, May 2001.

[7] J. Dean and M. R. Henzinger. Finding related pages in the World Wide Web. *Computer Networks*, 31(11–16):1467–1479, May 1999. Also in Proceedings of the 8th International World Wide Web Conference.

[8] J. Eggermont, J. N. Kok, and W. A. Kosters. Genetic programming for data classification: Refining the search space. In T. Heskes, P. Lucas, L. Vuurpijl, and W. Wiegerinck, editors, *Proceedings of the Fivteenth Belgium/Netherlands Conference on Artificial Intelligence (BNAIC'03)*, pages 123–130, Nijmegen, The Netherlands, 23-24 Oct. 2003.

[9] W. Fan, E. A. Fox, P. Pathak, and H. Wu. The effects of fitness functions on genetic programming-based ranking discovery for web search. *Journal of the American Society for Information Science and Technology*, 55(7):628–636, 2004.

[10] W. Fan, M. D. Gordon, and P. Pathak. Personalization of search engine services for effective retrieval and knowledge management. In *The Proceedings of the International Conference on Information Systems 2000*, pages 20–34, 2000.

[11] W. Fan, M. D. Gordon, and P. Pathak. Discovery of context-specific ranking functions for effective information retrieval using genetic programming. *IEEE Transactions on Knowledge and Data Engineering*, 16(4):523–527, 2004.

[12] W. Fan, M. D. Gordon, and P. Pathak. A generic ranking function discovery framework by genetic programming for information retrieval. *Information Processing and Management*, 2004. In press.

[13] W. Fan, M. D. Gordon, P. Pathak, W. Xi, and E. A. Fox. Ranking function optimization for effective web search by genetic programming: An empirical study. In *Proceedings of 37th Hawaii International Conference on System Sciences*, Hawaii, 2004. IEEE.

[14] W. Fan, M. Luo, L. Wang, W. Xi, and E. A. Fox. Tuning before feedback: combining ranking function

discovery and blind feedback for robust retrieval. In *Proceedings of the 27th Annual International ACM SIGIR Conference*, U.K., 2004. ACM.

[15] M. Fisher and R. Everson. When are links useful? Experiments in text classification. In F. Sebastianini, editor, *Proceedings of the 25th annual European conference on Information Retrieval Research, ECIR 2003*, pages 41–56. Springer-Verlag, Berlin, Heidelberg, DE, 2003.

[16] J. Furnkranz. Exploiting structural information for text classification on the WWW. In *Intelligent Data Analysis*, pages 487–498, 1999.

[17] L. Giles. Citeseer: An automatic citation indexing system. Dec. 16 1998.

[18] E. J. Glover, K. Tsioutsiouliklis, S. Lawrence, D. M. Pennock, and G. W. Flake. Using Web structure for classifying and describing Web pages. In *Proceedings of WWW-02, International Conference on the World Wide Web*, 2002.

[19] M. Gordon. Probabilistic and genetic algorithms for document retrieval. *Communications of the ACM*, 31(10):1208–1218, Oct. 1988.

[20] M. D. Gordon. User-based document clustering by redescribing subject descriptions with a genetic algorithm. *Journal of the American Society for Information Science*, 42(5):311–322, June 1991.

[21] N. Gövert, M. Lalmas, and N. Fuhr. A probabilistic description-oriented approach for categorizing web documents. In *Proceedings of the 8th International Conference on Information and Knowledge Management CIKM 99*, pages 475–482, Kansas City, Missouri, USA, November 1999.

[22] T. Joachims. Text categorization with support vector machines: learning with many relevant features. In *Proceedings of ECML-98, 10th European Conference on Machine Learning*, pages 137–142, Chemnitz, Germany, April 1998.

[23] T. Joachims, N. Cristianini, and J. Shawe-Taylor. Composite kernels for hypertext categorisation. In C. Brodley and A. Danyluk, editors, *Proceedings of ICML-01, 18th International Conference on Machine Learning*, pages 250–257, Williams College, US, 2001. Morgan Kaufmann Publishers, San Francisco, US.

