6

Evaluating Self-Explanations in iSTART: Word Matching, Latent Semantic Analysis, and Topic Models

Chutima Boonthum, Irwin B. Levinstein, and Danielle S. McNamara

6.1 Introduction

iSTART (Interactive Strategy Trainer for Active Reading and Thinking) is a web-based, automated tutor designed to help students become better readers via multimedia technologies. It provides young adolescent to college-aged students with a program of self-explanation and reading strategy training [19] called Self-Explanation Reading Training, or SERT [17, 21, 24, 25]. The reading strategies include (a) comprehension monitoring, being aware of one's understanding of the text; (b) paraphrasing, or restating the text in different words; (c) elaboration, using prior knowledge or experiences to understand the text (i.e., domain-specific knowledge-based inferences) or common sense, using logic to understand the text (i.e., domain-general knowledge based inferences); (d) predictions, predicting what the text will say next; and (e) bridging, understanding the relation between separate sentences of the text. The overall process is called "self-explanation" because the reader is encouraged to explain difficult text to him- or herself. iSTART consists of three modules: Introduction, Demonstration, and Practice. In the last module, students practice using reading strategies by typing self-explanations of sentences. The system evaluates each self-explanation and then provides appropriate feedback to the student. If the explanation is irrelevant or too short, the student is required to add more information. Otherwise, the feedback is based on the level of overall quality.

The computational challenge here is to provide appropriate feedback to the students concerning their self-explanations. To do so requires capturing some sense of both the meaning and quality of the self-explanation. Interpreting text is critical for intelligent tutoring systems, such as iSTART, that are designed to interact meaningfully with, and adapt to, the users' input. iSTART was initially proposed as using Latent Semantic Analysis (LSA; [13]) to capture the meanings of texts and to assess the students' self-explanation; however, while the LSA algorithms were being built, iSTART used simple word matching algorithms. In the course of integrating the LSA algorithms, we found that a combination of word-matching and LSA provided better results than either separately [18].

Our goal in evaluating the adequacy of the algorithms has been to imitate experts' judgments of the quality of the self-explanations. The current evaluation system predicts the score that a human gives on a 4-point scale, where 0 represents an

evaluation of the explanation as irrelevant or too short; 1, minimally acceptable; 2, better but including primarily the local textual context; and 3, oriented to a more global comprehension. Depending on the text, population, and LSA space used, our results have ranged from 55 to 70 percent agreement with expert evaluations using that scale. We are currently attempting to improve the effectiveness of our algorithms by incorporating Topic Models (TM) either in place of or in conjunction with LSA and by using more than one LSA space from different genres (science, narrative, and general TASA corpus). We present some of the results of these efforts in this chapter.

Our algorithms are constrained by two major requirements, speedy response times and speedy introduction of new texts. Since the trainer operates in real time, the server that calculates the evaluation must respond in 4 to 5 seconds. Furthermore the algorithms must not require any significant preparation of new texts, a requirement precisely contrary to our plans when the project began. In order to accommodate the needs of the teachers whose classes use iSTART, the trainer must be able to use texts that the teachers wish their students to use for practice within a day or two. This time limit precludes us from significantly marking up the text or gathering related texts to incorporate into an LSA corpus.

In addition to the overall 4-point quality score, we are attempting to expand our evaluation to include an assessment of the presence of various reading strategies in the student's explanation so that we can generate more specific feedback. If the system were able to detect whether the explanation uses paraphrasing, bridging, or elaboration we could provide more detailed feedback to the students, as well as an individualized curriculum based on a more complete model of the student. For example, if the system were able to assess that the student only paraphrased sentences while self-explaining, and never used strategies such as making bridging inferences or knowledge-based elaborations, then the student could be provided additional training to generate more inference-based explanations.

This chapter describes how we employ word matching, LSA, and TM in the iSTART feedback systems and the performance of these techniques in producing both overall quality and reading strategy scores.

6.2 iSTART: Feedback Systems

iSTART was intended from the outset to employ LSA to determine appropriate feedback. The initial goal was to develop one or more benchmarks for each of the SERT strategies relative to each of the sentences in the practice texts and to use LSA to measure the similarity of a trainee's explanation to each of the benchmarks. A benchmark is simply a collection of words, in this case, words chosen to represent each of the strategies (e.g., words that represent the current sentence, words that represent a bridge to a prior sentence). However, while work toward this goal was progressing, we also developed a preliminary "word-based" (WB) system to provide feedback in our first version of iSTART [19] so that we could provide a complete curriculum for use in experimental situations. The second version of iSTART has integrated both LSA and WB in the evaluation process; however, the system still provides only overall quality feedback. Our current investigations aim to provide feedback based on identifying specific reading strategies.

6.2.1 Word Matching Feedback Systems

Word matching is a very simple and intuitive way to estimate the nature of a self-explanation. In the first version of iSTART, several hand-coded components were built for each practice text. For example, for each sentence in the text, the "important words" were identified by a human expert and a length criterion for the explanation was manually estimated. Important words were generally content words that were deemed important to the meaning of the sentence and could include words not found in the sentence. For each important word, an association list of synonyms and related terms was created by examining dictionaries and existing protocols as well as by human judgments of what words were likely to occur in a self-explanation of the sentence. In the sentence "All thunderstorms have a similar life history," for example, important words are *thunderstorm, similar, life*, and *history*. An association list for *thunderstorm* would include *storms, moisture, lightning, thunder, cold, tstorm, t-storm, rain, temperature, rainstorms*, and *electric-storm*. In essence, the attempt was made to imitate LSA.

A trainee's explanation was analyzed by matching the words in the explanation against the words in the target sentence and words in the corresponding association lists. This was accomplished in two ways: (1) Literal word matching and (2) Soundex matching.

Literal word matching - Words are compared character by character and if there is a match of the first 75% of the characters in a word in the target sentence (or its association list) then we call this a literal match. This also includes removing suffix -s, -d, -ed, -ing, and -ion at the end of each words. For example, if the trainee's self-explanation contains 'thunderstom' (even with the misspelling), it still counts as a literal match with words in the target sentence since the first nine characters are exactly the same. On the other hand, if it contains 'thunder,' it will not get a match with the target sentence, but rather with a word on the association list.

Soundex matching - This algorithm compensates for misspellings by mapping similar characters to the same soundex symbol [1, 5]. Words are transformed to their soundex code by retaining the first character, dropping the vowels, and then converting other characters into soundex symbols. If the same symbol occurs more than once consecutively, only one occurrence is retained. For example, 'thunderstorm' will be transformed to 't8693698'; 'communication' to 'c8368.' Note that the later example was originally transformed to 'c888368' and two 8s were dropped ('m' and 'n' are both mapped to '8'). If the trainee's self-explanation contains 'thonderstorm' or 'tonderstorm,' both will be matched with 'thunderstorm' and this is called a soundex match. An exact soundex match is required for short words (i.e., those with fewer than six alpha-characters) due to the high number of false alarms when soundex is used. For longer words, a match on the first four soundex symbols suffices. We are considering replacing this rough and ready approach with a spell-checker.

A formula based on the length of the sentence, the length of the explanation, the length criterion mentioned below, the number of matches to the important words, and the number of matches to the association lists produces a rating of 0 (inadequate), 1 (barely adequate), 2 (good), or 3 (very good) for the explanation. The rating of 0 or inadequate is based on a series of filtering criteria that assesses whether the explanation is too short, too similar to the original sentence, or irrelevant. *Length*

is assessed by a ratio of the number of words in the explanation to the number in the target sentence, taking into consideration the length criterion. For example, if the length of the sentence is 10 words and the length priority is 1, then the required length of the self-explanation would be 10 words. If the length of the sentence is 30 words and the length priority is 0.5, then the self-explanation would require a minimum of 15 words. *Relevance* is assessed from the number of matches to important words in the sentence and words in the association lists. *Similarity* is assessed in terms of a ratio of the sentence and explanation lengths and the number of matching important words. If the explanation is close in length to the sentence, with a high percentage of word overlap, the explanation would be deemed too similar to the target sentence. If the explanation failed any of these three criteria (Length, Relevance, and Similarity), the trainee would be given feedback corresponding to the problem and encouraged to revise the self-explanation.

Once the explanation passes the above criteria, then it is evaluated in terms of its overall quality. The three levels of quality that guide feedback to the trainee are based on two factors: 1) the number of words in the explanation that match either the important words or association-list words of the target sentence compared to the number of important words in the sentence and 2) the length of the explanation in comparison with the length of the target sentence. This algorithm will be referred as *WB-ASSO*, which stands for *word-based with association list.*

This first version of iSTART (word-based system) required a great deal of human effort per text, because of the need to identify important words and, especially, to create an association list for each important word. However, because we envisioned a scaled-up system rapidly adaptable to many texts, we needed a system that required relatively little manual effort per text. Therefore, WB-ASSO was replaced. Instead of lists of important and associated words we simply used content words (nouns, verbs, adjectives, adverbs) taken literally from the sentence and the entire text. This algorithm is referred to as *WB-TT*, which stands for *word-based with total text.* The content words were identified using algorithms from Coh-Metrix, an automated tool that yields various measures of cohesion, readability, other characteristics of language [9, 20]. The iSTART system then compares the words in the self-explanation to the content words from the current sentence, prior sentences, and subsequent sentences in the target text, and does a word-based match (both literal and soundex) to determine the number of content words in the self-explanation from each source in the text. While WB-ASSO is based on a richer corpus of words than WB-TT, the replacement was successful because the latter was intended for use together with LSA which incorporates the richness of a corpus of hundreds of documents. In contrast, WB-ASSO was used on its own.

Some hand-coding remained in WB-TT because the length criterion for an explanation was calculated based on the average length of explanations of that sentence collected from a separate pool of participants and on the importance of the sentence according to a manual analysis of the text. Besides being relatively subjective, this process was time consuming because it required an expert in discourse analysis as well as the collection of self-explanation protocols. Consequently, the hand-coded length criterion was replaced with one that could be determined automatically from the number of words and content words in the target sentence (we called this *word-based with total text and automated criteria*, or *WB2-TT*). The change from WB-TT to WB2-TT affected only the screening process of the length and similarity criteria. Its lower-bound and upper-bound lengths are entirely based on the target sentence's

length. The overall quality of each self-explanation (1, 2, or 3) is still computed with the same formula used in WB-TT.

6.2.2 Latent Semantic Analysis (LSA) Feedback Systems

Latent Semantic Analysis (LSA; [13, 14]) uses statistical computations to extract and represent the meaning of words. Meanings are represented in terms of their similarity to other words in a large corpus of documents. LSA begins by finding the frequency of terms used and the number of co-occurrences in each document throughout the corpus and then uses a powerful mathematical transformation to find deeper meanings and relations among words. When measuring the similarity between text-objects, LSA's accuracy improves with the size of the objects. Hence, LSA provides the most benefit in finding similarity between two documents. The method, unfortunately, does not take into account word order; hence, very short documents may not be able to receive the full benefit of LSA.

To construct an LSA corpus matrix, a collection of documents are selected. A document may be a sentence, a paragraph, or larger unit of text. A term-document-frequency (TDF) matrix X is created for those terms that appear in two or more documents. The row entities correspond to the words or terms (hence the W) and the column entities correspond to the documents (hence the D). The matrix is then analyzed using Singular Value Decomposition (SVD; [26]), that is the TDF matrix X is decomposed into the product of three other matrices: (1) vectors of derived orthogonal factor values of the original row entities W, (2) vectors of derived orthogonal factor values of the original column entities D, and (3) scaling values (which is a diagonal matrix) S. The product of these three matrices is the original TDF matrix.

$$\{X\} = \{W\}\{S\}\{D\} \tag{6.1}$$

The dimension (d) of $\{S\}$ significantly affects the effectiveness of the LSA space for any particular application. There is no definite formula for finding an optimal number of dimensions; the dimensionality can be determined by sampling the results of using the matrix $\{W\}\{S\}$ to determine the similarity of previously-evaluated document pairs for different dimensionalities of $\{S\}$. The optimal size is usually in the range of 300-400 dimensions.

The similarity of terms is computed by taking the cosine of the corresponding term vectors. A term vector is the row entity of that term in the matrix W. In iSTART, the documents are sentences from texts and trainees' explanations of those sentences. These documents consist of terms, which are represented by term vectors; hence, the document can be represented as a document vector which is computed as the sum of the term vectors of its terms:

$$D_i = \sum_{t=1}^{n} T_{ti} \tag{6.2}$$

where D_i is the vector for the i^{th} document D, T_{ti} is the term vector for the term t in D_i, and n is number of terms in D. The similarity between two documents (i.e., the cosine between the two document vectors) is computed as

$$Sim(D1, D2) = \frac{\sum_{i=1}^{d}(D1_i \times D2_i)}{\sum_{i=1}^{d}(D1_i)^2 \times \sum_{i=1}^{d}(D2_i)^2} \tag{6.3}$$

Since the first versions of iSTART were intended to improve students' comprehension of science texts, the LSA space was derived from a collection of science texts [11]. This corpus consists of 7,765 documents containing 13,502 terms that were used in two or more documents. By the time the first version of the LSA-based system was created (referred to as *LSA1*), the original goal of identifying particular strategies in an explanation had been replaced with the less ambitious one of rating the explanation as belonging one of three levels [22]. The highest level of explanation, called "*global-focused*," integrates the sentence material in a deep understanding of the text. A "*local-focused*" explanation explores the sentence in the context of its immediate predecessors. Finally, a "*sentence-focused*" explanation goes little beyond paraphrasing. To assess the level of an explanation, it is compared to four benchmarks or bags of words. The rating is based on formulae that use weighted sums of the four LSA cosines between the explanation and each of the four benchmarks.

The four benchmarks include: 1) the words in the title of the passage ("title"), 2) the words in the sentence ("current sentence"), 3) words that appear in prior sentences in the text that are causally related to the sentence ("prior text"), and 4) words that did not appear in the text but were used by two or more subjects who explained the sentence during experiments ("world knowledge"). While the title and current sentence benchmarks are created automatically, the prior-text benchmark depends on a causal analysis of the conceptual structure of the text, relating each sentence to previous sentences. This analysis requires both time and expertise. Furthermore, the world-knowledge benchmark requires the collection of numerous explanations of each text to be used. To evaluate the explanation of a sentence, the explanation is compared to each benchmark, using the similarity function mentioned above. The result is called a cosine value between the self-explanation (SE) and the benchmark. For example, *Sim(SE, Title)* is called the *title LSA cosine*. Discriminant Analysis was used to construct the formulae that categorized the overall quality as being a level 1, 2, or 3 [23]. A score is calculated for each of the levels using these formulae. The highest of the three scores determines the predicted level of the explanation. For example, the overall quality score of the explanation is a 1 if the level-1 score is higher than both the level-2 and level-3 scores.

Further investigation showed that the LSA1 cosines and the factors used in the WB-ASSO approach could be combined in a discriminant analysis that resulted in better predictions of the values assigned to explanations by human experts. However, the combined approach was less than satisfactory. Like WB-ASSO, LSA1 was not suitable for an iSTART program that would be readily adaptable to new practice texts. Therefore, we experimented with formulae that would simplify the data gathering requirements to develop LSA2. Instead of the four benchmarks mentioned above, we discarded the world knowledge benchmark entirely and replaced the benchmark based on causal analysis of prior-text with one that simply consisted of the words in the previous two sentences. We could do this because the texts were taken from science textbooks whose argumentation tends to be highly linear argumentation in science texts; consequently the two immediately prior sentences

worked well as stand-ins for the set of causally related sentences. It should be noted that this approach may not succeed so well with other genres, such as narrative or history texts.

We tested several systems that combined the use of word-matching and LSA2 and the best one is LSA2/WB2-TT. In these combinatory systems, we combine a weighted sum of the factors used in the fully automated word-based systems and LSA2. These combinations allowed us to examine the benefits of using the world knowledge benchmark (in LSA1) when LSA was combined with a fully automated word-based system and we found that world knowledge benchmark could be dropped. Hence, only three benchmarks are used for LSA-based factors: 1) the words in the title of the passage, 2) the words in the sentence, and 3) the words in the two immediately prior sentences. From the word-based values we include 4) the number of content words matched in the target sentence, 5) the number of content words matched in the prior sentences, 6) the number of content words matched in the subsequent sentences, and 7) the number of content words that were not matched in 4, 5, or 6. One further adjustment was made because we noticed that the LSA approach alone was better at predicting higher values correctly, while the word-based approach was better at predicting lower values. Consequently, if the formulae of the combined system predicted a score of 2 or 3, that value is used. However, if the system predicted a 1, a formula from the word-based system is applied. Finally, level 0 was assigned to explanations that had negligible cosine matches with all three LSA benchmarks.

6.2.3 Topic Models (TM) Feedback System

The Topic Models approach (TM; [10, 27]) applies a probabilistic model in finding a relationship between terms and documents in terms of topics. A document is conceived of as having been generated probabilistically from a number of topics and each topic consists of number of terms, each given a probability of selection if that topic is used. By using a TM matrix, we can estimate the probability that a certain topic was used in the creation of a given document. If two documents are similar, the estimates of the topics they probably contain should be similar. TM is very similar to LSA, except that a term-document frequency matrix is factored into two matrices instead of three.

$$\{X_{normalized}\} = \{W\}\{D\} \tag{6.4}$$

The dimension of matrix $\{W\}$ is W x T, where W is the number of words in the corpus and T is number of topics. The number of topics varies, more or less, with the size of corpus; for example, a corpus of 8,000 documents may require only 50 topics while a corpus of 40,000 documents could require about 300 topics. We use the TM Toolbox [28] to generate the $\{W\}$ or TM matrix, using the same science corpus as we used for the LSA matrix. In this construction, the matrix $\{X\}$ is for all terms in the corpus, not just those appearing in two different documents. Although matrix $\{X\}$ is supposed to be normalized, the TM toolbox takes care of this normalization and outputs for each topic, the topic probability, and a list of terms in this topic along with their probabilities in descending order (shown in Table 6.1). This output is easily transformed into the term-topic-probability matrix.

Table 6.1. Results from Topic Models Toolbox: science corpus, 50 topics, seed 1, 500 iteration, default alpha and beta.

TOPIC 2 0.0201963151	TOPIC 38 0.0214418635
earth 0.1373291184	light 0.1238061875
sun 0.0883152826	red 0.0339683946
solar 0.0454833721	color 0.0307797075
atmosphere 0.0418036547	white 0.0262046347
moon 0.0362104843	green 0.0230159476
surface 0.0181062747	radiation 0.0230159476
planet 0.0166343877	wavelengths 0.0230159476
center 0.0148681234	blue 0.0184408748
bodies 0.0147209347	dark 0.0178863206
tides 0.0139849912	visible 0.0170544891
planets 0.0133962364	spectrum 0.0151135492
gravitational 0.0125131042	absorbed 0.0149749106
system 0.0111884060	colors 0.0148362720
appear 0.0110412173	rays 0.0116475849
mass 0.0100108964	eyes 0.0108157535
core 0.0083918207	yellow 0.0105384764
space 0.0083918207	absorption 0.0102611992
times 0.0079502547	eye 0.0095680064
orbit 0.0073614999	pigment 0.0092907293
...	...

To measure the similarity between documents based on TM, the Kullback Liebler distance (KL-distance: [27]) between two documents is recommended, rather than the cosine (which, nevertheless, can be used). A document can be represented by a set of probabilities that this document could contain topic i using the following

$$D_t = \sum_{i=1}^{n} T_{it} \tag{6.5}$$

where D_t is the probability of topic t in the document D, T_{it} is the probability of topic t of the term i in the document D, and n is number of terms appearing in the document D. The KL-distance between two documents (the similarity) is computed as follows:

$$KL(D1, D2) = \frac{1}{2} \sum_{t=1}^{T} D1_t log_2(D1_t/D2_t) + \frac{1}{2} \sum_{t=1}^{T} D2_t log_2(D2_t/D1_t) \tag{6.6}$$

Constructing a TM matrix involves making choices regarding a number of factors, such as the number of topics, the seed for random number generation, alpha, beta, and the number of iterations. We have explored these factors and constructed a number of TM matrices in an effort to optimize the resulting matrix; however, for this preliminary evaluation, we use a TM matrix of 50 topics and a seed of 1.

The first TM-based system we tried was simply used in place of the LSA-based factors in the combined-system. The three benchmarks are still the same but sim-

ilarity is computed in two ways: (1) using cosines — comparing the explanation and the benchmark using the cosine formula (Referred as TM1) and (2) using KL distances — comparing the explanation and the benchmark using the KL distance (Referred as TM2). As before, formulae are constructed using Discriminant Analysis in order to categorize the quality of explanation as Levels 1, 2, or 3.

6.2.4 Metacognitive Statements

The feedback systems include a metacognitive filter that searches the trainees' self-explanations for patterns indicating a description of the trainee's mental state such as "now I see ..." or "I don't understand this at all." While the main purpose of the filter is to enable the system to respond to such non-explanatory content more appropriately, we also used the same filter to remove "noise" such as "What this sentence is saying is ..." from the explanation before further processing. We have examined the effectiveness of the systems with and without the filter and found that they all perform slightly better with than without it. Thus, the systems in this chapter all include the metacognitive filter.

The metacognitive filter also benefits the feedback system. When a metacognitive pattern is recognized, its category is noted. If the self-explanation contains only a metacognitive statement, the system will respond to a metacognitive category such as *understanding, not-understanding, confirmation, prediction,* or *boredom* instead of responding irrelevantly. Regular expressions are used to define multiple patterns for each metacognitive category. If any pattern is matched in the self-explanation, words matching the pattern are removed before evaluation. Examples of regular expression are shown below:

```
NOTUNDERSTAND :i(?:.?m|\W+am)(?:\W+\w+)?\W+\W+(?:(?:not
                (?:\W+\w+)?\W+(?:sure|certain|clear))|
                un(?:sure|certain|clear))
UNDERSTAND :now\W+i\W+(?:know|knew|underst(?:an|oo)d|
                remember(?:ed)?|recall(?:ed)?|recogniz(?:ed)?|get|
                got|see)
CONF :(?:so\W+)?i\W+(?:was|got\W+it)\W+(?:right|correct)
```

The first pattern will include "I'm not sure," "I am uncertain"; second pattern includes "Now I understand," "Now I remembered"; and the last pattern includes "So, I was right." We originally constructed over 60 patterns. These were reduced to 45 by running them on a large corpus of explanations and eliminating those that failed to match and adding those that were missed.

6.3 iSTART: Evaluation of Feedback Systems

Two experiments were used to evaluate the performance of various systems of algorithms that vary as a function of approach (word-based, LSA, combination of word-based and LSA, and combination of word-based TM). In Experiment 1, we

compare all eight systems in terms of the overall quality score by applying each system to a database of self-explanation protocols produced by college students. The protocols had been evaluated by a human expert on overall quality. In Experiment 2, we investigated two systems using a database of explanations produced by middle-school students. These protocols were scored to identify particular reading strategies.

6.3.1 Experiment 1

Self-Explanations. The self-explanations were collected from college students who were provided with SERT training and then tested with two texts, Thunderstorm and Coal. Both texts consisted of 20 sentences. The Thunderstorm text was self-explained by 36 students and the Coal text was self-explained by 38 students. The self-explanations were coded by an expert according to the following 4-point scale: 0 = vague or irrelevant; 1 = sentence-focused (restatement or paraphrase of the sentence); 2 = local-focused (includes concepts from immediately previous sentences); 3 = global-focused (using prior knowledge).

The coding system was intended to reveal the extent to which the participant elaborated the current sentence. Sentence-focused explanations do not provide any new information beyond the current sentence. Local-focused explanations might include an elaboration of a concept mentioned in the current or immediately prior sentence, but there is no attempt to link the current sentence to the theme of the text. Self-explanations that linked the sentence to the theme of the text with world knowledge were coded as "global-focused." Global-focused explanations tend to use multiple reading strategies, and indicate the most active level of processing.

Results. Each of the eight systems produces an evaluation comparable to the human ratings on a 4-point scale. Hence, we calculated the correlations and percent agreement between the human and system evaluations (see Table 6.2). Additionally, d primes (d′s) were computed for each strategy level as a measure of how well the system could discriminate among the different levels of strategy use. The d′s were computed from hit and false-alarm rates. A hit would occur if the system assigned the same self-explanation to a category (e.g., global-focused) as the human judges. A false-alarm would occur if the system assigned the self-explanation to a category (e.g., global-focused) that was different from the human judges (i.e., it was not a global-focused strategy). d′s are highest when hits are high and false-alarms are low. In this context, d′s refer to the correspondence between the human and system in standard deviation units. A d′ of 0 indicates chance performance, whereas greater d′s indicate greater correspondence.

