Methods for Retrieving Alternative Contract Language Using a Prototype

Silviu Pitis
College of Computing
Georgia Institute of Technology
Atlanta, GA, USA 30332
spitis3@gatech.edu

ABSTRACT

This paper addresses the problem of searching for alternative contract language that is similar to, yet different from, a given provision (the prototype). While this is a core task in transactional legal work, generic search solutions do not offer an effective solution. We draw upon modern information retrieval research to propose and validate novel methods for retrieving alternative language using a prototype. Our solution accepts an entire provision as a prototype and retrieves variants on the language from a database of precedent contracts. In designing this solution, we propose two ordered proximity measures and demonstrate their effectiveness relative to existing techniques. Further, we examine the challenge posed by varying definitions of redundant search results and propose to resolve it with a user-tunable, dynamic approach to result clustering.

CCS CONCEPTS

• **Information systems** → *Retrieval models and ranking*; *Retrieval tasks and goals*; *Users and interactive retrieval*; • **Applied computing** → *Law*;

KEYWORDS

Transactional law, passage retrieval, contract clause retrieval, novelty detection, search result clustering

ACM Reference format:
Silviu Pitis. 2017. Methods for Retrieving Alternative Contract Language Using a Prototype. In *Proceedings of ICAIL '17, London, United Kingdom, June 12-16, 2017,* 9 pages.
https://doi.org/http://dx.doi.org/10.1145/3086512.3086530

1 INTRODUCTION

Searching for alternative language in a database of documents is an important and oft-repeated task in the day-to-day practice of transactional law. For instance, during the drafting and negotiation of a key contract provision, lawyers on both sides might spend hours researching similar provisions from precedent contracts in order to support their negotiating stance or find alternative wordings that are beneficial to their respective clients. When a lawyer's experience and personal collection of past deal documents fail to produce the required precedent for such a task, the lawyer typically consults a database of precedent, such as his or her firm's document management system.

The status quo for searching such a database, however, is not a specialized solution. Leading document management systems typically offer only generic search features that are not effective for locating alternative contract language.[1] For example, if you have access to such a contract database, you might try searching now for variations of the following provision (from a Registration Rights Agreement):

> The Company will use its best efforts to confirm that the rating of the Initial Securities obtained prior to the initial sale of such Initial Securities will also apply to the Securities covered by a Registration Statement. (A)

Generic search solutions do not return useful results when using the entire provision as a query. A phrase search retrieves only exact matches, and a regular search (e.g., ranked retrieval using Okapi BM25) typically retrieves many irrelevant (and often duplicate) results. Lawyers might instead resort to querying sub-phrases such as "confirm that the rating of the Initial Securities". While this may ameliorate the problem with full phrase search, it is not ideal. Crafting such sub-queries is time-consuming, and is still bound to miss relevant results and produce duplicates.

The contribution of this paper is twofold. First, we have identified a ubiquitous, yet unaddressed, search problem in the domain of transactional law, which we hope will spur legal technology providers to offer tailored solutions. Second, we draw upon modern information retrieval (IR) research to propose and validate novel methods for retrieving alternative language using a prototype. Our solution accepts an entire provision as a prototype and retrieves variants on the language from a database of precedent contracts. In designing this solution, we propose two ordered proximity measures and demonstrate their effectiveness relative to existing techniques. Further, we examine the challenge posed by varying definitions of redundant search results and propose to resolve it with a user-tunable and dynamic approach to result clustering.

We proceed as follows: in Section 2, we provide further motivation and context for the problem. In Section 3, we describe

ICAIL '17, June 12-16, 2017, London, United Kingdom

ACM ISBN 978-1-4503-4891-1/17/06…$15.00
https://doi.org/http://dx.doi.org/10.1145/3086512.3086530

[1] A small subset of providers have begun to recognize the importance of search for contract language. See, e.g., Lexis for Microsoft Office, Westlaw Sample Agreements and RealDealDocs, which offer model precedent and clause lookup.

the problem and discuss the aspects that may cause difficulties when implementing a solution. In Section 4, we outline the past research upon which we built our proposed solution. In Section 5, we describe our approaches to ranked retrieval, the data used to evaluate them, and our empirical results. In Section 6, we describe our approach to clustering redundant results, and in Section 7, we conclude.

2 PROBLEM MOTIVATION

2.1 For Drafting and Negotiation

Consider the process of drafting and negotiating an acquisition agreement involving two parties—a buyer and a seller. The initial drafting is the responsibility of a single party and typically begins with a copy of the final agreement from a past transaction, which is updated to reflect the current negotiation. This update will include changing the party names, deal-specific changes (to account for differences in deal structure, tax attributes, company characteristics, etc.), any tentative agreements reached during preliminary negotiations, and any additional changes the lawyers deem favorable to their client. Once prepared, the initial draft is communicated to the other party who, now holding the pen, strikes and replaces unfavorable language and counters with a redraft. This process repeats until the parties find a mutually agreeable middle ground and a final contract is executed.

Repeatedly during this process, the lawyers are faced with a problem: there is undesirable language that needs to be changed. As an example all readers can relate to, suppose you are a prospective tenant and the landlord has proposed a lease with a clause that allows them to enter the apartment at any time with or without your permission. You want to propose a change to this language, but not being a lawyer who specializes in landlord-tenant law, you're not sure what a fair alternative would be (it can't be that the landlord is absolutely denied entry at all times). What alternate language do you propose?

