2dSearch: a Visual Approach to Search Strategy Formulation

Tony Russell-Rose
2dSearch
3000 Cathedral Hill, Surrey
UK
tgr@uxlabs.co.uk

Phil Gooch
2dSearch
3000 Cathedral Hill, Surrey
UK
phil@contentinnovation.co.uk

ABSTRACT
Knowledge workers (such as healthcare information professionals, patent agents and media monitoring professionals) need to create and execute search strategies that are accurate, repeatable and transparent. The traditional solution is to use line-by-line ‘query builders’ such as those offered by proprietary database vendors. However, these offer limited support for error checking or query optimization, and their output can often be compromised by errors and inefficiencies. In this paper, we present a new approach to query formulation in which concepts are expressed as objects on a two-dimensional canvas. Relationships between objects are articulated by manipulating them using drag and drop. Automated search term suggestions are provided using a combination of knowledge-based and statistical natural language processing techniques. This approach has the potential to eliminate many sources of inefficiency, make the query semantics more transparent, and offers further opportunities for query refinement and optimisation.

CCS CONCEPTS
• Information systems → Query suggestion; • Information systems → Search interfaces

KEYWORDS
Visualization, Boolean, query expansion, professional search

INTRODUCTION
It has been claimed that knowledge workers spend as much as 2.5 hours per day searching for information [1]. Whether they find what they are looking for eventually or stop and make a sub-optimal decision, there can be a high cost to either outcome.

ACM Reference format:

Figure 1: A typical query builder.

Figure 2: An example patent search strategy.
What these professions have in common is a need to develop search strategies that are accurate, repeatable and transparent. The traditional solution to this problem is to use line-by-line query builders such as that shown in Figure 1. The output of these tools is a series of Boolean expressions consisting of keywords, operators and ontology terms, which are combined to form a multi-line search strategy such as that shown in Fig 2. However, most proprietary query builders offer limited support for error checking or query optimization, and the strategies produced are often compromised by mistakes and inefficiencies in the form of spelling errors, truncation errors, logical operator errors, incorrect query line references, and redundancy [6].

In this paper, we propose an alternative solution to the problem of search query formulation. Instead of a one-dimensional search box, concepts are expressed as objects on a two-dimensional canvas. Relationships between those objects are expressed by manipulating them using drag and drop. The use of a visual approach has the potential to eliminate many sources of syntactic error, helps to make the query semantics transparent, and offers further opportunities for query refinement and optimization.

2 RELATED WORK

2.1 Search query visualization

Previous studies have demonstrated that visual representations can communicate some kinds of information more rapidly and effectively than text, and these techniques have been productively applied to the presentation of search results [7]. However, the application of data visualization to search queries is much rarer.

The application of data visualization to search query formulation can offer significant benefits, such as fewer zero-hit queries, improved query comprehension, and better support for browsing within an unfamiliar database [8]. An early example of such an approach is that of Anick et al. [9], who developed a two-dimensional graphical representation of a user’s natural language query that supported reformulation via direct manipulation. Similarly, Fishkin and Stone [10] investigated the application of direct manipulation techniques to the problem of database query formulation, using a system of ‘lenses’ to refine and filter the data. Jones [11] developed VQuery, a query interface to the New Zealand Digital Library which exploits querying by Venn diagrams and integrated query result previews.

Later work includes that of Yi et al. [12], who explored the concept of a ‘dust and magnet’ metaphor applied to multivariate data visualization. Nitsche and Nürnberg [13] developed QUEST, a system based on a radial user interface that supports phrasing and interactive visual refinement of vague queries to search and explore large document sets. A further example is provided by Boolify1, which provides a dynamic drag and drop interface on top of Google’s search engine. Users build a query by dragging terms and operators onto a search surface. And more recently, de Vries et al [14] developed Spinque, which uses a visual canvas to allow users to graphically model a search engine using elementary building blocks. They describe this as searching ‘by strategy’, although the term is used more in the sense of defining (in advance) the behavior of a search engine, whereas in our case it refers to the run time execution of search expressions and operations.

