Improving Results for the INEX 2009 Thorough and 2010 Efficiency Tasks

A THESIS
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Radhika A. Banhatti

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

Donald B. Crouch

August, 2011

Acknowledgements

Many people have contributed towards the successful completion of this thesis. I would like to take this opportunity to express my heartfelt gratitude to them.

Firstly, I would like to thank Dr. Carolyn Crouch for her guidance, encouragement, support and good-humored disposition throughout the two years. I would also like to thank Dr. Donald Crouch for his suggestions and feedback.

I would like to thank the faculty at UMD including Dr. Pedersen, Dr. Rosandich, Dr. Colburn, Prof. Holtz, Dr. Willemsen, Dr. Kwon, Dr. Turner and Dr. Prince for imparting invaluable knowledge during the two years of my MS program.

I would also like to thank Lori Lucia and Clare Ford for their support and help throughout the two years. I would also like to thank Jim Luttienen for his help whenever it was needed.

I would like to thank my friends Natasha Acquilla, Bhagyashri Mahule, Reena Rachel for their help and support. I also thank my friends Sai Chittilla and Supraja Nagalla for their support in preparing the document collection.

Lastly, I would like to thank my parents, Manju Banhatti and Aniruddha Banhatti, for their constant support and encouragement in all my endeavors. Without their support none of this would have been possible.

Abstract

Information retrieval is the process of storing textual data, i.e., documents, and then retrieving information from these stored documents. Web-based retrieval systems that use XML documents can also retrieve specific portions of the documents which are considered to be relevant to a query. This is called *element* retrieval which is performed using *flexible retrieval* (Flex) [5] in this thesis.

In this thesis we focus on improving the results of the INEX 2009 Thorough Task and the INEX 2010 Efficiency Task. Experiments are performed, methodology is described, and results are reported. Since *flexible retrieval* (Flex) depends on *Lnu-ltu* term weighting [13], experiments which generate the best values for the required parameters (i.e., slope and pivot) are also briefly described. Using the best value of slope, the INEX 2009 Thorough Task and the INEX 2010 Efficiency Task are carried out and the results are reported.

Table of Contents

List of Tables

List of Figures

1 Introduction

Information retrieval is the process of storing textual data, i.e., documents, and then retrieving information from these stored documents. Though early retrieval systems retrieved documents or document references, web-based retrieval systems can also retrieve specific portions of a document (e.g., a paragraph, an image, a section, etc.). This is called *element* retrieval.

Retrieval systems are based on models. The Vector Space Model [11] was developed by Salton at Cornell University in the 1960s. In the Vector Space Model, both documents and queries are represented as vectors.

The first step in creating a vector is indexing the text. Document indexing consists of scanning a document, retaining the content-bearing words and disregarding non-substantiative words (such as *the*). The content-bearing words (terms) can be represented as a vector of terms which now represents the document. The vectors store the frequency of the terms (i.e., the number of occurrences of each term in the document) as well.

The next step is to weight the terms in the vectors. The weight is an indicator of the importance of the term in the document. The weight is usually a function of term frequency multiplied by a length normalization factor [12]. Length normalization is performed so that differences in document length do not unduly advantage longer vectors over shorter ones.

The next step in the retrieval process consists of finding the similarity between the query vector and the document vectors. Various similarity measures are used to do so (e.g., cosine, inner product). The method used in this research is inner product.

A primary advantage of the Vector Space Model is ranked retrieval. We use Smart [10], a retrieval system based on the Vector Space Model and developed by Buckley at Cornell, for basic retrieval functions (e.g., indexing and term weighing).

Flex [5] is used for the dynamic generation and retrieval of XML elements in our experiments. Flex generates the element vectors for the non-terminal nodes of the document tree [1] and then computes the correlation of every element vector with the query vector using *Lnu-ltu* term weighing and inner product. Flex outputs a ranked set of elements. Flex evolved through research in information retrieval by graduate students at UMD [6], [1].

The research in information retrieval at UMD is driven towards successfully completing the tasks defined by INEX (the INitiative for the Evaluation of XML Retrieval). INEX provides the participant institutes with a document collection along with tools for assessment and evaluation. The current collection is a set of 2,666,190 articles (50.7 GB) from Wikipedia. Every year INEX publishes various tasks to be performed on the document collection. The results of the year's research are submitted to INEX for evaluation.

2 Overview

This chapter provides an overview of INEX, and in particular, description of the INEX 2009 Thorough Task, the equivalent INEX 2010 Efficiency Task, and task evaluation. The 2009 and 2010 Ad Hoc Tasks are also briefly covered.

2.1 INEX

INEX [3], is an established evaluation forum for XML information retrieval (IR). Most web content today is in the form of XML (i.e., eXtensible Markup Language), which has made XML IR an active research area. INEX has over 100 registered organizations worldwide [3]. INEX provides a complete IR infrastructure, in the form of a large test collection, tasks, and appropriate scoring methods for the evaluation of IR strategies. INEX also provides a set of queries that can be used to test the accuracy of the participant institutes's IR methodology. UMD participates in the INEX Ad Hoc Retrieval Track.