The lawyer's solution is not usually to craft new language, but rather to find old language that fits the bill. Many situations, such as the simple landlord-tenant case just described, are so common that the experienced lawyer can provide the desired language from memory, or otherwise immediately dig up the right precedent. But for other situations, the lawyer is unaware of the relevant precedent. Thus, a junior lawyer ends up tasked with the following assignment: "Please locate precedent agreements that have this *type of* provision and see how others have dealt with this issue."

2.2 For Administration and Due Diligence

A firm that contracts with many similar parties will often need to deviate from its form agreement. For example, a commercial landlord might enter into hundreds of slightly different leases in order to accommodate the unique needs of its lessees, or a private equity fund might enter into hundreds of slightly different side agreements in order to accommodate the unique needs of each investor. For various administrative reasons it is often useful to know which such contracts contain a specific version of a given provision, or how many legally distinct versions of a given provision there are. In some cases, e.g., in satisfaction of a *most favored nations* clause[2] or due diligence request, there may be a legal obligation to procure such knowledge. Even in the absence of such obligation, the failure to find a relevant variant on a provision, e.g., during due diligence, could still result in a bad business outcome.

3 PROBLEM DESCRIPTION

In both the drafting and administrative scenarios discussed above, we would benefit from the ability to retrieve relevant provisions from a database using only a prototype provision as the query. Results from such a query should be ranked by similarity to the prototype, whilst simultaneously avoiding or clustering near duplicates. Consistent with past work involving retrieval of novel results and search result clustering [7, 36, 43], we divide this task into two components: a method to retrieve relevant results, and a method to reorder or cluster the results so as to avoid redundancy and promote diversity. The transactional law context, however, introduces nuances in both components.

One important nuance lies in the definition of "relevance". The usual meaning has been topical in nature; queries for TREC ad-hoc tasks, for instance, are formulated from TREC "topics" such as "law enforcement, dogs" [4]. When retrieving alternate contract language, there is also a strong *structural* component to "relevance". During a negotiation, lawyers are wary of introducing too much red ink to an existing draft. If a provision must be changed, light changes are always preferred—the provision is rarely replaced in its entirety. For this reason, the retrieval component should involve some amount of partial or fuzzy phrase detection so as to favor results with a smaller Levenshtein (edit) distance [22] to the query.

A second nuance lies in the amount and character of duplication in transactional databases [12]. As discussed in Section 2.1, new contracts are based on old contracts and often introduce small tweaks to the language. This results in numerous nearly identical, yet slightly different, provisions. Important differences are not necessarily lexical; they can be grammatical in nature and can amount to only a single changed character—an added comma or capitalized word can change a legal obligation. This immediately suggests that redundancy measures considering only token-level similarity (e.g., [43]) are insufficient. At the same time, an algorithm that does nothing but filter strict duplicates would be inadequate—most small changes are irrelevant and a diverse set of results is more useful to lawyers than a set of passages with only marginal differences. This is particularly true as the purpose of looking up variations of the same provision—and the very definition of a redundant result—will vary from case to case (cf. "dynamic relevance" [3]).

Finally, it is worth emphasizing that the prototype provisions used as queries may be quite long, containing up to 100 or more tokens. An effective solution must be capable of handling long queries efficiently. A conjunctive (AND) query is clearly inappropriate, but a disjunctive (OR) query may be slow to evaluate on larger databases. Moreover, the nature of legal writing makes it so that *most* query terms are extremely common. Consider provision A quoted in Section 1. If tokenized with a standard list of stop words, 20 out of 22 tokens appear in more than 40% of the contracts in

[2] A *most favored nations* (MFN) clause, named for its use in international trade, is common in, e.g., the private funds sphere. The recipient of MFN privileges is treated as the "most favored" contracting party, and is entitled to receive benefits from the grantor that are no worse than the benefits granted to any similarly situated party.

our dataset (described in Section 5). Even the rarest token, 'rating', appears in 12% of our contracts (the most common, 'such', appears in a whopping 94% of our contracts). This characteristic of "legalese" only serves to increase the importance of structural aspects of the query to relevance.

4 RELEVANT RESEARCH

Our proposal draws upon significant bodies of research for each of the two components discussed above.

4.1 Ranked Retrieval

Work on passage retrieval [5, 20, 21, 23, 33] has studied the scoring and retrieval of sub-documents (passages), as opposed to usual document-oriented approaches. Various types of passages have been explored, including those defined by document structure (e.g., sections or paragraphs) [5, 33] and those defined by fixed-length windows [5, 20]. Passage retrieval is a natural approach when the desired results are in fact passages, as is the case here—identification of the most relevant provision in each document will be automatic if scoring is done by provision. On the downside, this approach may be more expensive, as an order of magnitude more passages than documents must be scored. For this reason, it has been suggested that ranked retrieval be divided into two stages: first scoring documents, and then scoring only those passages within the top-k documents [5]. This runs the risk of missing certain relevant passages whose containing document does not rank highly. Passage retrieval is also not a complete solution: scoring passages with traditional *bag-of-words* scoring functions that assume term-wise independence (as in, e.g., [20, 23]) would ignore the structural component of relevance highlighted in Section 3.

A simple yet effective method for capturing the query structure is to use bigrams or higher order word n-grams instead of the standard unigram approach [4, 10, 27, 37]. Using bigrams, the query "net book value" would be tokenized as ['net book', 'book value'] in addition to (or instead of) the usual unigrams ['net', 'book', 'value']. Since the occurrence of higher order n-grams in documents is much sparser than the occurrence of individual words, the co-occurrence of multiple n-grams from a single query is strongly indicative of a partial phrase match or text reuse [10, 26]. Higher order n-grams have been used to detect near duplicate documents in large databases [12, 16], to detect passage-level text reuse [34, 35], and, in the legal context, to trace the development of legislative language [41]. One disadvantage of using higher order n-grams is that it greatly expands the size of the indexed vocabulary,[3] which has led to the development of specialized indexing methods [17]. Another disadvantage is that n-grams are composed of immediately adjacent terms and cannot accommodate certain "proximate" matches. For example, one would want a search for "aggregate net book value" to rank "aggregate book value" higher than "aggregate net sales", yet both results share only a single bigram with the query.