Our approach combines elements of the above including the use of graphical representations, support for direct manipulation, and real time results retrieval. However, it differs from the prior art in that it focuses specifically on the needs of professional searchers, offers a generic visual framework for the representation of Boolean expressions and semantic relationships, and provides automated query suggestions with support for saving, sharing and re-using query templates and best practices.

2.2 Automated term suggestion

Query expansion is the process of reformulating or augmenting a user’s query in order to increase query effectiveness, particularly with regard to recall [15]. Selection of candidate expansion terms can be automated or interactive (i.e. guided by the user), and methods can be either local (based on documents retrieved by the query) or global (using resource independent of the query).

Global methods involve the use of domain specific resources such as thesauri, controlled vocabularies or ontologies to identify related terms in the form of synonyms, hypernyms, hyponyms, etc. Such resources may be either manually curated or automatically generated from domain-specific corpora using collocation and co-occurrence analysis techniques. Global methods can increase recall significantly but may also reduce precision by adding irrelevant or out-of-domain terms to the query [15]. In the current implementation of our work (see Section 3.4), we will not always have access to the full text of the documents in the result set (other than result snippets), so local methods are less applicable.

Ontologies are considered most useful for query expansion when they are specific to the query domain. Universal resources such as WordNet are considered less useful as they are too general and may not distinguish class concepts from instances [16]. However, ontologies may offer a productive source of related terms in the form of gloss words, i.e. words occurring in the term definitions [17]. Moreover, in the biomedical domain, expanding queries with related MeSH terms has been shown to be useful [18], while adding synonyms from the larger and more comprehensive UMLS has been found to improve recall [19], at the expense of precision [20].

The development of efficient distributed word representations has revolutionized unsupervised natural language processing techniques for finding synonyms [21][22]. Given the value of distributed word representations in identifying related terms, a number of researchers have considered the utility of word embeddings for query expansion. Kuzi [23], Roy [24] and Diaz [25] all used local embeddings trained on TREC corpora, with differing results. While Kuzi [23] found that local word embeddings outperformed the standard RM3 relevance model, Roy [24] found the opposite. Diaz [25] compared local

---

1 www.boolify.com
embeddings (TREC corpus) with global (generic Gigaword corpus) and found that local embeddings provided significantly better results for query expansion than global embeddings.

The fundamental problem with most query expansion techniques is that as many queries may be harmed (e.g. by introducing noise) as may be improved [26]. In addition, the user is unable to control how the expansion terms are used in the query. Cao et al [27] argue that previous work, irrespective of approach used, only considers the effect of the complete set of expansion terms on retrieval, and ignores the issue of how to distinguish useful expansion terms from useless or even harmful terms within that set. We address both these issues by treating query expansion as a recommendation task rather than an information retrieval task, i.e. given one or more query terms already entered by the user, can we provide a list of further recommended terms. Reframing the task in this way is particularly significant, since the visual approach offers an unprecedented opportunity for the user to engage meaningfully with candidate expansion terms and exercise more informed judgement regarding their value and contribution to the current search strategy.

3 CONCEPT AND DESIGN

At the heart of 2dSearch is a graphical editor which allows the user to create search strategies using a visual framework in which concepts are expressed as objects on a two-dimensional canvas (Figure 3). It is currently implemented as a Java desktop app using the JavaFX UI library, although in future work, we hope to deploy a browser-based version, using a combination of JavaScript libraries plus HTML and CSS.

2dSearch is aimed at knowledge workers who share a need to create search strategies that are repeatable, transparent and comprehensive [28]. The key design principles are to:

- Guide the user toward the formulation of syntactically correct expressions
- Present the semantics of the expressions in a transparent manner
- Facilitate the re-use of query templates and subcomponents
- Reduce the need for users to ‘translate’ their search strategy between different databases

Taken together these principles reduce the likelihood of common errors such as spelling errors, truncation errors, logical operator errors, incorrect query line references, and redundancy without rationale [6].