2.2 INEX 2009 Thorough Task

The main task is to retrieve all elements that are relevant to the query. It specifies no restriction in the form of the retrieved element; i.e., the element can be, for example, a body, a paragraph or a subsection. It also allows “overlapping” elements. An example of overlapping elements is given in Figure 1 :

article[1]/bdy[1]/section[1]
article[1]/bdy[1]/section[1]/p[1]

Figure 1 : Example of “overlapping” Elements

These are called overlapping elements because *article[1]/bdy[1]/section[1]/p[1]* is

a child of *article[1]/bdy[1]/section[1]*. The output is a list of elements ranked by perceived relevance to the given query. The Thorough Task is defined as “retrieval that retrieves all elements along the path of a retrieved element” [7, p.32].

2.3 INEX 2010 Efficiency Task

In the INEX 2010 Ad Hoc Retrieval Track, the Thorough Task is called the Efficiency Task. The main task is to retrieve the top-ranked 15, 150 and 1500 elements. The elements are ranked by their correlation with the query. Also, run times and I/O costs for evaluating each query as well as general statistics about the hardware and software environment used for generating the results are to be submitted [3].

2.4 INEX 2009 and 2010 Thorough Task Evaluation

The measure used to evaluate the Thorough Task is MAiP, i.e., mean average interpolated precision. This is calculated over 101 standard recall points (e.g., 0.00, 0.01, 0.02, ..., 1.00). The aim of choosing this metric is to measure overall performance. The formula for MAiP is derived by using the formulas given in Figures 2, 3, 4, 5 and 6. See [7].

$$R[r] = \frac{\sum_{i=1}^{r} rsize(p_i)}{T_{rel}(q)}$$

where,

R[r] is the recall score at rank r

p_r is the document part assigned to rank r in ranked list of documents

$rsize(p_r)$ is the length of relevant text contained by p_r in characters

$T_{rel}(q)$ is the total amount of relevant text for topic q

Figure 2 : Recall at Rank r

$$P[r] = \frac{\sum_{i=1}^{r} rsize(p_i)}{\sum_{i=1}^{r} size(p_i)}$$

where,

P[r] is precision at rank r

p_r is the document part assigned to rank r in ranked list of documents

$rsize(p_r)$ is the length of relevant text contained by p_r in characters

$size(p_r)$ is the total number of characters contained by p_r

Figure 3 : Precision at Rank r

$$iP[x] = \begin{cases} \max\limits_{1 \le r \le |L_q|} (P[r] \wedge R[r] \ge x) & \text{if } x \le R[|L_q|] \\ 0 & \text{if } x > R[|L_q|] \end{cases}$$

where,

$P[r]$ is the precision at rank r

$R[r]$ is the recall score at rank r

L_q is the ranked list of document parts returned by a retrieval system for a topic q

Figure 4 : Interpolated Precision at Recall value x

$$AiP = \frac{1}{101} \cdot \sum_{x=0.00,0.01,\ldots,1.00} iP[x]$$

where, $iP[x]$ is the interpolated precision at recall value of x

Figure 5 : Formula for Average Interpolated Precision

$$MAiP = \frac{1}{n} \sum_{t} AiP(t)$$

where,

n is the number of topics,

$AiP(x)$ is the average interpolated precision for recall value of x

Figure 6 : Formula for Mean Average Interpolated Precision

For a more detailed analysis of the MAiP metric, refer to [2] and [4].

2.5 Ad Hoc Tracks for INEX 2009 and 2010

In 2009-10, UMD participated in the Ad Hoc Retrieval Track. The Thorough Task is discussed in detail in Sections 2.2 and 2.3. The remaining tasks in the Ad Hoc Retrieval Track are covered below.

Focused Task (2009)

The task is to retrieve non-overlapping (or focused) elements that are relevant to the query. Overlapping of elements occurs when both a child element and a parent element are retrieved. In the case of parent and child elements having equal correlations, the child is preferred. The result must be a rank-ordered list in descending order of correlation. The metric is interpolated precision at 1% recall, i.e., (iP[0.01]).

Relevant in Context Task (2009)

The task is to retrieve a rank-ordered list of articles that are relevant to the query; associated with each article is a ranked-ordered list of focused elements. Results are grouped by document and rank-ordered by element within the document. The elements within the document must be non-overlapping. The metric is mean average generalized precision (MAgP).

Restricted Focused Task (2010)

The task is identical to the 2009 Focused Task, with the following exception : the retrieved elements are restricted to 1,000 chars per query. The metric is interpolated precision at 1% recall, i.e., (iP[0.01]).

Relevant in Context Task (2010)

The task is identical to the 2009 Relevant in Context Task. The metric remains the same, i.e., MAgP.

Restricted Relevant in Context Task (2010)

The task is similar to the Relevant in Context Task described above, with the additional restriction that an element may contain maximally 500 characters. The metric is mean average generalized precision (MAgP).

3 Background Details for Experiments

This chapter provides a background for the steps required to do an element retrieval for the Thorough Task. Each step is described in detail along with its input and output.