Within the legal domain, Rosso et al. [32] show that n-gram based re-ranking of retrieved passages can be effective for a number of tasks, including answering questions about legislation, searching a patent database, and retrieving conflicting contract provisions. By contrast, our work addresses the distinct problem of retrieval by prototype and investigates the process of ranked retrieval itself rather an algorithm for re-ranking search results.

A significant body of works examines other methods of modeling term-wise dependence, with particular focus on the proximity of query terms within results. While many approaches are probabilistic [25, 28, 44], others are heuristic [14, 31, 39]. The latter group, which employ ad-hoc structures such as covers or spans (i.e., sequences of tokens in a document that "cover" or "span" some set of query terms) [8, 38], or alternatively, measures of distance between query terms in scored documents [31, 39], are more relevant to our work given that we seek to model a precise query structure. Nevertheless, due to the topical nature of relevance in traditional IR (as opposed to the structural component that we are interested in), few researchers have examined *order* in addition to proximity (among the few are [2, 28]).

Finally, we note that the structural component to relevance bridges the gap between topical IR and approximate string matching [29]. For instance, Okazaki and Tsujii [30] describe and implement SimString, an efficient, open-source algorithm for locating approximate strings in large datasets. Approximate string matching generally utilizes a character-level n-gram index [30], and considers the relevance of results to be solely based on various measures of string distance (e.g., Levenshtein distance [22]). Character-level indexing offers more granularity than necessary for the task we address, while measures of string distance potentially ignore the topical component of relevance that is still present.

4.2 Novelty Detection and Result Clustering

IR researchers have proposed a variety of methods for introducing novelty and avoiding redundancy in search results [1, 6, 9, 43], typically as a follow-up step that re-orders or filters results from ranked retrieval [36]. By and large, novelty and redundancy have been treated as "opposite ends of a continuous scale" [1, 43], where the novelty of a result is defined as the new information it contains relative to, or its dissimilarity to, higher ranked results [6, 43].

Numerous measures of similarity have been proposed for purposes of novelty detection, including ones based on word count, set difference, geometric distance, and distributional similarity [1, 43]. One might even use the same similarity measure as used for ranked retrieval [6]. Measures intended to detect paraphrasing [13, 19], which include string similarity measures like Levenshtein distance [22] and n-gram overlap, may also prove useful.

With respect to transactional law specifically, Conrad and Raymond [12] have examined the problem of deduplicating provisions returned by a search in a contract database. Although the context of our work overlaps with [12], there are significant differences in both the core problems addressed and our proposed solutions. We specifically target searches for variations on a prototype provision, using that prototype as the query. By contrast, [12] use queries formed by real users of their database, for which the search intent is unclear. Indeed, judging by the sample queries provided in [12], which include "put option" and "sale of collateral", it is unlikely that most users were searching for variations on language. This is further reflected by the definition of a "duplicate" result used in

[3] E.g., Büttcher et al. [4] report that the 426GB TREC GOV2 collection contains 4.4×10^7 unique words, but 5.2×10^8 unique bigrams and 2.3×10^9 unique trigrams

[12], which states that "two texts are duplications if they retain much of the same language and are at least 90% similar". In our view, such a rigid definition strictly fails for our purposes; as stated in Section 3 and further expanded upon in Section 6, the definition of redundancy may shift from search to search, and even a single character difference can be of interest to a lawyer. For this reason, our proposed solution, which relies on result clustering, is of a different character than that of [12].

In recognition of the multidimensional concept of relevance [3], Clarke et al. [9] distinguish diversity from novelty: diversity refers to results from different topics, whereas novelty refers to results that contribute new information about a topic already covered by higher ranked results. Although result diversity can be treated by the strategic ordering of a flat result list, as suggested by [9], work on search result clustering provides a relevant alternative to the flat list that organizes results into clusters of similar results [7]. A variety of approaches and algorithms for clustering have been proposed (see [7] for a detailed survey), wherein clusters have generally been organized by topic (e.g., [15, 42]). Clustering has also been examined within the legal domain, but at the document level (pre-retrieval), which is less relevant to our work [11, 24].

5 METHODS FOR RANKED RETRIEVAL

5.1 Data

To compare and test the various methods proposed in this section, we have compiled a dataset of 20,236 contracts that were obtained by crawling the U.S. Securities and Exchange Commission (SEC) EDGAR database of public company filings. These contracts were uploaded by companies with public filing obligations under the Securities Act of 1933 and the Securities Exchange Act of 1934, who have an obligation to file copies of certain contracts as exhibits to their public filings. This includes copies of "material" contracts, as described in Item 601 of Regulation S-K. During our crawl we found over 1 million (not necessarily unique) contracts. Our chosen subset consists only of those contracts for which a title was easily extracted from the main filing they were attached to, and whose title included one of the terms "merger", "purchase", "asset" or "acquisition".

We created both a unigram and bigram index of the selected contracts. Contracts retrieved in html format were stripped of the html markup and converted to plain text before indexing. Tokenization involved stripping punctuation, passing the text through a lowercase filter, removing a standard list of stop words, and, in the case of the bigram index, applying a Porter stemmer to reduce bigram sparsity.