The range of relationships that may be expressed include traditional Boolean operators (e.g. AND, OR and NOT), but this can be extended to support other operators as required by the application context (such as ADJ, NEAR, etc.). Concepts may be combined to form aggregate structures, such as lists (unordered sets sharing a common operator) or composites (nested structures containing a combination of sub-elements). By nesting components within each other it is possible to create logical expressions of arbitrary complexity.

3.1 Managing complexity

2dSearch facilitates the adoption of approaches from object-oriented programming (OOP) that have been shown to help manage complexity in software development [29]. These include abstraction (e.g. creating generic templates from individual instances) and modularity (e.g. applying standard naming conventions). It encourages the use of meaningful names to identify and apply re-usable components, analogous to the creation of classes and objects in OOP.

It is quite common for text-based search strategies to extend over several pages, particularly in media monitoring applications. Consequently, there are instances when the visual equivalent would be too large to fit within the visible canvas. A naive solution is simply to zoom in and out of the canvas, magnifying or shrinking the display accordingly. However, this can render the text unreadable. Instead, a better approach is to incrementally control the level of abstraction so that more or less of the detail of each component is exposed. This strategy is augmented by the use of a ‘canvas map’ which provides an overview of the current query indicating where it extends beyond the viewport (Figure 4).
3.2 Query editing

2dSearch provides support for common graphical editing operations, such as move, copy, cut and paste, undo, redo etc. Composite expressions are created by combining elements: when one element is dragged over another, they are combined using an operator of the user’s choice. Queries can be persisted as external files, and then opened, imported etc. on demand. Legacy queries (i.e. text-based Boolean expressions and search strategies developed for proprietary databases) can also be opened and displayed as editable objects on the canvas.

3.3 Query execution

By default, 2dSearch will refresh the search results whenever an editing operation changes the semantics of the canvas content. However, the user is also able to execute individual query elements on demand. For example, to investigate the effect of a particular element within a larger expression, it can be executed in isolation and the results examined. Conversely, to remove a particular element from consideration without permanently deleting it, it can be temporarily disabled (analogous to commenting out a section of code). 2dSearch also offers a function to show ‘hit counts’ for individual query elements, so that their contribution to the overall search strategy can be understood in context.

2dSearch functions as a meta-search engine, and in principle is agnostic of any particular search technology or platform. In practice however, to execute a given query and retrieve results, the semantics of the canvas content need to be mapped to the API of the underlying database. This has required the development of an abstraction layer or ‘adapter’ for common search platforms such as Bing, PubMed, Elastic, etc.

Search results are displayed in a separate pane, which can be rendered adjacent to the canvas or in a separate window (Figure 4). The results pane also includes a tab to display the outbound query (which is generated specifically for the API of the selected database) and a further tab to display any errors or warnings returned to 2dSearch, e.g. if the query uses features or operators that the underlying API does not support. In due course we will explore ways to provide automated support for the resolution of such errors or warnings (perhaps following the example of software development environments in offering line help to resolve compiler errors or warnings).

3.4 Query suggestions

The ability to generate useful query expansion terms for a given query against a third party search engine without access to the source documents presents a challenge. Based on the review in Section 2.2, we decided to implement and evaluate three approaches to query expansion:

1. Global, using ontologies via a variety of SPARQL endpoints
2. Global, using word embeddings created from a variety of corpora
3. Local, using clustering and topic modelling of result snippets with Carrot2 [30]

**Query expansion using ontologies**

We developed a service that executes SPARQL queries to extract hypernyms, hyponyms, related terms, and term definitions from DBpedia, WebISA [31], MeSH, and other public SPARQL endpoints. Keywords from term definitions are extracted using a variety of algorithms (TF-IDF weighted noun phrases, textrank [32], sgrank [33], RAKE [34], and neoclassical combining forms (NCF) [35]. Keywords plus ontology terms are then ranked and aggregated to form query expansion suggestions.

