

A Genetic Algorithm for Segmentation and Information Retrieval of SEC Regulatory Filings

Joshua Carroll
School of Engineering and Applied Science
University of Pennsylvania
Philadelphia, PA

carrollk@seas.upenn.edu

Thomas Lee
The Wharton School
University of Pennsylvania
Philadelphia, PA
+1-215-898-3266

thomas.lee@wharton.upenn.edu

ABSTRACT

A principal mechanism by which the SEC fulfills its missions of investor protection and market efficiency is the widespread dissemination of the information that publicly traded firms submit for disclosure. The continuing evolution of reporting standards like the International Financial Reporting Standards (IFRS) and the global convergence on XBRL as a syntax for sharing data address the quantitative side of the equation. This work complements the ongoing research on financial disclosure by helping investors learn from the textual, narrative portions of the filing. In structured retrieval, terms are differentially weighted based upon the document segments in which a term appears. Our objective is to automatically segment SEC 10-K financial regulatory filings to facilitate structured retrieval and querying. We leverage the regulatory instructions provided by the SEC to identify a set of semantic labels such as "Legal Proceedings" or "Management's Discussion and Analysis" that segment a 10-K annual report. We frame the problem of document segmentation as a search for semantic labels and use a genetic algorithm to segment each filing. We evaluate the genetic algorithm on a test set of 112 randomly selected regulatory filings and compare those results to a simple, greedy approach for information extraction and segmentation.

Categories and Subject Descriptors

J1 [Administrative Data Processing]: Business, Government
H3 [Information Storage and Retrieval]: Online Information Services – *Web-based services*; Information Search and Retrieval – *search process, retrieval models*

General Terms

Management, Documentation.

Keywords

Genetic algorithms, document segmentation, structured information retrieval, SEC.

DRAFT ONLY. SUBMITTED FOR CONFERENCE PUBLICATION
AND UNDER REVIEW

1. INTRODUCTION

"The mission of the U.S. Securities and Exchange Commission (SEC) is to protect investors, maintain fair, orderly, and efficient markets, and facilitate capital formation [16]." To fulfill this mission, the SEC "requires public companies to disclose meaningful financial and other information to the public [16]." The impending adoption of XBRL (eXtensible Business Reporting Language) as a mandatory format for regulatory filings promises to enable automated comparisons of data between firms and over time [21].

But SEC disclosures comprise two elements: financial facts, and textual narratives written in "plain English." With all of the attention directed towards XBRL adoption, it is easy to overlook the role of the text narrative as a source of investor information. However, the narrative includes critical information not documented in tables and figures such as "forward-looking" statements and factors affecting market risk. The current XBRL data initiatives address only a few, limited textual elements of regulatory filings (see, for example, XBRL and the proposed data elements for Institutional Controls [19, 21]). While the SEC has implemented a full-text search engine [17] to help navigate textual narratives, the engine only includes the four most recent years of filings and makes extensive use of HTML tags within the filing(s). Moreover, HTML tagging is used inconsistently between firms and even by the same firm over time. Many earlier filings do not include HTML at all. Finally, HTML does not support the semantics of structured retrieval and querying because HTML does not reveal whether a search term appears in the General Business description, a discussion of Legal Proceedings, or the Management's Discussion and Analysis (MD&A).

We introduce a novel approach for automatically segmenting regulatory filings to facilitate full-text, structured searching and querying of SEC 10-K reports. Rather than learning from a manually labeled training set of representative filings, we obtain a semantic label for each document segment from the regulatory instructions. Using a genetic algorithm (GA), we frame the text segmentation problem as a search for the sequence that correctly identifies and orders segment labels within each filing. Our objective is to facilitate searching and querying of SEC regulatory filings by automatically segmenting documents in the presence of inconsistent markup or the absence of any markup whatsoever. We evaluate our approach on a test set of 112 randomly selected regulatory filings and compare those results to a simple, greedy approach for information extraction and segmentation.

This paper includes both a theoretical and pragmatic contribution. First, we use GAs to automatically segment filings for structured retrieval. This is the first work that we know of to apply GAs to segmentation in information retrieval. The research differs from prior work in structured retrieval that learns weights associated with segments that are pre-defined. Second, this work enables more robust investor access to information by supporting structured search of SEC regulatory filings in the presence of inconsistent or nonexistent markup. Looking forward, the work complements ongoing SEC supported research in XBRL markup for text narratives [21] by automatically identifying the narrative elements.

In the remainder of this paper, we provide some motivating background, describe our approach, detail some preliminary experiments, contrast our approach to related work, and discuss future directions.