3.1 Collection Preparation

Scrubbing

The first step is to acquire the document collection. The collection this research uses is a 2,666,190 article dump of Wikipedia, provided by INEX. The files in this collection are in XML format. (XML, a set of encoding rules, is a tag-based language used to store and transmit data.) Content in an XML file is usually contained within tags (e.g., <article>). Every XML tag must have a corresponding closing tag (e.g., </article>).

Figure 7 presents an excerpt of a sample document from the collection. The collection then is *scrubbed* to remove unwanted text. Tags and text that do not help in retrieval are usually scrubbed.

As seen in Figure 7, the tags *creation* and *recording* are empty (i.e., they have no content). Figure 8 presents an except of a scrubbed document. As seen in Figure 8, creation, recording and other empty tags are scrubbed. The *article* tag is retained as the root tag. These scrubbed tags are later patched again (i.e., re-inserted) before evaluation takes place.

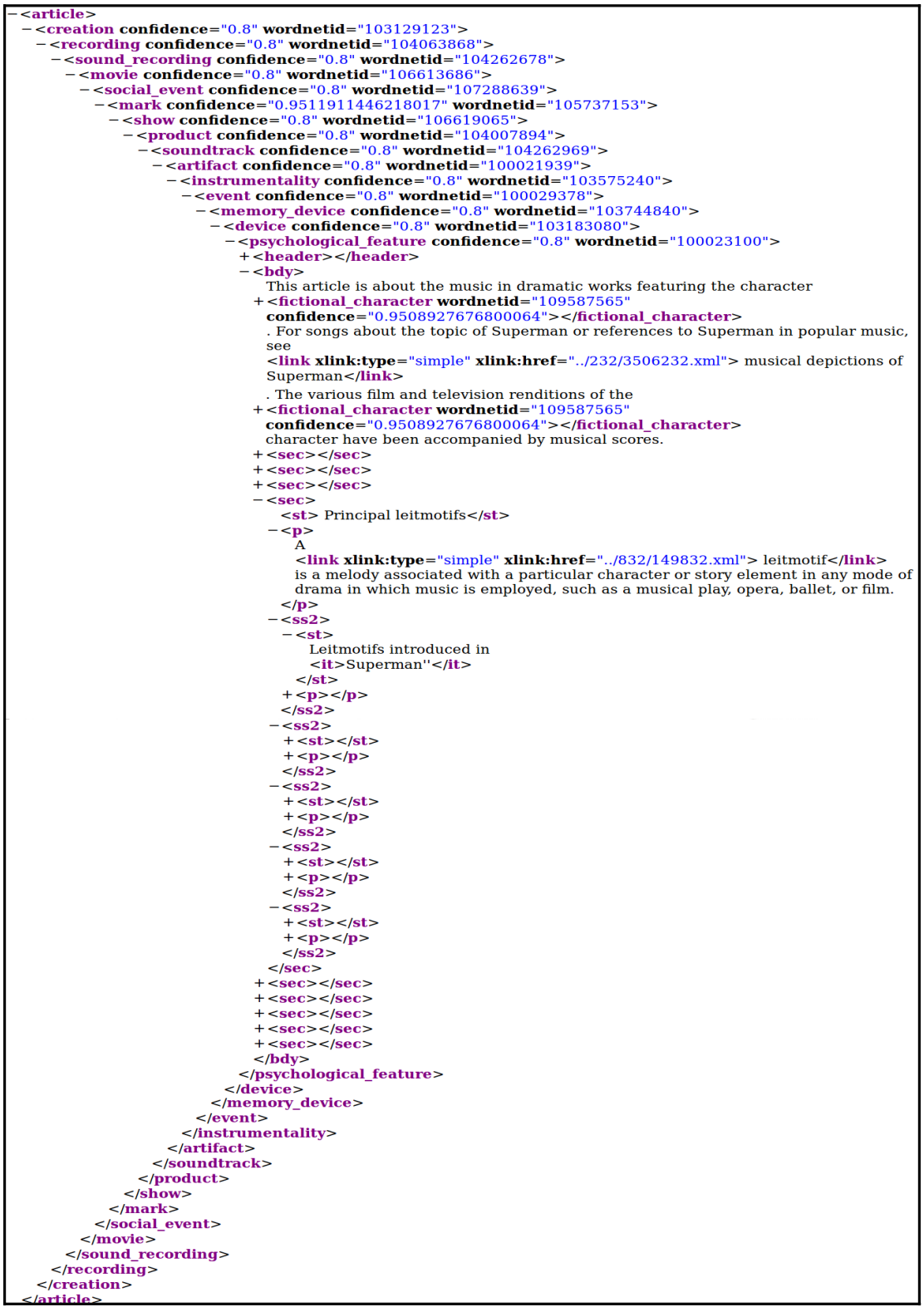

Figure 7 : Excerpt of Original INEX Document – 3242386

```
<article>
 <header>
  <title> Superman music </title>
  <categories>
   Superman films Superman television series Superman music Film soundtracks Film scores
  </categories>
 </header>
 <bdy>
  <sec>
   <st> Radio cartoons early films </st>
   <p>
    The radio shows of the early 1940s already had the famous phrases Faster than a speeding bullet It s a bird it s a plane it s Superman uttered by studio announcer Jackson Beck Initially the radio series had no theme tune under its introductory lines
   </p>
   <p></p>
   <p></p>
   <p></p>
   <p></p>
  </sec>
  <sec>
   <st> Television and broadway </st>
   <p>
    The TV theme for the 1950s series Adventures of Superman starring George Reeves had the unusual lead in of a harp playing a kind of stringed drumroll as the camera moved through space segueing into a dramatic brass triad accompanied by cymbals drums etc at the moment when a shooting star explodes on the screen and the title card appears A variation on the classic Faster than a speeding bullet was rendered by deep voiced actor Bill Kennedy
   </p>
   <p></p>
   <p></p>
   <p></p>
   <p></p>
   <p></p>
   <p></p>
   <p></p>
   <p></p>
   <p></p>
  </sec>
  <sec></sec>
  <sec>
   <st> Principal leitmotifs </st>
   <p>
    A leitmotif is a melody associated with a particular character or story element in any mode of drama in which music is employed such as a musical play opera ballet or film
   </p>
   <ss2>
    <st> Leitmotifs introduced in Superman </st>
    <p>
     Superman Fanfare A short triad based motif played just before the Main Theme or as a standalone when Superman appears in a quick cut on screen Also restated many times in the Superman March Superman March or Superman Main Theme Used over opening and closing credits It consists of two sections an A theme which is the main part of the melody and a B theme which is a bit lighter in mood and which often connects the March to the Fanfare Can You Read My Mind or the soaring Love Theme 1 Typically used when Lois and Superman or sometimes Clark find themselves alone together A portion of is introduced as an interlude in the midst of the Superman March Lyrics for the melody were written by longtime John Williams collaborator Leslie Bricusse for the purpose of having a song during the film s extended flying sequence Margot Kidder who plays Lois Lane speaks the lyrics in the film but cover versions of the song have been recorded by Maureen McGovern Shirley Bassey and others Krypton fanfare Used as the viewer zooms in on Krypton and again with the self construction of the Fortress of Solitude Krypton crystal motif or the Secondary Krypton motif Mysterious sounding theme associated with the physicality