One thing to note in Table 6.3 is that there is general improvement according to all of the measures going from left to right. As might be expected, the systems with LSA fared far better than those without LSA, and the combined systems were the most successful. The word-based systems tended to perform worse as the evaluation level increased (from 0 to 3), but performed relatively well at identifying poor self-explanations and paraphrases. All of the systems, however, identified the sentence-focused (i.e., 2's) explanations less successfully. However, the d′s for the sentence focused explanations approach 1.0 when LSA is incorporated, particularly when LSA is combined with the word-based algorithms.

Apart from better performance with LSA than without, the performance is also more stable with LSA. Whereas the word-based systems did not perform equally

Table 6.2. Measures of agreement for the Thunderstorm and Coal texts between the eight system evaluations and the human ratings of the self-explanations in Experiment 1.

Thunderstorm Text	WB-ASSO	WB-TT	WB2-TT	LSA1	LSA2	LSA2/ WB2-TT	TM1	TM2
Correlation	0.47	0.52	0.43	0.60	0.61	0.64	0.56	0.58
% Agreement	48%	50%	27%	55%	57%	62%	59%	60%
d' of 0's	2.21	2.26	0.97	2.13	2.19	2.21	1.49	2.37
d' of 1's	0.84	0.79	0.66	1.32	1.44	1.45	1.27	1.39
d' of 2's	0.23	0.36	-0.43	0.47	0.59	0.85	0.74	0.70
d' of 3's	1.38	1.52	1.41	1.46	1.48	1.65	1.51	1.41
Avg d'	1.17	1.23	0.65	1.34	1.43	1.54	1.25	1.23

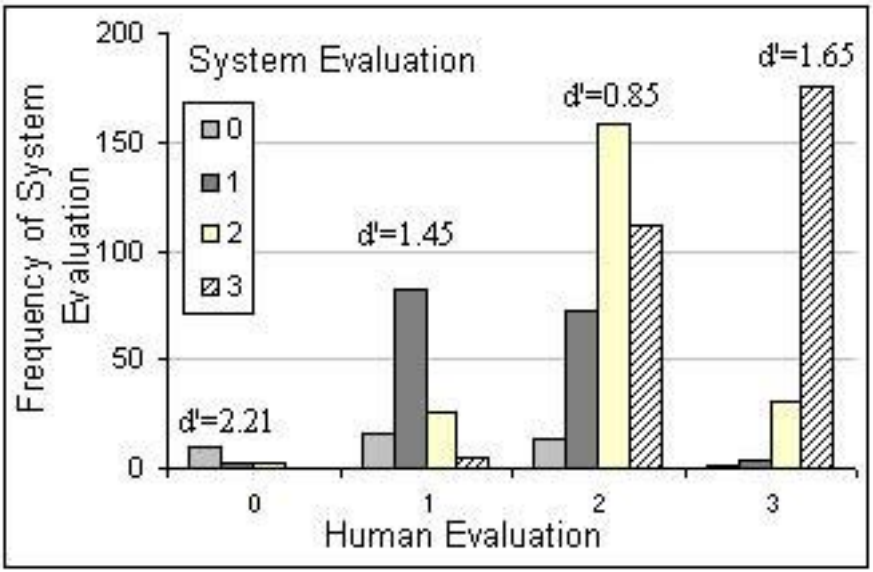
Coal Text	WB-ASSO	WB-TT	WB2-TT	LSA1	LSA2	LSA2/ WB2-TT	TM1	TM2
Correlation	0.51	0.47	0.41	0.66	0.67	0.71	0.63	0.61
% Agreement	41%	41%	29%	56%	57%	64%	61%	61%
d' of 0's	4.67	4.73	1.65	2.52	2.99	2.93	2.46	2.05
d' of 1's	1.06	0.89	0.96	1.21	1.29	1.50	1.38	1.52
d' of 2's	0.09	0.13	-0.37	0.45	0.49	0.94	0.74	0.61
d' of 3's	-0.16	1.15	1.28	1.59	1.59	1.79	1.60	1.50
Avg d'	1.42	1.73	0.88	1.44	1.59	1.79	1.54	1.42

well on the Thunderstorm and Coal texts, there is a high-level of agreement for the LSA-based formulas (i.e., the results are virtually identical in the two tables). This indicates that if we were to apply the word-based formulas to yet another text, we have less assurance of finding the same performance, whereas the LSA-based formulas are more likely to replicate across texts.

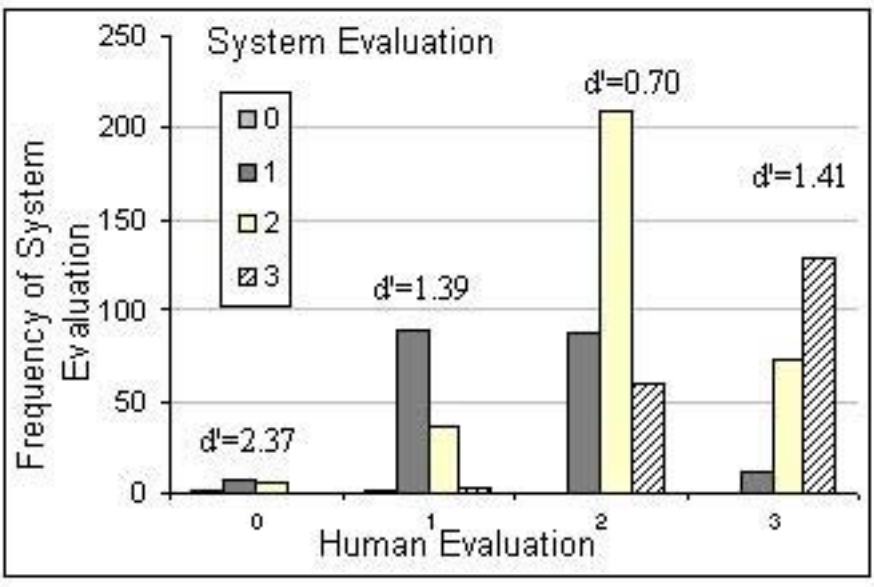
Figure 6.1.a provides a closer look at the data for the combined, automated system, LSA2/WB2-TT and Figure 6.1.b for the TM2 system. As the d′s indicated, both systems' performance is quite good for explanations that were given human ratings of 0, 1, or 3. Thus, the system successfully identifies poor explanations, paraphrases, and very good explanations. It is less successful for identifying explanations that consist of paraphrases in addition to some information from the previous sentence or from world knowledge. As one might expect, some are classified as paraphrases and some as global by the system. Although not perfect, we consider this result a success because so few were misclassified as poor explanations.

6.3.2 Experiment 2

Self-Explanations. The self-explanations were collected from 45 middle-school students (entering 8th and 9th grades) who were provided with iSTART training and then tested with two texts, Thunderstorm and Coal. The texts were shortened versions of the texts used in Experiment 1, consisting of 13 and 12 sentences, respectively. This chapter presents only the data from the Coal text.



a) LSA2/WB2-TT — LSA with Word-based



b) TM2 — Topic Models with KL distance

Fig. 6.1. Correspondence between human evaluations of the self-explanations and the combined system (LSA2/WB2-TT and TM2) for Thunderstorm text. Explanations were evaluated by humans as vague or irrelevant (0), sentence-focused (1), local-focused (2), or global (3).

The self-explanations from this text were categorized as paraphrases, irrelevant elaborations, text-based elaborations, or knowledge-based elaborations. Paraphrases did not go beyond the meaning of the target sentence. Irrelevant elaborations may have been related to the sentence superficially or tangentially, but were not related to the overall meaning of the text and did not add to the meaning of the text. Text-based elaborations included bridging inferences that made links to information presented in the text prior to the sentence. Knowledge-based elaborations included the use of prior knowledge to add meaning to the sentence. This latter category is analogous to, but not the same as, the global-focused category in Experiment 1.

Results. In contrast to the human coding system used in Experiment 1, the coding system applied to this data was not intended to map directly onto the iSTART evaluation systems. In this case, the codes are categorical and do not necessarily translate to a 0-3 quality range. One important goal is to be able to assess (or discriminate) the use of reading strategies and improve the system's ability to appropriately respond to the student. This is measured in terms of percent agreement with human judgments of each reading strategy shown in Table 6.3.

Table 6.3. Percent agreement to expert ratings of the self-explanations to the Coal text for the LSA2/WB2-TT and TM2 combined systems for each reading strategy in Experiment 2.

Reading Strategy	LSA2/WB2-TT	TM2
Paraphrase Only	69.9	65.8
Irrelevant Elaboration Only	71.6	76.0
Current Sentence Elaboration Only	71.9	71.2
Knowledge-Based Elaboration Only	94.6	90.3
Paraphrase + Irrelevant Elaboration	79.7	76.6
Paraphrase + Current Sentence Elaboration	68.2	67.3
Paraphrase + Knowledge-Based Elaboration	84.6	81.2

The results show that both systems perform very well, with an average of 77% for the LSA2/WB2-TT system and 75% for the TM2 system. This approaches our criteria of 85% agreement between trained experts who score the self-explanations. The automated systems could be thought of as 'moderately trained scorers.' These results thus show that either of these systems would guide appropriate feedback to the student user.

The score for each strategy score (shown in Table 6.3) can be coded either 0=present or 1=present. With the current coding scheme, only one strategy (out of seven) will be given a value of 1. We are currently redefining the coding scheme so that each reading strategy will have its own scores. For example, if the explanation contains both paraphrase and current sentence elaboration, with the current coding scheme, "Paraphrase + Current Sentence Elaboration" will be coded as a 1. On the other hand, with the new coding scheme, we will have at least 3 variables: (1) "Paraphrase" will be coded as a 1 for *present*, (2) "Elaboration" coded as a 1 for *present*, and (3) "Source of Elaboration" coded as a 2 for *current sentence elaboration*.

6.4 Discussion

The purpose of this chapter has been to investigate the ability of topic model algorithms to identify the quality of explanations as well as specific reading strategies in comparison to word-based and LSA-based algorithms. We found in Experiment 1 that TM systems performed comparably to the combined systems, though not quite as well. In Experiment 2, we found that the TM models performed nearly as well as the combined system in identifying specific strategies. These results thus broaden the scope of NLP models that can be applied to problems such as ours — providing real-time feedback in a tutoring environment. Indeed, the performance of both systems in Experiment 2 was highly encouraging. These results indicate that future versions of iSTART will be able to provide specific feedback about reading comprehension strategy use with relatively high confidence.

Our future work with the TM systems will be to attempt to combine the TM algorithms with the LSA and word-based algorithms. To venture toward that goal, we need to first identify the strengths of the TM algorithms so that the combined algorithm capitalizes on the strengths of the TM — much as we did when we created the combined word-based and LSA-based system. This will require that we analyze a greater variety of protocols, including self-explanations from a greater variety of texts and text genres. We are in the process of completing that work.

These NLP theories and their effectiveness have played important roles in the development of iSTART. For iSTART to effectively teach reading strategies, it must be able to deliver valid feedback on the quality of the self-explanations that a student types during practice. In order to deliver feedback, the system must understand, at least to some extent, what a student is saying in his or her self-explanation. Of course, automating natural language understanding has been extremely challenging, especially for non-restrictive content domains like self-explaining a text in which a student might say one of any number of things. Algorithms such as LSA opened up a horizon of possibilities to systems such as iSTART — in essence LSA provided a 'simple' algorithm that allowed tutoring systems to provide appropriate feedback to students (see [14]). The results presented in this chapter show that the topic model similarly offers a wealth of possibilities in natural language processing.

6.5 Acknowledgments

This project was supported by NSF (IERI Award number: 0241144) and its continuation funded by IES (IES Award number: R305G020018). Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF and IES.

References

1. Birtwisle, M. (2002) The Soundex Algorithm. Retrieved from: http://www.comp.leeds.ac.uk/matthewb/ar32/basic_soundex.htm
2. Bransford, J., Brown, A., & Cocking, R., Eds. (2000). How people learn: Brain, mind, experience, and school. Washington, D.C.: National Academy Press. Online at: http://www.nap.edu/html/howpeople1/

3. Chi, M. T. H., De Leeuw, N., Chiu, M., & LaVancher, C. (1994). Eliciting self-explanations improves understanding. Cognitive Science, 18, 439-477.
4. Chi, M. T. H., Bassok, M., Lewis, M. W., Reimann, R., & Glaser, R. (1989). Self-explanation: How students study and use examples in learning to solve problems. Cognitive Science, 13, 145-182.
5. Christian. P. (1998) Soundex — can it be improved? Computers in Genealogy, 6 (5)
6. Graesser, A. C., Penumatsa, P., Ventura, M., Cai, Z., & Hu, X. (2005). Using LSA in AutoTutor: Learning through mixed-initiative dialogue in natural language. In T. Landauer, D.S., McNamara, S. Dennis, & W. Kintsch (Eds.), LSA: A Road to Meaning. Mahwah, NJ: Erlbaum.
7. Graesser, A. C., Hu, X., & McNamara, D. S. (2005). Computerized learning environments that incorporate research in discourse psychology, cognitive science, and computational linguistics. In A. F. Healy (Ed.), Experimental Cognitive Psychology and its Applications: Festschrift in Honor of Lyle Bourne, Walter Kintsch, and Thomas Landauer. Washington, D.C.: American Psychological Association.
8. Graesser, A. C., Hu, X., & Person, N. (2001). Teaching with the help of talking heads. In T. Okamoto, R. Hartley, Kinshuk, J. P. Klus (Eds.), Proceedings IEEE International Conference on Advanced Learning Technology: Issues, Achievements and Challenges (460-461).
9. Graesser, A. C., McNamara, D. S., Louwerse, M. M., & Cai, Z. (2004). CohMetrix: Analysis of text on cohesion and language. Behavior Research Methods, Instruments, and Computers, 36, 193-202.
10. Griffiths, T., & Steyvers, M. (2004). Finding Scientific Topics. Proceedings of the National Academy of Science, 101 (suppl. 1), 5228-5235.
11. Kurby, C.A., Wiemer-Hastings, K., Ganduri, N., Magliano, J.P., Millis, K.K., & McNamaar, D.S. (2003). Computerizing Reading Training: Evaluation of a latent semantic analysis space for science text. Behavior Research Methods, Instruments, and Computers, 35, 244-250.
12. Kintsch, E., Caccamise, D., Dooley, S., Franzke, M., & Johnson, N. (2005). Summary street: LSA-based software for comprehension and writing. In T. Landauer, D.S., McNamara, S. Dennis, & W. Kintsch (Eds.), LSA: A Road to Meaning. Mahwah, NJ: Erlbaum.
13. Landauer, T. K., Foltz, P. W., & Laham, D. (1998). Introduction to Latent Semantic Analysis. Discourse Processes, 25, 259-284.
14. Landauer, T. K., McNamara, D. S., Dennis, S., & W. Kintsch. (2005) LSA: A Road to Meaning, Mahwah, NJ: Erlbaum.
15. Louwerse, M. M., Graesser, A. C., Olney, A., & the Tutoring Research Group. (2002). Good computational manners: Mixed-initiative dialog in conversational agents. In C. Miller (Ed.), Etiquette for Human-Computer Work, Papers from the 2002 Fall Symposium, Technical Report FS-02-02, 71-76.
16. Magliano, J. P., Todaro, S., Millis, K. K., Wiemer-Hastings, K., Kim, H. J., & McNamara, D. S. (2004). Changes in reading strategies as a function of reading training: A comparison of live and computerized training. Submitted for publication.
17. McNamara, D. S. (2004). SERT: Self-explanation reading training. Discourse Processes, 38, 1-30.
18. McNamara, D. S., Boonthum, C., Levinstein, I. B., & Millis, K. K. (2005) Using LSA and word-based measures to assess self-explanations in iSTART. In

T. Landauer, D.S. McNamara, S. Dennis, & W. Kintsch (Eds.), LSA: A Road to Meaning, Mahwah, NJ: Erlbaum.
19. McNamara, D. S., Levinstein, I. B., & Boonthum, C. (2004). iSTART: Interactive strategy training for active reading and thinking. Behavior Research Methods, Instruments, & Computers, 36, 222-233.
20. McNamara, D. S., Louwerse, M. M., & Graesser, A. C. (2002). Coh-Metrix: Automated cohesion and coherence scores to predict text readability and facilitate comprehension. Technical report, Institute for Intelligent Systems, University of Memphis, Memphis, TN.
21. McNamara, D. S., & Scott, J. L. (1999). Training reading strategies. In M. Hahn & S. C. Stoness (Eds.), Proceedings of the Twenty-first Annual Meeting of the Cognitive Science Society (pp. 387-392). Hillsdale, NJ: Erlbaum.
22. Millis, K. K., Kim, H. J., Todaro, S. Magliano, J. P., Wiemer-Hastings, K., & McNamara, D. S. (2004). Identifying reading strategies using latent semantic analysis: Comparing semantic benchmarks. Behavior Research Methods, Instruments, & Computers, 36, 213-221.
23. Millis, K. K., Magliano, J. P., Wiemer-Hastings, K., Todaro, S., & McNamara, D. S. (2005). Assessing comprehension with Latent Semantic Analysis. In T. Landauer, D.S. McNamara, S. Dennis, & W. Kintsch (Eds.), LSA: A Road to Meaning, Mahwah, NJ: Erlbaum.
24. O'Reilly, T., Best, R., & McNamara, D. S. (2004). Self-Explanation reading training: Effects for low-knowledge readers. In K. Forbus, D. Gentner, T. Regier (Eds.), Proceedings of the Twenty-sixth Annual Meeting of the Cognitive Science Society (pp. 1053-1058). Mahwah, NJ: Erlbaum.
25. O'Reilly, T., Sinclair, G. P., & McNamara, D. S. (2004). Reading strategy training: Automated verses live. In K. Forbus, D. Gentner, T. Regier (Eds.), Proceedings of the Twenty-sixth Annual Meeting of the Cognitive Science Society (pp. 1059-1064). Mahwah, NJ: Erlbaum.
26. Press, W.M., Flannery, B.P., Teukolsky, S.A., & Vetterling, W.T. (1986). Numerical recipes: The art of scientific computing. New York, NY: Cambridge University Press.
27. Steyvers, M., & Griffiths, T. (2005) Probabilistic topic models. In T. Landauer, D.S. McNamara, S. Dennis, & W. Kintsch (Eds.), LSA: A Road to Meaning, Mahwah, NJ: Erlbaum.
28. Steyvers, M., & Griffiths, T. (2005) Matlab Topic Modeling Toolbox 1.3. Retrieved from http://psiexp.ss.uci.edu/research/programs_data/toolbox.htm
29. Streeter, L., Lochbaum, K., Psotka, J., & LaVoie, N. (2005). Automated tools for collaborative learning environments. In T. Landauer, D.S., McNamara, S. Dennis, & W. Kintsch (Eds.), LSA: A Road to Meaning. Mahwah, NJ: Erlbaum.

7

Textual Signatures: Identifying Text-Types Using Latent Semantic Analysis to Measure the Cohesion of Text Structures

Philip M. McCarthy, Stephen W. Briner, Vasile Rus, and
Danielle S. McNamara

7.1 Introduction

Just as a sentence is far more than a mere concatenation of words, a text is far more than a mere concatenation of sentences. Texts contain pertinent information that co-refers across sentences and paragraphs [30]; texts contain relations between phrases, clauses, and sentences that are often causally linked [21, 51, 56]; and texts that depend on relating a series of chronological events contain temporal features that help the reader to build a coherent representation of the text [19, 55]. We refer to textual features such as these as cohesive elements, and they occur within paragraphs (locally), across paragraphs (globally), and in forms such as referential, causal, temporal, and structural [18, 22, 36]. But cohesive elements, and by consequence cohesion, does not simply feature in a text as dialogues tend to feature in narratives, or as cartoons tend to feature in newspapers. That is, cohesion is not present or absent in a binary or optional sense. Instead, cohesion in text exists on a continuum of presence, which is sometimes indicative of the text-type in question [12, 37, 41] and sometimes indicative of the audience for which the text was written [44, 47]. In this chapter, we discuss the nature and importance of cohesion; we demonstrate a computational tool that measures cohesion; and, most importantly, we demonstrate a novel approach to identifying text-types by incorporating contrasting rates of cohesion.

7.2 Cohesion

Recent research in text processing has emphasized the importance of the cohesion of a text in comprehension [5, 44, 43]. Cohesion is the degree to which ideas in the text are explicitly related to each other and facilitate a unified situation model for the reader. As McNamara and colleagues have shown, challenging text (such as science) is particularly difficult for low-knowledge students. These students are cognitively burdened when they are forced to make inferences across texts [22, 34, 35, 38, 44]. Adding cohesion to text alleviates this burden by filling conceptual and structural gaps. Recent developments in computational linguistics and discourse processing have now made it possible to measure this textual cohesion. These developments

have come together in a computational tool called *Coh-Metrix* [22] that approximates over 200 indices of textual cohesion and difficulty. Armed with this technology, text-book writers and teachers have the opportunity to better assess the appropriateness of a text for particular students [47], and researchers have the opportunity to assess cohesion patterns in text-types so as to better understand what constitutes a prototypical text from any given domain, genre, register, or even author [37, 42, 12, 14].

7.3 Coh-Metrix

Coh-Metrix assesses characteristics of texts by using parts of speech classifiers [4, 49, 7, 8, 9, 10, 11], and latent semantic analysis [32, 33]. The indices generated from Coh-Metrix offer an assessment of the cohesiveness and readability of any given text . These indices have been used to indicate textual cohesion and difficulty levels in a variety of studies. For example, Ozuru et al. [47] used Coh-Metrix to rate high and low cohesion versions of biology texts, the study showing that participants benefited most from the high cohesion versions. And Best, Floyd, and McNamara [1] used Coh-Metrix to compare 61 third-graders' reading comprehension for narrative and expository texts, the study suggesting that children with low levels of world knowledge were more inclined to have comprehension problems with expository texts. While research into assessing the benefits of cohesion continues apace, the utility of the Coh-Metrix tool has also allowed the pursuit of other avenues of textual investigation.

One of these alternative avenues is text identification. For example, Louwerse et al. [37] used Coh-Metrix to investigate variations in cohesion across written and spoken texts, finding evidence for a significant difference between these modes. McCarthy, Lewis, et al. [42] showed that Coh-Metrix indices were versatile enough to distinguish between authors even within the same register. Crossley et al.[12] used a wide variety of Coh-Metrix indices to show significant differences between authentic English language texts and the simplified versions used in texts designed for English language learners. And McCarthy, Lightman et al. [41] used Coh-Metrix to investigate variations in cohesion and difficulty across units of science and history texts, finding evidence that while difficulty scores for textbooks reflect the grade to which they are assigned, the cohesion rates differed significantly depending upon domain. In this chapter, we build on these approaches to identification of textual characteristics by demonstrating how cohesion indices produced by Coh-Metrix can be used to form prototypical models of text-types that we call *textual signatures.*

7.4 Approaches to Analyzing Texts

Traditional approaches to categorizing discourse have tended to treat text as if it were a homogeneous whole. These wholes, or bodies of text, are analyzed for various textual features, which are used to classify the texts as belonging to one category or another [2, 3, 26, 27, 37, 42]. To be sure, such approaches have yielded impressive findings, generally managing to significantly discriminate texts into categories such as dialect, domain, genre, or author. Such discrimination is made possible because

different kinds of texts feature different quantities of features. For example, Biber [2] identified *if clauses* and singular person pronoun use as key predictors in distinguishing British- from American-English. Louwerse et al. [37] used cohesion scores generated from Coh-Metrix to distinguish both spoken from written texts and narratives from non-narratives. And Stamatatos, Fakotatos, and Kokkinakis [50] used a number of style markers including punctuation features and frequencies of verb- and noun-phrases to distinguish between the authors of a variety of newspaper columns. Clearly, discriminating texts by treating them as homogenous wholes has a good track record. However, texts tend to be *heterogeneous*, and treating them as such may substantially increase the power of corpus analyses.

The *parts* of a text serve the textual whole either by function or by form. In terms of *function*, Propp [48] identified that texts can be comprised of fundamental components, fulfilled by various characters, performing set functions. In terms of *form*, numerous theories of text structure have demonstrated how textual elements are inter-related [24, 25, 31, 40]. Labov's narrative theory, to take one example, featured six key components: the abstract (a summary), the orientation (the cast of characters, the scene, and the setting), the action (the problem, issue, or action), the evaluation (the story's significance), the resolution (what happens, the denouement), and the coda (tying up lose ends, moving to the present time and situation).

Unfortunately for text researchers, the identification of the kinds of discourse markers described above has proven problematic because the absence or ambiguity of such textual markers tends to lead to limited success [39]. This is not to say that there has been no success at all. Morris and Hirst [46], for example, developed an algorithm that attempted to uncover a hierarchical structure of discourse based on lexical chains. Although their algorithm was only manually tested, the evidence from their study, suggesting that text is structurally identifiable through themes marked by chains of similar words, supports the view that the elements of heterogeneous texts are identifiable. Hearst [23] developed this idea further by attempting to segment expository texts into topically related parts. Like Morris and Hirst [46], Hearst used term repetition as an indicator of topically related parts. The output of his method is a linear succession of topics, with topics able to extend over more than one paragraph. Hearst's algorithm is fully implementable and was also tested on magazine articles and against human judgments with reported precision and recall measures in the 60th percentile, meaning around 60% of topic boundaries identified in the text are correct (precision) and 60% of the true boundaries are identified (recall).