5.2 Task

We evaluated the methods by comparing the top 10 search results for each of 20 tested queries. Queries were chosen to be diverse, and took the form of a complete or partially complete provision from a contract within the dataset. Provision A from Section 1 is one of the shorter queries used, with the average query being 89 words long.

Results were compared by measuring the normalized discounted cumulative gain (nDCG) [18] of the top 10 results. For purposes of computing gain vectors, we graded the results according to five relevance categories:

- returned passage is an exact match (5 points)
- returned passage is a close match; e.g., "best efforts" has been replaced by "commercially reasonable efforts", insignificant clauses have been added or deleted, or similar minor modifications (4 points)
- returned passage is a partial match or otherwise shares significant language with the query; e.g., part of the query is present, significant additional parentheticals or other contextual elements have been added (3 points)
- returned passage is topically relevant to the query but there is no significant text reuse (2 points)
- returned passage is irrelevant (0 points)

Results were returned together with the surrounding context (i.e., there was no automatic paragraph or section segmentation), and irrelevant context was ignored when grading the relevance of the result. Table 1 provides examples of results (without any irrelevant context) graded at the 4 point, 3 point and 2 point levels in response to the query consisting of provision A from Section 1.

As this experiment was focused solely on ranked retrieval, we did not penalize the gain contributions of duplicate results. The kth slot in each gain vector was discounted by dividing by $log_2(1 + k)$. The ideal gain vector, for purposes of normalization, was obtained by sorting the set of all graded results by relevance.

5.3 Methods

In all, we tested four different scoring algorithms, each using both a unigram index and a bigram index, for a total of eight distinct approaches to ranked retrieval.

Our baseline scoring algorithm was a standard implementation of Okapi BM25 (as reported in [4]), evaluated at the document level. Multiple occurrences of the same term in the query were treated (here and throughout) as distinct terms. For the reasons outlined in Section 3, we expected poor results using BM25 on the unigram index, even though we believe this to be comparable to the generic search solutions available to transactional lawyers. As document-level BM25 returns documents and not passages, we ran an additional passage-level evaluation to find the top ranking passage in each document for purposes of relevance grading.

Second, we tested passage retrieval on fixed-width windows with a simplified BM25-like scoring function. The width of the passage window was set at query-time to equal 1.5 times the query length (to allow for added language mid-passage), and our algorithm allowed passages to start on arbitrary tokens (see arbitrary passage retrieval in [20, 21]). To be precise about the scoring function used, given a query, q, and a passage, p, contained in a document, d, the passage was scored with the formula:

$$\sum_{t \in q} \log\left(\frac{N}{N_t}\right) \cdot \frac{2.2 f_{t,p}}{f_{t,p} + 1.2} \tag{1}$$

where t is a term in query q, N is the total number of documents in the collection, N_t is the number of documents containing term t, and $f_{t,p}$ is the frequency of term t in passage p. This formula reflects BM25 with parameter $k = 1.2$ applied at the passage-level, but using an inverse document frequency ($\log(N/N_t)$) computed at the document level, which is justified on the basis that we are only considering the top passage from each document (i.e., we will never include two passages from the same document in the results).

Relevance grade	Blackline of result against provision A
4 points	The Company will use ~~its best~~commercially reasonable efforts to confirm that the rating of the Initial Securities obtained prior to the initial sale of such Initial Securities will also apply to the Securities covered by a Registration Statement.
3 points	The Company will use its best efforts (i) if the Securities have been rated prior to the initial sale of such Securities, to confirm ~~that the~~such ratings ~~of the Initial Securities obtained prior to the initial sale of such Initial Securities~~ will also apply to the Securities or the New Securities, as the case may be, covered by a Registration Statement; or (ii) ….
2 points	[E]ach of the issuers shall: … in the case of a shelf registration, use its reasonable best efforts to cause the transfer restricted securities covered by the registration statement to be rated with the appropriate rating agencies, if so requested by the holders of a majority …

Table 1: Samples of relevance graded results using provision A as the query

For our third and fourth scoring algorithms, we propose two novel heuristic measures of proximity, one based on term-wise distance [31, 39] and the other based on covers [8, 39]. Unlike prior measures, our proposals explicitly model the *order* of query terms so as to maintain the query's structure. Below, we refer to the following sample document adapted from [39] to illustrate our proposed measures:

$$d_{\text{sample}} = t_1, t_2, t_1, t_3, t_5, t_6, t_2, t_3, t_4 \tag{2}$$

A high score with these measures is strongly suggestive of a matching passage even when scoring is done at the document-level, and so we did not evaluate them at the passage-level during ranking (though this is possible, at the price of some additional computation). Therefore, as with document-level BM25, we ran an additional passage-level evaluation to find the top ranking passage in each document for purposes of relevance grading.

5.3.1 Position-adjusted minimum distance. This measure, inspired by Tao and Zhai's *MinDist* [39], computes the minimum position-adjusted distance between pairs of matched query terms. For a pair of matched query terms, (t_i, t_j), in a document, d, we define this measure, $\delta((t_i, t_j), d)$, as the difference between their directed minimum distance in the document and their directed distance in the query. It can be efficiently computed by first adjusting the positions vector[4] of one term by the distance in the query and then computing the minimum distance as in [39]. The minimum for each pair of terms is taken across all matched pairs in the document. For instance, given a query $q = t_1, t_2, t_3$ and the sample document (2), the minimum position-adjusted distances between the pairs (t_1, t_2), (t_1, t_3), and (t_2, t_3) are 0, 1, and 0 respectively.