of the planet Krypton both the crystals sent by Jor El to Earth with his son and the radioactive kryptonite which is deadly to Superman Personal motif A melody related to the duality of Superman and Clark Kent which musically connects the Fanfare to the Love Theme Smallville or Growing Up Theme A Coplandesque Americana melody used during the Smallville sequences which in some ways is a simpler or undeveloped version of the the March s A theme It bears a similarity to a theme written by John Williams for the 1972 John Wayne western The Cowboys The March of the Villains or Lex Luthor theme A comedic Prokofiev inspired march associated with the villain Lex Luthor and his henchman Otis
    </p>
   </ss2>
   <ss2>
    <st> Leitmotifs introduced in Superman II </st>
    <p></p>
   </ss2>
   <ss2></ss2>
   <ss2></ss2>
   <ss2></ss2>
  </sec>
  <sec></sec>
  <sec></sec>
  <sec></sec>
  <sec></sec>
  <sec></sec>
  <mt>
   This article is about the music in dramatic works featuring the character Superman 32 32 For songs about the topic of Superman or references to Superman in popular music see musical depictions of Superman 32 32 The various film and television renditions of the Superman character have been accompanied by musical scores
  </mt>
 </bdy>
</article>
```


Figure 8 : Excerpt of Scrubbed INEX Document – 3242386

Parsing

Parsing is the process of removing unwanted text and tags, producing a reduced view of the document wherein only needed elements are retained. Various parses are performed. For *all-element* retrieval, performed here to calculate appropriate values of slope and pivot for *Lnu-ltu* term weighting, the following parses must be generated. At the highest level 0, is the *article* parse, then at level 1 are the *bdy* and *header* parses, at levels 2-7 are the section/subsection parses and last is the terminal node parse (*para+mt* parse). Consider the sample document in Figure 8 and four of its corresponding parses, namely *bdy* parse (level 1), section parse (level 2), subsection parse (level 3) and *para+mt* parse.

***bdy* parse** - This parse identifies the *bdy* element of each document and retains only the text contained within *bdy*. The content preserved serves as a component of the *all-element* parse. Figure 9 presents an excerpt of a *bdy* parse.

```
/article[1]/header[1]/
Superman music Superman films Superman television series
Superman music Film soundtracks Film scores
</header[1]>

/article[1]/bdy[1]/
Radio  cartoons  early films The radio shows of the early 1940s
already had the famous phrases Faster than a speeding bullet
Its a bird its a plant its Superman uttered by studio announcer
Jackson Beck Initially the radio series had no theme tune under
its introductory lines The  Superman cartoon
</bdy[1]>
```

Figure 9 : Excerpt of a *bdy* parse for INEX Document - 3242386

section parse - This parse identifies the section elements of each document and retains only the text contained within the section. Figure 10 presents an excerpt of a section parse.

```
/article[1]/bdy[1]/sec[1]/
Radio cartoons early films The radio shows of the early 1940s already had the famous phrases Faster than a speeding bullet It s a bird its a plane its Superman uttered by studio announcer Jackson Beck
</sec[1]>

/article[1]/bdy[1]/sec[2]/
Television and broadway The TV theme for the 1950s series Adventures of Superman starring George Reeves had the unusual lead in of a harp playing a kind of stringed drumroll as the camera moved through space
</sec[2]>
```

Figure 10 : Excerpt of a section parse for INEX Document - 3242386

subsection parse – This parse identifies the subsections within the document and retains only the text within the subsections. The content preserved is used for the *all-element* indexing. Figure 11 presents an excerpt of a subsection parse.