The limited success in identifying textual segments may be the result of searching for a reliable fine grained analysis before a courser grain has first been established. For example, a courser approach acknowledges that texts have easily identifiable *beginnings*, *middles*, and *ends*, and these *parts* of a text , or at least a sample from them, are not at all difficult to locate. Indeed, textual analysis using such parts has proved quite productive. For example, Burrows [6] found that the introduction section of texts rather than texts as a whole allowed certain authorship to be significantly distinguished. And McCarthy, Lightman et al.[41] divided high-school science and history textbook chapters into sections of beginnings, middles, and ends, finding that reading difficulty scores rose with significant regularity across these sections as a chapter progressed.

If we accept that texts are comprised of parts, and that the text (as a whole) is dependent upon the presence of each part, then we can form the hypothesis that

the parts of the text are inter-dependent and, therefore, are likely to be structurally inter-related. In addition, as we know that cohesion exists in texts at the clausal, sentential, and paragraph level [22], it would be no surprise to find that cohesion also existed across the parts of the text that constitute the whole of the text . If this were *not* the case, parts of text would have to exist that bore no reference to the text as a whole. Therefore, if we measure the cohesion that exists across identifiable parts of the text, we can predict the degree to which the parts co-refer would be indicative of the kind of text being analyzed. In Labov's [31] narrative model, for example, we might expect a high degree of coreference between the second section (the orientation) and the sixth section (the coda): Although the two sections are textually distant, they are semantically related in terms of the textual elements with both sections likely to feature the characters, the motive of the story, and the scene in which the story takes place. In contrast, we might expect less coreference between the forth and fifth sections (evaluation and resolution): While the *evaluation* and *resolution* are textually juxtaposed, the *evaluation* section is likely to offer a more global, moral and/or abstracted perspective of the story. The *resolution*, however, is almost bound to be local to the story and feature the characters, the scene, and the outcome. Consequently, semantic relations between these two elements are likely to be less marked.

By forming a picture of the degree to which textual parts inter-relate, we can build a representation of the structure of the texts, a prototypical model that we call *the textual signature*. Such a signature stands to serve students and researchers alike. For students, their work can be analyzed to see the extent to which their paper reflects a prototypical model. Specifically, a parts analysis may help students to see that sections of their papers are under- or over-represented in terms of the global cohesion. For researchers, a text-type signature should help significantly in mining for appropriate texts. For example, the first ten web sites from a Google search for a text about cohesion (featuring the combined keywords of *comprehension*, *cohesion*, *coherence*, and *referential*) yielded papers from the field of composition theory, English as a foreign language, and cognitive science, not to mention a disparate array of far less academic sources. While the specified keywords that were entered may have occurred in each of the retrieved items, the organization of the parts of the retrieved papers (and their inter-relatedness) would differ. Knowing the signatures that distinguishes the text types would help researchers to locate more effectively the kind of resources that they require. A further possible benefit of textual signatures involves Question Answering (QA) systems [45, 52]. Given a question and a large collection of texts (often in gigabytes), the task in QA is to draw a list of short answers (the length of a sentence) to the question from the collection. The typical architecture of a modern QA system includes three subsystems: question processing, paragraph retrieval and answer processing. Textual signatures may be able to reduce the search space in the paragraph retrieval stage by identifying more likely candidates.

7.5 Latent Semantic Analysis

To assess the inter-relatedness of text sections we used latent semantic analysis (hereafter, LSA). An extensive review of the procedures and computations involved in LSA is available in Landauer and Dumais [32] and Landauer et al. [33]. For this

chapter, however, we offer only an overview of the theory of LSA, its method of calculations, and a summary of some of the many studies that have incorporated its approach.

LSA is a technique that uses a large corpus of texts together with singular value decomposition to derive a representation of world knowledge [33]. LSA is based on the idea that any word (or group of words) appears in some contexts but not in others. Thus, words can be compared by the aggregate of their co-occurrences. This aggregate serves to determine the degree of similarity between such words [13]. LSA's practical advantage over shallow word overlap measures is that it goes beyond lexical similarities such as *chair/chairs* or *run/ran*, and manages to rate the relative semantic similarity between terms such as *chair/table*, *table/wood*, and *wood/forest*. As such, LSA does not only tell us whether two items are the same, it tells us how similar they are. Further, as Wolfe and Goldman [54] report, there is substantial evidence to support the notion that the reliability of LSA is not significantly different from human raters when asked to perform the same judgments.

As a measure of semantic relatedness, LSA has proven to be a useful tool in a variety of studies. These include computing ratings of the quality of summaries and essays [17, 29], tracing essay elements to their sources [15], optimizing texts-to-reader matches based on reader knowledge and projected difficulty of unread texts [53], and for predicting human interpretation of metaphor difficulty [28]. For this study, however, we adapted the LSA cohesion measuring approach used by Foltz, Kintsch & Landauer [16]. Foltz and colleagues formed a representation of global cohesion by using LSA to analyze the relationship of ever distant textual paragraphs. As the distances increased, so the LSA score of similarity decreased. The results suggested that LSA was a useful and practical tool for measuring the relative degrees of similarity between textual sections. In our study, however, we replace Foltz and colleagues comparison of paragraphs with a comparison of journal sections, and rather than assuming that cohesion would decrease relative to distance, we made predictions based on the relative similarity between the sections of the article.

7.6 Predictions

The *abstract* section was selected as the primary source of comparison as it is the only section whose function is specifically to relate the key elements of each other section of the paper. But the abstract does not relate to each other section of the paper equally. Instead, the abstract outlines the theme of the study (introduction); it can refer to the basic method used in the study (methods); it will briefly state a prominent result from the study (results); and it will then discuss the relevance of the studys findings (discussions). This definition allowed us to make predictions as to the signature generated from such comparisons. Specifically, we predicted that *abstracts* would feature far greater reference to the *introduction* (AI comparison type) and *discussion* sections (AD comparison type), less reference to the results section (AR comparison type), and less reference still to the methods section (AM comparison type). The reason for such predictions is that abstracts would take more care to set the scene of the paper (the introduction) and the significance of the findings (discussions). The results section, although important, tends to see its key findings restated in the discussion section, where it is subsumed into the significance of the

paper. We predicted that the abstract to methods comparison type (AM) would form the weakest co-reference as experimental methods, although essential to state clearly in the body of a paper, tend to follow well-established patterns and are of little interest to the reader of an abstract who needs to understand quickly and succinctly the gist of the paper.

7.7 Methods

Using freely available on-line psychology papers from five different journals (see Appendix), we formed a corpus of 100 texts. For the purposes of simplification and consistency, we extracted from this corpus only the texts that were comprised of five author-identified sections: *abstract*, *Introduction*, *methods*, *results*, *discussion*. This left 67 papers in the analysis. We then removed titles, tables, figures, and footnotes before forming the paired sections outlined above (AI, AM, AR, AD). Each of the pairs from the 67 papers was then processed through the Coh-Metrix version of LSA .

7.8 Results of Experiment 1

To examine differences in relatedness of the abstract to each of the text sections, we conducted a repeated measures Analysis of Variance (ANOVA) on the LSA cosines including the within-text factors of AI (M=.743, SD=.110), AM (M=.545, SD=.151), AR (M=.637, SD=.143), and AD (M=.742, SD=.101). As shown in Figure 7.1, the results confirmed our predictions. There was a main effect of comparison type, $F(3,66)$= 54.701, MSE=.011, p<.001. Pairwise contrasts (see Table 7.1) indicated that all of the differences were reliable except for the difference between the AI and AD comparisons. The pattern depicted in Figure 7.1 is what we will refer to as the *textual signature* for scientific reports such as those we have analyzed in this study.

Table 7.1. Pairwise Comparisons of the Relatedness of Text Sections to the Abstract

	Method (AM)	Results (AR)	Discussion (AD)
Introduction (AI)	Diff=.198 (.021)*	Diff=.106 (.021)*	Diff=.001(.010)
Method (AM)		Diff=-.092 (.019)*	Diff=-.197(.019)*
Results (AR)			Diff=-.105 (.018)*

Notes: Diff denotes the average difference between the cosines; * $p<.01$

While the signature from Experiment 1 confirmed our prediction, one possibility is that the differences may simply reflect the relative length of the textual sections. To test this possibility, we examined differences in relatedness of the abstract to each of the text sections by conducting a repeated measures ANOVA on

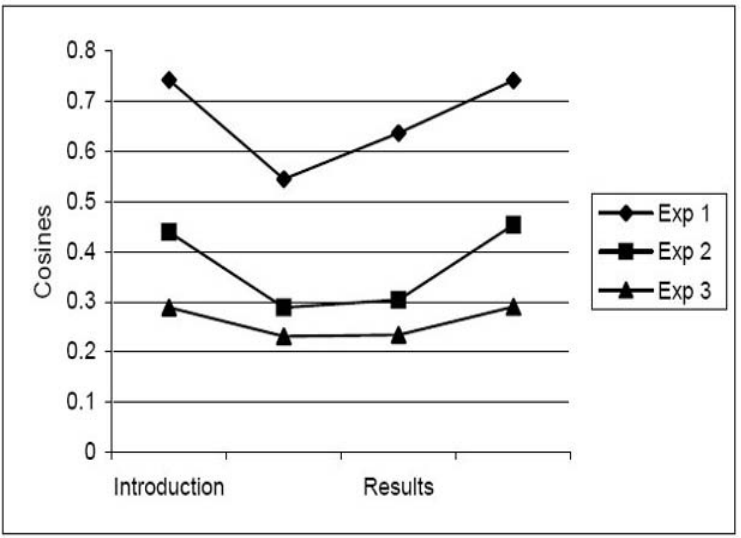


Fig. 7.1. Textual signature formed from means of the abstract to other sections for Experiments 1, 2 and 3.

the number of words in each text section including the within-text factors of Introduction (*M=1598.015, SD=871.247*), Method (*M*=1295.791, *SD*=689.756), Results (*M*=1408.627, *SD*=841.185), and Discussion (*M*=1361.284, *SD*=653.742). There was a main effect of comparison type, $F(3,66) = 2.955$, *MSE*=382691.182, p=.034. Pairwise contrasts (see Table 7.2) indicated that the only significant differences were between the AI/AM comparison types and the AI/AD comparison type. The results confirmed that the section *length* signature does not reflect the LSA signature (see Figure 7.2). Removing words that LSA does not account for from this analysis (such as numbers) made no significant difference to the results.

Table 7.2. Pairwise Comparisons of the Relatedness of Text Sections to the Abstract for text length

	Method (AM)	Results (AR)	Discussion (AD)
Introduction (AI)	Diff=302.22 (113.13)*	Diff=189.39 (107.91)	Diff=236.73 (94.37)*
Method (AM)		Diff=-112.84 (120.59)	Diff=-65.49 (105.67)
Results (AR)			Diff=47.34 (97.41)

Notes: Diff denotes the average difference between the lengths; * $p<.01$

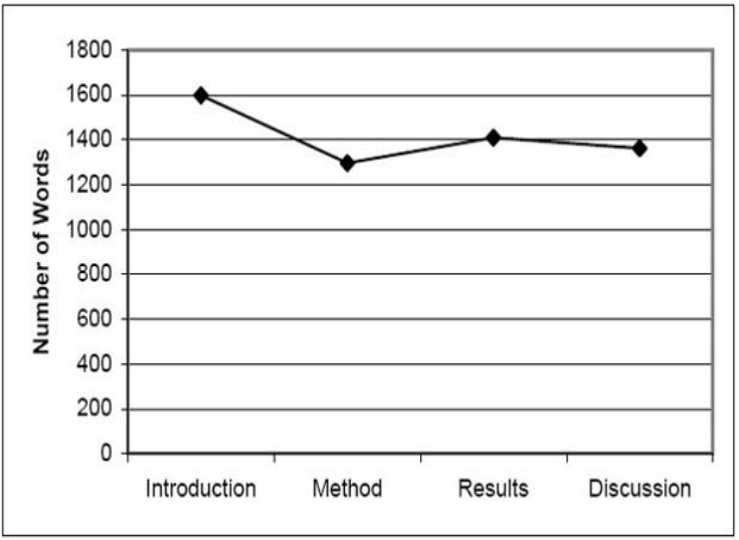


Fig. 7.2. Average length in terms of number of words in each text section.

7.9 Experiment 2

If LSA compares relative similarities between sections, then it is reasonable to assume that comparing similarly *themed* papers would produce signatures similar to those observed in Experiment 1. Because less of a relationship is expected from papers with overlapping themes as compared to those from the same paper, we predicted a relative decline in the LSA comparison scores. We also expected a reduced relationship between the abstract and results section because they are from different studies.

To test this prediction, we composed an entirely new corpus of 20 similar articles: all themed as *working memory and intelligence*. We then extracted the abstracts of these texts from the bodies and randomly reassigned the abstract to a different articles parts. As in Experiment 1, we conducted a repeated measures ANOVA on the LSA cosines including the within-text factors of AI (*M=.439, SD=.136*), AM (*M*=.289, *SD*=.110), AR (*M*=.304, *SD*=.144), and AD (*M*=.454, *SD*=.162). There was a main effect of comparison type, $F(3,19)= 16.548$, $MSE=.009$, $p<.001$. Pairwise contrasts (see Table 7.3) indicated that all differences were reliable except between AM and AD and between AM and AR.

As shown in Figure 7.1, the pattern of cosines is similar to that of Experiment 1, with reduced scores overall compared to Experiment 1, and a reduction in the relationship between the abstracts and results section. These results allow us to predict that LSA can produce prototypical signatures that are able to differentiate between sections from the same articles, and those articles that are merely similar in theme.

Table 7.3. Pairwise comparison of abstract to similar-themed body

	Method (AM)	Results (AR)	Discussion (AD)
Introduction (AI)	Diff=.150 (.032)*	Diff=.135 (.032)*	Diff=-.015 (.020)
Method (AM)		Diff=-.015 (.026)	Diff=-.165 (.036)*
Results (AR)			Diff=.150 (.032)*

Notes: Diff denotes the average difference between the cosines; * $p<.01$

7.10 Experiment 2a

One potential weakness of Experiment 2 was the relatively small size of the corpus (i.e., 20 texts). To alleviate the concern that the results were a function of the size of text corpora, we split the original 67-text corpus from Experiment 1 into three random groups of 20 texts and re-analyzed the results. If 20 texts were a sufficiently sized corpus , then the analysis should yield the same pattern as observed in Experiment 1. This analysis produced three sets of scores for each of the four comparison types (AI, AM, AR, and AD). As can be seen from Figure 7.3, the three new signatures map almost perfectly to the original signature from Experiment 1. There were also no significant differences within the corresponding section comparisons from the three 20-text corpora.

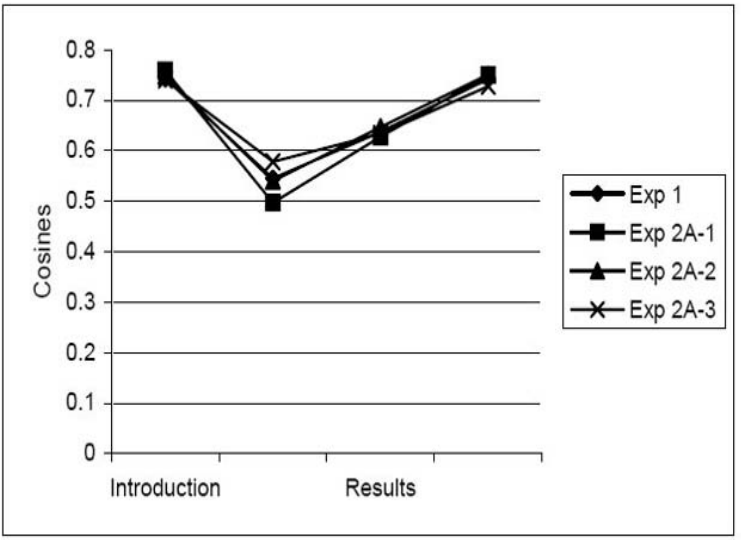


Fig. 7.3. Comparison of Experiment 1 to three sets of 20 texts taken from the original corpora.

7.11 Experiment 3

In Experiment 1, we showed that LSA may be able to provide a textual signature based on the relationships between the abstract of the paper and the sections within the paper. We will refer to this kind of signature as type *same paper* (SP). In Experiment 2 we showed that LSA can also produce prototypical signatures indicative of articles of a similar theme. We will refer to this kind of signature as type *same theme* (ST). In Experiment 3, we show that LSA based signatures can also indicate papers that are differently themed. We will refer to this kind of signature as type *different theme* (DT).

Based on the findings from Experiment 2, we predicted that the DT signature would more closely match that of Experiment 2. However, because the themes of Experiment 3's abstracts are different from the sections they were being compared to, we predicted the differences of the LSA scores for the AI and AD comparison types over the AM and AR comparison types would be less pronounced. To test this prediction, we randomly replaced the abstracts from Experiment 1 with the thematically consistent abstracts from Experiment 2.

To examine differences in relatedness of the abstract to each of the text sections for the DT corpus, we conducted a repeated measures ANOVA on the LSA cosines including the within-text factors of AI (*M=.289, SD=.138*), AM (*M*=.231, *SD*=.141), AR (*M*=.234, *SD*=.143), and AD (*M*=.298, *SD*=.150). There was a main effect of comparison type, $F(3,19) = 9.278$, $MSE=.002$, $p<.001$. Pairwise contrasts (see Table 7.4) indicated that the DT signature (like the ST signatures) resulted in the AR and AM comparisons being significantly different from the AI and AD comparison types. However, also like the ST signature, the AI comparisons did not significantly differ from the AD comparisons. Thus, despite the appearance of Figure 7.1 producing a lower cosine signature, we could not be sure from these results whether the DT corpus significantly differed from the ST corpus.

Table 7.4. Pairwise comparison of abstract to dissimilarly themed body

	Method (AM)	Results (AR)	Discussion (AD)
Introduction (AI)	Diff=.058 (.013)*	Diff=.055 (.014)*	Diff=-.002 (.013)
Method (AM)		Diff=-.003 (.015)	Diff=-.060 (.019)*
Results (AR)			Diff=-.056 (.017)*

Notes: Diff denotes the average difference between the cosines; * $p<.01$

To examine whether the SP, ST, and DT corpora were significantly different from one another, we ran mixed ANOVA, including the within-text factor of comparison type and the between-text factor of corpora. Because the SP corpus contained 67 papers, and the other corpora were comprised of 20 papers, we randomly selected a 20-paper corpus from Experiment 2a to represent the SP corpus.

As shown in the previous studies, there was a main effect of comparison type, $F(3,171) = 41.855$, $MSE=.007$, $p<.001$. There was also a main effect of corpus, $F(2,57) = 67.259$, $MSE=.013$, $p<.001$. A post hoc Bonferroni test between the corpora indicated that there was a significant difference between each of the corpora: SP

to ST ($p<.001$); SP to DT ($p<.001$); and ST to DT ($p<.05$). Most importantly, the interaction between corpus and section was significant, $F(6,171) = 4.196$, $MSE=.007$, $p<.001$, indicating that the differences between the sections depends on the type of corpora.

The results from this experiment indicate that the corpus using the same papers for the comparisons (SP) shows greater internal difference than do those with either similar or different themes (i.e., ST, DT). This result is largely due to the stronger AR comparison generated in the SP corpus. While the signatures generated from the ST and DT corpora are internally similar, the results of this experiment offer evidence that the degree of similarity between sections within the corpora is significantly different.

These results allowed us to extend our signature assumption to predicting that LSA can differentiate three text-types: the same paper, similarly themed papers, and differently themed papers.

7.12 Discussion

The results of this study suggest that LSA comparisons of textual sections can produce an identifiable textual signature. These signatures serve as a prototypical model of the text-type and are distinguishable from those produced by texts which are merely similar in theme (ST), or similar in field (DT).

Textual signatures of the type produced in this study have the potential to be used for a number of purposes. For example, students could assess how closely the signature of their papers reflected a prototypical signature. The discrepancies between the two LSA cosines may indicate to the student where information is lacking, redundant, or irrelevant. For researchers looking for supplemental material, the signatures method could be useful for identifying texts from the same field, texts of the same theme, and even the part of the text in which the researcher is interested. Related to this issue is a key element in Question Answering systems: as textual signatures stand to identify thematically related material, the retrieval stage of QA systems may be better able to rank its candidate answers.

Future research will focus on developing a range of textual signatures beyond the abstract comparisons outlined in this chapter. Specifically, comparisons of section parts from the perspective of the *introduction*, *methods*, *results*, and *discussions* sections need to be examined. This broader scope offers the possibility of greater accuracy in textual identification. For example, papers that were only thematically related would likely have higher overlaps generated from introduction sections than from other sections. Introductions feature a review of the literature which would likely be highly consistent across papers within the same theme, whereas the other sections (especially the results section) would likely be significantly different from paper to paper.

In addition to extending the perspectives of signatures, we also need to consider how other indices may help us to better identify textual signatures. Coh-Metrix generates a variety of alternative lexical similarity indices such as *stem*, *lemma*, and *word* overlap. While these indices do not compare semantic similarities such as *table/chair* or *intelligence/creativity* (as LSA does), they do compare lexical overlaps such as *produce/production*, *suggest/suggests* and *investigate/investigated*. Indices

such as these, and the signatures they generate, may come to form a web of soft constraints that could help us improve the confidence we have that a retrieved text or textual unit matches a target set of constraints.

If future research offers continued efficacious signatures then an array of indices can be imagined. Once achieved, a discriminant analysis between corpora such as the SP, ST, and DT outlined in this study could be conducted. Such testing would lend substantial support to a textual signatures approach to text identification.

Looking even further ahead, we would also like to extend our signatures research beyond the type of texts presented in this study. For example, we need to consider the signatures generated from articles with multiple experiments as well as articles, essays, and reports from other fields. It is reasonable to expect that any identifiable genre is composed of elements, and that those elements exposed to methods such as those used in this study will produce identifiable and therefore distinguishable signatures.

While a great deal of work remains to be done, we believe that LSA-based textual signatures contributes to the field by offering a useful and novel approach for computational research into text mining.

7.13 Acknowledgments

This research was supported by the Institute for Education Sciences (IES R3056020018-02). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the IES. We would also like to thank David Dufty and Mike Rowe for their contributions to this study.

References

1. Best, R.M., Floyd, R.G., & McNamra, D.S. (2004). Understanding the fourth-grade slump: Comprehension difficulties as a function of reader aptitudes and text genre. Paper presented at the 85th Annual Meeting of the American Educational Research Association.
2. Biber, D. (1987). A textual comparison of British and American writing. American Speech, 62, 99-119.
3. Biber, D. (1988). Linguistic features: algorithms and functions in variation across speech and writing. Cambridge: Cambridge University Press.
4. Brill, E. (1995). Unsupervised learning of disambiguation rules for part of speech tagging. In Proceedings of the Third Workshop on Very Large Corpora, Cambridge, MA.
5. Britton, B. K., & Gulgoz, S. (1991). Using Kintschs computational model to improve instructional text: Effects of inference calls on recall and cognitive structures. Journal of Educational Psychology, 83, 329-345
6. Burrows, J. (1987). Word-patterns and story-shapes: The statistical analysis of narrative style. Literary and Linguistic Computing, 2, 6170.
7. Charniak, E. (1997) Statistical Parsing with a context-free grammar and word statistics Proceedings of the Fourteenth National Conference on Artificial Intelligence, Menlo Park: AAAI/MIT Press

8. Charniak, E. (2000) A Maximum-Entropy-Inspired Parser. Proceedings of the North-American Chapter of Association for Computational Linguistics, Seattle, WA
9. Charniak, E. & Johnson, M. (2005) Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (pp. 173-180). Ann Arbor, MI
10. Collins, M. (1996) A New Statistical Parser Based on Bigram Lexical Dependencies. Proceedings of the 34th Annual Meeting of the ACL, Santa Cruz, CA
11. Collins, M. (1997) Three Generative, Lexicalised Models for Statistical Parsing Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics, Madrid, Spain.
12. Crossley, S., Louwerse, M.M., McCarthy, P.M., & McNamara, D.S. (forthcoming 2007). A linguistic analysis of simplified and authentic texts. Modern Language Journal, 91, (2).
13. Dennis, S., Landauer, T., Kintsch, W. & Quesada, J. (2003). Introduction to Latent Semantic Analysis. Slides from the tutorial given at the 25th Annual Meeting of the Cognitive Science Society, Boston.
14. Duran, N., McCarthy, P.M., Graesser, A.C., McNamara, D.S., (2006). An empirical study of temporal indices. Proceedings of the 28th annual conference of the Cognitive Science Society, 2006.
15. Foltz, P. W., Britt, M. A., & Perfetti, C. A. (1996). Reasoning from multiple texts: An automatic analysis of readers' situation models. In G. W. Cottrell (Ed.) Proceedings of the 18th Annual Cognitive Science Conference (pp. 110-115). Lawrence Erlbaum, NJ.
16. Foltz, P. W., Kintsch, W., & Landauer, T. K. (1998). The measurement of textual Coherence with Latent Semantic Analysis. Discourse Processes, 25, 285-307.
17. Foltz, P. W., Gilliam, S., & Kendall, S. (2000). Supporting content-based feedback in on-line writing evaluation with LSA. Interactive Learning Environments, 8, 111-127.
18. Gernsbacher, M.A. (1990). Language comprehension as structure building. Hillsdale, NJ: Erlbaum.
19. Givn, T. (1995). Coherence in the text and coherence in the mind. In Gernsbacher, M.A. & Givn, T., Coherence in spontaneous text. (pp. 59-115). Amsterdam/Philadelphia, John Benjamins.
20. Graesser, A.C. (1993). Inference generation during text comprehension. Discourse Processes, 16, 1-2.
21. Graesser, A.C., Singer, M., & Trabasso, T. (1994). Constructing inferences during narrative text comprehension. Psychological Review, 101, 371-95.
22. Graesser, A.C., McNamara, D., Louwerse, M., & Cai, Z. (2004). Coh-Metrix: CohMetrix: Analysis of text on cohesion and language. Behavioral Research Methods, Instruments, and Computers, 36, 193-202.
23. Hearst, M.A. (1994) Multi-paragraph Segmentation of Expository Text. Proceedings of the Association of Computational Linguistics, Las Cruces, NM.
24. Hobbs, J.R. (1985). On the coherence and structure of discourse. CSLI Technical Report, 85-37. Stanford, CA.
25. Hovy, E. (1990). Parsimonious and profligate approaches to the question of discourse structure relations. Proceedings of the Fifth International Workshop on Natural Language generation, East Stroudsburg, PA, Association for Computational Linguistics.