[24] M. M. Kessler. Bibliographic coupling between scientific papers. *American Documentation*, 14(1):10–25, January 1963.

[25] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999.

[26] J. R. Koza. *Genetic programming: On the programming of computers by natural selection*. MIT Press, Cambridge, Mass., 1992.

[27] S. Lawrence, C. L. Giles, and K. Bollacker. "Digital Libraries and Autonomous Citation Indexing". *IEEE Computer*, 32(6):67–71, 1999.

[28] S. Lawrence, C. L. Giles, and K. D. Bollacker. Autonomous citation matching. In O. Etzioni, J. P. Müller, and J. M. Bradshaw, editors, *Proceedings of the Third International Conference on Autonomous Agents (Agents'99)*, pages 392–393, Seattle, WA, USA, 1999. ACM Press.

[29] M. J. Martin-Bautista, M. Vila, and H. L. Larsen. A fuzzy genetic algorithm approach to an adaptive information retrieval agent. *American Society for Information Science*, 50:760–771, 1999.

[30] A. K. McCallum and K. Nigam. Employing EM and pool-based active learning for text classification. In *Proc. 15th International Conf. on Machine Learning*, pages 350–358. Morgan Kaufmann, San Francisco, CA, 1998.

[31] F. C. Misch, editor. *Webster's Ninth New Collegiate Dictionary*. Merriam-Webster Inc., Springfield, Massachusetts, 1988.

[32] H.-J. Oh, S. H. Myaeng, and M.-H. Lee. A practical hypertext catergorization method using links and incrementally available class information. In *Proceedings of the 23rd annual international ACM SIGIR conference on research and development in information retrieval*, pages 264–271. ACM Press, 2000.

[33] P. Pathak, M. Gordon, and W. Fan. Effective information retrieval using genetic algorithms based matching function adaptation. In *Proceedings of the 33rd Hawaii International Conference on System Science (HICSS)*, Hawaii, USA, 2000.

[34] V. V. Raghavan and B. Agarwal. Optimal determination of user-oriented clusters: an application for the reproductive plan. In J. J. Grefenstette, editor, *Proceedings of the 2nd International Conference on Genetic Algorithms and their Applications*, pages 241–246, Cambridge, MA, July 1987. Lawrence Erlbaum Associates.

[35] S. E. Robertson, S. Walker, and M. M. Beaulieu. Okapi at TREC-4. In *NIST Special Publication 500-236: The Fourth Text REtrieval Conference (TREC-4)*, pages 73–96, 1995.

[36] M. Saar-Tsechansky and F. Provost. Active learning for class probability estimation and ranking. In B. Nebel, editor, *Proceedings of the Seventeenth International Conference on Artificial Intelligence (IJCAI-01)*, pages 911–920, San Francisco, CA, Aug. 4–10 2001. Morgan Kaufmann Publishers, Inc.

[37] G. Salton. *Automatic Text Processing*. Addison-Wesley, Boston, Massachusetts, USA, 1989.

[38] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988.

[39] H. G. Small. Co-citation in the scientific literature: A new measure of relationship between two documents. *Journal of the American Society for Information Science*, 24(4):265–269, July 1973.

[40] A. Srinivasan. A study of two sampling methods for analysing large datasets with ILP. *Data Mining and Knowledge Discovery*, 3(1):95–123, 1999.

[41] A. Sun, E.-P. Lim, and W.-K. Ng. Web classification using support vector machine. In *Proceedings of the fourth international workshop on Web information and data management*, pages 96–99. ACM Press, 2002.

[42] Y. Yang. Expert network: effective and efficient learning from human decisions in text categorisation and retrieval. In W. B. Croft and C. J. van Rijsbergen, editors, *Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval*, pages 13–22,

Dublin, IE, 1994. Springer Verlag, Heidelberg, DE.

[43] Y. Yang, S. Slattery, and R. Ghani. A study of approaches to hypertext categorization. *Journal of Intelligent Information Systems*, 18(2-3):219–241, 2002.