/article[1]/bdy[1]/sec[4]/ss2[1]/
Leitmotifs introduced in Superman Superman Fanfare A short triad based motif played just before the Main Theme or as a standalone when Superman appears in a quick cut on screen Also restated many times in the Superman March Superman March or Superman Main Theme
</ss2[1]>

/article[1]/bdy[1]/sec[4]/ss2[2]/
Leitmotifs introduced in Superman II Composer arranger Ken Thorne was mandated to reuse the first films themes for Superman II He based the music for the Kryptonian villains on the Williams material associated with Krypton and the Fortress of Solitude He also added a descending three note motif for the villains and a briefly heard ominous melody associated with General Zod
</ss2[2]>

Figure 11 : Excerpt of a subsection parse for INEX Document - 3242386

***para+mt* parse**

Many times a document has untagged text. We refer to such text as *magic text (mt).* In the terminal node or *para+mt* parse, a node is created which identifies each terminal node, including the *mt* node, which is created to hold all the untagged text of the parent node. (The *mts* are essential to the operation of Flex as all terminal nodes must be present to generate the parent node properly. For details see [9]). Figure 12 presents an excerpt of a *para+mt* parse.

/article[1]/header[1]/title[1]/
Superman music
</title[1]>

/article[1]/header[1]/categories[1]/
Superman films Superman television series Superman music Film soundtracks Film scores
</categories[1]>

/article[1]/bdy[1]/sec[1]/st[1]/
Radio cartoons early films
</st[1]>

/article[1]/bdy[1]/sec[1]/p[1]/
The radio shows of the early 1940s already had the famous phrases Faster than a speeding bullet Its a bird its a plane its Superman uttered by studio announcer Jackson Beck Initially the radio series had no theme tune under its introductory lines
</p[1]>

/article[1]/bdy[1]/sec[1]/p[2]/
The Superman cartoon series produced by the Fleischer Studios during the 1940s included a triad based theme composed by Fleischer musical director Sammy Timberg The cartoons were clearly intended to extend the characters from radio as Jackson Beck again provided the introduction voice
</p[2]>

/article[1]/bdy[1]/sec[1]/p[3]/
The two Superman Columbia Pictures serials of the late 1940s starring Kirk Alyn featured a theme that began with a triad repeated once The rest of the theme was a standard orchestral march in a minor key that did not refer back to the original triad This theme was composed by Mischa Bakaleinikoff who scored a number of the Columbia serials themes
</p[3]>

/article[1]/bdy[1]/mt[1]/
This article is about the music in dramatic works featuring the character Superman For songs about the topic of Superman or references to Superman in popular music see musical depictions of Superman The various film and television renditions of the Superman character have been accompanied by musical scores
</mt[1]>

Figure 12 : Excerpt of *paras+mt* parse for INEX Document - 3242386

Indexing

Indexing is the process that converts text into a corresponding (term frequency) vector. The article parse is the input to article indexing, which produces article vectors (i.e., the article index).

The indexing phase results in the creation of *nnn* vectors (i.e., term frequency vectors). This weighting takes into consideration only the frequency of occurrence of the terms. Many weighing schemes are available in Smart – in particular *Lnu-ltu* [13], which is used in our experiments. Vectors are compared

on the basis of similarity; common similarity measures are cosine and inner product. The *nnn* document/element vector is converted to a *Lnu* vector, and the query vector is converted to a *ltu* vector using Smart. These weights take into consideration pivoted normalization. Pivoted normalization requires two constants, slope and pivot. Pivot is average length of the element vectors over the whole collection. Slope is the point where the normalization value is to be 'tilted.' The idea is to increase the probability of retrieval for shorter vectors in an environment where the lengths of the vectors vary widely. Slope is calculated through empirical experiments with the document collection. For a more detailed description of pivoted normalization, see [6].

The formula for *Lnu* is given in Figure 13 [6].

$$\frac{\frac{1+\log(tf)}{1+\log(average\ tf\ \ in\ text)}}{(1-slope)\ \ +slope\times\frac{\#\ unique\ terms\ in\ text}{pivot}}$$

Where *tf* is the term frequency obtained from the nnn element vectors.

average tf in text is the average of the tf of all the terms in this element

#unique terms in text is the number of distinct terms in this element

slope and *pivot* are the empirically determined parameters

Figure 13 : Formula for *Lnu*

The formula for *ltu* is given in Figure 14 [6].

$$\frac{(1+\log(tf)) \times \left(\log \frac{N+1}{df}\right)}{(1-slope) + slope \times \frac{\#unique\ terms\ in\ text}{pivot}}$$

Where *tf* is the term frequency which is obtained from the nnn vectors.