As the time complexity of computing distances for all pairs of terms in an n-term query is $O(n^2)$ and typical queries might be very long (see discussion in Section 3), we restrict scoring to terms that are adjacent in the ordered list of matched terms for a document, which is $O(n)$. To improve upon the limited usefulness of measuring the position-adjusted distance between pairs of overlapping bigrams (as adjacent bigrams often are), we expand the definition of adjacent bigram tokens to include tokens separated by a third token. Thus, if all terms in the query t_1, t_2, t_1, t_3 (which is tokenized as t_1-t_2, t_2-t_1, t_1-t_3) are matched, the token t_1-t_2 is considered adjacent to t_1-t_3 in addition to the usual t_2-t_1. Note that our definition of adjacency allows tokens (both unigram and bigram) to be adjacent in the list of matched terms without being adjacent in the query.

To turn this distance measure into a document score, we use the following function (cf. π from [39]):

$$\pi(q, d) = \sum_{(t_i, t_j) \in q} \max\left(0, k - \delta((t_i, t_j), d)\right) \tag{3}$$

where each (t_i, t_j) is a pair of query terms that are adjacent in the list of matched terms and k is a distance threshold. In our experiments we set $k = 3$.

Using position-adjusted minimum distance to score documents indexed with unigrams can be thought of as using "fuzzy" bi-grams; given a query of "aggregate book value", the phrase "aggregate net book value" scores higher than the phrase "book value", even though both share the same single bigram with the query. When used with a bigram index, position-adjusted minimum distance simulates the effect of using even higher-order fuzzy n-grams.

5.3.2 Maximum ascending m-cover. This measure, inspired by the cover and span based approaches of [8, 39] (among others), scores a document based on the size of its maximum ascending m-cover. We define an m-cover as a span of tokens in a document that contains or "covers" some a set of m distinct query terms, with repeat query terms treated as being distinct. An *ascending* m-cover is an m-cover whose m matched query terms are in order of their appearance in the query. Using this as a measure for ordered proximity only makes sense if ascending m-covers are limited to a certain length (in tokens), which we set to equal twice the query length. The size of an m-cover, used for scoring and taking the maximum, refers to the magnitude of m, and not the cover's length in tokens.

For example, given the query t_1, t_3, t_4, the sample document (2) has five 2-covers, but no 3-covers (the span from positions 3 to 9

[4]The positions vector of a term within a document lists the positions at which the term appears in that document. The positions vector of t_1 in the sample document (2) is [1, 3].

Method	nDCG
Document-level BM25 (unigram)	0.613 ± 0.106
Document-level BM25 (bigram)	0.953 ± 0.030
Passage retrieval (unigram)	0.929 ± 0.057
Passage retrieval (bigram)	0.990 ± 0.007
Position-adj. min dist. (unigram)	0.950 ± 0.027
Position-adj. min dist. (bigram)	0.989 ± 0.011
Max ascending m-cover (unigram)	0.945 ± 0.034
Max ascending m-cover (bigram)	0.977 ± 0.021

Table 2: nDCG by retrieval method (± 95% confidence)

would be a 3-cover, except that it exceeds the maximum m-cover length of $2 \times 3 = 6$).

Our algorithm for computing the maximum ascending m-cover merges the positions vectors for n query terms using a min-heap, and then iterates through the merged positions, keeping track, for each of the n query tokens, the maximum ascending m-cover ending in that token within the last s tokens, where s is the maximum m-cover length in tokens. With respect to n, the merge has time complexity $O(n \log n)$, and the iteration has time complexity $O(n^2)$. As with the computation of position-adjusted minimum distance for all pairs of query terms, complexity that is polynomial in n is unacceptable given the long expected query length.

Therefore, instead of computing the maximum ascending m-cover with respect to all query tokens, we split the tokens into non-overlapping groups of length not less than k, and sum the scores from each group. In our case we set $k = 5$. If g_i represents a group of query terms $t_i \dots t_{i+k}$, and $\text{score}(g_i)$ is the maximum ascending m-cover with respect to the sub-query $t_i \dots t_{i+k}$, then the score of a document d given the full query q is computed as:

$$\pi(q, d) = \sum_{g_i \in q} \text{score}(g_i) \tag{4}$$

Each group is scored in constant time with respect to n, so that the grouped scoring algorithm has a total time complexity with respect to n of $O(n)$. Note that there is no guarantee that grouped covers will be local to a single passage. On the other hand, if A and B are phrases, then given the query AB, the grouped approach will score the document BA higher than would the non-grouped approach.

5.4 Results

Table 2 shows the mean nDCG of the top 10 results for each of the 8 methods over the 20 tested queries, together with 95% confidence intervals that were computed using a t distribution. As expected, document-level BM25 using a unigram index, which we believe is representative of current generic search tools available to transactional lawyers, performed very poorly. Every other method obtained respectable results.

Among methods using a unigram index, position-adjusted minimum distance achieved the best results. Even though it does not consider the inverse document frequency of terms and is not strictly local (minimum distances can occur anywhere in a document), it beat unigram-based passage retrieval (although not by a statistically significant margin). This reflects the structural component to relevance discussed in Section 3: while unigram-based passage retrieval considers proximity of terms, it does not also consider order.

This structural component of relevance is even more clearly illustrated by the sizable gap between the performance of a unigram and bigram index. Applying paired t-tests shows that the gap is statistically significant for all four methods.

Using a bigram index had the additional benefit of allowing queries to be executed an order of magnitude faster than with a unigram index. This is due to the relative sparsity of bigrams. Consider again our comment from Section 3 that 20 out of 22 unigrams in Provision A appear in more than 40% of the contracts in our dataset. By contrast, the most common (stemmed) bigram in Provision A is 'registr-statement', which appears in 32% of the contracts in our dataset. 15 out of 21 bigrams in Provision A appear in less than 10% of contracts—a welcome change that significantly improves query time.