2. MOTIVATION

The objective of this research is to segment and semantically label individual filings to facilitate structured information retrieval (IR) and semistructured querying. Structured IR exploits the intuition that knowing *where* terms appear within a document impacts relevance. It has been shown in the literature that retrieval results can improve when document representations weight terms differentially based upon where they appear in a document [2]. For example, comments about risk factors may have greater significance if they appear within the MD&A. Labeling each segment with the headings specified in the regulatory instructions also enables semantic search. Additionally, in the manner of semistructured querying, users can limit their search to explicit segments such as Property or MD&A rather than searching the entire filing. The current SEC full-text engine supports a hybrid model of structured retrieval and querying. Users cannot specify where in the filing a search term should appear. However, rather than listing documents, the search result is a list of document fragments where fragments are delimited by enclosing HTML elements. There are no fixed semantics to the tags, so some searches are delimited by HTML denoting sub-section headings, section headings, or even simply page separators.

In their regulatory instructions, the SEC actually stipulates specific segments and their corresponding labels that must appear within a 10-K filing [18]. The labels are semantic in nature, describing the contents of the numerical tables or textual narratives contained within. Labels include: Item 1. Business (a narrative description of the general business) or Item 7. MD&A as well as Item 8. Financial Statements and Supplementary Data (numerical tables and figures).

Unfortunately, exploiting these segments and labels for use in structured IR and semistructured querying is not straightforward. Regulatory filings are prepared by independent companies. Despite a common set of instructions, segment labels may vary in subtle or even glaring ways between different submissions. Individual filers might also vary the order in which segments appear.

Figure 1 contains a portion of the regulatory instructions for SEC 10-K filings valid from 1/00 through 2/05 with excerpted submissions of three firms from 2005. Any HTML markup is intentionally omitted from Figure 1 to illustrate some of the ways in which filers deviate from the regulatory instructions:

- Insertions: the word 'Consolidated' in Item 8.

- Deletions: 'and Supplementary Data' in Item 8.
- Substitutions: 'Disclosure' v. "Disclosures" and 'of' v. 'About' in Item 7A.
- Transpositions: 'Qualitative and Quantitative' v. 'Quantitative and Qualitative' in Item 7A.

In addition, segment ordering may vary between filings, so Item 14 in one filing might appear as Item 15 in another. Finally, because of cross-referencing, a single submission might contain a label's text string many times. Thus, the problem of segmentation involves not only discovering labels in the presence of inconsistent naming, but also of selecting the correct instance of a label within the text.

<p>SEC General Instructions for Form 10-K, last updated 12/05</p> <p>Item 7A. Quantitative and Qualitative Disclosures About Market Risk. Furnish the information required by Item 305 of Regulation S-K (§ 229.305 of this chapter)</p> <p>Item 8. Financial Statements and Supplementary Data. Furnish financial statements meeting the requirements of Regulation S-X (§ 210 of this chapter)</p>
<p>2005 Federated Department Stores Inc Acc No. 0000950152-05-002623 Item 8. Consolidated Financial Statements and Supplementary Data.</p>
<p>2005 BERKSHIRE BANCORP INC Acc No. 0000950117-05-001186 ITEM 7A. Quantitative and Qualitative Disclosure About Market Risk.</p>
<p>2005 Cherokee Inc Acc No. 0001104659-05-016467 Item 7A. QUALITATIVE AND QUANTITATIVE DISCLOSURES OF MARKET RISK Item 8. CONSOLIDATED FINANCIAL STATEMENTS</p>

Figure 1. Comparing the text of actual 10-K filings to the SEC General Instructions [18]

While a human reader can easily resolve subtle naming inconsistencies and distinguish cross-references from mis-ordered segments, automated strategies are typically limited by the available training examples. Changes in regulations over time introduce new segments and rename or reorder existing segments further exacerbating the challenge.

Markup provides little value in this context. Figure 2 depicts the same filings listed in Figure 1, but with the markup included. Not all firms submit filings in HTML (2005 Berkshire Bankcorp), but even among those who do, the markup is used inconsistently (2005 Federated v. 2005 Cherokee). Even the same firm may vary its use of markup over time (2004 Federated v. 2005 Federated).

3. APPROACH

Our problem calls for segmenting and labeling each filing into the elements named in the regulatory instructions. We propose to address this problem using a genetic algorithm (GA). In constrained optimization, GAs simulate biological evolution to efficiently search a solution space. Every parameter of the objective function is represented by one *gene*. A *chromosome* is a sequence of genes. A candidate solution is thus modeled as a chromosome comprising one value for each input to the objective

random text strings within the filing and bear no resemblance to the actual labels in the regulatory instructions.

Instead of updating the fitness function, we generate a set of valid candidates. By analogy, a single DNA strand is comprised of nucleotide bases. A DNA nucleotide base must be one of cytosine, guanine, adenine, or thymine. The set of valid candidates is therefore C, G, A, or T. Likewise, for each label, if we generate a set of valid candidates, we can initialize genes by only drawing from offsets that are sure to bear some similarity to our desired segment labels.