N is the total number of elements in the collection

df is the document frequency of the term

#unique terms in text is the number of terms in the element

slope and *pivot* are the empirically determined parameters

Figure 14 : Formula for *ltu*

3.2 Retrieval

In these experiments, we use Flex along with the doc-trees to generate each document from the bottom up. Flex builds the tree and correlates the query with each element (some of which may be *mt*s). The output of Flex is a rank-ordered list of elements for each document.

Removing *mt*s

The retrieval results in a ranked list of element vectors; it may contain *mt*s. These elements do not exist *per se* in the actual document but rather are produced

artificially by grouping together the untagged text at each level in the document and enclosing it within *mt* tags. Such elements, however, cannot be processed further as they do not exist as such in the original document and hence have no xpath associated with them.

An xpath is a path in the document which identifies the element. For example, *3260094/article[1]/bdy[1]/p[2]/* is an xpath that identifies the second paragraph of the article 3260094. An *mt* is an artificially created element and has no xpath associated with it.

Format Conversions

The output produced by removing mt elements is a ranked list of elements. This output is first converted to INEX format and then to TREC format via Perl scripts. This is required for submission to INEX.

Patching Xpaths

The next step is to attach the high order path or tags which were removed before parsing. An example of an xpath before expansion is */article[1]/bdy[1]/sec[1]/* whereas its expanded form is */article[1]/creation[1]/recording[1]/sound_recording[1]/movie[1]/social_event[1]/mark[1]/show[1]/product[1]/soundtrack[1]/artifact[1]/instrumentality[1]/event[1]/memory_device[1]/device[1]/psychological_feature[1]/body[1]/sec[1]/.*

3.3 Result Evaluation

Evaluation of results is performed using two tools provided by INEX. These tools are in the JAR (Java Archived) file format. The first of the two tools converts the xpaths to File Offset and Length (FOL) format. A sample FOL file is shown in Figure 15.

In Figure 15, first column represents the topic number (year and query number), second column represents the query number within that topic, this is currently unused and should be Q0, third column is the document id, fourth column is the rank of the retrieved document, fifth column is the inverse rank, sixth column is called the “run tag” identifying the group, seventh column is the offset in the file where the element begins and the eighth column is the number of characters present in the retrieved element following the offset.

```
2009001 Q0 3260094 1 1500 1 280 4081
2009001 Q0 3260094 2 1499 1 139 4296
2009001 Q0 21201 3 1498 1 151 32952
2009001 Q0 21201 4 1497 1 15396 5929
2009001 Q0 21201 5 1496 1 25100 5166
2009001 Q0 21201 6 1495 1 25091 5177
2009001 Q0 21201 7 1494 1 18304 2168
2009001 Q0 21201 8 1493 1 13080 2315
2009001 Q0 21201 9 1492 1 18344 896
2009001 Q0 21201 10 1491 1 21326 3331
```

Figure 15 : Excerpt of a FOL File

The second tool calculates the score for each task using its metric. The input to this tool are the qrels file provided by INEX and the FOL format file produced by the first INEX evaluation tool. The qrels file is created from manual relevance assessments and is used as the baseline for evaluation. The final evaluation result is shown in Figure 16.

```
<eval run-id="1" file="1500_Ele/FOL.txt">
num_q          all    68
num_ret        all    7319
num_rel        all    4858
num_rel_ret    all    1745
ret_size       all    131439385
rel_size       all    18838137
rel_ret_size   all    7435460
iP[0.00]       all    0.5794364927223699
iP[0.01]       all    0.5626358131851333
iP[0.05]       all    0.5123586035456653
iP[0.10]       all    0.46242888927705605
MAiP           all    0.2313537079330074
ircl_prn.0.00  all    0.5794364927223699
ircl_prn.0.01  all    0.5626358131851333
ircl_prn.0.02  all    0.5431486538798717
ircl_prn.0.03  all    0.5265931851472455
ircl_prn.0.04  all    0.5158921751049688
```

Figure 16 : Excerpt of an Evaluation File

The MAiP score, as seen in Figure 16, is of primary interest for the INEX 2009 Thorough Task and the INEX 2010 Efficiency Task.

4 Experiments, Results and Analysis

This chapter provides detailed description of the experiments performed. The experiments performed are slope-pivot experiments for the INEX 2009 query collection, the INEX 2009 Thorough Task and the INEX 2010 Efficiency Task.

4.1 Slope-Pivot Experiments with 2009 Query Collection

The experiments aim at finding a slope value that produces the maximum MAiP value. These experiments were performed using the INEX 2009 query collection, which consists of 115 queries. Pivot is fixed at 38. (See Section 3.1). The value of slope used previously was 0.11 [7]. To find a good value of slope, the slope was increased as well as decreased in steps of 0.05. This resulted in slope ranging from 0.01 to 0.21 in steps of 0.05. Thorough element retrieval using Flex as described in Section 3.2 was performed for each slope value. The results were evaluated using the method for Thorough Task evaluation to produce a MAiP value. The top 1000 and the top 1500 documents from the Reference Run were used for article retrieval (i.e., to identify the top *n* documents). See [14] for details of the Reference Run. The results for the specified values of slope with the top 1000 documents and top 1500 documents taken from the Reference Run, are presented in Tables 1 and 2, respectively.