26. Karlsgren J. & Cutting, D. (1994). Recognizing text genres with simple metrics using discriminant analysis. International Conference on Computational Linguistics Proceedings of the 15th conference on Computational linguistics - Volume 2 (pp. 1071-1075). Kyoto, Japan.
27. Kessler, Nunberg, G., & Schutze, H. (1997). Automatic detection of text genre. In Proceedings of 35th Annual Meeting of Association for Computational Linguistics, and in 8th Conference of European Chapter of Association for Computational Linguistics (pp. 32-38). Madrid, Spain.
28. Kintsch, W. & Bowles, A. (2002) Metaphor comprehension: What makes a metaphor difficult to understand? Metaphor and Symbol, 2002, 17, 249-262.
29. Kintsch, E., Steinhart, D., Stahl, G., LSA Research Group, Matthews, C., & Lamb, R. (2000). Developing summarization skills through the use of LSA-based feedback. Interactive Learning Environments 8, 87-109.
30. Kintsch, W., & van Dijk, T.A. (1978). Toward a model of text comprehension and production. Psychological Review, 85, 363-394.
31. Labov, W. (1972). The Transformation of Experience in Narrative Syntax, In W. Labov (ed.), Language in the Inner City, 1972, University of Pennsylvania Press, Philadelphia.
32. Landauer, T. K., & Dumais, S. T. (1997). A solution to Plato's problem: The Latent Semantic Analysis theory of the acquisition, induction, and representation of knowledge. Psychological Review, 104, 211-240.
33. Landauer, T. K., Foltz, P. W., & Laham, D. (1998). Introduction to Latent Semantic Analysis. Discourse Processes, 25, 259-284.
34. Lehman, S., & Schraw, G. (2002). Effects of coherence and relevance on shallow and deep text processing. Journal of Educational Psychology, 94, 738-750.
35. Linderholm, T., Everson, M.G., van den Broek, Mischinski, M., Crittenden, A., & Samuels, J. (2000). Effects of causal text revisions on more and less skilled readers comprehension of easy and difficult text. Cognition and Instruction, 18, 525-556.
36. Louwerse, M.M. (2002). Computational retrieval of themes. In M.M. Louwerse & W. van Peer (Eds.), Thematics: Interdisciplinary Studies (pp. 189-212). Amsterdam/Philadelphia: John Benjamins.
37. Louwerse, M. M., McCarthy, P. M., McNamara, D. S., & Graesser, A. C. (2004). Variation in language and cohesion across written and spoken registers. In K. Forbus, D. Gentner, & T. Regier (Eds.), Proceedings of the 26th Annual Meeting of the Cognitive Science Society (pp. 843-848). Mahwah, NJ: Erlbaum.
38. Loxterman, J.A., Beck, I. L., & McKeown, M.G. (1994). The effects of thinking aloud during reading on students' comprehension of more or less coherent text. Reading Research Quarterly, 29, 353-367.
39. Mani, I. & Pustejovsky, J. (2004). Temporal discourse markers for narrative structures.ACL Workshop on Discourse Annotation, Barcelona, Spain. East Stoudsburg, PA, Association for Computational Linguistics.
40. Mann, W. C. & Thompson, S. A. (1988). Rhetorical Structure Theory: Toward a functional theory of text organization. Text, 8 (3). 243-281
41. McCarthy, P.M., Lightman, E.J., Dufty, D.F. & McNamara (in press). Using Coh-Metrix to assess distributions of cohesion and difficulty in high-school textbooks. Proceedings of the 28th annual conference of the Cognitive Science Society.

42. McCarthy, P.M., Lewis, G.A., Dufty, D.F., & McNamara, D.S. (2006). Analyzing Writing Styles with Coh-Metrix. 19th International FLAIRS Conference 2006.
43. McNamara, D.S., Kintsch, E., Songer, N.B., & Kintsch, W. (1996). Are good texts always better? Text coherence, background knowledge, and levels of understanding in learning from text. Cognition and Instruction, 14, 1-43.
44. McNamara, D. S. (2001). Reading both high and low coherence texts: Effects of text sequence and prior knowledge. Canadian Journal of Experimental Psychology, 55, 51-62.
45. Moldovan, D., Harabagiu, S., Pasca, M., Mihalcea, R., Girju, R., Goodrum, R., & Rus, V. (2000): The Structure and Performance of an Open-Domain Question Answering System, in Proceedings of ACL 2000, Hong Kong, October
46. Morris, J., Hirst, G. (1991) Lexical cohesion computed by thesaural relations as an indicator of the structure of text, Computational Linquistics, 17, 21-48.
47. Ozuru, Y., Dempsey, K., Sayroo, J., & McNamara, D. S. (2005). Effects of text cohesion on comprehension of biology texts. Proceedings of the 27th Annual Meeting of the Cognitive Science Society (pp. 1696-1701). Hillsdale, NJ: Erlbaum.
48. Propp, V. (1968). Morphology of the folk tale. Baltimore: Port City Press, pp 19-65.
49. Ratnaparkhi, A. (1996), A maximum entropy model for part-of-speech tagging. Proceedings of Conference on Empirical Methods in Natural Language Processing, University of Pennsylvania.
50. Stamatatos, E., Fakotatos, N., & Kokkinakis, G. (2001). Computer-based authorship attribution without lexical measures. Computers and the Humanities, 35, 193-214.
51. Trabasso, T., & van den Broek, P. (1985). Causal thinking and the representation of narrative events. Journal of Memory and Language, 24, 612-630.
52. Voorhees, E. M. & Tice, D.M. (2000). Building a question answering test collection. Proceedings of the Twenty-Third Annual International ACM SIGIR Conference on Research and Development in Information Retrieval
53. Wolfe, M. B., Schreiner, M. E., Rehder, B., Laham, D., Foltz, P. W., Kintsch, W., & Landauer, T. K. (1998). Learning from text: Matching readers and text by Latent Semantic Analysis. Discourse Processes, 25, 309-336.
54. Wolfe, M. B.W., & Goldman S.R. (2003). Use of latent semantic analysis for predicting psychological phenomena: Two issues and proposed solutions. Behavior Research Methods, Instruments, & Computers, 35, 22-31.
55. Zwaan, R.A.(1996). Processing narrative time shifts. Journal of Experimental Psychology: Learning, Memory, and Cognition, 22, 1196-1207.
56. Zwaan, R.A. & Radvansky, G.A. (1998). Situation models in language comprehension and Memory. Psychological Bulletin, 123, 162-185.

Appendix (Journals Analyzed)

Table 7.5.

Journal Name	Articles	Publication Date Range
Acta Psychologica	11	2000-2004
Biological Psychology	14	2000-2004
Cognition	4	2000-2001
Intelligence	20	2000-2003
Journal of Applied Psychotherapy Research	17	2002-2003

8

Automatic Document Separation: A Combination of Probabilistic Classification and Finite-State Sequence Modeling

Mauritius A. R. Schmidtler, and Jan W. Amtrup

8.1 Introduction

Large organizations are increasingly confronted with the problem of capturing, processing, and archiving large amounts of data. For several reasons, the problem is especially cumbersome in the case where data is stored on paper. First, the weight, volume, and relative fragility of paper incur problems in handling and require specific, labor-intensive processes to be applied. Second, for automatic processing, the information contained on the pages must be digitized, performing Optical Character Recognition (OCR). This leads to a certain number of errors in the data retrieved from paper. Third, the identities of individual documents become blurred. In a stack of paper, the boundaries between documents are lost, or at least obscured to a large degree.[1]

As an example, consider the processing of loan documents in the mortgage industry: Usually, documents originate at local branch offices of an organization (e.g., bank branches, when a customer fills out and signs the necessary forms and provides additional information). All loan documents finalized at a local office on a given day are collated into one stack of paper (called a *batch*) and sent via surface mail to a centralized processing facility. At that facility, the arriving packets from all over the country are opened and the batches are scanned. In order to define the boundaries and identities of documents, *separator sheets* are manually inserted in the batches. Separator sheets are special pages that carry a barcode identifying the specific loan document that follows the sheet, e.g., Final Loan Application or Tax Form, etc. The separation and identification of the documents is necessary for archival and future retrieval of specific documents. It is also a precondition for further processing, for instance in order to facilitate the extraction of certain key information, e.g., the loan number, property address and the like.

The problem we are addressing in this chapter is the process of manually inserting separator sheets into loan files. A person must take a loan file, leaf through the stack of paper (hundreds of pages), and insert appropriate separators at the correct boundary points. This work is both tedious and challenging. It is tedious, since no important new information is created, but only information that previously

[1] Notwithstanding physical markers such as staples, etc. Those are usually removed as a first step in document processing.

existed is re-created. It is challenging, since the person needs to have a fair amount of knowledge of loan documents (hundreds of document categories) and work with a high degree of attention to detail. Nevertheless, the error rate for this process can be as high as 8%. The cost for the insertion is also significant, both in terms of labor and material; it is estimated that 50% of the document preparation cost is used for sorting and the insertion of separator sheets. One customer estimates the printing cost for separator sheets alone to be in excess of $1M per year.

In the automated solution presented here [1], the loan files still need to be collected and shipped to a central facility for processing.[2] At the facility, the batches are scanned in their entirety, without inserting separator sheets beforehand. The result of this process is a long sequence of images of pages, up to 2000 images per batch. Next, the text on each page is read by an OCR engine. A classification engine (see Section 8.4.2) determines the likely document types of loan documents (e.g., Appraisals, Tax Forms, etc.), and a separation mechanism (see Section 8.4.3) inserts virtual boundaries between pages to indicate where one document ends and the next one begins. The separated documents are then labeled accordingly and delivered for further processing, e.g., the extraction of relevant information.

8.2 Related Work

Traditionally, the processing of scanned paper forms has concentrated on the handling of structured forms. These are paper documents that have well-defined physical areas in which to insert information, such as the social security number and income information on tax forms. Ideally, for these forms the separation problem does not even arise, since the documents are of a specified length. If, however, a sequence of documents needs to be separated, it is usually enough to concentrate a recognition process on the first page to find out which document is present. This information defines the number of pages in the document and thus the separation information with certainty. The recognition process is often done indirectly, coupled with a subsequent extraction system. Extraction rules define areas of interest on a form and how to gather data from those *zones*. For instance, an extraction rule for a tax form could first specify how to identify a box on the top left corner of the document that contains the text "1040." Then the rule would search down the form to find a rectangular box labeled "SSN" and extract the nine digits contained in a grid directly to the right of the label. If this recognition rule succeeds, i.e., text can be found and recognized with sufficient confidence, the document is identified as a two-page 1040 tax form and the social security number is extracted.

While such local, forms-based rules work extremely well in their area of application, the extension of this approach to less structured forms or even forms that exist in a large number of variations is highly effort-intensive and error-prone. For instance, the example above treated federal tax forms, of which there are only a few varieties. However, there are at least fifty varieties of state tax forms, and defining

[2] We do not discuss distributed scanning operations here. The principle in this case is that no paper documents are ever shipped, but that each local office scans the documents that are created locally. The images of the documents are then transferred to a central facility. This operational schema presents some of the same and some additional complications.

the form and contents of every such form is a major undertaking. But even then, the layout of the forms is known and can, in principle, be described in advance. For other semi-structured forms, this is not the case. For instance, appraisals (as in the case of mortgage loan applications) always contain roughly the same type of information (property address, value, comparable objects in the vicinity of the property in question, etc.). However, recognizing an appraisal based on very local information about specific structural properties of the form is extremely difficult. The layout of appraisals from different sources can not be foreseen. As such, the search for specific items on a page, using this information as an indication of what form is present and, more importantly for the case discussed here, the length of the document, are highly uncertain.

This is even more pronounced for so-called unstructured forms which have no specific layout considerations. Those also appear in concrete business cases, such as legal documents, waivers, riders, etc. Here, a layout-based definition of forms is highly unlikely to succeed.

The conclusion is that for a large variety of important documents, a rule-driven layout-based recognition is possibly inferior to a content-based recognition, as is used in the present solution. This is still true if a subsequent extraction step is used to gather information from the documents. Distinguishing between the separation step and the extraction step can facilitate the process of writing rules for information extraction, since the identity of documents can now be taken for granted.[3]

The cost of maintaining a solution for separation (and extraction) also needs to be considered, since it is highly likely that the layout of forms changes over time. Except in very specific circumstances, the extent and form of the change is out of control of the maintainer of a separation solution. This entails monitoring the incoming forms for such changes and the rules governing recognition must be modified immediately once a change is observed.

From the preceding discussion, it seems to us that treating separation and extraction as two distinct steps is advantageous. Furthermore, we favor content-based and example-based methods over manually written layout rules. The exact form of features used (e.g., image-based or text-based) is unspecified in principle. However, based on the experiences in our application domain, we prefer text-based features (see below).

The most direct approach to document separation would treat the task as a straightforward segmentation problem. Maximum Entropy (ME) methods have proven very successful in the area of segmentation of natural language sentences [2, 3]. Each boundary (in our case the point between two pages) is characterized by features of its environment (e.g., by the words used on the preceding and following page). An ME classifier is then used to solve the binary problem (boundary/non-boundary) for new, unseen page transitions. We are unaware of any publication using this approach for automatic document separation.

Instead of looking for boundaries, one could also attempt to ascertain that two consecutive pages belong in the same document, thus indirectly establishing borders.

[3] Note that this still assumes that rules are used to identify local information on a page. It may also be possible to handle the extraction step in a content-based manner, focusing not on the layout of a page, but on the words on it. The respective merits of each of these method is beyond the scope of the present chapter.

An instance of this method is described in [4]. They define a similarity measure between two pages that takes document structure (text in headers and footers, esp. page numbers), layout information (font structure), and content (text on pages) into account. They use a single-linkage agglomerative clustering method to group pages together. The clustering process is bounded by manually set thresholds. They report a maximum separation accuracy of 95.68%, using a metric from [5] that measures the correctness of the number of separation points between non-adjacent pages. Since our data is different and we solve a combined problem of classification and separation ([4] only perform separation), their results cannot directly be compared to ours.

8.3 Data Preparation

The input to our separation solution is the text delivered by an OCR engine of scanned page images. We are primarily reporting on data from the mortgage processing industry, hence the document types (Appraisal, Truth in Lending, etc.). Our sample here contains documents from 30 document types. The quality of the images varies based on their origin (original or photocopy) and treatment (fax). Figures 8.1 and 8.2 show two sample images (one from a Loan Application, one from a Note) and some of the OCR text generated from them.

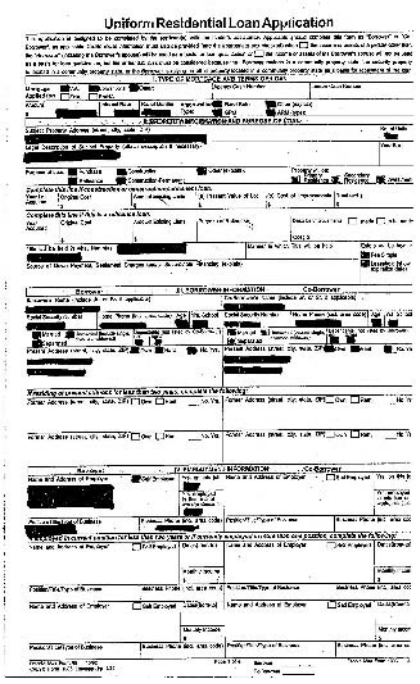
In order to be prepared for the core classification algorithms (see below), the input text is tokenized and stemmed. Tokenization uses a simple regular expression model that also eliminates all special characters. Stemming for English is based on the Porter algorithm.[6][4]

The stream of stemmed tokens isolated from a scanned image is then converted into a feature vector. We are using a *bag of words* model of text representation; each token type is represented by a single feature and the value of that feature is the number of occurences of the token on the page. In addition, the text is filtered using a stopword list. This filtering removes words that are very common in a language; for instance, in English the list includes all closed-class words such as "the," "a," "in," "he," etc. Table 8.1 shows some of the features and their values extracted from a Note. The entries in the table indicate the processing that the text underwent.

Note that these three processes introduce two significant abstractions over the input text:

- By stemming, we assume that the detailed morphological description of words is irrelevant for the purpose of classification. For instance, we are unable to tell whether the feature "address" in Table 8.1 came from the input "address," "addresses," or "addressing." Inflectional and part-of-speech information is lost.
- Using bags of words, we are abstracting from the linear structure of the input text. We pose that there is little value in knowing which word appeared before or near another and the only important information is in knowing which word appears more frequently than others.
- The application of a stopword list, finally, de-emphasizes the value of syntactic information even further, since many syntactically disambiguating words are ignored.

[4] We only apply stemming for English text. Text in other languages is used without morphological processing.

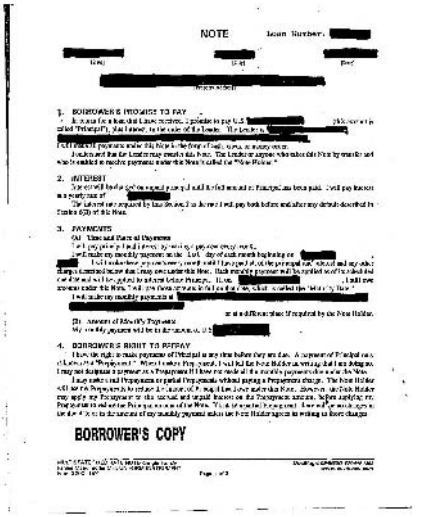


Uniform Residential Loan Application I TYPE GF'MORTGAGE-AHD TERMS OF LOAN Mortgage flvAB—Ccnvniltiai Applied fof: E¿HA Agency Case Number Lender Case Number Amount 5 I No. of Months Amortization Fixed Rate Typo: I Other (explain): I ARM (type): / LOAN Subject Property Addresi (street, city, state, ZIP) Legal Descripllon of Subject Property (attach description If necessary) No. of Units Year Built Purpose of Loan Construction Construction-Permanent Otner (explain): Property will be: Primary " "an. Compl&te this line If construction or construction-permanent loan. Secondary Investment Year Lot Acquired Original Cost S Amount Exjsling Uens $ (a) Present Value of Lot $ (b) Cost o(Improvements $ Total (a+b) S Complete this line if this Is a ra/fiuncfl loan. Year Acquired Original Cost Amount Existing Uens Title will be held in what Name(s) Purpose of Refinance Describe ImprovamanU

Fig. 8.1. Image and OCR text from a sample loan application. While the forms themselves are authentic, we redacted the information contained on them to ensure privacy.

It has been shown [7] that, for certain classifiers and texts, these abstractions do not reduce accuracy. In addition, abstraction reduces the number of parameters that need to be estimated during training, which in turn reduces the number of training samples that need to be provided. This aspect is of particular importance for us. In order to deploy a separation solution, customers must prepare a certain number of samples for each document type. Given the classification technology outlined in section 8.4.2, we achieve acceptable results with as little as twenty to thirty examples per document type. If we were using a classifier that takes word sequence information into account, for instance a Bayesian classifier over word n-grams, we would need hundreds of samples per document type; this would pose a severe entrance barrier for customers.[5]

[5] However, for certain types of problems, sequence-aware modeling is superior and even necessary. In one deployment, we encountered a fixed form with two broad columns into which data could be entered. Depending on whether only one column or both were filled out, the documents were categorized as different types. The classification model had difficulties distinguishing between these two document types. In an experiment, we collected enough sample data to train a word n-gram classifier and were then able to reliably assess the correct type.


NOTE Loan Number: [redacted]

[redacted]

1. BORROWER'S PROMISE TO PAY

[redacted]

2. INTEREST

[redacted]

3. PAYMENTS

(A) Time and Place of Payments

[redacted]

(B) Amount of My Monthly Payments

[redacted]

4. BORROWER'S RIGHT TO PREPAY

[redacted]

BORROWER'S COPY

4. BORROWER'S RIGHT TO PREPAY

I have the right to make payments of Principal at any time before they are due. A payment of Principal only is known as a "Prepayment." When I make a Prepayment, I will tell the Note Holder in writing that I am doing so. I may not designate a payment as a Prepayment if 1 have not made all the monthly payments due under the Note. I may make a full Prepayment or partial Prepayments without paying a Prepayment charge. The Note Holder will use my Prepayments to reduce the amount of Principal that I owe under this Note. However, the Note Holder may apply my Prepayment to the accrued and unpaid interest on the Prepayment amount, before applying my Prepayment to reduce the Principal amount of the Note. If I make a partial Prepayment, there willpe no changes in the due date or in the amount of my monthly payment unless the Note Holder agrees in writing to those changes.

Fig. 8.2. Image and OCR text from a sample Note

Table 8.1. Some of the features extracted from a Note

Token	#Occur	Token	#Occur	Token	#Occur
accru	1	chang	2	fanni	1
acm	1	charg	3	fix	1
address	1	check	1	form	3
agre	1	citi	1	freddi	1
ani	3	compani	1	ftill	1
anyon	1	date	5	holder	6
appli	4	day	1	home	1
august	1	dbs	1	howev	1
befor	4	default	1	initi	1
begin	1	describ	2	ink	1
borrow	2	design	1	instrument	1
bowi	1	differ	1	interest	9
box	1	entitl	1	juli	1
burtonsvil	1	everi	2	known	1
cash	1	famili	1	la	1

From the examples in Figures 8.1 and 8.2, it can be seen that the OCR process introduces a significant degree of noise into the textual data that all further processes are operating on. We have not undertaken experiments specifically designed to evaluate the degradation in accuracy of either classification or separation that this OCR noise induces. Such experiments could be set up to work from cleaned-up OCR text. Since this implies a large amount of manual labor, the change in document quality could be simulated by printing electronic documents and manipulating the pages, e.g., by copying them repeatedly.

Table 8.2. Some of the features related to the stem *borrow*

Token	#Occur	Token	#Occur	Token	#Occur
borrnu	1	borronv	4	borrovv	8
borrnwer	1	borrotr	1	borrovvcr	1
borro	92	borrou	3	borrovvef	1
borroa	1	borrov	14	borrovvei	1
borroaer	1	borrovc	1	borrovvfir	1
borroh	4	borrovcf	1	borrovvi	1
borroi	1	borrovd	1	borrovw	1
borroifril	1	borrovi	3		
borrojv	1	borrovj	3		
borrokbr	1	borrovjar	1		
borrom	2	borrovl	1		
borromad	1	borrovrti	1		
borromicrl	1	borrovt	1		
borromr	1	borrovti	1		
borron	1	borrovu	1		

OCR noise also affects the size of training sets negatively. Under the bag-of-words model, the text for each page is converted into a feature vector with a dimensionality equal to the number of distinct words (or stems) in the training corpus. Noise introduced during OCR multiplies this number by generating many seemingly distinct, spurious words. Table 8.2 shows a small number of features related to the stem "borrow."