Generating a set of valid candidates requires identifying the string patterns within a submission that correspond to the labels within the regulatory instructions. The simplest way of generating valid candidates is to match the raw label text as a string literal. In this case, the *raw* function f returns a set containing the single string pattern equal to the raw text. Unfortunately, because independent filers deviate from the instructions in both simple and in unexpected ways, the simplest pattern is not always reliable.

As a baseline pattern, we use a tokenized version of the label. The *baseline* function f takes as input the raw text of a label and returns a set containing the single pattern of corresponding tokens:

- All string literals are generalized to a mixed-case equivalent i
- All punctuation marks are reduced to an optional token p
- All continuous strings of white-space separators are reduced to a single s

The baseline pattern accounts for the simplest ways in which filers deviate from the regulations: inconsistent use of case, white-spacing, and punctuation within labels. Figure 1 provided examples of the more complex ways in which filings deviate. We summarized those deviations as some combination of insertion, deletion, substitution, and transposition errors. In this paper, we focus on deviations due to a single insertion, deletion, or substitution error. Transposition may be modeled as a sequence of insertions and deletions while this preliminary work focuses only on single errors.

In a singleton insertion error, filers introduce an extra literal into a label. In Figure 1, some filers have introduced the string 'Consolidated' into the label for Item 8. The *insertion* function f processes a baseline pattern and returns a set of patterns where each pattern inserts an unspecified literal S into a whitespace. The set of patterns returned by the *insertion* function is captured by the Mealy machine[9] depicted as Figure 4a.

By contrast, deletion errors occur when filers remove one or more literals from the label. In Figure 1, one filer omitted the literals 'and', 'Supplementary', and 'Data' from Item 8. For singleton deletion errors, the *deletion* function f processes a baseline pattern and returns a set of patterns where each pattern omits one literal i . Recall that for pattern matching purposes, successive white-space tokens ss are treated as a single token s . The set of patterns returned by the *deletion* function is represented by the transducer depicted as Figure 4b.

Substitution errors may be seen as a general case of deletion. In deletion, a single literal i is substituted with the empty string. More generally, we can substitute an unspecified literal S for any single literal i . Figure 1 included the substitution 'of' in place of 'about' in Item 7A. Note also that typos are captured as substitution. Figure 1 also included the substitution of the singleton 'Disclosure' for the plural 'Disclosures' in Item 7A. A

transducer for the patterns represented by the *substitution* function is depicted as Figure 4c.

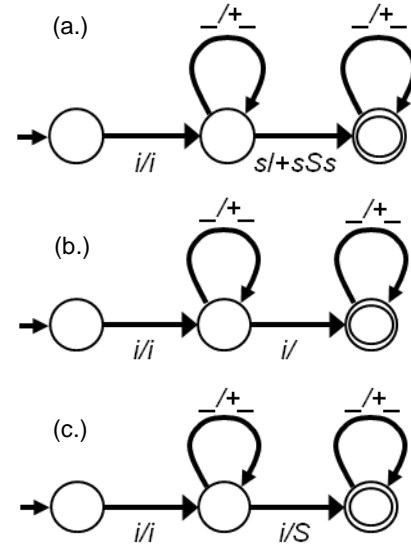


Figure 4. Generating robust patterns for (a) insertion (b) deletion and (c) substitution.

3.3 Specificity

For a given filing and a given label (gene), we can draw an offset from the set of valid candidates at random. The result is a text string that closely matches the regulatory instructions. However, offsets are discovered by matching patterns within the filing text; not all patterns are equal. The *specificity* of an offset is equal to the most specific pattern that matches the corresponding text. Depending upon specificity, the text may be closer to or farther from the original instructions. The pattern that produces offsets with the best (lowest) specificity score is simply the *raw* pattern. Depending upon the wildcards introduced, specificity deteriorates (gets larger). In our GA, offsets matched by the *raw* pattern have a specificity score of 1; the specificity of a candidate gene (offset) increases with the deviations.

3.4 Initialization

To initialize the GA, we create a population of *chromosomes* or candidate solutions. Each chromosome is a sequence of *genes*, a possible offset (x,y) for each label l_i . A value for each gene is drawn from its corresponding set of valid candidates. By introducing specificity into the objective function, we can incorporate a preference for offsets that match with higher specificity. The intuition is that as patterns become more general, they are more likely to yield offsets for irrelevant strings within the text. Therefore offsets with higher specificity are more likely to be correct.