Table 1 : Results for Slope-Pivot Experiments with the Top 1000 Documents from the Reference Run for 2009 Query Collection

Slope	MAiP
0.01	0.2156
0.06	0.2249
0.11	0.2314
0.16	0.2007
0.21	0.1949

Table 2 : Results for Slope-Pivot Experiments with the Top 1500 Documents from the Reference Run for 2009 Query Collection

Slope	MAiP
0.01	0.2126
0.06	0.2204
0.11	0.2302
0.16	0.1930
0.21	0.1904

We find that the best results are obtained when we use a slope value of 0.11. This value of slope is used by Smart for term weighting when performing the INEX 2009 and INEX 2010 Ad Hoc Track Tasks.

4.2 INEX 2009 Thorough Task

For carrying out the INEX 2009 Thorough Task, the article reference list provided by INEX is used. Top-ranked elements are retrieved using the reference list and Flex. Flex output is evaluated as described in Section 3.2 to obtain the final MAiP value. These runs evaluate 50, 100, 150, 200, 250, 500, 1000 and 1500 elements retrieved from 25, 50, 100, 150, 200, 250, 500 and 1000 documents. Table 3 presents the MAiP values for the INEX 2009 Thorough Task.

Table 3 : Results for INEX 2009 Thorough Task

Elements -> / Documents	50	100	150	200	250	500	1000	1500
25	0.1082	0.1315	0.1506	0.1655	0.1758	0.2003	0.2061	0.2061
50	0.1082	0.1315	0.1506	0.1655	0.1758	0.2049	0.2206	0.2220
100	0.1082	0.1315	0.1506	0.1655	0.1758	0.2052	0.2254	0.2304
150	0.1082	0.1315	0.1506	0.1655	0.1758	0.2052	0.2255	0.2313
200	0.1082	0.1315	0.1506	0.1655	0.1758	0.2052	0.2255	0.2313
250	0.1082	0.1315	0.1506	0.1655	0.1758	0.2052	0.2255	0.2313
500	0.1082	0.1315	0.1506	0.1655	0.1758	0.2052	0.2255	**0.2314**
1000	0.1082	0.1315	0.1506	0.1655	0.1758	0.2052	0.2255	**0.2314**

Observations

The observations from the experiments are listed below :

1. For the INEX 2009 Thorough Task, the best MAiP score (0.2314) is produced using a slope value of 0.11. It is observed that results for the other INEX 2009 Ad Hoc Track Tasks are also are superior for this slope value [1], [8].

2. As seen in Table 4, the results obtained in these experiments indicate that the University of Minnesota Duluth would rank 7th in the top 10 participant scores. See [2].

Using a confidence interval of 95% in a one-tailed t-test, we found that our INEX 2009 Thorough Task results were not significantly different than the results of the other ten participants.

Table 4 : INEX 2009 Thorough Task Results Compared to the Top Participant Scores

Rank	Participant	MAiP
1	p48-LIG-2009-thorough-3T	0.2855
2	p6-UAmsIN09article	0.2818
3	p5-BM25thorough	0.2585
4	p92-Lyon3LIAmanlmnt	0.2496
5	p60-UJM 15494	0.2435
6	p346-utCASartT09	0.2350
*	**p72-UMD**	**0.2314**
7	p10-MPII-CASThBM	0.2133
8	p167-09RefT	0.1390
9	p68-I09LIP6OWATh	0.0630
10	p25-ruc-base-coT	0.0577

4.3 INEX 2010 Efficiency Task

For carrying out the INEX 2010 Efficiency Task, the reference list provided by INEX is used. The procedure is similar to the INEX 2009 Thorough Task. These runs evaluate 15, 50, 100, 150, 200, 250, 500, 1000 and 1500 elements retrieved from 25, 50, 100, 150, 200, 250, 500 and 1000 documents. Table 5 presents the MAiP values for the INEX 2010 Efficiency Task.

Table 5 : Results for INEX 2010 Efficiency Task

Elements-> / Documents	15	50	100	150	200	250	500	1000	1500
25	0.0497	0.0762	0.0943	0.1040	0.1169	0.1211	0.1385	0.1410	0.1410
50	0.0497	0.0762	0.0943	0.1040	0.1169	0.1213	0.1408	0.1534	0.1555
100	0.0497	0.0762	0.0943	0.1040	0.1169	0.1213	0.1409	0.1576	0.1662
150	0.0497	0.0762	0.0943	0.1040	0.1169	0.1213	0.1409	0.1577	**0.1675**
200	0.0497	0.0762	0.0943	0.1040	0.1169	0.1213	0.1409	0.1577	**0.1675**
250	0.0497	0.0762	0.0943	0.1040	0.1169	0.1213	0.1409	0.1577	**0.1675**
500	0.0497	0.0762	0.0943	0.1040	0.1169	0.1213	0.1409	0.1577	**0.1675**
1000	0.0497	0.0762	0.0943	0.1040	0.1169	0.1213	0.1409	0.1577	**0.1675**

Observations

The observations from the experiments are listed below :

1. For the INEX 2010 Efficiency Task, the best MAiP score (0.1675) is produced using a slope value of 0.11. It is observed that results for the other INEX 2010 Ad Hoc Track Tasks are also superior for this slope value [1], [8].