We note that using a bigram index alone, as we did for our tests, may hurt the precision of searches when considering longer result lists (e.g., precision with respect to the top 100 results). For example, the 2 point result from Table 1 shares only five bigrams with the Provision A, three of which are quite common (appearing about 30% of contracts). For this reason, a mixture of unigram and bigram models may be best in practice. The faster query time of a bigram index could still be leveraged by executing and returning the results of the bigram and unigram searches asynchronously. As our experiments show that bigram searches produce near-ideal top search results, the user will not have to wait for the unigram search to complete in order to start using the results.

On the topic of query efficiency, it is worth noting that while our implementation of arbitrary passage retrieval had acceptable query time, it was noticeably slower than other methods. For this reason, we are tentative about recommending passage retrieval over position-adjusted minimum distance on a bigram index even though it performed better (although not by a statistically significant margin). A worthwhile compromise might be to adopt a two-step passage retrieval process as suggested in [5].

6 DYNAMIC RESULT CLUSTERING

Text reuse is so common within transactional law that any ranked retrieval solution would be incomplete without a discussion about duplicate results. Even using our relatively small database of 20,236 contracts, the majority of tested queries turned up at least a few near duplicate results.

In this section, we propose to avoid the difficulties posed by redundant results by introducing a dynamic search result clustering method that is specifically tailored to our task. Due to the interactive nature of our solution, which provides users with significant flexibility in organizing search results, an empirical test of its effectiveness would require a longer term user-focused study. We leave such an evaluation for future work. We emphasize, however, that our proposal, is an abstraction of, and can therefore emulate, traditional redundancy-based filtering of search results. As with

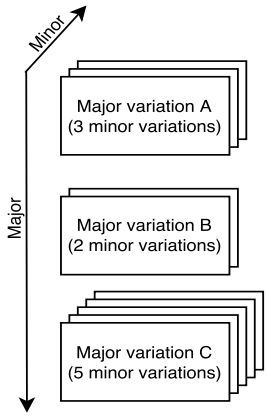

Figure 1: Organization of results into major and minor variations

search result clustering generally, our approach should be viewed as "complementary" to traditional methods [7].

6.1 The Definition of Redundancy

In many cases, lawyers will seek a diverse set of variations on a provision. As discussed in Section 3, however, near duplicates are not always redundant, and so filtering out all near duplicates is not an effective solution. In fact, even exact duplicates might at times be relevant, as a lawyer may be seeking to determine the number or types of parties that have agreed to a certain variant of a provision.

This problem is not necessarily unique to the legal field. Several authors have observed that users have varying requirements. Carbonell and Goldstein [6] argue that a "user-tunable" approach to novelty reordering is best, and Zhang et al. [43] observe that human assessors sometimes disagree about the very definition of redundancy. This latter observation was also made specifically within the transactional law domain by Conrad and Raymond [12]. We assert that redundancy in the legal context is even more dynamic than that: the definition of redundancy varies not only across lawyers, but also across situations for the same lawyer.

As a result, any static solution will be suboptimal in at least some situations. We therefore adopt the suggestion of Carbonell and Goldstein [6] and propose a user-tunable method. First, however, we present a static version of our proposal that takes advantage of search result clustering.

6.2 Major and Minor Variations

Inspired by Clarke et al.'s [9] distinction between diversity and novelty, our proposal draws a distinction between "major" and "minor" variations on contract language. For the moment, let us assume that major and minor variations are well defined, perhaps by relevance grades 3 and 4 from Section 5.2, respectively. We will revisit these definitions shortly.

We propose to treat major and minor variations as two separate dimensions of returned results, as depicted in Figure 1. In effect, we sidestep the difficulties discussed in Section 6.1 by employing search result clustering [7]. Rather than use the more traditional approach of filtering and/or reordering redundant results, our method clusters the "redundant" results (minor variations) under the "novel" results (major variations) that they correspond to. Users of our proposed system are able to step into or out of clusters of minor variations in real-time.

The key advantage over a flat list is that two conflicting definitions of redundancy can be simultaneously satisfied. A lawyer searching for minor variations can easily access them, whereas a lawyer searching only for major variations can ignore minor variations altogether.

Note that although the proposed approach is an instance of search result clustering, the transactional law context once again introduces an important structural element that makes our proposal different from prior work: as suggested by the label "variations", clusters are similar in *language* rather than *topic*.

6.3 User-Tunable Parameters

We now return to the definitions of a major and minor variations, which we propose be set according to dynamic, user-tunable parameters.

Formally, if $\mathbb{P}$ is the space of provisions, then given $p, q \in \mathbb{P}$ and some measure of the difference between two provisions, $\Delta : \mathbb{P} \times \mathbb{P} \rightarrow \mathbb{R}$, which may be asymmetric [19], we make the following definitions:

- **Redundant**: q is redundant with respect to p if $\Delta(p, q) < r$, where $r \in \mathbb{R}$ is the redundancy threshold.
- **Major variation:** q is a major variation of p if $\Delta(p, q) \geq m$, where $m \geq r$, $m \in \mathbb{R}$ is the major variation threshold.
- **Minor variation:** q is a minor variation or p if it is neither a major variation of p nor redundant with respect to p (i.e., $r \leq \Delta(p, q) < m$).

In line with past work [6, 43], we rank the major axis of the results such that lower results are novel with respect to all higher results. Thus, the nth major variation in the result list is a major variation with respect to *all* higher ranked major variations. For this reason, the major axis of results contains only a subset of all major variations of the original provisions.