For some data sets, the number of OCR-induced variations becomes so high that the size of the training set exceeds reasonable memory sizes (e.g., > 2 GB). In those cases, we apply a preliminary feature selection step that removes features with low occurrence counts until we arrive at a small enough feature set. In general, though, we prefer to keep all features available for the classification mechanism and not perform any initial feature selection. Only in cases when size or performance require it, we apply feature selection to reduce the size of the feature set. We use basic frequency filtering and information-gain or mutual information as selection means.

8.4 Document Separation as a Sequence Mapping Problem

Automatic Document Separation adds two pieces of information to a stream of unlabeled pages. It inserts boundaries, so that documents are kept together, and it assigns labels to those documents that indicate their type. The problem can be seen as the mapping of an input sequence to an output sequence, i.e., a sequence of scanned paper pages is mapped to a sequence of document types. The mapping of sequences is a well known problem in Computer Science and there exist many different applications. For example, compilers, speech recognition, information extraction, and machine translation are all instances that have some aspect that deals with the problem of sequence mapping: A sequence of human readable program statements to a sequence of machine code, a sequence of acoustic signals to a sequence of words, a sequence of words to a sequence of tags, and a sequence of, e.g., Spanish words to a sequence of French words.

In addition, probabilistic models are often employed in order to determine the probabilities of possible output sequences given a particular input sequence. In this chapter, we utilize these concepts to solve the problem of document separation: Map a given sequence of pages to all possible output sequences, i.e., sequences of document types, determine for each output sequence its probability given the input sequence, find the most likely output sequence (sequence of document types), and, thus, effectively separate the sequence of input pages by document type.

8.4.1 Sequence Model

Formally, the procedure described above can be modeled as a Markov chain. Denoting the input sequence of ordered pages[6] p^c by $\mathcal{P} = (p_1^c, \ldots, p_n^c)$ and the output sequence of document types by $\mathcal{D} = (d_1, \ldots, d_n)$, the probability of a specific sequence of document types $\mathcal{D}$ given the input sequence of pages can be written as

$$p(\mathcal{D}|\mathcal{P}) = \prod_{j=1}^{n} p(d_j|\mathcal{D}_{j-1}, \mathcal{P}) \,, \tag{8.1}$$

where d_j denotes the document type of the j-th page and $\mathcal{D}_{j-1}$ the output sequence of document types up to the $(j-1)$-th page. In many practical applications, independence assumptions regarding the different events d_j, $\mathcal{D}_{j-1}$, and $\mathcal{P}$ hold at some level of accuracy and allow estimations of the probability $p(\mathcal{D}|\mathcal{P})$ that are efficient yet accurate enough for the given purpose.

We started by assuming that the document type d_j at time step j only depends on the page content p_j^c at time step j and gradually increased the complexity of the models by taking into account the document types of previous time steps. In particular, we considered

$$p(\mathcal{D}|\mathcal{P}) \approx \prod_{j=1}^{n} p(d_j|p_j^c) \tag{8.2}$$

as well as the following approximation, which is very common and has been widely used in several fields, e.g., for information extraction [8],

[6] The superscript c indicates that the *content* of the pages is considered.

$$p(\mathcal{D}|\mathcal{P}) \approx \prod_{j=1}^{n} p(d_j|d_{j-1}, p_j^c) \tag{8.3}$$

and finally

$$p(\mathcal{D}|\mathcal{P}) \approx \prod_{j=1}^{n} p(d_j|d_{j-1}, d_{j-2}, p_j^c)\,. \tag{8.4}$$

Instead of trying to approximate the probality of $p(\mathcal{D}|\mathcal{P})$ ever more accurately by relaxing the independence assumptions one also can describe pages in more detail by breaking up the document types based on the page position within a document. Functionally, this is achieved by altering the output language. In the extreme, this would lead to a model of the data in which the symbols of the output language are different for each page number within the document. You would have symbols like *TaxForm*$_1$, *TaxForm*$_2$, *TaxForm*$_3$, etc., for the different page numbers within a tax form. Here, we increased the alphabet of the original output language threefold. Every document type symbol is split into three symbols: *Start*, *middle*, and *end* page of the document type. In our experience, forms often have distinctive first and last pages, e.g., forms ending with signature pages and starting with pages identifying the form, whereas middle pages of forms do not contain as much discriminating information. Accordingly, the sequences of the new output language are now sequences of the type $\mathcal{D}'$, where $\mathcal{D}'$ is given by $\mathcal{D}' = (d'_1, \ldots, d'_n)$ with d'_j denoting the document type as well as the page type. The definitions of the page type events $\{start, middle, end\}$ are:

$$\begin{aligned} start:&\ \{p_{j,t}^c | t = 1, t \leq l\} \\ middle:&\ \{p_{j,t}^c | t > 1, t < l\} \\ end:&\ \{p_{j,t}^c | t > 1, t = l\} \end{aligned} \tag{8.5}$$

where j is the global page number within the batch and t is the local page number within a document of length l.

One of the models considered using the new output language is

$$p(\mathcal{D}'|\mathcal{P}) \approx \prod_{j=1}^{n} p(d'_j|p_j^c)\,, \tag{8.6}$$

under the constraint that the sequence of page types is consistent with the definitions given by Eq. 8.5, e.g., every document has to end with the *end* page type with the exception of one-page documents. The last model has, owing to this constraint, many similarities with the model given by Eq. 8.4. The main difference between the two models is that the model of Eq. 8.4 determines boundaries between documents based on the previous document types, whereas the model of Eq. 8.6 relies mainly on the difference of *start*, *middle*, and *end* pages within the document type to identify boundaries. Accordingly, the model of Eq. 8.6 can separate subsequent instances of the same document type, whereas the model of Eq. 8.4 cannot.

Finally, we also tested models that conditioned the output symbol at a given time step not only on the content of the current page but also on the previous and the next page

$$p(\mathcal{D}|\mathcal{P}) \approx \prod_{j=1}^{n} p(d_j|d_{j-1}, d_{j-2}, p^c_{j-1}, p^c_j, p^c_{j+1}) \tag{8.7}$$

$$p(\mathcal{D}|\mathcal{P}) \approx \prod_{j=1}^{n} p(d'_j|p^c_{j-1}, p^c_j, p^c_{j+1})\,, \tag{8.8}$$

where the model given by Eq. 8.8 has the same constrained output language as the model of Eq. 8.6, i.e., an output language consistent with the definitions of the events $\{start, middle, end\}$ given by Eq. 8.5.

8.4.2 Sequence Model Estimation

The problem of determining the different sequence models introduced in the previous section is given by estimating a probability of the form $p(x|p^c, y)$ with e.g., x denoting a document type and y a history of document types. As outlined in Section 8.3, a *bag of words* model is used for the page content[7] p^c, i.e., $p^c = \{(c_1, w_1), \ldots, (c_n, w_n)\}$ with c_j denoting the number of occurences of word w_j on the page, yielding

$$p(x|p^c, y) = p(p^c|x, y)\frac{p(x, y)}{p(p^c, y)} \propto \prod_{j=1}^{n} p(w_j|x, y)^{c_j} p(x, y)\,, \tag{8.9}$$

whereby in the last step the constant factor $1/p(p^c, y)$ has been omitted. As can be seen from Eq. 8.9, the sequence model estimation is reduced to the determination of the probabilities $p(w_j|x, y)$ and $p(x, y)$. These probabilities are estimated empirically by using sample documents (training examples) for the various events (x, y). For a typical training corpus , provided by the customer, the statistics for determining the word probabilities $p(w|x, y)$ are very low.[8] Given such statistics, overfitting to the training data is a common problem. Smoothing techniques, like those developed for language modeling [9], are a common tool to address the problem of low statistics by reserving some probability mass for unobserved events. In the case of determining the conditioned word probabilities $p(w|x, y)$, words that have been observed in the training data would be assigned lower probabilites than the maximum likelihood estimates, whereas unobserved words would be assigned higher probabilities than their maximum likelihood estimates. Statistical learning methods, e.g., [10, 11], utilizing methods of regularization theory, allow us to determine the tradeoff between memorization and generalization more principled than the smoothing techniques mentioned above. The learning method adopted here for estimating the sequence model is a Support Vector Machine[10] (SVM). It is commonly known that Support Vector Machines are well suited for text applications given a small number of training examples [12]. This is an important aspect for the commercial use of the system, since the process of gathering, preparing, and cleaning up training examples is time consuming and expensive.

[7] Here, p^c indicates both content models we are considering: The page content at a given time step as well as the content of the pages $p^c_{j-1}, p^c_j, p^c_{j+1}$ at a time step j.

[8] For a typical training corpus, almost all words occur rarely with words counts of one to two.

Table 8.3. Classification Results

	Optimized			Not optimized		
	precision	recall	F1-value	precision	recall	F1-value
Micro averages	**0.95**	**0.95**	**0.95**	**0.90**	**0.90**	**0.90**
Macro averages	**0.94**	**0.86**	**0.87**	**0.82**	**0.79**	**0.78**

Support Vector Machines solve a binary classification problem. The SVM score associated with an instance of the considered events is its signed distance to the separating hyperplane in units of the SVM margin. In order to solve multiclass problems, a series of Support Vector Machines have to be trained, e.g., in the case of a one-vs-all training schema, the number of SVMs trained is given by the number of classes. The scores between these different machines are not directly comparable and the scores must be calibrated such that at least for a given classification instance the scores are on an equal scale. In this application, the scores not only must be comparable between classes for a given classification instance (page), but also between different classification instances (pages), i.e., the SVM scores must be mapped to probabilities. Platt [13] uses SVM scores that are calibrated to class membership probabilities by adopting the interpretation of the score being proportional to the logarithmic ratio of class membership probability. He determines the class membership probability as a funcion of the SVM score by fitting a sigmoid function to the empirically observed class membership probabilities as a function of the SVM score. The fit parameters are the slope of the sigmoid function and/or a translational offset. The latter parameter, given the interpretation of the SVM scores discussed above, is the logarithmic ratio of the class prior probabilities. The method used here [14] fixes the translational offset and only fits the slope parameter. In addition, the Support Vector Machines are trained using cost factors for the positive as well as for the negative class and optimize the two costs independently. Empirical studies performed by the authors showed that cost factor optimization in conjunction with fitting the slope parameter of the mapping function from SVM scores to probabilities yields superior probability estimates than fitting the slope and the translational offset without cost factor optimization, fitting the slope and the translational offset with cost factor optimization, and fitting the slope only.

Table 8.3 summarizes the classification results for different loan forms. The results shown in the *Optimized* heading are the classification results obtained with the class membership probabilities using cost factor optimization and fitting the slope of the sigmoid function. Using SVM scores directly without calibration and cost factor optimization yields the results under the heading *Not Optimized*. The macro averages, especially, illustrate the effectiveness of the elected method. The observed improvement is a combined effect of using probabilities instead of SVM scores and cost factor optimization. An added benefit of optimizing the positive and negative cost factors is an improved handling of the OCR noise. As discussed in section 8.3, OCR increases the feature space considerably and cost factor optimization becomes important in order to avoid overfitting to the training corpus.

In summary, the effects of cost factor optimization can be interpreted as follows: The ratio of positive to negative cost factors determines the right class prior prob-

States				
Appraisal	0.9	0.5	0.5	0.8
TaxForm	0.7	0.9	0.9	0.8
...				
Note	0.5	0.7	0.7	0.3
				time
	page 1	page 2	page 3	page 4

Fig. 8.3. A trellis for the model of Eq. 8.2.

ability and thus enables an effective mapping of SVM scores to probabilities. The absolute value of the cost factor is an estimate of the optimal tradeoff between memorization and generalization and thus, enables an efficient handling of the noisy data. This together with mapping the scores to probabilities allows us to effectively utilize Support Vector Machines with their superior learning paradigm for the estimation of the sequence models.

8.4.3 Sequence Processing

In the previous two sections, we outlined the different probability models that can be applied to the problem of document separation and the approach to classification that we have taken to arrive at probabilities for categories attached to pages. All probability models were based on viewing the classification and separation process as a sequence mapping problem, described formally as a Markov chain as in Eq. 8.1. Experience from information extraction and speech recognition (e.g., [15]) shows that the results of such mappings and the search for the best sequence can be represented as a *trellis*. A trellis is a two-dimensional graph in which the horizontal axis represents time steps. For speech recognition, this would be incoming acoustic feature vectors. For information extraction, it could be words and, in our case, each time step is an incoming page within a batch. The vertical axis represents the states in which the mapping process may find itself and also the possible output symbols it may generate.

Transitions from states for one time step to the next denote the larger structure of the problem. These transitions can also be annotated with probabilities.

Consider the very simple model of Eq. 8.2, in which the probability of a document type only depends on the content of a page. In the trellis, there are as many states as there are document types. The interesting value for such a state is the probability that the page is of the associated document type. There are transitions from each state for one time step to all states for the next time step, indicating that each following state is equally probable. Figure 8.3 shows part of such a trellis.

The question of what a "best sequence" is can easily be answered: Since the individual scores delivered are probabilities (due to calibration), the probability of the complete sequence can be modeled using the product of all scores encountered on a path from the first page to the last. This sequence can be computed using *Viterbi*

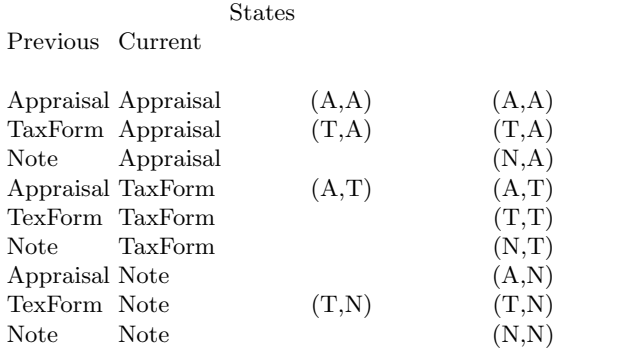


Fig. 8.4. Partial connection structure for the trellis for model Eq. 8.2.

search. At each time step, it records the locally best path to each state. Due to the independence of each time step, no locally suboptimal path can be part of the global solution. We can use Viterbi search to establish the best sequence of document types according to the model of Eq. 8.2. In fact, in the case of such a simple model, it suffices to identify the best document type for each page, which automatically will be a member of the best overall sequence. However, for any non-trivial model, this is not the case.

The models according to Eq. 8.3 and Eq. 8.4 introduce context into the decision process. This context or history needs to be reflected in the states of the trellis. For instance, the states reflecting the model of Eq. 8.3 are annotated with pairs of document types, the first one denoting the conditioning on the document type of the previous page and the second one denoting the decision for the current page. Moreover, the transition structure of the trellis needs to be modified as to ensure the consistency of paths. For instance, a state marked with "Appraisal" as the current decision can only be connected to following states that have "Appraisal" in their history. Figure 8.4 shows part of the connection structure for the model according to Eq. 8.3.

The extension of the context increases the number of states in a trellis. For the model of Eq. 8.4, we use triples instead of pairs of document types as state names.

Model 8.6 describes pages in more detail, adding a page type (Start, Middle, End) to the document type. Thus, for each document type relevant for a specific problem, we would have three states in the trellis. In addition, the transition structure needs to be carefully crafted as to only allow paths that describe complete documents, i.e., follow Eq. 8.5. For instance, in order to be a valid document boundary, a state associated with an end page must be immediately followed by a state associated with a start page.

For reasons of simplicity and extensibility, it would be advantageous if these sequence constraints could be formulated in isolation from the trellis containing the classification results themselves. Pereira and Riley [16] show that speech recognition

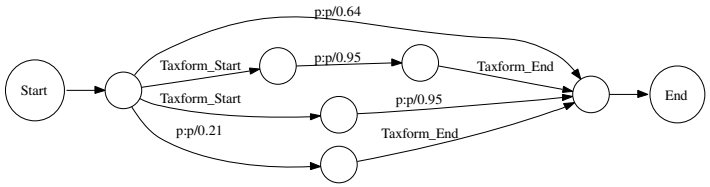


Fig. 8.5. Classification results for one page

can be interpreted as a sequence of weighted finite state transducers (WFSTs) that are combined using the composition operation. We adopt this view by associating our trellis of page classification results with an acoustic model applied to some input in speech recognition. The probabilities for an individual page to be of some class correspond to the emission probabilities represented in the recognition trellis of a speech recognizer. The restriction we placed on only allowing complete documents to be part of a sequence of documents corresponds to the use of a language model that renders certain word sequences more likely than others.[9] The "language model" we use currently only contains binary probability values, modeling hard constraints. However, similar to language models used in speech recognition, we could employ graded constraints represented by probabilities on language model transitions. This could be useful, for instance, in modeling the different likelihoods of sequences of documents, should such sequences exist.

In order to apply this analogy, we need to define the topology and contents of two finite state transducers. For the document type/page type model, the classification results can be represented in an FST as shown in Figure 8.5. The transitions of a classification transducer are of two kinds:

- Transitions that represent physical pages contain a symbol indicating a physical page on the lower and upper level and a classification score as weight. Which score is attached to the page depends on the topology of the transducer, which is defined by the next type of transitions.
- Transitions with an empty lower level denote boundary information about documents. There are transitions for the start and the end of a document. The occurrence of these transitions thus defines the type of page and the type of score that should be used. For instance, in Figure 8.5, the topmost transition (with score 0.64) indicates a middle page, since there are no boundaries given. The second transition chain belongs to a form that contains only a single page and consequently is bounded by both a start indicator and an end indicator. The third and fourth transitions belong to start and end pages respectively.

Figure 8.5 contains the information necessary to represent the classification results for one page with regard to one document type. The complete FST representing a problem with three document types and four pages is shown in Figure 8.6. Note

[9] On a more basic level, the document sequence restrictions can also be likened to the use of a pronunciation dictionary within a speech recognizer. However, acoustic modeling and pronunciation dictionary are usually combined into one processing step, while we explicitly distinguish between these.

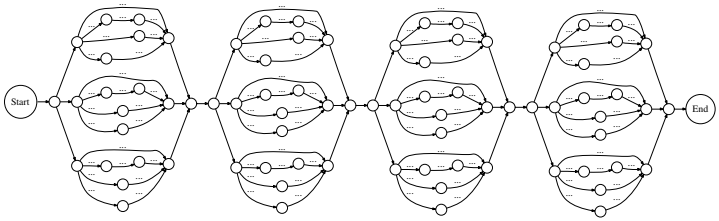


Fig. 8.6. Classification results as an FST

that using this FST, it is still possible to generate invalid page sequences. For instance, a start page for an Appraisal could be followed by another such start page. A restriction transducer plays a role analogous to a language model and ensures that such invalid paths are not present in the final search trellis.

A restriction (or rule) transducer contains labels on both the lower and upper side. The transformation from page content to document type/page type symbols has already been defined by the classification FST. The rule transducer has no informational role anymore, but a pure filtering role of eliminating impossible sequences of document and page types. Figure 8.7 shows the rules for a sequence containing documents of three document types. Note that all weights on the transitions are given as 1.0, so as not to modify the probabilities from the page information.

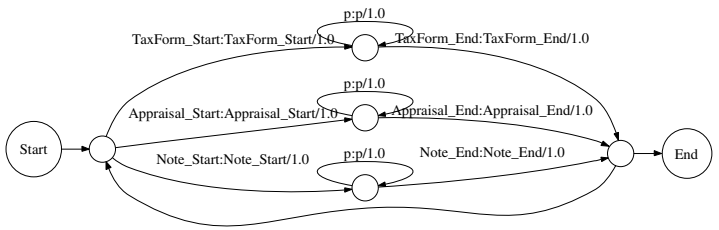


Fig. 8.7. FST for three document types

The composition operation on finite state transducers can now be used to generate a transducer that only contains such sequences of pages that result in a valid sequence of documents. Composition treats the "output" of one transducer (the upper level) as the "input" (lower level) of the other transducer. If we compose the classification FST with the rule FST, we achieve the desired result, an FST that contains the probabilities for specific pages, but only contains valid sequences. The output of this operation can be quite large, though. In the worst case, the composed FST has a number of states that is the product of the number of states of both arguments. In practice, this upper limit is not reached, but the size of the composed FST is still a concern, which we will address below.

Algorithm 1 General document separation

Require: An ordered list of pages (a batch), consisting of pages p_1 to p_n.
Require: An FST *rules* that describes the possible sequences of documents, as in Figure 8.7.
Ensure: An ordered list of documents d_k, each of which consists of an ordered list of pages.

```
    {Perform Classification}
 2: for all pages p_i, 1 ≤ i ≤ n in the batch do
 3:    for all classes c_j do {Three classes per document type (start/middle/end)}
 4:       c_ij ← probability that p_i is of class c_j
 5:    end for
 6: end for
    {Perform Separation}
 8: pg ← The FST representing the classification results, as in Figure 8.6
 9: sg ← pg composed with rules
10: sg ← the best path through sg
    {Create documents}
12: D ← ∅
13: d ← ∅
14: for all Transitions in sg, in topological order do
15:    if The transition is labeled with "_End" then
16:       D+ = d
17:    else if The transition is labeled with a page then
18:       d+ = p
19:    end if
20: end for
```

The goal is to find a path through the composed FST from the start state to an end state under the constraint that we are interested in the highest possible overall probability. Any graph search method can be applied. However, the topology of the input graph simplifies the problem somewhat (see below). For convenience, we represent the result of the search also as an FST. The separation algorithm can now be described by Algorithm 1.

Representing rules about the sequence of document types as a graph has more far-reaching applications than just to make sure that document boundaries are observed. For instance, limits about the size of documents can now be easily introduced. Figure 8.8 shows a rule FST that prescribes all Tax forms must be at least two pages and at most five pages long.

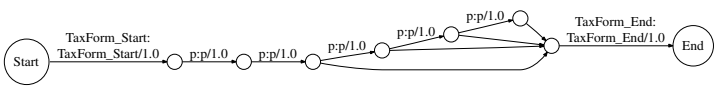


Fig. 8.8. A rule FST restricting tax forms to between two and five pages long

Similarly, the rule FST can be used to make demands about the order of documents, depending on the application. For instance, a mortgage application could prescribe that an Appraisal is always directly followed by a Note. Using a powerful representation mechanism such as finite state transducers simplifies the introduction of additional functionality significantly.

However, there are also drawbacks to the naive implementation of operations over finite state transducers. Additional representational power in this case comes at a price, mainly in terms of memory consumption and secondary in processing time. A typical mortgage application defines somewhere between 50 and 200 document types and consequently between 150 and 600 classes for which probabilities and graphs have to be produced. A batch may be as long as 1000 pages. The composition of the resulting classification graph with the rule graph requires a great deal of space to hold and time to construct. We experienced graph sizes of over one gigabyte and runtime reached tens of minutes, clearly insufficient for the successful application in industry.

This situation is again similar to speech recognition with language modeling. A language model represents an extremely large number of possible and probable word sequences. Incorporating this knowledge into the basic recognition trellis is infeasible due to both space and time restrictions. Thus, both knowledge sources are kept separate. The probabilities delivered by the language model are taken into account by the search for the best word sequence. For our problem, we also notice that the composition of the classification and rules FSTs is transient; in the end, we are interested only in the best path through the composition FST. Thus, it is unnecessary to completely unfold the composition. Instead, we use a technique similar to delayed composition [17] combined with beam search [18] to extract the final result. This reduces memory usage significantly for large problems. Table 8.4 shows the memory usage for a few data points, all of which are well within the limits of our requirements. The runtime is also much lower; for 1000 pages and 200 categories, the processing time decreases from roughly 16 minutes for the naive approach to less than 3 minutes for the advanced procedure.

Table 8.4. Memory usage for separation

Batch size in pages	Memory Usage 200 categories	300 categories
1000	118 MB	151 MB
2000	211 MB	363 MB

8.5 Results

We evaluated the performance of all probability models for sequences that we described in Section 8.4.2. Table 8.5 shows the $F1$-values for all models. The table suggests three main results:

Table 8.5. Comparison of separation and classifcation results of the various sequence models.

Seqence model	Probability	Micro-averaged F1-value
Eq. 8.2	$p(d_j \mid p_j^c)$	0.63
Eq. 8.3	$p(d_j \mid d_{j-1}, p_j^c)$	0.74
Eq. 8.4	$p(d_j \mid d_{j-1}, d_{j-2}, p_j^c)$	0.83
Eq. 8.6	$p(d'_j \mid p_j^c)$	0.84
Eq. 8.7	$p(d_j \mid d_{j-1}, d_{j-2}, p_{j-1}^c, p_j^c, p_{j+1}^c)$	0.86
Eq. 8.8	$p(d'_j \mid p_{j-1}^c, p_j^c, p_{j+1}^c)$	0.87

- The inclusion of a history of document types improves performance. This is not surprising, given the fact that forms are, on average, longer than one page. For instance, using a trigram model instead of a unigram yields an improvement of 31%.
- Specializing page descriptions improves performance. This confirms our earlier reasoning that forms often exhibit specific start and end pages. It also allows the model to separate two consecutive instances of the same document type.
- Conditioning on the content of surrounding pages improves performance. Comparing the last two rows in Table 8.5 with their counterparts without the content of the surrounding pages in the condition indicates a boost of around 3.5% in $F1$-value.