2. As seen in Table 5, the MAiP score remains constant for retrieval of 15 elements at a value of 0.0497 and also remains constant for retrieval of 150 elements at a value of 0.1040. It increases from a MAiP score of 0.1410 for 1500 elements from 25 documents to a MAiP score of 0.1675 for 1500 elements from 150 documents and remains constant thereafter.

3. As seen in Table 6, the results obtained in these experiments indicate that the University of Minnesota Duluth would rank 4th in the top 5 participant scores. See [3].

We are awaiting data from INEX for the significance testing of the INEX 2010 Efficiency Task results.

Table 6 : INEX 2010 Efficiency Task Results
Compared to the Top Participant Scores

Rank	Participant	MAiP
1	p167-18P167	0.2354
2	p4-OTAGO-2010-10topk-18	0.2304
3	p68-LIP6-OWPCRefRunTh	0.2196
*	**p72-UMD**	**0.1675**
4	p29-ISI2010 thorough	0.0846
5	p98-I10LIA4FBas	0.0417

5 Conclusions and Future Work

Conclusions

From the slope-pivot experiments carried out, we have concluded that the slope that works best for this collection is 0.11. The INEX 2009 Thorough Task and the INEX 2010 Efficiency Task give good results when carried out with this value of slope. The other Ad Hoc Track Tasks for INEX 2009 and 2010 also produce good results with this value of slope [1], [8].

Future Work

The Ad Hoc Track has been dropped from INEX 2011 and has been replaced by the Snippet Retrieval Track. This track is very similar to the Restricted Focused Task and Restricted Relevant in Context Tasks in the INEX 2010 Ad Hoc Track. The IR research group at UMD has the necessary techniques to successfully complete the tasks in the INEX 2011 Snippet Retrieval Track.

References

[1] Acquilla, N. Improving Results for the 2009 and 2010 INEX Focused Tasks, MS Thesis, Department of Computer Science, University of Minnesota, Duluth, 2011.
http://www.d.umn.edu/cs/thesis/Acquilla.pdf

[2] Arvola P., Geva S., Kamps J., Trotman A. Overview of the INEX 2009 Ad Hoc Track
http://www.cs.otago.ac.nz/homepages/andrew/2009-13.pdf

[3] Arvola P., Geva S., Kamps J., Trotman A. Overview of the INEX 2010 Ad Hoc Track
http://www.cs.otago.ac.nz/homepages/andrew/2010-13.pdf

[4] Bhirud, D. Focused Retrieval using Upper Bound Methodology, MS Thesis, Department of Computer Science, University of Minnesota, Duluth, 2009.
http://www.d.umn.edu/cs/thesis/Bhirud.pdf

[5] Crouch, C. Dynamic element retrieval in structured environment. *ACM TOIS*, 24(4): 437-454, 2006.

[6] Khanna, S., Design and Implementation of a Flexible Retrieval System. MS Thesis, Department of Computer Science, University of Minnesota Duluth, 2008.
http://www.d.umn.edu/cs/thesis/Khanna.pdf

[7] Mahule, A. Improving Results for the INEX Thorough Task, MS Thesis, Department of Computer Science, University of Minnesota, Duluth, 2010.
http://www.d.umn.edu/cs/thesis/Mahule.pdf

[8] Narendravarapu, R. Improving Results for the 2009 and 2010 INEX Relevant in Context Tasks, MS Thesis, Department of Computer Science, University of Minnesota, Duluth, 2011.
http://www.d.umn.edu/cs/thesis/Narendravarapu.pdf

[9] Paranjape, D. Improving Focused Retrieval. MS Thesis, Department of Computer Science, University of Minnesota Duluth, 2008.
http://www.d.umn.edu/cs/thesis/Paranjape.pdf

[10] Salton, G., ed. *The Smart Retrieval System – Experiments in Automatic Document Processing*. Prentice-Hall,1971.

[11] Salton, G., Wong, A., and Yang, C. A vector space model for information retrieval. *Communications of the ACM*, vol. 18, n. 11, pages 613–620, 1975.

[12] Salton, G., and Buckley, C. Term Weighting Approaches in Automatic Text Retrieval. *Information Processing and Management, Vol.32 (4)*, p. 431-443, 1996.

[13] Singhal, A., Buckley, C. and Mitra, M. Pivot document length normalization. *Proceedings of the 19th Annual International ACM Special Interest Group in Information Retrieval (SIGIR) Conference*. Zurich, Switzerland. 19-21, 1996.

[14] Website for INEX.
https://inex.mmci.uni-saarland.de/