Each label l_i has a set of valid offsets. Let $\max(spec_i)$ represent the highest (worst) specificity score for any offset in the set of valid candidates. If $spec_i$ is the specificity of one offset, then the weight associated with that offset is:

$$w_i = \frac{1}{\max(spec_i) + 1 - spec_i}$$

The specificity weight of a single chromosome is then:

$$w = \prod_{v_i} w_i$$

and we can update the fitness function as:

$$\min_w \sum_{v_i} \left(\frac{|s_i|}{|d|} - \frac{1}{n} \right)^2$$

3.5 Evolution

The process of evolution in the GA involves initializing a parent population with a fixed number of chromosomes. A parameterized threshold creates a parent pool by selecting the top scoring chromosomes from the current generation with respect to the fitness function. The GA then defines mutation and crossover operations to generate off-spring that become parents in the next iteration of the algorithm.

In crossover, a pair of chromosomes is drawn according to some distribution, with replacement, from the parent pool. At a parameterized crossover rate, the pair is either selected for crossover or not. If the pair is selected for crossover, a crossover point (in this case, a label) is drawn uniformly from all possible crossover points. The crossover point splits each parent chromosome into a head and a tail. The head of one parent is paired with the tail of the second parent producing two new parents. In Figure 5a, two parent chromosomes are represented as solid and dashed lines respectively. The shapes indicate the label offsets that define each chromosome. In Figure 5b, the circle label has been selected as the crossover point. Notice that the tail of the second parent has now become the tail of the first parent. At the limit, the crossover point is either the beginning or end-of-file marker and the two parents proceed to the next generation unchanged.

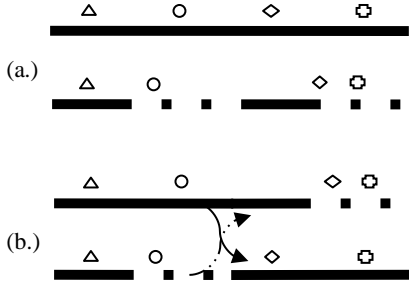


Figure 5. Crossover between two chromosomes

In mutation, a single chromosome is drawn according to some distribution, with replacement, from the parent pool. At a parameterized mutation rate, the chromosome is either selected for mutation or not. If the chromosome is selected for mutation, then a label is selected uniformly from all possible labels. A new offset is drawn from the valid candidate set of the label selected for mutation. In Figure 6a, a single parent chromosome is depicted as a solid line. The unfilled geometric shapes represent the offsets for different labels in the chromosome and the solid shapes represent alternate offsets in the valid candidate set for the circle label. In Figure 6b, the circle has been selected for mutation. In the next generation, the mutated chromosome uses the new label offset.

4. EVALUATION

In a set of ongoing experiments to evaluate the efficacy of applying genetic algorithms to segmentation for structured retrieval and querying, we have focused on the set of SEC 10-K

submissions for 2005. These filings correspond to a common set of regulatory instructions issued in November 2000 and valid through December 2005 [18]. For this period, the regulatory instructions define 23 labels corresponding to 17 segments containing substantive content. The remaining labels define the beginning of the filing, the ending of the filing, and several non-content sections such as a list of external documents incorporated into the filing by reference. We evaluate our approach in two ways: First, we calculate absolute performance by calculating precision and recall statistics for GA-based segmentation. Second, we compare the performance of GA-based segmentation to a previously published "greedy hill-climbing algorithm" for automatic segmentation [12].

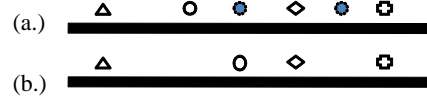


Figure 6. Mutation within one chromosome

To calculate absolute performance, we randomly selected 112 10-K filings from Q1 of 2005 and applied the GA to automatically segment the documents. In our experiments, the set of valid candidates for each gene ranges in specificity from 1 to 4 depending upon whether the offsets matched the raw pattern, a single insertion, a single deletion, or a single substitution. The population size is set to 1000 chromosomes. The threshold for the parent pool is set to 100% so that all chromosomes are eligible to serve as parents in the next generation. Chromosomes in the parent pool are ranked in increasing order according to their fitness with the fittest chromosome ranked 1. If $|C|$ is the size of the parent pool (e.g. the number of unique chromosomes), then a chromosome c with fitness ranking $rank(c)$ is drawn from the parent pool to produce offspring with probability:

$$P(c) = \frac{e^{-rank(c)}}{\sum_{v \in C} e^{-rank(c)}}$$

The crossover rate is set to 1.0 so every pair is eligible for crossover with the crossover point selected uniformly over all labels. Every chromosome selected from the parent pool is then subject to mutation at rate 0.2. The GA runs through repeated generations until the fitness of the best chromosome fails to improve over several generations. We repeated the GA beginning with a different initial population ten times in an attempt to account for local minima.