Together with each major variation, which serves as the prototype for its cluster, we cluster all of its minor variations. Note that clusters are soft: the same provision might appear as a minor variation in two separate clusters.

This flexible setup allows users to tune results in real-time by adjusting the threshold parameters r and m (cf. λ in [6]). Increasing r decreases the size of each cluster, leaving the major variation axis unchanged. Increasing m increases the size of each cluster and decreases the number of clusters, as the divide between major variations grows. By setting $r = m$, users can eliminate the minor axis and achieve a result identical to traditional redundancy-based filtering.

A straightforward algorithm can be used for clustering and reclustering (when parameters r and m are changed). Given n results to be sorted into m clusters, the algorithm iterates over the results, selecting cluster prototypes by comparing each result to all previously selected cluster prototypes. Assuming constant comparison time, this iteration has time complexity $O(nm)$. Having selected

cluster prototypes, the other results are each compared to each prototype for clustering. This latter operation is also $O(nm)$, for a combined time complexity of $O(nm)$.

6.4 The Δ Function

With regards to the Δ function, we argue that functions based on Levenshtein distance (at the character- and/or word-level) are preferable to other methods due to their ability to provide interpretable guarantees. As noted in Section 4.2, a variety of measures might be used, including the measure used during ranked retrieval (e.g., BM25 at the passage-level). Generally speaking, however, measures of novelty or redundancy are not easily interpretable. For example, consider a measure involving the percentage overlap in term vectors between two passages, which was the primary measure used in Conrad and Raymond [12]. Although this is one of the more interpretable measures available, a lawyer exploring a cluster grouped according to such measure would have only a vague idea of the cluster's boundaries. By contrast, if a cluster is determined according to a Levenshtein distance of 5 characters, this would provide the lawyer with an easily interpretable guarantee: all passages within 5 characters of the cluster prototype are included in the cluster.

This notion of interpretable guarantees is of particular importance in legal search, and is one of the reasons why the use of Boolean connectors and proximity operators is still popular among lawyers despite the advent of powerful free text search [40].

7 CONCLUDING REMARKS

In this paper we examined the problem of finding alternative contract language using a prototype. We identified the characteristics that make this problem different from traditional IR tasks, and empirically compared several alternative retrieval models. We also outlined a method for organizing typically numerous, near duplicate results.

We compared four different scoring algorithms, each using both a unigram index and a bigram index, for a total of eight distinct approaches to ranked retrieval. Two of the retrieval models, position-adjusted minimum distance and maximum ascending m-cover, are novel measures that extend past work to take the order of query terms into account. We found that passage retrieval on a bigram index was the most effective method in terms of retrieval performance, but noted that our implementations of comparable methods, such as position-adjusted minimum distance on a bigram index, had faster query times. Among methods using a unigram index, the proposed position-adjusted minimum distance measure achieved the best retrieval performance.

We observed that search results in contract databases can contain many near duplicate provisions. Rather than filter or reorder such near duplicates, we proposed to solve the problem by strategically organizing the results into major and minor variations. Our proposal is user-tunable and therefore flexible with regards to search intent, and our proposed use of character- or word-level Levenshtein distance for grouping variations provides lawyers using the system with interpretable guarantees about search results.

Both aspects of our work—ranked retrieval and result organization—leave open important questions that will require attention, either in practice or in future work. With respect to ranked retrieval, our inquiry was limited to the top results, and it would be helpful to run a more involved evaluation that also examines lower ranked results. With respect to result organization, a user study would be helpful for determining a set of prototypical use cases, setting default parameters, and optimizing the interface with regards to user satisfaction and efficiency.

REFERENCES

[1] James Allan, Courtney Wade, and Alvaro Bolivar. 2003. Retrieval and novelty detection at the sentence level. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*. ACM, 314–321.

[2] Jing Bai, Yi Chang, Hang Cui, Zhaohui Zheng, Gordon Sun, and Xin Li. 2008. Investigation of partial query proximity in web search. In *Proceedings of the 17th international conference on World Wide Web*. ACM, 1183–1184.

[3] Pia Borlund. 2003. The concept of relevance in IR. *Journal of the American Society for information Science and Technology* 54, 10 (2003), 913–925.

[4] Stefan Büttcher, Charles Clarke, and Gordon V Cormack. 2010. *Information Retrieval: Implementing and Evaluating Search Engines*. The MIT Press.

[5] James P Callan. 1994. Passage-level evidence in document retrieval. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*. Springer-Verlag New York, Inc., 302–310.

[6] Jaime Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 335–336.

[7] Claudio Carpineto, Stanislaw Osiński, Giovanni Romano, and Dawid Weiss. 2009. A survey of web clustering engines. *ACM Computing Surveys (CSUR)* 41, 3 (2009), 17.

[8] Charles LA Clarke, Gordon V Cormack, and Elizabeth A Tudhope. 2000. Relevance ranking for one to three term queries. *Information processing & management* 36, 2 (2000), 291–311.

[9] Charles LA Clarke, Maheedhar Kolla, Gordon V Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. 2008. Novelty and diversity in information retrieval evaluation. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 659–666.

[10] Paul Clough. 2003. Old and new challenges in automatic plagiarism detection. *National Plagiarism Advisory Service* (2003). http://scholar.google.com/scholar?hl=en

[11] Jack G Conrad, Khalid Al-Kofahi, Ying Zhao, and George Karypis. 2005. Effective document clustering for large heterogeneous law firm collections. In *Proceedings of the 10th international conference on Artificial intelligence and law*. ACM, 177–187.