**Query expansion with word embeddings**

In addition to open-source, publicly available word embeddings for Wikipedia [36], GoogleNews [22], and PubMed [37][38], we created new embeddings from the cleaned body text of around 900,000 full-text, open-access PubMed papers, optimized for multi-word expressions that typically occur in healthcare. Terms
most similar in vector space to the input query terms are ranked and aggregated to form new query expansion suggestions.

**Query expansion using clustering**

As an initial experiment with healthcare data, we implemented Carrot2 as a service, and the cluster labels from PubMed search result snippets for the input query terms are ranked and returned as candidate query expansion suggestions.

The system architecture for the deployment of these NLP services is shown in Figure 5.

### 4 EVALUATION

Due to its nature as a professional search tool, recruiting suitable participants to take part in a qualitative evaluation of 2dSearch can be challenging. However, we have undertaken a quantitative evaluation of the query suggestion API, using an offline, Cranfield-style approach in combination with a publicly available test set and traditional precision/recall metrics. It is important to recognize that the query suggestion task in 2dSearch differs from traditional query expansion tasks in a number of important ways:

- The primary use case for 2dsearch is recall-oriented professional search tasks, so evaluation methods that focus on the effect of query expansion on search engine ranking are less appropriate.
- The suggested terms are being added to an existing term or set of terms within a larger search strategy, rather than to a single natural language query. This means that their effect must be considered within the context of that specific set of terms.
- Since the visual approach allows the user to select individual expansion terms and apply them in isolation, it is important that any evaluation method considers the individual contribution of each candidate term, and not just the overall effect of an entire candidate set.

We have therefore based our evaluation on an approach that measures the extent to which the query suggestion API can generate terms found in existing (published) search strategies. For example, given the term rodent in the strategy of Figure 1, can it generate the related terms rat, rats, mouse, and mice (and only those terms). This particular search strategy consists of five such disjunctions (lines 2, 3, 6, 7 and 10), each of which offers an opportunity to apply and evaluate the query suggestion API.

For our test collection, we used data from the CLEF 2017 eHealth Lab, which includes a set of 20 topics for Diagnostic Test Accuracy (DTA) reviews. Each of these includes a search strategy (manually constructed by subject matter experts). Our overall evaluation approach is as follows: for every strategy in our test collection, iterate over each disjunction calling the query suggestions API on each term and calculating P & R based on the overlap between the search strategy term set and the suggested term set. We then repeat this process for each of the query expansion services offered by our API, and calculate macro precision, recall and F-measure. The results for the Ontology-based services are shown in Table 1. These ontologies were selected based on their likely coverage of the terminology in the CLEF data set.

**Table 1: Precision, recall and F for Ontology-based related terms**

<table>
<thead>
<tr>
<th>Service</th>
<th>P</th>
<th>R</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBPEDIA</td>
<td>0.017</td>
<td>0.040</td>
<td>0.024</td>
</tr>
<tr>
<td>WEBISA</td>
<td>0.001</td>
<td>0.003</td>
<td>0.002</td>
</tr>
<tr>
<td>MeSH</td>
<td>0.045</td>
<td>0.012</td>
<td>0.019</td>
</tr>
<tr>
<td>BNF</td>
<td>0.002</td>
<td>0.001</td>
<td>0.001</td>
</tr>
</tbody>
</table>

At first glance these results appear quite low, since even the best performing ontology (DBPEDIA) returns an F measure of 0.024. However, this is in line with previous studies on recommendation system evaluation where precision in the range 0.5-7% can be expected using offline methods [39]. Precision is relatively high for MeSH (0.045), reflecting the highly specialized nature of this resource (medical subject headings). Recall is relatively high for DBPEDIA (0.04), reflecting the broad coverage of this resource.

As mentioned in Section 3, DBPEDIA also provides term definitions that can serve as a further source of query expansion terms. These were extracted using a variety of algorithms (Table 2). Here, the best performing algorithm was NCF regex, in line with previous results using this approach to extract entities from biomedical text [40]. However, it is still inferior to the results obtained using the DBPEDIA ontology terms (Table 1). This contrasts with the results of [17], who found gloss terms to be more useful than ontology terms for query expansion (although in [17] WordNet was used).