We measured the precision and recall of the segmentation on the 17 substantive elements of each filing. In our context, segmentation requires correctly identifying the label of both the segment in question and the label of the following (disjoint) segment. Therefore, where relevant, we also report the precision and recall related to non-substantive segments that serve as separators. Results for the final set of 20 elements are reported in Table 1.

Precision measures the number of segments that correctly match the actual text. Recall measures the number of actual segments that we correctly labeled. Note that recall is necessary in this context because, despite the instructions, many filings may omit one or more segments. As a consequence, the denominator for recall is not simply the number of documents in the test set. Moreover, segmentation can fail in at least two ways. We might select the wrong label to name a segment at the *beginning*, or we might incorrectly identify the label text at the *end*.

In earlier work, a greedy hill-climbing algorithm (greedy) for automatic text segmentation was developed and applied to regulatory filings [12]. In the greedy approach, the segment order, defined in the regulatory instructions, are treated as a hard constraint. Beginning with the first label, the greedy approach proceeds through consecutive labels, taking the first offset that conforms to the segment order constraint. Two heuristics were necessary to improve the performance of the greedy approach. First, many filings include a table of contents (toc) that comprises a list of all segments and segment labels. A simple, greedy algorithm would match every gene to the toc. To avoid being confounded by an optional toc, the hill-climbing worked in reverse order, starting with the last label in the document rather than the first. Second, the greedy approach internalized the intuition behind specificity, always attempting to first use offsets that correspond to the best specificity score.

Table 1. Precision and recall for the GA

	Item 1	Item 2	Item 3	Item 4	Part II
P	0.877	0.887	0.896	0.848	0.437
R	0.861	0.870	0.864	0.764	0.421
	Item 5	Item 6	Item 7	Item 7A	Item 8
P	0.923	0.819	0.905	0.869	0.837
R	0.436	0.796	0.796	0.827	0.821
	Item 9	Item 9A	Part III	Item 10	Item 11
P	0.950	0.970	0.877	0.922	0.610
R	0.872	0.942	0.869	0.905	0.615
	Item 12	Item 13	Item 14	Part IV	Item 15
P	0.952	0.924	0.889	0.229	0.813
R	0.925	0.915	0.830	0.218	0.241

To compare the performance of the greedy hill-climber (GH) to the GA, we applied greedy to the same 112 randomly selected filings from 2005 Q1. We compared performance using the F measure, the weighted harmonic mean of precision and recall, with $\beta = 1$ (i.e. Precision and Recall are equally weighted):

$$F = \frac{(1 + \beta^2) * R * P}{\beta^2 * P + R}$$

Results are reported as Table 2.

5. DISCUSSION

Before commenting on the absolute performance, it is worth noting that the GA segments by selecting offsets for each gene from a set of valid candidates rather than by selecting offsets entirely at random throughout the entire document. While selecting from a set of valid candidates may greatly speed convergence, if the correct offsets are not in the set of valid candidates, the GA cannot possibly succeed. The set of valid candidates establishes an upper-bound on precision and recall. To document the limits of this upper-bound, we analyzed the same 112 filings used in the experiment to see whether the candidate set even contained the correct offset. Recall that there were four levels of offset specificity. In Figure 7, we plot the percentage of the time that the correct offset had specificity 1 and the percentage of the time that the correct offset had a specificity level of 4. Offsets with a specificity level of 2 or 3 did not significantly

improve performance over patterns with offset levels of 1 and so are omitted.

Table 2. F-measure comparison between the greedy-hill climber (GH) and the genetic algorithm (GA)

	Item 1	Item 2	Item 3	Item 4	Part II
GH	0.846	0.860	0.856	0.839	0.429
GA	0.885	0.870	0.856	0.849	0.429
	Item 5	Item 6	Item 7	Item 7A	Item 8
GH	0.571	0.800	0.814	0.843	0.792
GA	0.571	0.829	0.834	0.893	0.832
	Item 9	Item 9A	Part III	Item 10	Item 11
GH	0.871	0.894	0.831	0.890	0.560
GA	0.871	0.894	0.870	0.910	0.900
	Item 12	Item 13	Item 14	Part IV	Item 15
GH	0.897	0.906	0.820	0.197	0.286
GA	0.906	0.906	0.870	0.207	0.301

First, the analysis in Figure 7 confirms the anecdotal observation that, despite a common set of regulatory instructions, actual filings deviate from the instructions in both small and large ways. Second, as might be expected, the labels where the set of valid candidates fail (e.g. Item 5, Item 15), are the same labels that are the longest by character and/or word count. Longer counts imply more opportunities for deviation from the instructions.

Establishing the upper-bound helps to explain the precision and recall performance in Table 1. For example, recall is particularly poor for Item 5 and Item 15. In light of Figure 7, the poor recall is expected because the set of valid candidates sets a low upper-bound. Relative to that lower-bound, however, notice that precision is quite high. Thus, when a label is discovered, the GA does comparatively well in selecting it correctly. However, based upon the low recall, there may also be many filings for which there are no valid candidates for the GA to select.