[12] Jack G Conrad and Edward L Raymond. 2007. Essential Deduplication Functions for Transactional Databases in Law Firms. In *Proceedings of the 11th international conference on Artificial intelligence and law*. 261–270.

[13] Joao Cordeiro, Gael Dias, and Pavel Brazdil. 2007. A metric for paraphrase detection. In *Computing in the Global Information Technology, 2007. ICCGI 2007. International Multi-Conference on*. IEEE, 7–7.

[14] Ronan Cummins and Colm O'Riordan. 2009. Learning in a pairwise term-term proximity framework for information retrieval. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 251–258.

[15] Marti A Hearst and Jan O Pedersen. 1996. Reexamining the cluster hypothesis: scatter/gather on retrieval results. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 76–84.

[16] Monika Henzinger. 2006. Finding near-duplicate web pages: a large-scale evaluation of algorithms. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 284–291.

[17] Samuel Huston, Alistair Moffat, and W Bruce Croft. 2011. Efficient indexing of repeated n-grams. In *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 127–136.

[18] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems* 20, 4 (2002), 422–446. https://doi.org/10.1145/582415.582418

[19] Cordeiro João, Dias Gaël, and Brazdil Pavel. 2007. New functions for unsupervised asymmetrical paraphrase detection. *Journal of Software* 2, 4 (2007), 12–23.

[20] Marcin Kaszkiel and Justin Zobel. 1997. Passage retrieval revisited. *ACM SIGIR Forum* 31 (1997), 178–185. https://doi.org/10.1145/278459.258561

[21] Marcin Kaszkiel and Justin Zobel. 2001. Effective ranking with arbitrary passages. *Journal of the American Society for Information Science and Technology* 52, 4 (2001),

344–364.

[22] Vladimir I Levenshtein. 1966. Binary codes capable of correcting deletions, insertions and reversals. In *Soviet physics doklady*, Vol. 10. 707.

[23] Xiaoyong Liu and W Bruce Croft. 2002. Passage retrieval based on language models. In *Proceedings of the eleventh international conference on Information and knowledge management*. ACM, 375–382.

[24] Qiang Lu, Jack G Conrad, Khalid Al-Kofahi, and William Keenan. 2011. Legal document clustering with built-in topic segmentation. In *Proceedings of the 20th ACM international conference on Information and knowledge management*. ACM, 383–392.

[25] Yuanhua Lv and ChengXiang Zhai. 2009. Positional language models for information retrieval. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 299–306.

[26] Caroline Lyon, Ruth Barrett, and James Malcolm. 2004. A theoretical basis to the automated detection of copying between texts, and its practical implementation in the Ferret plagiarism and collusion detector. *Plagiarism: Prevention, Practice and Policies* (2004).

[27] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.

[28] Donald Metzler and W Bruce Croft. 2005. A Markov random field model for term dependencies. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 472–479.

[29] Gonzalo Navarro. 2001. A guided tour to approximate string matching. *ACM computing surveys (CSUR)* 33, 1 (2001), 31–88.

[30] Naoaki Okazaki and Jun'ichi Tsujii. 2010. Simple and Efficient Algorithm for Approximate Dictionary Matching. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*. Beijing, China, 851–859. http://www.aclweb.org/anthology/C10-1096

[31] Yves Rasolofo and Jacques Savoy. 2003. Term proximity scoring for keyword-based retrieval systems. In *European Conference on Information Retrieval*. Springer, 207–218.

[32] Paolo Rosso, Santiago Correa, and Davide Buscaldi. 2011. Passage retrieval in legal texts. *The Journal of Logic and Algebraic Programming* 80, 3-5 (2011), 139–153.

[33] Gerard Salton, James Allan, and Chris Buckley. 1993. Approaches to passage retrieval in full text information systems. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 49–58.

[34] Jangwon Seo and W Bruce Croft. 2008. Local text reuse detection. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 571–578.

[35] David A Smith, Ryan Cordell, Elizabeth Maddock Dillon, Nick Stramp, and John Wilkerson. 2014. Detecting and modeling local text reuse. In *Proceedings of the 14th ACM/IEEE-CS Joint Conference on Digital Libraries*. IEEE Press, 183–192.

[36] Ian Soboroff and Donna Harman. 2005. Novelty detection: the TREC experience. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 105–112.

[37] Fei Song and W Bruce Croft. 1999. A general language model for information retrieval. In *Proceedings of the eighth international conference on Information and knowledge management*. ACM, 316–321.

[38] Ruihua Song, Michael J Taylor, Ji-Rong Wen, Hsiao-Wuen Hon, and Yong Yu. 2008. Viewing term proximity from a different perspective. In *European Conference on Information Retrieval*. Springer, 346–357.

[39] Tao Tao and ChengXiang Zhai. 2007. An exploration of proximity measures in information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 295–302.

[40] Howard Turtle. 1994. Natural language vs. Boolean query evaluation: A comparison of retrieval performance. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*. Springer-Verlag New York, Inc., 212–220.

[41] John Wilkerson, David Smith, and Nicholas Stramp. 2015. Tracing the flow of policy ideas in legislatures: A text reuse approach. *American Journal of Political Science* 59, 4 (2015), 943–956.

[42] Oren Zamir and Oren Etzioni. 1999. Grouper: a dynamic clustering interface to Web search results. *Computer Networks* 31, 11 (1999), 1361–1374.

[43] Yi Zhang, Jamie Callan, and Thomas Minka. 2002. Novelty and redundancy detection in adaptive filtering. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 81–88.

[44] Jinglei Zhao and Yeogirl Yun. 2009. A proximity language model for information retrieval. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 291–298.