The last model (Eq. 8.8) is the best model in our experiments. However, it presents a serious drawback in that it uses roughly three times the number of features to describe a page (namely, the content of the page itself and that of the two surrounding pages). Given the increased CPU and memory usage during training, this seemed too high a price to pay for a 3% gain in performance. Thus, for deployment into customer production systems, we decided to use the model according to Eq. 8.6. It is the best of the one-page-content models, and the distinction of page types not only makes the model more efficient, but also helps with the integration of the separation workflow in a broader, extraction-oriented system owing to its capability of separating two consecutive identical forms.

Table 8.6 shows detailed results for the final deployment model. For each document type, the table shows the absolute counts of the results, precision, recall, and $F1$-value in two different scenarios. The first six columns show results on the page level: For each page, the predicted document type is compared with the true document type, and results calculated from that. The last six columns show values on the sequence level, taking into account full documents rather than pages. Each document (i.e., the sequence of pages from start page to end page) is compared with a gold standard; if both the extent of the document and its type match, the document is counted as correct. If either the document type or the pages contained in the document do not match, the document is counted as incorrect. These measures are much more strict than the page-level measures, as can be seen from the micro- and macro averages. Note that Table 8.6 reports on an experiment with 30 document types; however, the method scales well, and we achieve similar results with much larger numbers of categories.

Table 8.6. Number of true positive (TP), false positive (FP), false negative (FN), precision (P), recall (R), and F-measure (F) on a page as well as on a sequence level after separation. Final Model.

	Page level						Sequence level					
	TP	FP	FN	P	R	F1	TP	FP	FN	P	R	F1
Form A	207	4	1	0.98	1.00	0.99	108	9	8	0.92	0.93	0.93
Form B	13	0	0	1.00	1.00	1.00	4	0	0	1.00	1.00	1.00
Form C	151	9	7	0.94	0.96	0.95	79	24	13	0.77	0.86	0.81
Form D	171	0	10	1.00	0.94	0.97	108	10	12	0.92	0.90	0.91
Form E	2	0	0	1.00	1.00	1.00	2	0	0	1.00	1.00	1.00
Form F	15	0	0	1.00	1.00	1.00	12	0	0	1.00	1.00	1.00
Form G	100	0	0	1.00	1.00	1.00	22	0	0	1.00	1.00	1.00
Form H	56	0	0	1.00	1.00	1.00	53	0	0	1.00	1.00	1.00
Form I	6	0	0	1.00	1.00	1.00	6	0	0	1.00	1.00	1.00
Form J	14	0	0	1.00	1.00	1.00	14	0	0	1.00	1.00	1.00
Form K	10	3	0	0.77	1.00	0.87	10	3	0	0.77	1.00	0.87
Form L	74	11	12	0.87	0.86	0.87	21	9	7	0.70	0.75	0.72
Form M	52	11	10	0.83	0.84	0.83	18	6	4	0.75	0.82	0.78
Form N	13	0	0	1.00	1.00	1.00	13	0	0	1.00	1.00	1.00
Form O	2	1	3	0.67	0.40	0.50	1	2	3	0.33	0.25	0.29
Form P	167	8	4	0.95	0.98	0.97	106	23	11	0.82	0.91	0.86
Form Q	22	0	0	1.00	1.00	1.00	22	0	0	1.00	1.00	1.00
Form R	51	0	0	1.00	1.00	1.00	16	0	0	1.00	1.00	1.00
Form S	1	5	0	0.17	1.00	0.29	1	5	0	0.17	1.00	0.29
Form T	226	0	0	1.00	1.00	1.00	66	0	0	1.00	1.00	1.00
Form U	64	4	2	0.94	0.97	0.96	48	8	4	0.86	0.92	0.89
Form V	4	2	1	0.67	0.80	0.73	0	6	2	0.00	0.00	0.00
Form W	9	3	1	0.75	0.90	0.82	9	3	1	0.75	0.90	0.82
Form X	55	0	0	1.00	1.00	1.00	28	0	0	1.00	1.00	1.00
Form Y	376	4	13	0.99	0.97	0.98	248	18	15	0.93	0.94	0.94
Form Z	26	0	1	1.00	0.96	0.98	6	3	2	0.67	0.75	0.71
Form AA	326	8	8	0.98	0.98	0.98	26	18	7	0.59	0.79	0.68
Form AB	26	0	0	1.00	1.00	1.00	21	5	1	0.81	0.95	0.88
Form AC	332	0	2	1.00	0.99	1.00	20	2	2	0.91	0.91	0.91
Form AD	17	2	0	0.89	1.00	0.94	17	2	0	0.89	1.00	0.94
Σ	2588	75	75	–	–	—	1105	156	92	–	–	–
Micro averages	–	–	–	0.97	0.97	0.97	–	–	–	0.88	0.92	0.90
Macro averages	–	–	–	0.91	0.95	0.92	–	–	–	0.82	0.89	0.84

The training for this model required at least 20 examples per category, 10 each for the training and as a hold-out set. The maximum number of examples per category was capped at 40. Initially, the feature space had a dimensionality of 620,455. We reduced this number to at most 20,000 features per category by applying mutual information feature selection.

A comparison between the different problems and the models we apply is instructive. In Table 8.3, we reach an $F1$-value of 95% for the classification of documents. There, the boundaries are given, and the classifier is able to use all words from all pages in the document. In the experiments we report in Table 8.5, the problem is more complex: Each page must be classified separately and document boundaries inferred. Applying a comparable model (Eq. 8.2) in this situation, we only reach an $F1$-value of 63%. Only by careful selection of an appropriate probability model, we are able to raise the performance to an $F1$-value of 92% on the page level with model (Eq. 8.6).

One should note that the scores delivered by the SVM multi-class classifier are calibrated and represent class membership probabilities. Thus, thresholding can be applied to control the amount of errors that a customer expects from automatic decisions, and to control the amount of manual review of decisions that have been rejected. Using this technique, we can achieve precision of $> 95\%$ while simultaneously keeping the recall above 80%.

8.5.1 Production Deployments

The deployment of an automatic document separation solution is a lengthy process, as is common for any workflow-changing installation in large organizations. Most often, a proof-of-concept phase precedes the deployment proper. This part of a project can be pre-sales in order to demonstrate the feasibility of the approach to the customer or it can be as the first step in a deployment to find out how much automation can be introduced with high accuracy. In a proof of concept (POC), only a small subset of document types are considered for classification and separation. This poses a set of unique problems to consider: The document separator is normally set up to classify all documents into a set of well-known and well-defined document types. In a POC, only a subset of document types (say, 10 out of 50) is relevant. However, the incoming batches still contain documents of all types. The challenge here is to "actively ignore" the remaining document types without adverse effects on the classification and separation results for the document types on which we are concentrating.

The deployment of the production version of the separation solution can take as long as six months for a medium-sized organization (separating between five and ten million pages per month). Of this time, two to three months are usually spent on configuring the software. This includes the setup of the training data for the separator but also the development of an extraction mechanism that is usually part of the larger workflow. The rest of the time is used for the purchase and installation of hardware (possibly new scanners, processing machines, and review stations) and the retraining of the review personnel. It is good practice to introduce the new workflow and automated solution in increments, first converting one or two production lines to the automated separation solution and reviewing the efficiency of the process. Once the hardware, software and workflow function satisfactorily, the remaining lines are activated.

Using an automated classification and separation solution yields significant benefits for an organization. There are large cost-savings associated with the process (in a manual solution, 50% of the preparation cost is spent on sorting and inserting separator sheets) and the accuracy is superior. In a typical setting with hundreds of document types, at least 95-98% precision can be attained at a recall level of at least 80%. This means that only a fraction of the original data must be reviewed and no operations have to be performed on the physical paper pages that are at the source of the process.

8.6 Conclusion

In this chapter, we presented an automatic solution for the classification and separation of paper documents. The problem is to ingest a long sequence of images of paper pages and to convert those into a sequence of documents with definite boundaries and document types. In a manual setting, this process is costly and error prone. The automatic solution we describe prepares the incoming pages by running them through an OCR process to discover the text on the page. Basic NLP techniques for segmentation and morphological processing are used to arrive at a description of a page that associates stems with occurrence counts for a page (bag-of-words model). An SVM classifier is applied to generate probabilities that pages are of a given document and page type. After obtaining all classification probabilities, we are using a finite state transducer-based approach to detect likely boundaries between documents. Viewing this process as a sequence-mapping problem with well-defined subareas such as probabilistic modeling, classification and sequence processing allows us to fine-tune several aspects of the approach.

There were several major challenges in the development of this set of algorithms. The outside constraints prescribed a solution with high performance, both in terms of process accuracy and resource efficiency (time and hardware in setup and production). These requirements have significant ramifications for the choice of algorithms and models. For instance, Bayesian classifiers based on word n-grams are primarily unsuited due to their high training data demands. Also, the composition and search during separation had to be implemented in an on-demand fashion to comply with memory size requirements.

The overall result is a system that — although relatively simple in its basic components and methods — is very complex in its totality and its optimizations on a component level. We consistently reach high performance of greater than 95% precision with more than 80% recall and use the solution described here in large deployments with several million pages throughput a month.

8.7 Acknowledgments

Developing and validating technology solutions that can eventually be turned into successful products in the marketplace is an endeavor that includes many people. The authors would like to thank all who participated in exploring the technological and engineering problems of automatic document separation, in particular Tristan Juricek, Scott Texeira, and Sameer Samat.

References

1. Schmidtler, M., Texeira, S., Harris, C., Samat, S., Borrey, R., Macciola, A.: Automatic document separation. United States Patent Application 20050134935, US Patent & Trademark Office (2005)
2. Ratnaparkhi, A.: A Simple Introduction to Maximum Entropy Models for Natural Language Processing. IRCS Report 97-08, University of Pennsylvania, Philadelphia, PA (1997)
3. Reynar, J., Ratnaparkhi, A.: A Maximum Entropy Approach to Identifying Sentence Boundaries. In: Proceedings of the ANLP97, Washington, D.C. (1997)
4. Collins-Thompson, K., Nickolov, R.: A Clustering-Based Algorithm for Automatic Document Separation. In: SIGIR 2002 Workshop on Information Retrieval and OCR. (2002)
5. Pevzner, L., Hearst, M.: A Critique and Improvement of an Evaluation Metric for Text Segmentation. Computational Linguistics **28** (2002) 19–36
6. Porter, M.: An Algorithm for Suffix Stripping. Program **14** (1980) 130–130
7. Joachims, T.: Learning to Classify Text using Support Vector Machines: Methods, Theory, and Algorithms. Kluwer (2002)
8. McCallum, A., Freytag, D., Pereira, F.: Maximum entropy markov models for information extraction and segmentation. Technical report, Just Research, AT&T Labs — Research (2000)
9. Goodman, J.: A bit of progress in language modeling. Technical Report MSR-TR-2001-72, Machine Learning and Applied Statistics Group Microsoft Research (2001)
10. Vapnik, V.: Statistical Learning Theory. JOHN WILEY & SONS, INC (1998)
11. Jaakola, T., Meila, M., Jebara, T.: Maximum entropy discrimination. Technical report, MIT AI Lab, MIT Media Lab (1999)
12. Joachims, T.: Text Categorization with Support Vector Machines: Learning with Many Relevant Features. Technical Report LS-8 Report 23, Universitat Dortmund Fachbereich Informatik Lehrstuhl VIII Kunstliche Intelligenz (1997)
13. Platt, J.: Probabilistic outputs for support vector machines and comparison to regularised likelihood methods. Technical report, Microsoft Research (1999)
14. Harris, C., Schmidtler, M.: Effective multi-class support vector machine classification. United States Patent Application 20040111453, US Patent & Trademark Office (2004)
15. Jelinek, F.: Statistical Methods for Speech Recognition. Language, Speech and Communication. MIT Press, Cambridge, Massachusetts (1998)
16. Pereira, F., Riley, M.: Speech recognition by composition of weighted finite automata. Technical report, AT&T Labs — Research (1996)
17. Mohri, M., Pereira, F.C.N., Riley, M.: A Rational Design for a Weighted Finite-State Transducer Library. In: Workshop on Implementing Automata. (1997) 144–158
18. Lowerre, B.T.: The HARPY Speech Recognition System. PhD thesis, Carnegie Mellon University (1976)

9

Evolving Explanatory Novel Patterns for Semantically-Based Text Mining *

John Atkinson

9.1 Motivation

An important problem with mining textual information is that in this unstructured form is not readily accessible to be used by computers. This has been written for human readers and requires, when feasible, some natural language interpretation. Although full processing is still out of reach with current technology, there are tools using basic pattern recognition techniques and heuristics that are capable of extracting valuable information from free text based on the elements contained in it (i.e., keywords). This technology is usually referred to as **Text Mining**, and aims at discovering unseen and interesting patterns in textual databases [8, 19].

These discoveries are useless unless they contribute valuable knowledge for users who make strategic decisions (i.e., managers, scientists, businessmen). This leads then to a complicated activity referred to as **Knowledge Discovery from Texts** (KDT) which, like *Knowledge Discovery from Databases* (KDD), correspond to *"the non-trivial process of identifying valid, novel, useful and understandable patterns in data"* [6].

KDT can potentially benefit from successful techniques from Data Mining or Knowledge Discovery from Databases (KDD) [14] which have been applied to relational databases. However, Data Mining techniques cannot be immediately applied to text data for the purposes of TM as they assume a structure in the source data which is not present in free text. Hence new representations for text data have to be used. Also, while the assessment of discovered knowledge in the context of KDD is a key aspect for producing an effective outcome, the evaluation/assessment of the patterns discovered from text has been a neglected topic in the majority of the KDT approaches. Consequently, it has not been proven whether the discoveries are novel, interesting, and useful for decision makers.

Despite the large amount of research over the last few years, few research efforts worldwide have recognized the need for high-level representations (i.e., not just

* This research is sponsored by the National Council for Scientific and Technological Research (FONDECYT, Chile) under grant number 1040469 "*Un Modelo Evolucionario de Descubrimiento de Conocimiento Explicativo desde Textos con Base Semantica con Implicaciones para el Analisis de Inteligencia.*"

keywords), for taking advantage of linguistic knowledge, and for special purpose ways of producing and assessing the unseen knowledge. The rest of the effort has concentrated on doing text mining from an *Information Retrieval* (IR) perspective and so both representation (keyword based) and data analysis are restricted.

The most sophisticated approaches to text mining or KDT are characterised by an intensive use of external electronic resources including ontologies, thesauri, etc., which highly restricts the application of the unseen patterns to be discovered, and their domain independence. In addition, the systems so produced have few metrics (or none at all) which allow them to establish whether the patterns are interesting and novel.

In terms of data mining techniques, Genetic Algorithms (GA) for Mining purposes has several promising advantages over the usual learning/analysis methods employed in KDT: the ability to perform global search (traditional approaches deal with predefined patterns and restricted scope), the exploration of solutions in parallel, the robustness to cope with noisy and missing data (something critical in dealing with text information as partial text analysis techniques may lead to imprecise outcome data), and the ability to assess the goodness of the solutions as they are produced.

In this paper, we propose a new model for KDT which brings together the benefits of shallow text processing and GAs to produce effective novel knowledge. In particular, the approach combines *Information Extraction* (IE) technology and multi-objective evolutionary computation techniques. It aims at extracting key underlying linguistic knowledge from text documents (i.e., rhetorical and semantic information) and then hypothesising and assessing interesting and unseen explanatory knowledge. Unlike other approaches to KDT, we do not use additional electronic resources or domain knowledge beyond the text database.

9.2 Related Work

Typical approaches to text mining and knowledge discovery from texts are based on simple bag-of-words (BOW) representations of texts which make it easy to analyse them but restrict the kind of discovered knowledge [2]. Furthermore, the discoveries rely on patterns in the form of numerical associations between concepts (i.e., these terms will be later referred to as *target concepts*) from the documents, which fails to provide explanations of, for example, why these terms show a strong connection. Consequently, no deeper knowledge or evaluation of the discovered knowledge is considered and so the techniques become merely "adaptations" of traditional DM methods with an unproven effectiveness from a user viewpoint.

Traditional approaches to KDT share many characteristics with classical DM but they also differ in many ways: many classical DM algorithms [19, 6], are irrelevant or ill suited for textual applications as they rely on the structuring of data and the availability of large amounts of structured information [7, 18, 27]. Many KDT techniques inherit traditional DM methods and keyword-based representation which are insufficient to cope with the rich information contained in natural-language text. In addition, it is still unclear how to rate the novelty and/or interestingness of the knowledge discovered from texts.

Some people suggest that inadequacy and failure to report novel results are likely because of the confusion between finding/accessing information in texts (i.e., using

IR and data analysis techniques) and text mining: the goal of information access is to help users find documents that satisfy their information needs, whereas KDT aims at discovering or deriving novel information from texts, finding patterns across the documents [17]. Here, two main approaches can be distinguished: those based on Bag-of-Words representations, and those based on more structured representations.

9.2.1 Bag-of-Words-Based Approaches

Some of the early work on TM came from the Information Retrieval community, hence the assumption of text represented as a *Bag-of-Words* (BOW), and then to be processed via classical DM methods [7, 27]. Since there is additional information beyond these keywords and issues such as their order do not matter in a BOW approach, it will usually be referred to as non-structured representation.

Once the initial information (i.e., terms, keywords) has been extracted, KDD operations can be carried out to discover unseen patterns. Representative methods in this context have included *Regular Associations* [6], *Concept Hierarchies* citeFeldman98b, *Full Text Mining* [27], *Clustering*, *Self-Organising Maps*.

Most of these approaches work in a very limited way because they rely on surface information extracted from the texts, and on its statistical analysis. As a consequence, key underlying linguistic information is lost. The systems may be able to detect relations or associations between items, but they cannot provide any description of what those relations are. At this stage, it is the user's responsibility to look for the documents involved with those concepts and relations to find the answers. Thus, the relations are just a "clue" that there is something interesting but which needs to be manually verified.

9.2.2 High-Level Representation Approaches

Another main stream in KDT involves using more structured or higher-level representations to perform deeper analysis so to discover more sophisticated novel / interesting knowledge. Although in general, the different approaches have been concerned with either performing exploratory analysis for hypothesis formation, or finding new connections/relations between previously analysed natural language knowledge, it has also involved using term-level knowledge for other purposes than just statistical analysis.

Some early research by Swanson on the titles of articles stored in MEDLINE [28] used an augmented low-level representation (the words in the titles) and exploratory data analysis to discover hidden connections [30, 32] leading to very promising and interesting results in terms of answering questions for which the answer was not currently known. He showed how chains of causal implication within the medical literature can lead to hypotheses for causes of rare diseases, some of which have received scientific supporting evidence.

Other approaches using *Information Extraction* (IE) which inherited some of Swanson's ideas to derive new patterns from a combination of text fragments, have also been successful. Essentially, IE is a Natural-Language (NL) technology which analyses an input NL document in a shallow way by using defined patterns along with mechanisms to resolve implicit discourse-level information (i.e., anaphora, coreference, etc.) to match important information from the texts. As a result, an IE task

produces an intermediate representation called "templates" in which information relevant has been recognised, for example: names, events, entities, etc., or high-level linguistic entities: noun phrases, etc.

Using IE techniques and electronic linguistic resources, Hearst [19] proposes a domain-independent method for the automatic discovery of WordNet-style lexicosemantic relations by searching for corresponding lexicosyntactical patterns in unrestricted text collections. This technique is meant to be useful as an automated or semi-automated aid for lexicographers and builders of domain-dependent knowledge bases. Also, it does not require an additional knowledge base or specific interpretation procedures in order to propose new instances of WordNet relations [9]. Once the basic relations (i.e., hyponyms, hypernyms, etc.) are obtained, they are used to find common links with other "similar" concepts in WordNet [9] and so to discover new semantic links [18]. However, there are tasks which need to be performed by hand such as deciding on a lexical relation that is of interest (i.e., hyponym) and a list of word pairs from WordNet this relation is known to hold between.

One of the main advantages of this method is its low cost for augmenting the structure of WordNet and its simplicity of relations. However, it also has some drawbacks including its dependence on the structure of a general-purpose ontology which prevents it from reasoning about specific terminology/concepts, the restricted set of defined semantic relations (i.e., only relations contained in WordNet are dealt with), its dependence on WordNet's terms (i.e., only terms present in WordNet can be related and any novel domain-specific term will be missed), the kind of inference enabled (i.e., it is only possible to produce direct links; what if we wish to relate different terms which are not in WordNet?), etc.

A natural further important step would be using knowledge base such as WordNet to support text inference to extract relevant, unstated information from the text. Harabagiu and Moldovan [15] address this issue by using WordNet as a commonsense knowledge base and designing relation-driven inference mechanisms which look for common semantic paths in order to draw conclusions. One outstanding feature of their method is that from these generated inferences, it is easy to ask for unknown relations between concepts. This has proven to be extremely useful in the context of Question-Answering Systems. However, although the method exhibits understanding capabilities, the commonsense facts discovered have not been demonstrated to be novel and interesting from a KDD viewpoint.

Mooney and colleagues [25] have also attempted to bring together general ontologies, IE technology and traditional machine learning methods to mine interesting patterns. Unlike previous approaches, Mooney deals with a different kind of knowledge, e.g., prediction rules. In addition, an explicit measure of novelty of the mined rules is proposed by establishing semantic distances between rules' antecedents and consequents using the underlying organisation of WordNet. Novelty is then defined as the average (semantic) distance between the words in a rule's antecedent and consequent. A key problem with this is that the method depends highly on WordNet's organisation and idiosyncratic features. As a consequence, since a lot of information extracted from the documents are not included in WordNet the predicted rules will lead to misleading decisions on their novelty.

The discussed approaches to TM/KDT use a variety of different "learning" techniques. Except for cases using Machine Learning techniques such as Neural Networks (e.g., SOM), decision trees, and so on, which have also been used in traditional DM, the real role of "learning" in the systems is not clear. There is no learning which

enables the discovery but instead a set of primitive search strategies which do not necessarily explore the whole search space due to their dependence on the kind of semantic information previously extracted.

Although DM tasks have been commonly tackled as learning problems, the nature of DM suggests that the problem of DM (i.e., finding unseen, novel and interesting patterns) should be seen as involving search (i.e., different hypotheses are explored) and optimization (i.e., hypotheses which maximize quality criteria should be preferred) instead.

Despite there being a significant and successful number of practical search and optimization techniques [24, 5], there are some features that make some techniques more appealing to perform this kind of task than others, in terms of representation required, training sets required, supervision, hypothesis assessment, robustness in the search, etc.

In particular, the kind of evolutionary computation technique known as *Genetic Algorithms* (GA) has proved to be promising for search and optimization purposes. Compared with classical search and optimization algorithms, GAs are much less susceptible to getting stuck in local suboptimal regions of the search space as they perform global search by exploring solutions in parallel. GAs are robust and able to cope with noisy and missing data, they can search spaces of hypotheses containing complex interacting parts, where the impact of each part on overall hypothesis fitness may be difficult to model [13].

In order to use GAs to find optimal values of decision variables, we first need to represent the hypotheses in binary strings (the typical pseudo-chromosomal representation of a hypothesis in traditional GAs). After creating an initial population of strings at random, genetic operations are applied with some probability in order to improve the population. Once a new string is created by the operators, the solution is evaluated in terms of its measure of individual goodness referred to as *fitness*.

Individuals for the next generation are selected according to their fitness values, which will determine those to be chosen for reproduction. If a termination condition is not satisfied, the population is modified by the operators and a new (and hopefully better) population is created. Each interaction in this process is called a *generation* and the entire set of generations is called a *run*. At the end of a run there is often one or more highly fit chromosomes in the population.

One of the major contributions of evolutionary algorithms (e.g., GAs) for an important number of DM tasks (e.g., rule discovery, etc.) is that they tend to cope well with attribute interactions. This is in contrast to the local, greedy search performed by often-used rule induction and decision-tree algorithms [3, 14]. Most rule induction algorithms generate (prune) a rule by selecting (removing) one rule condition at a time, whereas evolutionary algorithms usually evaluate a rule as a whole via the fitness function rather than evaluating the impact of adding/removing one condition to/from a rule. In addition, operations such as crossover usually swap several rule conditions at a time between two individuals.

Typical tasks for GAs in DM have included [12, 34]: *Classification*; in which the goal is to predict the value (the class) of a user-defined goal attribute based on the values of other attributes; *Discovery of Association rules*; where binary attributes (items) contained in data instances (i.e., records) are used to discover associations of the form IF-THEN, *Rule discovery/prediction*; in which the system can produce many different combinations of attributes. (even if the original attributes do not

have much predictive power by themselves, the system can effectively create "derived attributes" with greater predictive power) to come up with new rules.