Contrasting the low-recall high-precision performance for the longer labels, Part II and Part IV are poor in both precision and recall. Moreover, Figure 7 suggests that their upper-bound is near 1.0. However, Figure 7 also depicts the sequence order, revealing that Item 5 follows Part II and Item 15 follows Part IV. Because the correctness of a segment depends both upon the correct offset of its own label and the label of the next segment, it is easy to see how poor recall in Item 5 and Item 15 leads to poor recall of Part II and Part IV. Moreover, the high precision of Item 5 and Item 15 indicate that there may be many instances where no offset for Item 5 or Item 15 are found. As a consequence, even if their starting labels are correct, the segments for Part II and Part IV skip over their ending labels resulting in poor precision as well as poor recall.

While Items 5 and 15 have similar document context as per Figure 7, by F-measure (see Table 2), Item 15 does much worse. Although not apparent from the graphs, Item 15 and Part IV are often transposed meaning that filers submit documents where Item 15 appears in Part III rather than in Part IV. This observation highlights a more general distinction between the greedy approach and the GA. Namely, that the GA is more robust in the face of transpositions where segments appear in an order that deviates

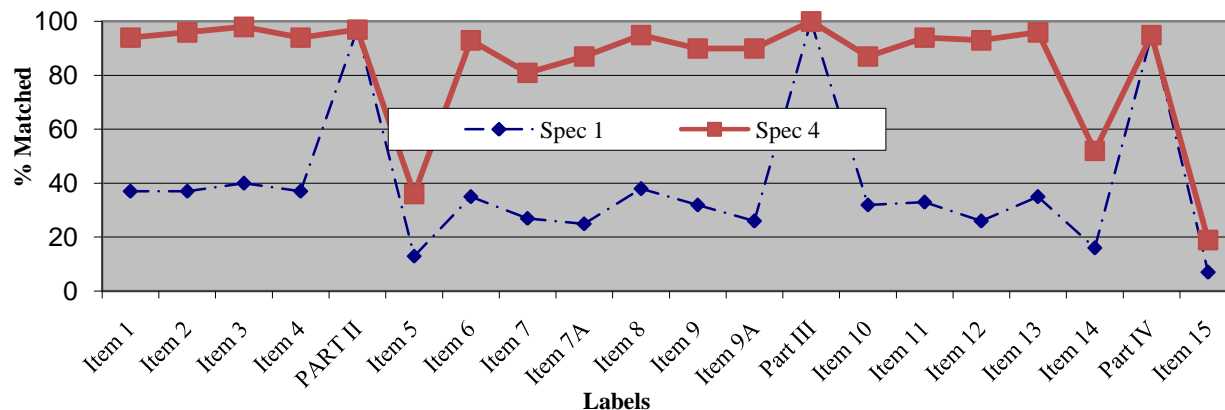


Figure 7. The upper bound on recall for each segment based upon the specificity of the set of valid candidates

from the regulatory instructions. Differences between the GA and GH performance are also due to cross-references. In particular, GH fails on segments that include a cross-reference to the next document fragment whereas the GA, by using deviations from expected segment lengths, is more suited to resolving such discrepancies.

Having commented on the differences between GA and GH, it is also worth noting their similarities. The GA fitness function uses the variance in segment length as a general mechanism for, among other things, discriminating against a toc at the beginning of a filing. However, by processing filings in reverse order, the GH also is able to avoid such local solutions.

In addition, the GH treats the selection of an offset for each label as a greedy hill-climbing search through the space of possible offsets where offsets are effectively ordered first by specificity and second by their sequential place in the filing [12]. The GA draws from a valid set of offsets and weighs the objective function score by specificity. The GA selects for the most specific offsets just as the GH explicitly selects the most specific offsets.

6. RELATED WORK

This work is intended to support both the structured retrieval and querying of narrative segments in regulatory filings. Structured retrieval exploits the intuition that knowing *where* terms appear within documents can affect relevance [2]. In the case of regulatory filings, it is difficult to attribute a search term to a particular document segment if the labels separating fragments cannot be accurately identified. By identifying label patterns for individual filings, we hope to facilitate the structural indexing of content that previously lacked relevant cues. In the same way, users can write explicit queries that restrict their search to explicit segments based upon the semantics of the labels. Queries might be limited to MD&A or to Risk Factors. However, we are not aware of work in structured IR that focuses on automatically learning the segments and segment identifiers. Instead, this work borrows from work in genetic algorithms and work in information extraction/segmentation.