**Table 2: Precision, recall and F for gloss terms extracted from DBPEDIA definitions**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>P</th>
<th>R</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>NCF regex</td>
<td>0.011</td>
<td>0.025</td>
<td>0.015</td>
</tr>
<tr>
<td>nltk-np</td>
<td>0.007</td>
<td>0.015</td>
<td>0.010</td>
</tr>
<tr>
<td>textrank</td>
<td>0.011</td>
<td>0.018</td>
<td>0.014</td>
</tr>
<tr>
<td>sgrank</td>
<td>0.009</td>
<td>0.014</td>
<td>0.011</td>
</tr>
<tr>
<td>rake</td>
<td>0.003</td>
<td>0.005</td>
<td>0.003</td>
</tr>
</tbody>
</table>

We then evaluated a number of publicly available word embedding models and the bespoke models that we created (as described in Section 3). The results are shown in Table 3, with our two bespoke models in the final two rows. The performance of our first model (PubMed unigram) is slightly greater but comparable to that of [37], which provides some evidence for the repeatability of the approach. The best performing model overall was our second bespoke model (PubMed trigram), which suggests that using higher order ngrams improves both precision and recall. A further contributory factor may have been our creation and use of a relatively clean corpus, which included only body text (no figures, headers, footers etc.) and removed numbers, punctuation, and other non-alphabetic elements.
Finally, we evaluated the use of the topic labels generated by Carrot2 clustering search results from PubMed. The results for Carrot2’s three clustering algorithms are shown in Table 4.

<table>
<thead>
<tr>
<th>Service</th>
<th>P</th>
<th>R</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lingo</td>
<td>0.002</td>
<td>0.005</td>
<td>0.003</td>
</tr>
<tr>
<td>STC</td>
<td><strong>0.010</strong></td>
<td><strong>0.019</strong></td>
<td><strong>0.013</strong></td>
</tr>
<tr>
<td>kMeans</td>
<td>0.008</td>
<td>0.016</td>
<td>0.011</td>
</tr>
</tbody>
</table>

Overall the STC (suffix tree clustering) algorithm performs best, although the F-measure is still some way short of that of the word embeddings and DBPEDIA. Moreover, Carrot2’s results are much harder to replicate, as it relies on sending live queries to Pubmed which is subject to database updates, timeouts, latency issues etc.

5 DISCUSSION AND CONCLUSIONS

In this paper we have described 2dSearch, a new approach to search strategy formulation. The use of a visual approach has the potential to eliminate many sources of syntactic error, makes the query semantics more transparent, and offers further opportunities for query refinement and optimisation. We have also described and evaluated an NLP API that returns query suggestions as recommendations to the end user.

The results from our evaluation are positive in the sense that our PubMed Word2vec model returns results comparable with typical offline recommender evaluation tasks and outperforms the best publicly available embedding model. However, we should note a number of caveats. Firstly, we have assumed that all disjunctions in the data set are equal, whereas in reality some may contain synonyms while others contain terms associated in some other way. Clearly the nature of those relations will have a bearing on the most effective expansion approach. Secondly, we have assumed that the CLEF data is gold standard, in the sense that it includes all (and only) the ‘correct’ terms in each disjunction.

However, there may be instances where a particular suggestion may actually be accepted by a human expert, even though it was absent from the data. This implies that our current results may be an underestimate of the actual live performance, although only an interactive evaluation (or a comparison with human performance on the same task) could formally establish this. Thirdly, many of the query terms are polysemous, whereas our work so far has been agnostic of word sense. Evidently, there are many ways to utilize context to better disambiguate query terms, and this is suggested as an area for future work. Finally, our evaluation concerns only one data set and one domain. In future work we will extend this to other data sets and domains.

ACKNOWLEDGMENTS

Part of this work was funded by Technology Strategy Board grant 131641 “A visual framework for search query formulation” and Innovate UK grant 102975 “Intelligent search assistance”.

REFERENCES