A common representation used for this kind of task encodes attributes and values of a rule in a binary string of rule conditions and rule consequent. Suppose that an individual represents a rule antecedent with a single attribute-value condition, where the attribute *Marital_status* and its values can be "single," "married," "divorced," and "widow." A possible representation would be a condition involving this attribute encoded by four bits, so the string "0110" (i.e., the second and third values of the attribute are present) would represent the antecedent *IF marital_status=married OR divorced)* using internal disjunctions (i.e., logical OR).

One general aspect worth noting in applying GAs for DM tasks is that both the representation used for the discovery and the evaluation carried out assume that the source data are properly represented in a structured form (i.e., database) in which the attributes and values are easily handled.

When dealing with text data, these working assumptions are not always plausible because of the complexity of text information. In particular, mining text data using evolutionary algorithms requires a certain level of representation which captures knowledge beyond discrete data (i.e., semantics). Thus there arises the need for new operations to create knowledge from text databases. In addition, fitness evaluation also imposes important challenges in terms of measuring novel and interesting knowledge which might be implicit in the texts or be embedded in the underlying semantics of the extracted data.

Applying evolutionary methods to TM/KDT is a very recent research topic. With the exception of the work of [1] on the discovery of semantic relations no other research effort is under way as far as we know as the most promising KDT techniques have been tackled with more traditional search/learning methods.

The advantage over a similar approach for discovery of unseen relations as in [16], is that this approach provides more robust results in a way that exploits a wider number of possible hypotheses in the search space. In addition, the IE patterns finally used for the extraction are automatically learned, whereas for [16], these need to be handcrafted. Although the obtained relations have been evaluated in terms of their coverage in WordNet, the subjective quality of this unseen knowledge has not been assessed from a KDD viewpoint as no user has been involved in the process.

9.3 A Semantically Guided Model for Effective Text Mining

We developed a semantically guided model for evolutionary Text Mining which is domain-independent but genre-based. Unlike previous approaches to KDT, our approach does not rely on external resources or descriptions hence its domain-independence. Instead, it performs the discovery only using information from the original corpus of text documents and from the training data generated from them. In addition, a number of strategies have been developed for automatically evaluating the quality of the hypotheses ("novel" patterns). This is an important contribution on a topic which has been neglected in most of KDT research over the last years.

We have adopted GAs as central to our approach to KDT. However, for proper GA-based KDT there are important issues to be addressed including representa-

tion and guided operations to ensure that the produced offspring are semantically coherent.

In order to deal with issues regarding representation and new genetic operations so to produce an effective KDT process, our working model has been divided into two phases. The first phase is the preprocessing step aimed to produce both training information for further evaluation and the initial population of the GA. The second phase constitutes the knowledge discovery itself, in particular this aims at producing and evaluating explanatory unseen hypotheses.

The whole processing starts by performing the IE task (Figure 9.1) which applies extraction patterns and then generates a rule-like representation for each document of the specific domain corpus. After processing a set of n documents, the extraction stage will produce n rules, each one representing the document's content in terms of its conditions and conclusions. Once generated, these rules, along with other training data, become the "model" which will guide the GA-based discovery (see Figure 9.1).

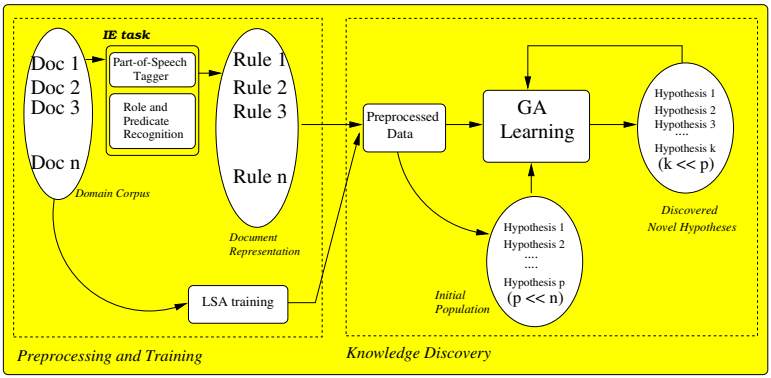


Fig. 9.1. The Evolutionary Model for Knowledge Discovery from Texts

In order to generate an initial set of hypotheses, an initial population is created by building random hypotheses from the initial rules, that is, hypotheses containing predicate and rhetorical information from the rules are constructed. The GA then runs for a number of generations until a fixed number of generations is achieved. At the end, a small set of the best hypotheses are obtained.

The description of the model is organised as follows: Section 9.3.1 presents the main features of the text preprocessing phase and how the representation for the hypotheses is generated. In addition, training tasks which generate the initial knowledge (semantic and rhetorical information) to feed the discovery are described. Section 9.3.2 describes constrained genetic operations to enable the hypotheses discovery, and proposes different evaluation metrics to assess the plausibility of the discovered hypotheses in a multi-objective context.

9.3.1 Text Preprocessing and Training

The preprocessing phase has two main goals: to extract important information from the texts and to use that information to generate both training data and the initial population for the GA.

In terms of text preprocessing (see first phase in Figure 9.1), an underlying principle in our approach is to be able to make good use of the structure of the documents for the discovery process. It is well-known that processing full documents has inherent complexities [23], so we have restricted our scope somewhat to consider a scientific genre involving scientific/technical abstracts. These have a well-defined macro-structure (genre-dependent rhetorical structure) to "summarise" what the author states in the full document (i.e., background information, methods, achievements, conclusions, etc).

Unlike patterns extracted for usual IE purposes such as in [18, 19, 20], this macro-structure and its roles are domain-independent but genre-based, so it is relatively easy to translate it into different contexts.

As an example, suppose that we are given the following abstract where bold sequences of words indicate the markers triggering the IE patterns:

The current study **aims to provide**
GOAL { the basic information about the fertilisers system, specially in its nutrient dynamics.
OBJECT { Long-term trends of the soil's chemical and physical fertility
were also analysed. The **methodology is based on** the
METHOD { study of lands' plots using different histories of usage of crop rotation with fertilisers
in order to detect long-term changes. ... Finally, a deep checking of data allowed us **to conclude that**
CONCLUSION { soils have improved after 12 years of continuous rotation.

From such a structure, important constituents can be identified:

- *Rhetorical Roles (discourse-level knowledge):* these indicate important places where the author makes some "assertions" about his/her work (i.e., the author is stating the goals, used methods, achieved conclusions, etc.). In the example above, the roles are represented by `goal, object of study, method` and `conclusion`.
- *Predicate Relations:* these are represented by actions (predicate and arguments) which are directly connected to the role being identified and state a relation which holds between a set of terms (words which are part of a sentence), a predicate and the role which they are linked to. Thus, for the example, they are as follows: **provide('the basic information ..'), analyse('long-term trends ...'), study('lands plot using ...'), improve('soil ..improved after ..')**
- *Causal Relation(s):* Although there are no explicit causal relations in the above example, we can hypothesise a simple rule of the form:
  IF the current goals are G1,G2, .. and the means/methods used M1,M2, .. (and any other constraint/feature) THEN it is true that we can achieve the conclusions C1,C2, ..
  Finally, the sample abstract may be represented in a rule-like form as follows:

```
IF    goal(provide('the basic information ..'))
  AND object(analyse('long-term trends ...'))
  AND method(study('lands plot using  ...'))
THEN  conclusion(improve('soil ..improved after ..'))
```

Note that causal relations are extracted from individual abstracts. In order to extract this initial key information from the texts, an IE module was built. Essentially, it takes a set of text documents, has them tagged through a previously trained Part-of-Speech (POS) tagger (i.e., Brill Tagger), and produces an intermediate representation for every document (i.e., template, in an IE sense) which is then converted into a general rule. A set of hand-crafted domain-independent extraction patterns were written and coded.

In addition, key training data are captured from the corpus of documents itself and from the semantic information contained in the rules. This can guide the discovery process in making further similarity judgements and assessing the plausibility of the produced hypotheses.

- *Training Information from the Corpus:*
  It has been suggested that huge amounts of texts represent a valuable source of semantic knowledge. In particular, in *Latent Semantic Analysis* (LSA) [21] it is claimed that this knowledge is at the word level.
  Following work by [21] on LSA incorporating structure, we have designed a semi-structured LSA representation for text data in which we represent predicate information (i.e., verbs) and arguments (i.e., set of terms) separately once they have been properly extracted in the IE phase. For this, the similarity is calculated by computing the closeness between two predicates (and arguments) based on the LSA data (function $SemSim(P_1(A_1), P_2(A_2))$).
  We propose a simple strategy for representing the meaning of the predicates with arguments. Next, a simple method is developed to measure the similarity between these units.
  Given a predicate `P` and its argument `A`, the vectors representing the meaning for both of them can be directly extracted from the training information provided by the LSA analysis. Representing the argument involves summing up all the vectors representing the terms of the argument and then averaging them, as is usually performed in semi-structured LSA. Once this is done, the meaning vector of the predicate and the argument is obtained by computing the sum of the two vectors as used in [33]. If there is more than one argument, then the final vector of the argument is just the sum of the individual arguments' vectors.
  Next, in making further semantic similarity judgements between two predicates $P_1(A_1)$ and $P_2(A_2)$ (i.e., **provide('the basic information ..')**), we take their corresponding previously calculated meaning vectors and then the similarity is determined by how close these two vectors are. We can evaluate this by computing the *cosine* between these vectors which gives us a closeness measure between -1 (complete unrelatedness) and 1 (complete relatedness) [22].
  Note however that training information from the texts is not sufficient as it only conveys data at a word semantics level. We claim that both basic knowledge at a rhetorical, semantic level, and co-occurrence information can be effectively computed to feed the discovery and to guide the GA.
  Accordingly, we perform two kinds of tasks: creating the initial population and computing training information from the rules.

a) *Creating the initial population of hypotheses:*
once the initial rules have been produced, their components (rhetorical roles, predicate relations, etc.) are isolated and become a separate "database." This information is used both to build the initial hypotheses and to feed the further genetic operations (i.e., mutation of roles will need to randomly pick a role from this database).

b) *Computing training information (in which two kinds of training data are obtained)*:

   a) *Computing correlations between rhetorical roles and predicate relations:* the connection between rhetorical information and the predicate action constitutes key information for producing coherent hypotheses. For example, is, in some domain, the *goal* of some hypothesis likely to be associated with the *construction* of some component? In a health context, this connection would be less likely than having "*finding* a new medicine for .." as a *goal*.
   In order to address this issue, we adopted a Bayesian approach where we obtain the conditional probability of some predicate `p` given some attached rhetorical role `r`, namely $Prob(p \mid r)$. This probability values are later used to automatically evaluate some of the hypotheses' criteria.

   b) *Computing co-occurrences of rhetorical information:*
   One could think of a hypothesis as an abstract having text paragraphs which are semantically related to each other. Consequently, the meaning of the scientific evidence stated in the abstract may subtly change if the order of the facts is altered.
   This suggests that in generating valid hypotheses there will be rule structures which are more or less desirable than others. For instance, if every rule contains a "goal" as the first rhetorical role, and the GA has generated a hypothesis starting with some "conclusion" or "method," it will be penalised and therefore, it is very unlikely for that to survive in the next generation. Since the order matters in terms of affecting the rule's meaning, we can think of the p roles of a rule, as a sequence of tags: $< r_1, r_2, ..r_p >$ such that r_i precedes r_{i+1}, so we generate, from the rules, the conditional probabilities $Prob(r_p \mid r_q)$, for every role r_p, r_q. The probability that r_q precedes r_p will be used in evaluating new hypotheses, in terms that, for instance, its coherence.

9.3.2 Knowledge Discovery and Automatic Evaluation of Patterns

Our approach to KDT is strongly guided by semantic and rhetorical information, and consequently there are some soft constraints to be met before producing the offspring so as to keep them coherent.

The GA will start from a initial population, which in this case, is a set of semi-random hypotheses built up from the preprocessing phase. Next, constrained GA operations are applied and the hypotheses are evaluated. In order for every individual to have a fitness assigned, we use a evolutionary multi-objective optimisation strategy based on the *Strength Pareto Evolutionary Algorithm* (SPEA) algorithm [35]. SPEA deals with the diversity of the solutions (i.e., niche formation) and the fitness assignment as a whole in a representation-independent way. An attractive

feature of SPEA is that in order to create niches, this does not define a neighborhood by means of a distance metric on the genotypic or phenotypic space. Instead, the classes of solutions are grouped according to the results of a clustering method which uses the vector of objective functions of the individuals, and not the individuals themselves.

Once the offspring is produced, the population update is performed using a steady-state strategy. Here, each individual from a small number of the worst hypotheses is replaced by an individual from the offspring only if the latter are better than the former.

For semantic constraints, judgements of similarity between hypotheses or components of hypotheses (i.e., predicates, arguments, etc.) are carried out using the LSA training data and predicate-level information previously discussed in the training step.

Hypothesis Discovery

Using the semantic measure above and additional constraints discussed later on, we propose new operations to allow guided discovery such that unrelated new knowledge is avoided, as follows:

- *Selection:* selects a small number of the best parent hypotheses of every generation (*Generation Gap*) according to their fitness. Note that the notion of optimum (and *best*) is different here as there is more than one objective to be traded off. Accordingly, this is usually referred to as a *"Pareto Optimum"* [29]. Assuming a minimization problem (i.e., "worse" involves smaller values), a decision vector (i.e., vector of several objectives) is a Pareto optimal if there exists no feasible vector which would increase some objective without causing a simultaneous decrease in at least one other objective. Unfortunately, this concept almost always gives not a single solution, but rather a set of solutions called the *Pareto Optimal set*. The decision vectors corresponding to the solutions included in the Pareto optimal set are called non-dominated, and the space of the objective functions whose nondominated vectors are in the Pareto optimal set is called the *Pareto front* [4, 5, 11].
- *Crossover:* a simple recombination of both hypotheses' conditions and conclusions takes place, where two individuals swap their conditions to produce new offspring (the conclusions remain).
  Under normal circumstances, crossover works on random parents and positions where their parts should be exchanged. However, in our case this operation must be restricted to preserve semantic coherence. We use soft semantic constraints to define two kinds of recombinations:
  a) *Swanson's Crossover:* based on Swanson's hypothesis [30, 31] we propose a recombination operation as follows:
     If there is a hypothesis (AB) such that "IF A THEN B" and another one (BC) such that "IF B' THEN C," (B' being something semantically similar to B) then a new interesting hypothesis "IF A THEN C" can be inferred via LSA if the conclusions of AB have high semantic similarity with the conditions of hypothesis BC.
     The above principle can be seen in Swanson's crossover between two learned hypotheses as shown in figure 9.2

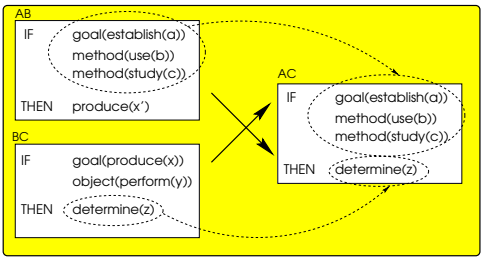


Fig. 9.2. Semantically guided Swanson Crossover

b) *Default Semantic Crossover:* if the previous transitivity does not apply then the recombination is performed as long as both hypotheses as a whole have high semantic similarity which is defined in advance by providing minimum thresholds.

- *Mutation:* aims to make small random changes on hypotheses to explore new possibilities in the search space. As in recombination, we have dealt with this operation in a constrained way, so we propose three kinds of mutations to deal with the hypotheses' different objects:
  a) *Role Mutation:* one rhetorical role (including its contents: relations and arguments) is selected and randomly replaced by a random one from the initial role database.
  b) *Predicate Mutation:* one inner predicate and its argument is selected and randomly replaced with another predicate-argument pair from the initial predicate databases.
  c) *Argument Mutation:* since we have no information about arguments' semantic types, we choose a new argument by following a guided procedure in which the former argument is randomly replaced with that having a high semantic similarity via LSA. [33].
- *Population Update:* we use a non-generational GA in which some individuals are replaced by the new offspring in order to preserve the hypotheses' good material from one generation to other, and so to encourage the improvement of the population's quality.

Evaluation

Since each hypothesis in our model has to be assessed by different criteria, usual methods for evaluating fitness are not appropriate. Hence *Evolutionary Multi-Objective Optimisation* (EMOO) techniques which use the multiple criteria defined for the hypotheses are needed. Accordingly, we propose EMOO-based evaluation metrics to assess the hypotheses' fitness in a domain-independent way and, unlike other approaches, without using any external source of domain knowledge. The different metrics are represented by multiple criteria by which the hypotheses are assessed.

In order to establish evaluation criteria, we have taken into account different issues concerning plausibility (Is the hypothesis semantically sound?, Are the GA operations producing something coherent in the current hypothesis?), and quality

itself (How is the hypothesis supported from the initial text documents? How interesting is it?). Accordingly, we have defined eight evaluation criteria to assess the hypotheses (i.e., in terms of Pareto dominance, it will produce a 8-dimensional vector of objective functions) given by: **relevance, structure, cohesion, interestingness, coherence, coverage, simplicity, plausibility of origin**.

The current hypothesis to be assessed will be denoted as H, and the training rules as R_i. Evaluation methods (criteria) by which the hypotheses are assessed and the questions they are trying to address are as follows:

- **Relevance**
  Relevance addresses the issue of how important the hypothesis is to target concepts. This involves two concepts (i.e., terms), as previously described, related to the question:
  What is the best set of hypotheses that explain the relation between $< term1 >$ *and* $< term2 >$*?*
  Considering the current hypothesis, it turns into a specific question: how good is the hypothesis in explaining this relation?
  This can be estimated by determining the semantic closeness between the hypothesis' predicates (and arguments) and the target concepts[2] by using the meaning vectors obtained from the LSA analysis for both terms and predicates. Our method for assessing relevance takes these issues into account along with some ideas of Kintsch's Predication. Specifically, we use the concept of *Strength* [21]: $strength(A, I) = f(SemSim(A, I), SemSim(P, I)))$ between a predicate with arguments and surrounding concepts (target concepts in our case) as a part of the relevance measure, which basically decides whether the predicate (and argument) is relevant to the target concepts in terms of the similarity between both predicate and argument, and the concepts.
  We define the function f as proposed by [21] to give a relatedness measure such that high values are obtained only if both the similarity between the target concept and the argument (α), and target concept and the predicate (β) exceed some threshold. Next, we highlight the closeness by determining the square difference between each similarity value and the desired value (1.0). If we take the average square difference, we obtain an error metric which is a *Mean Square Error* (MSE). As we want to get low error values so to encourage high closeness, we subtract MSE from 1. Formally, $f(\alpha, \beta)$ is therefore computed as the function:

$$f(\alpha, \beta) = \begin{cases} 1 - MSE(\{\alpha, \beta\}) & \text{if both } \alpha \text{ and } \beta > threshold \\ 0 & \text{Otherwise} \end{cases}$$

  where the MSE is the *Mean Square Error* between the similarities and the desired value ($Vd = 1.0$), is calculated as:

$$\text{MSE}(\{\text{list of n values } v_i\}) = \frac{1}{n} \sum_{i=1}^{n} (v_i - Vd)^2$$

  In order to account for both target concepts, we just take the average of **strength** for both terms. So, the overall relevance becomes:

$$relevance(H) = \frac{\frac{1}{2} \sum_{i=1}^{|H|} strength(P_i, A_i, <term1>) + strength(P_i, A_i, <term2>)}{|H|}$$

[2] Target concepts are relevant nouns in our experiment. However, in a general case, these might be either nouns or verbs.

in which | H | denotes the length of the hypothesis H, that is, the number of predicates.
Note that pairs of target concepts are provided by a domain experts so as to guide the search process.

- **Structure** (*How good is the structure of the rhetorical roles?*): measures how much of the rules' structure is exhibited in the current hypothesis.
Since we have previous pre-processed information for bi-grams of roles, the structure can be computed by following a Markov chain [23] as follows:

$$Structure(H) = Prob(r_1) * \prod_{i=2}^{|H|} Prob(r_i \mid r_{i-1})$$

where r_i represents the $i-th$ role of the hypothesis H, $Prob(r_i \mid r_{i-1})$ denotes the conditional probability that role r_{i-1} immediately precedes r_i. $Prob(r_i)$ denotes the probability that no role precedes r_i, that is, it is at the beginning of the structure (i.e., $Prob(r_i \mid < start >)$).

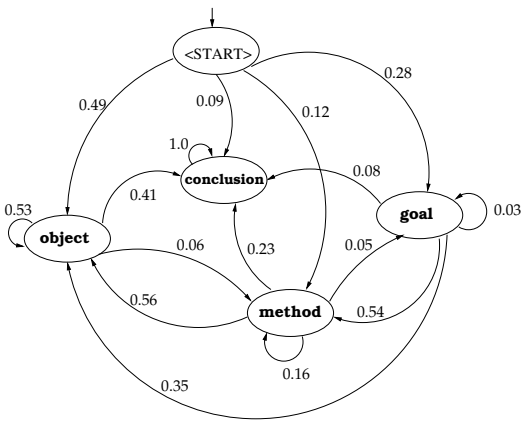


Fig. 9.3. Markov Model for Roles Structure Learned from sampled technical documents

For example, part of a Markov chain of rhetorical roles learned by the model from a specific technical domain can be seen in figure 9.3. Here it can be observed that some structure tags are more frequent than others (i.e., the sequence of rhetorical roles goal-method (0.54) is more likely than the sequence goal-conclusion (0.08)).

- **Cohesion** (*How likely is a predicate action to be associated with some specific rhetorical role?*): measures the degree of "connection" between rhetorical information (i.e., roles) and predicate actions. The issue here is how likely (according to the rules) some predicate relation P in the current hypothesis is to be associated with role r. Formally, *cohesion* for hypothesis H is expressed as:

$$\text{cohesion(H)} = \sum_{r_i, P_i \in H} \frac{Prob(P_i|r_i)}{|H|}$$

where $Prob(P_i \mid r_i)$ states the conditional probability of the predicate P_i given the rhetorical role r_i.

- **Interestingness** (*How interesting is the hypothesis in terms of its antecedent and consequent?*):
  Unlike other approaches to measure "interestingness" which use an external resource (e.g., WordNet) and rely on its organisation, we propose a different view where the criterion can be evaluated from the semi-structured information provided by the LSA analysis. Accordingly, the measure for hypothesis H is defined as a degree of unexpectedness as follows:

  `interestingness(H)= <Semantic Dissimilarity between Antecedent and Consequent>`

  That is, the lower the similarity, the more interesting the hypothesis is likely to be, so the dissimilarity is measured as the inverse of the LSA similarity. Otherwise, it means the hypothesis involves a correlation between its antecedent and consequent which may be an uninteresting known common fact [26].
- **Coherence:** This metrics addresses the question whether the elements of the current hypothesis relate to each other in a semantically coherent way. Unlike rules produced by DM techniques in which the order of the conditions is not an issue, the hypotheses produced in our model rely on pairs of adjacent elements which should be semantically sound, a property which has long been dealt with in the linguistic domain, in the context of *text coherence* [10].
  As we have semantic information provided by the LSA analysis which is complemented with rhetorical and predicate-level knowledge, we developed a simple method to measure coherence, following work by [10] on measuring text coherence.
  Semantic coherence is calculated by considering the average semantic similarity between consecutive elements of the hypothesis. However, note that this closeness is only computed on the semantic information that the predicates and their arguments convey (i.e., not the roles) as the role structure has been considered in a previous criterion. Accordingly, the criterion can be expressed as follows:

  $$\text{Coherence(H)} = \sum_{i=1}^{(|H|-1)} \frac{SemSim(P_i(A_i), P_{i+1}(A_{i+1}))}{(|H|-1)}$$

  where $(\mid H \mid -1)$ denotes the number of adjacent pairs, and $SemSim$ is the LSA-based semantic similarity between two predicates.
- **Coverage:** The coverage metric tries to address the question of how much the hypothesis is supported by the model (i.e., rules representing documents and semantic information).
  Coverage of a hypothesis has usually been measured in KDD approaches by considering some structuring in data (i.e., discrete attributes) which is not present in textual information. Besides, most of the KDD approaches have assumed the use of linguistic or conceptual resources to measure the degree of coverage of the hypotheses (i.e., match against databases, positive examples).
  In order to deal with the criterion in the context of KDT, we say that a generated hypothesis H covers an extracted rule R_i (i.e., rule extracted from the original training documents, including semantic and rhetorical information) only if the predicates of H are roughly (or exactly, in the best case) contained in R_i.
  Formally, the rules covered are defined as:

  $$\texttt{RulesCovered(H)} = \{\ R_i \in RuleSet \mid \forall P_j \in R_i \quad \exists HP_k \in HP : (\texttt{SemSim}(HP_k, P_j) \geq \texttt{threshold} \wedge \texttt{predicate}(HP_k) = \texttt{predicate}(P_j))\}$$

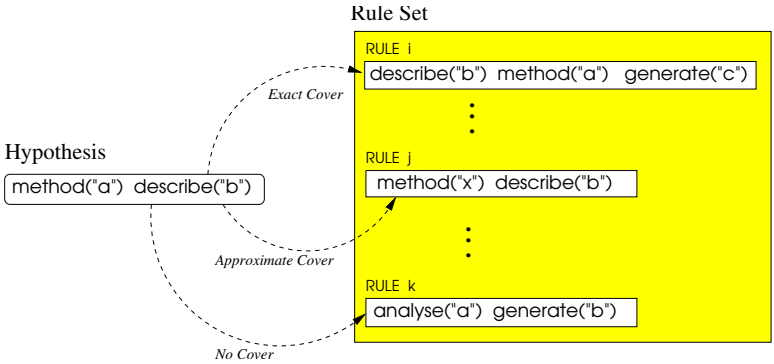


Fig. 9.4. Computing Hypothesis Covering of Documents' rules

Where $SemSim(HP_k, P_j)$ represents the LSA-based similarity between hypothesis predicate HP_k and rule predicate P_j, `threshold` denotes a minimum fixed user-defined value, $RuleSet$ denotes the whole set of rules, HP represents the list of predicates with arguments of H, and P_j represents a predicate (with arguments) contained in R_i. Once the set of rules covered is computed, the criterion can finally be computed as:

$$Coverage(H) = \frac{|RulesCovered(H)|}{|RuleSet|}$$

Where | *RulesCovered* | and | *RuleSet* | denote the size of the set of rules covered by H, and the size of the initial set of extracted rules, respectively.