6.1 Genetic Algorithms

Genetic algorithms have traditionally been applied to the study of IR in one of three ways. One body of work focuses on the use of GAs for query reformulation. For example, in query expansion, a GA might be used to discover additional search terms that would

improve relevancy for ranking results or add to the result set [22]. A second application of GAs in IR considers a reformulation of the document representation [8]. Different document features are selected as possible descriptors and individual chromosomes represent novel combinations of descriptors. A GA searches for an optimal document description subject to a particular retrieval (similarity) strategy. Finally, GAs and genetic programming (a functional variant of GAs) have been applied to the design of an optimal function for calculating the similarity between a query and a document representation [15]. For example, Genetic programs have been applied to the discovery of functional combinations of ranking scores [6]. However, to our knowledge this is the first effort to apply GAs to the problem of text segmentation.

While GAs are new to the problem of text segmentation and we are not aware of segmentation strategies for structured IR and querying, automated segmentation for the purpose of information extraction and database integration has a long history. In the context of information extraction, segmentation is the process of finding delimiters that fragment a body of text into labeled fields. Work in segmentation and extraction has focused on identifying author, title, and conference (or journal) in bibliographic records, contact and scheduling information from talk announcements, or contact information and product characteristics from classified ads.

6.2 Segmentation and information extraction

Research in information extraction and segmentation varies along at least two dimensions: the degree of machine-learning supervision and the exploitation of document structure, varying from highly-structured mark-up to the uncertainty of English (language) grammar rules [3, 20].

The majority of the work in IE adopts a supervised, machine learning approach. A hand-coded set of positive and negative examples is generated from a representative set of documents. In either a top-down [7, 20] or bottom-up [5] fashion, a set of disjunctive rules is generated to cover the greatest number of positive labels while excluding negative instances. Refinements draw upon context such as where items appear relative to a specified label or relative to one-another [11, 14].

Instead of focusing on separators, by learning distinguishing text features of the text attributes (e.g. line length, whitespace, percentage of alphanumeric characters), Hidden Markov Models

(HMM) could be trained for identifying transitions into and out of a single field [13] or multiple fields comprising an entire schema [4].

For generating a valid candidates, Our approach is most similar to those who use transducers to model document context [10]. The differences in our work stem from the unique challenges of managing submissions from thousands of independent filers. Rather than a training set of instances, we use policy documents to manually identify the relevant labels and label ordering.

Second, context elements are used in the prior literature to identify explicit regions of text within which a separate extraction process might take place. Instead, we use context, in the form of sequence order and specificity, to resolve the ambiguity that can arise from cross-references or transpositions among segments in the text.

Third, others have used external references to help discover the defining characteristics of their target text [1]. For example, one might use a geographic database containing city, state, and country names to help train a HMM for parsing a list of addresses. We use external an external reference in the form of the regulatory instructions to help learn segment separators (e.g. the labels) as well as segment order.

7. FUTURE WORK

Our future work is focused in three directions. First, we are currently engaged in active evaluation of information retrieval results based upon our automatically generated segments. Second, we are already pursuing several avenues for improving the performance of our GA-based segmentation. Finally, we hope to explore several new avenues of research based upon the ability to automatically identify semantic elements within textual narratives of financial filings, as represented by the segment labels.

Our objective is to automatically segment financial regulatory filings to facilitate structured retrieval and querying. However, our evaluation has focused on the accuracy of that segmentation. We are currently evaluating the effectiveness of retrieval on automatically generated segments by comparing the results from searches over our automatically generated segments to results from the full-text retrieval engine publicly accessible from the SEC. The evaluation is not perfectly straightforward because the SEC does not support structured retrieval in the traditional sense where a search returns a document and relevance is weighted relative to where terms appear within the document. Instead, the SEC returns document fragments. Fragments are defined, not semantically based upon segment labels where search terms appear, but based upon the supplied HTML markup within the text.

Coincident to the work on evaluation, we are extending the GA in two ways. First, we are considering several refinements to the fitness function. We are considering both supervised approaches to generate a sample mean for each segment independently, and a Bayesian approach with a Dirichlet prior to explicitly model the mean for each segment separately. We are also introducing *transformations* as an explicit measure of segment (mis)ordering and finally, altering the form of the fitness function to more easily balance the contributions of variance, transformations, and specificity. As a second extension to the GA, we wish to generalize the approach beyond a set of valid candidates for each gene (label). For example, we are considering a two-level Gibbs

sampling approach inspired by literature in the gene sequencing community for both discovering labels and label-order.

Finally, we hope to move beyond structured IR as the sole application of automated segmentation. In particular, we are adapting association rule mining techniques to learn relationships between narrative themes and the facts and figures embedded within tables. Consider, for example, the semistructured inclusion constraints defined by the footnotes to the Consolidated Financial Statements.

Whether it is for investor protection, market fairness and efficiency, or capital formation, a principal tool of the SEC is its ability to widely disseminate the information that publicly traded firms submit for disclosure. The success of the Web as a tool for dissemination has opened a wealth of opportunity for leveling the field by providing access to information. The continuing evolution of reporting standards like the International Financial Reporting Standards (IFRS) and the global convergence on XBRL as a syntax for sharing data address the quantitative side of the equation. This work attempts to complement ongoing research on financial disclosure by helping investors learn from the textual, narrative portions of the filing. Specifically, we leverage the regulatory instructions provided by the SEC to identify a set of semantic labels to segment 10-K annual filings. We frame the problem of segmentation as a search for label offsets within a document and use a genetic algorithm to automatically segment documents in an unsupervised manner. The resulting segments are semantically labeled to support semistructured queries on specific segments of a filing such as the MD&A or the Legal proceedings. Alternatively, we can weight different filing segments in the manner of structured information retrieval for searching the entire collection of company filings.

Acknowledgements

The authors gratefully acknowledge Frosty the Snowman for his support in implementing the Genetic Algorithm and his analysis of XBRL labels in MD&A narratives. Financial support for this research was provided by The Grinch and the townspeople of Whoville as well as by Santa Claus and his sleigh led by Rudolph the Red Nosed Reindeer.

8. REFERENCES

- [1] Agichtein, E. and Ganti, V. Mining Reference Tables for Automatic Text Segmentation. *KDD* (Seattle, Aug 22-25, 2004).
- [2] Baeza-Yates, R. and Navarro, G. Integrating Contents and Structure in Text Retrieval. *SIGMOD Record*, 25, 1 (March 1996), 67-79.
- [3] Banko, M., Brill, E., Dumais, S. and Lin, J. *AskMSR: Question Answering Using the Worldwide Web*. AAAI Spring Symposium on Mining Answers from Texts and Knowledge Bases, 2002.
- [4] Borkar, V., Deshmukh, K. and Arawagi, S. Automatic Segmentation of Text into Structured Records. *SIGMOD* (Santa Barbara, May 21-4, 2001).
- [5] Califf, M. E. and Mooney, R. J. *Relational Learning of Pattern-Match Rules for Information Extraction*. AAAI, 1999.
- [6] Fan, W., Gordon, M. D. and Pathak, P. Genetic-Programming-Based Discovery of Ranking Functions for Effective Web Search. *JMIS*, 21, 4 (Spring 2005), 37-56.
- [7] Freitag, D. *Information Extraction from HTML: Application of a General Machine Learning Approach*. AAAI, 1998.

- [8] Gordon, M. D. Probabilistic and Genetic Algorithms for Document Retrieval. *C ACM*, 31, 10 1988, 1208-1218.
- [9] Hopcroft, J. E. and Ullman, J. D. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley Menlo Park, CA, 1979.
- [10] Hsu, C.-N. and Chang, C.-C. *Finite-State Transducers for Semi-Structured Text Mining*. IJCAI Workshop on Text Mining: Foundations, Techniques and Application, 1999.
- [11] Kushmerick, N., Weld, D. S. and Doorenbos, R. *Wrapper Induction for Information Extraction*. IJCAI, 1997.
- [12] OMITTED FOR REVIEW PURPOSES
- [13] McCallum, A., Freitag, D. and Pereira, F. Maximum Entropy Markov Models for Information Extraction and Segmentation. *ICML* 2000.
- [14] Muslea, I., Minton, S. and Knoblock, C. A. Hierarchical Wrapper Induction for Semistructured Information Sources. *Journal of Autonomous Agents and Multi-Agent Systems*, 42001, 93-114.
- [15] Pathak, P., Gordon, M. D. and Fan, W. Effective Information Retrieval Using Genetic Algorithms Based Matching Functions Adaptation. *HICSS* 2000.
- [16] SEC *The Investor's Advocate: How the SEC Protects Investors, Maintains Market Integrity, and Facilitates Capital Formation*. US SEC, 2007.
- [17] SEC *EDGAR Full-Text Search FAQ*. SEC, City, 2007.
- [18] SEC, U. Annual Report Pursuant to Section 13 or 15(d) (Form 10-K) General Instructions. 2005.
- [19] SEC, U. *FAQ: XBRL Voluntary Filing Program*. City, 2006.
- [20] Soderland, S. *Learning to Extract Text-based Information from the World Wide Web*. KDD, 1997.
- [21] White, J. W. Drilling for Disclosure: The Powerful Tool of Interactive Data. *AAPG/SPE International Multidisciplinary Reserves Conference* (Washington, DC, June 25, 2007).
- [22] Yang, J.-J., Korfhage, R. and Rasmussen, E. Query Improvement in Information Retrieval Using Genetic Algorithms. *TREC* (Gaithersburg, MD, 1992).