- **Simplicity** (*How simple is the hypothesis?*): shorter and/or easy-to-interpret hypotheses are preferred. Since the criterion has to be maximised, the evaluation will depend on the length (number of elements) of the hypothesis.
- **Plausibility of Origin** (*How plausible is the hypothesis produced by Swanson's evidence?*): If the current hypothesis was an offspring from parents which were recombined by a Swanson's transitivity-like operator, then the higher the semantic similarity between one parent's consequent and the other parent's antecedent, the more precise is the evidence, and consequently worth exploring as a novel hypothesis. If no better hypothesis is found so far, the current similarity is inherited from one generation to the next.
  Accordingly, the criterion for a hypothesis H is simply given by:

$$Plausibility(H) = \begin{cases} S_p & \texttt{If H was created from a Swanson's crossover} \\ 0 & \texttt{If H is in the original population or is a} \\ & \quad \texttt{result of another operation} \end{cases}$$

Note that since we are dealing with a multi-objective problem, there is no simple way to get independent fitness values as the fitness involves a set of objective functions to be assessed for every individual. Therefore, the computation is performed by comparing objectives of one individual with others in terms of *Pareto dominance* [5] in which non-dominated solutions (Pareto individuals) are searched for in every generation.

We took a simple approach in which an approximation to the Pareto optimal set is incrementally built as the GA goes on. The basic idea is to determine whether a

solution is better than another in global terms, that is, a child is better if this is a becomes a non-dominated hypothesis.

Next, since our model is based on a multi-criteria approach, we have to face three important issues in order to assess every hypothesis' fitness: Pareto dominance, fitness assignment and the diversity problem [5]. Despite an important number of state-of-the-art methods to handle these issues [5], only a small number of them has focused on the problem in an integrated and representation-independent way. In particular, Zitzler [35] proposes an interesting method, *Strength Pareto Evolutionary Algorithm* (SPEA) which uses a mixture of established methods and new techniques in order to find multiple Pareto-optimal solutions in parallel, and at the same time to keep the population as diverse as possible. We have also adapted the original SPEA algorithm to allow for the incremental updating of the Pareto-optimal set along with our steady-state replacement method.

9.4 Analysis and Results

In order to assess the quality of the discovered knowledge (hypotheses) by the model a Prolog-based prototype has been built. The IE task has been implemented as a set of modules whose main outcome is the set of rules extracted from the documents. In addition, an intermediate training module is responsible for generating information from the LSA analysis and from the rules just produced. The initial rules are represented by facts containing lists of relations both for antecedent and consequent.

For the purpose of the experiments, the corpus of documents has been obtained from the *AGRIS* database for agricultural and food science. We selected this kind of corpus as it has been properly cleaned-up, and builds upon a scientific area which we do not have any knowledge about so to avoid any possible bias and to make the results more realistic. A set of 1000 documents was extracted from which one third were used for setting parameters and making general adjustments, and the rest were used for the GA itself in the evaluation stage.

Next, we tried to provide answers to two basic questions concerning our original aims:

a) How well does the GA for KDT behave?
b) How good are the hypotheses produced according to human experts in terms of text mining's ultimate goals: interestingness, novelty and usefulness, etc.

In order to address these issues, we used a methodology consisting of two phases: the system evaluation and the experts' assessment.

a) *System Evaluation:* this aims at investigating the behavior and the results produced by the GA.
   We set the GA by generating an initial population of 100 semi-random hypotheses. In addition, we defined the main global parameters such as *Mutation Probability* (0.2), *Crossover Probability* (0.8), *Maximum Size of Pareto set* (5%), etc. We ran five versions of the GA with the same configuration of parameters but different pairs of terms to address the quest for explanatory novel hypotheses.
   The different results obtained from running the GA as used for our experiment are shown in the form of a representative behavior in figure 9.5, where the

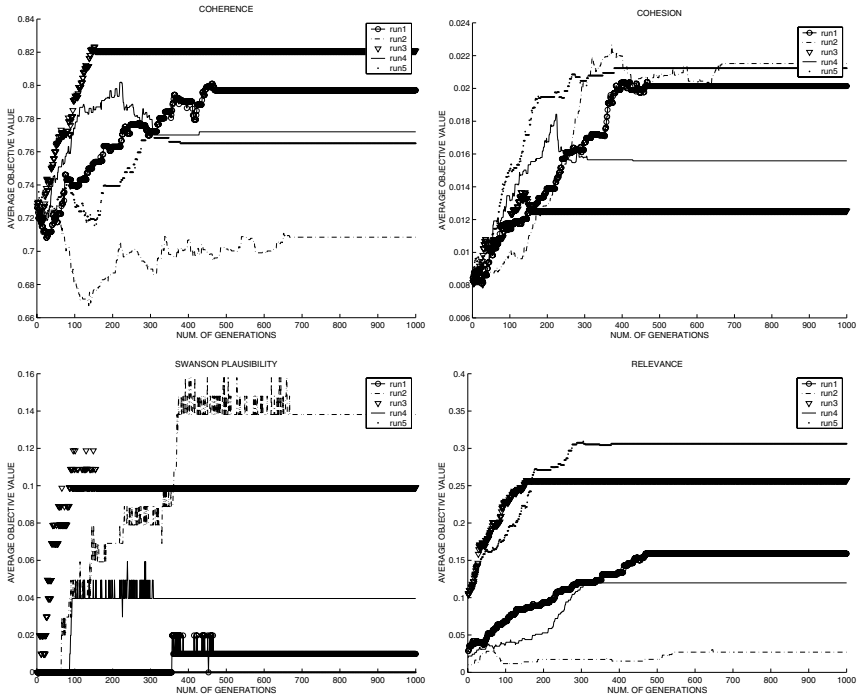


Fig. 9.5. GA evaluation for some of the criteria

number of generations is placed against the average objective value for some of the eight criteria.

Some interesting facts can be noted. Almost all the criteria seem to stabilise after (roughly) generation 700 for all the runs; that is, no further improvement beyond this point is achieved and so this may give us an approximate indication of the limits of the objective function values.

Another aspect worth highlighting is that despite a steady-state strategy being used by the model to produce solutions, the individual evaluation criteria behave in unstable ways to accommodate solutions which had to be removed or added. As a consequence, it is not necessarily the case that all the criteria have to monotonically increase.

In order to see this behavior, look at the results for the criteria for the same period of time, between generations 200 and 300 for run 4. For an average hypothesis, *Coherence*, *Cohesion*, *Simplicity* and *Structure* get worse, whereas *Coverage*, *Interestingness* and *Relevance*, improve and *Plausibility* shows some variability. Note that not all the criteria are shown in the graph.

b) *Expert Assessment:* this aims at assessing the quality (and therefore, effectiveness) of the discovered knowledge on different criteria by human domain experts. For this, we designed an experiment in which 20 human experts were involved and each assessed 5 hypotheses selected from the Pareto set. We then asked the experts to assess the hypotheses from 1 (worst) to 5 (best) in terms of the

following criteria: Interestingness (INT), Novelty (NOV), Usefulness (USE) and Sensibleness (SEN).

In order to select worthwhile terms for the experiment, we asked one domain expert to filter pairs of target concepts previously related according to traditional clustering analysis (see Table 9.1 containing target concepts used in the experiments). The pairs which finally deserved attention were used as input in the actual experiments (i.e., **degradation** and **erosive**).

Run	**Term 1**	**Term 2**
1	enzyme	zinc
2	glycocide	inhibitor
3	antinutritious	cyanogenics
4	degradation	erosive
5	cyanogenics	inhibitor

Table 9.1. Pairs of target concepts used for the actual experiments

Once the system hypotheses were produced, the experts were asked to score them according to the five subjective criteria. Next, we calculated the scores for every criterion as seen in the overall results in Table 9.2 (for length's sake, only some criterion are shown).

The assessment of individual criteria shows some hypotheses did well with scores above the average (50%) on a 1-5 scale. Overall, this supports the claim that the model indeed is able to find *nuggets* in textual information and to provide some basic explanation about the hidden relationships in these discoveries. This is the case for 3 hypotheses in terms of INT, 2 hypotheses in terms of SEN, 5 hypotheses in terms of USE, and 1 hypothesis in terms of NOV, etc.

Criterion	**No. of Hypotheses**	
	Negative $<$ Average	Positive $\geq$ Average
ADD	20/25 (80%)	5/25 (20 %)
INT	19/25 (76%)	6/25 (24 %)
NOV	21/25 (84%)	4/25 (16 %)
SEN	17/25 (68%)	8/25 (32 %)
USE	20/25 (80%)	5/25 (20 %)

Table 9.2. Distribution of Experts' assessment of Hypothesis per Criteria

These results and the evaluation produced by the model were used to measure the correlation between the scores of the human subjects and the system's model evaluation. Since both the expert and the system's model evaluated the results based on several criteria, we first performed a normalisation aimed at producing a single "quality" value for each hypothesis as follows:

- *For the expert assessment:* the scores of the different criteria for every hypothesis[3] are averaged. Note that this will produce values between 1 and 5, with 5 being the best.
- *For the model evaluation:* for every hypothesis, both the objective value and the fitness are considered as follows: whereas the lower the fitness score, the better the hypothesis, the higher the objective value, the better the hypothesis. Therefore, we subtract the fitness from 1 for each hypothesis and then we add this to the average value of the objective values for this hypothesis. Note that this will produce values between 0 and 2, with 2 being the best.

We then calculated the pair of values for every hypothesis and obtained a (Spearman) correlation $r = 0.43$ ($t - test = 23.75, df = 24, p < 0.001$). From this result, we see that the correlation shows a good level of prediction compared to humans. This indicates that for such a complex task (knowledge discovery), the model's behavior is not too different from the experts' (see Figure 9.6).

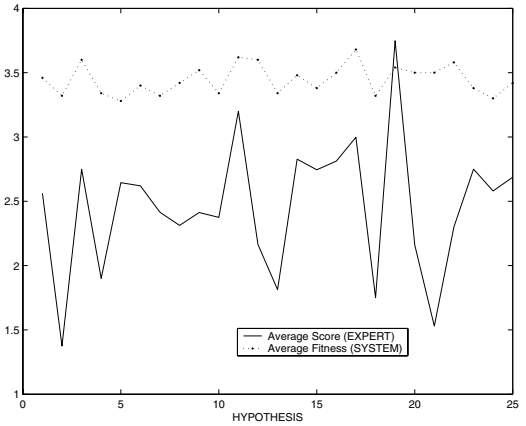


Fig. 9.6. Correlation between human and system evaluation of discovered hypotheses

Note that in Mooney's experiment using simple discovered rules, a lower human-system correlation of $r = 0.386$ was obtained. Considering also that the human subjects were not domain experts as in our case, our results are encouraging as these involve a more demanding process which requires further comprehension of both the hypothesis itself and the working domain. In addition, our model was able to do it better without any external linguistic resources as in Mooney's experiments [26].

In order to show what the final hypotheses look like and how the good characteristics and less desirable features above are exhibited, we picked one of the best hypotheses as assessed by the experts (i.e., we picked one of the best 25 of the 100 final hypotheses) based on the average value of the 5 scores they assigned. For example, hypothesis 65 of run 4 looks like:

[3] ADD is not considered here as this does not measure a typical KDD aspect,

```
IF goal(perform(19311)) and goal(analyze(20811))
THEN establish(111)
```

Where the numerical values represent internal identifiers for the arguments and their semantic vectors, and its resulting criteria vector is $[\mathbf{0.92}, 0.09, 0.50, 0.005, 0.7, 0.00, 0.30, 0.25]$ (the vector's elements represent the values for the criteria relevance, structure, coherence, cohesion, interestingness, plausibility, coverage, and simplicity) and obtained an average expert's assessment of 3.74. In natural-language text, this can roughly be interpreted as (each item of the following NL description represents a predicate-level information of hypothesis above):

- IF the work **aims** at **performing** the genetic grouping of seed populations and investigating a tendency to the separation of northern populations into different classes, AND
- The **goal** is to **analyse** the vertical integration for producing and selling Pinus Timber in the Andes-Patagonia region.
- THEN as a **consequence**, the best agricultural use for land lots of organic agriculture must be **established** to promote a conservationist culture in priority or critical agricultural areas.

The hypothesis appears to be more relevant and coherent than others (relevance = 92%). However, this is not complete in terms of cause-effect. For instance, the methods are missing. It is also important to highlight that the high value for the coherence of the pattern (50%) is consistent with the contents of the predicates of the hypothesis. The three key paragraphs containing rhetorical knowledge indeed relate to the same topic: testing and producing specific Pinus trees. Even more important is the fact that despite having zero plausibility (novelty), the pattern is still regarded as interesting by the model (70%) and above the average by the experts. As for the target concepts (**degradant** and **erosive**) and the way the discovered hypothesis attempts to explain the link between them, it can be seen that the contents of this patterns try to relate these terms with "agricultural areas," "seed populations," etc., so the discovery makes a lot of sense.

Another of the discovered patterns is given by hypothesis 88 of run 3, which is represented as follows:

```
IF goal(present(11511)) AND
   method(use(25511))
THEN effect(1931,1932)
```

and has a criteria vector $[0.29, 0.18, 0.41, 0.030, 0.28, \mathbf{0.99}, 0.30, 0.50]$ and obtained an average expert's assessment of 3.20. In natural-language text, this can roughly be interpreted as:

- IF the **goal** is to **present** a two-dimensional scheme for forest restoration in which two regression models with Pinus and without Pinus are identified by inspiring in the natural restoring dynamics, AND
- The **method** is based on the **use** of micro-environments for capturing the kind of farm mice called *Apodemus Sylvaticus*i, and on the use of capture traps at a rate of 1464 traps per night.

- THEN, in vitro digestion of three cutting ages in six ecotypes has an **effect** on "Bigalta" cuttings which got their higher performance in a 63-day period.

This hypothesis looks more complete (goal, methods, etc.) but is less relevant than the previous hypothesis despite its close coherence. Note also that the plausibility is much higher than for hypothesis 65, but the other criteria seemed to be a key factor for the experts.

The hypothesis concerns the production and cutting of a specific kind of tree (Pinus) and forests where these lie. However, the second role ("the method is based...") discusses a different topic (mice capture) which apparently has nothing to do with the main issue and that is the reason for the pattern's coherence to be scored lower than the previous hypothesis (41% vs. 50%). The model also discovered that there are organisms (and issues related to them) which are affecting the Pinus (and forest) restoration (i.e., mice). This fact has received a higher value for *Plausibility of Origin* or the Novelty of the pattern (99%) and consequently, it is correlated with the experts opinion of the pattern (score=3.20).

Another example of discovered patterns is a low-scored hypothesis given by the following hypothesis 52:

```
IF object(perform(20611)) AND
   object(carry_out(2631))
THEN  effect(1931,1932)
```

and has a criteria vector $[\mathbf{0.29}, \mathbf{0.48}, 0.49, 0.014, 0.2, 0, 0.3, 0.5]$ and obtained an average expert's assessment of 1.53.

The structure of this pattern (48%) is better than for hypothesis 88. However, since the hypothesis is not complete, this has been scored lower than the previous one. This might be explained because the difference in structure between object-object and goal-method (Figure 9.3) is not significant and as both hypotheses (88 and 52) become final solutions, the expert scored best those which better explain the facts. Note that as the model relies on the training data, this does not ensure that every hypothesis is complete. In fact, previous experimental analyses of recall show that only 26% of the original rules representing the documents contain some sort of "method."

In natural-language text, the pattern can roughly be interpreted as:

- IF the **object** of the work is to **perform** the analysis of the fractioned honey in Brazil for improving the producers' income and profitability, AND
- The **object** of the work is to **carry out** observations for the study of Pinus hartwegii at the mexican snowed hills so to complement the previously existing information about the development status of *Adjunctus* and its biology.
- THEN in vitro digestion of three cutting ages in six ecotypes has an **effect** on bigalta cuttings which got their higher performance in a 63-day period.

This hypothesis shows the same relevance as the previous one (29%) indicating that both attempt to explain the connection between the target concepts and that contained in the pattern. Note also that coherence is not very high (49%) considering that one part of the pattern discusses the "honey production" and issues, and the other parts deal with the investigation, production and cutting of Pinus trees. Accordingly the degree of interestingness and novelty of the patterns has been low

scored which is well-correlated with the expert's assessment. Nevertheless, the hypothesis is successful in detecting hidden relations between certain areas (*Mexican snowed hills*) and the Pinus production and cutting.

9.5 Conclusions

Unlike traditional approaches to Text Mining, in this chapter we contribute an innovative way of combining additional linguistic information and evolutionary learning techniques in order to produce novel hypotheses which involve explanatory and effective novel knowledge.

From the experiments and results, it can be noted that the approach supports the claim that the evolutionary model to KDT indeed is able to find *nuggets* in textual information and to provide basic explanations about the hidden relationships in these discoveries.

We also introduced a unique approach for evaluation which deals with semantic and Data Mining issues in a high-level way. In this context, the proposed representation for hypotheses suggests that performing shallow analysis of the documents and then capturing key rhetorical information may be a good level of processing which constitutes a trade off between completely deep and keyword-based analysis of text documents. In addition, the results suggest that the performance of the model in terms of the correlation with human judgements are slightly better than approaches using external resources as in [26]. In particular criteria, the model shows a very good correlation between the system evaluation and the expert assessment of the hypotheses.

The model deals with the hypothesis production and evaluation in a very promising way which is shown in the overall results obtained from the experts evaluation and the individual scores for each hypothesis. However, it is important to note that unlike the experts who have a lot of experience, preconceived concept models and complex knowledge in their areas, the system has done relatively well only exploring the corpus of technical documents and the implicit connections contained in it.

From an evolutionary KDT viewpoint, the correlations and the quality of the final hypotheses show that the GA operations and the system's evaluation of the individuals may be effective predictions of really useful novel knowledge from a user perspective.

References

1. A. Bergstron, P. Jaksetic, and P. Nordin. Acquiring Textual Relations Automatically on the Web Using Genetic Programming. *EuroGP 2000, Edinburgh, Scotland*, pages 237–246, April 2000.
2. Michael Berry. *Survey of Text Mining: Clustering, Classification, and Retrieval.* Springer, 2004.
3. M. Berthold and D. Hand. *Intelligent Data Analysis.* Springer, 2000.
4. C. Coello. A Short Tutorial on Evolutionary Multiobjective Optimisation. *ACM Computing Surveys*, 2001.

5. Kalyanmoy Deb. *Multi-objective Optimization Using Evolutionary Algorithms*. Wiley, 2001.
6. U. Fayyad, G. Piatesky-Shapiro, and P. Smith. From Data Mining to Knowledge Discovery: An Overview. In *Advances in Knowledge Discovery and Data Mining*, pages 1–36. MIT Press, 1996.
7. R. Feldman. Knowledge Management: A Text Mining Approach. *Proc. of the 2nd Int. Conference on Practical Aspects of Knowledge Management (PAKM98), Basel, Switzerland*, October 1998.
8. R. Feldman and I. Dagan. Knowledge Discovery in textual databases (KDT). *Proceedings of the first international conference on knowledge discovery and data mining (KDD-95), Montreal, Canada*, pages 112–117, August 1995.
9. C. Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.
10. P. Foltz, W. Kintsch, and T. Landauer. The Measurement of Textual Coherence with Latent Semantic Analysis. *Discourse processes*, 25(2):259–284, 1998.
11. C. Fonseca and P. Fleming. An Overview of Evolutionary Algorithms in Multiobjective Optimisation. *Evolutionary Computation*, 3(1):1–16, 1995.
12. A. Freitas. A Survey of Evolutionary Algorithms for Data Mining and Knowledge Discovery. *Advances in Evolutionary Computation, Springer-Verlag*, 2002.
13. D. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley, 1989.
14. J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan-Kaufmann, 2001.
15. S. Harabagiu and D. Moldovan. Knowledge processing on an extended wordnet. In *WordNet: An Electronic Lexical Database*, pages 379–403. MIT Press, 1998.
16. M. Hearst. Automated Discovery of WordNet Relations. In *WordNet: An Electronic Lexical Database*, pages 131–151. MIT Press, 1998.
17. M. Hearst. Text Data Mining: Issues, Techniques and the Relation to Information Access. Technical report, Univerity of California at Berkeley, 1998.
18. M. Hearst. Untangling Text Data Mining. *Proceedings of the 37th Annual Meeting of the ACL, University of Maryland (invited paper)*, June 1999.
19. M. Hearst. Text Mining Tools: Instruments for Scientific Discovery. *IMA Text Mining Workshop, USA*, April 2000.
20. C. Jacquemin and E. Tzoukermann. NLP for Term Variant Extraction: Synergy between Morphology, Lexicon, and Syntax. In *Natural Language Information Retrieval*. Kluwer Academic, 1999.
21. W. Kintsch. Predication. *Cognitive Science*, 25(2):173–202, 2001.
22. T. Landauer, P. Foltz, and D. Laham. An Introduction to Latent Semantic Analysis. *Discourse Processes*, 10(25):259–284, 1998.
23. C. Manning and H. Schutze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.
24. M. Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, 1996.
25. U. Nahm and R. Mooney. Using Information Extraction to Aid the Discovery of Prediction Rules from Text. *Proceedings of the 6th International Conference on Knowledge Discovery and Data Mining (KDD-2000) Workshop on Text Mining*, August 2000.
26. U. Nahm and R. Mooney. Text Mining with Information Extraction. *AAAI 2002 Spring Symposium on Mining Answers from Texts and Knowledge Bases, Stanford, USA*, 2002.

27. M. Rajman and R. Besancon. Text Mining: Knowledge Extraction from Unstructured Textual Data. *6th Conference of the International federation of classification societies (IFCS-98), Rome, Italy*, July 1998.
28. P. Srinivasan. Text mining: Generating hypotheses from medline. *Journal of the American Society for Information Science*, 55(4):396–413, 2004.
29. W. Stadler. *Fundamentals of Multicriteria Optimization*. Plenum Press, New York, 1988.
30. D. Swanson. Migraine and Magnesium: Eleven Neglected Connections. *Perspectives in Biology and Medicine*, n/a(31):526–557, 1988.
31. D. Swanson. On the Fragmentation of Knowledge, the Connection Explosion, and Assembling Other People's ideas. *Annual Meeting of the American Society for Information Science and Technology*, 27(3), February 2001.
32. M. Weeber, H. Klein, L. de Jong, and R. Vos. Using concepts in literature-based discovery: Simulating swanson's raynaud-fish oil and migraine-magnesium discoveries. *Journal of the American Society for Information Science*, 52(7):548–557, 2001.
33. P. Wiemer-Hastings. Adding Syntactic Information to LSA. *Proceedings of the Twenty-second Annual Conference of the Cognitive Science Society*, pages 989–993, 2000.
34. G. Williams. Evolutionary Hot Spots Data Mining. *3rd Pacific-Asia Conference, PAKDD-99, Beijing, China, April*, pages 184–193, 1999.
35. E. Zitzler and L. Thiele. An Evolutionary Algorithm for Multiobjective Optimisation: The Strength Pareto Approach. Technical Report 43, Swiss Federal Institute of Technology (ETH), Switzerland, 1998.