ABSTRACT
The paper proposes the Indri model considering different term weights and PageRank score as prior. Then pseudo-relevance feedback (PRF) is performed. The proposed approaches are useful in extensive experiments, and the proposed PRF with soft selection can stably increase the performance. Very encouraging results are obtained.

Keywords
Indri Language, Pseudo-Relevance Feedback (PRF)

1. INTRODUCTION
The web page retrieval is the famous research topic, and it exists a lot of different retrieval models and approaches. Because the amount of web page is very large, there are some noisy information, and it's not easy to retrieve the relevant documents. Traditional approaches were proposed, including [5, 6, 1], and Google also proposed a famous approach, PageRank [4], which utilizes the links between the web pages to improve the performance, and the retrieval models can be refined and combined with a lot of approaches.

The paper proposes an approach, which uses weighted Indri retrieval model based on language model and combines different approaches; the performance can be improved by the proposed approach.

2. PROPOSED FRAMEWORK
The proposed framework is shown in Figure 3. The red block is the original information retrieval approach. After document preprocessing, indexing can be applied and then the retrieval model can be constructed, which are described in Section 3 and 4. When a user enter a query \( q \), we proposed an approach to compute term weights (left yellow block) after preprocessing. The detail about query processing is described in Section 5. Then the retrieval model can compute the relevance scores, \( S(d_j) \), for all documents, and the documents can be ranked according to relevance scores to form a first-pass result.

The blue block is the re-ranking approach. We apply pseudo-relevance feedback (PRF) to re-rank the documents. In this part, we utilize not only the conventional PRF approach, selecting fixed number of pseudo-relevant documents to re-retrieve the documents, called hard selection, but also the proposed approach, soft selection, flexibly selecting feedback number of documents. Section 8.4 describes pseudo-relevance feedback (PRF) in detail.

3. DOCUMENT PREPROCESSING
The terms in the corpus need to be preprocessed so that time of the query process can be decreased. We first apply word stemming to each term in the corpus and construct the inverted files.

3.1 Word Stemming
Considering the same stem of different terms can represent the same meaning, for each term in corpus we change it into its stem. With word stemming, the query terms can be easily matched so that the recall of performance can be increased. However, word stemming is not always useful; because it may delete some important information, the meanings of the terms would become less precise. For example, two terms, "use" and "user", have the same stem "use". After word stemming, the precise meaning of "user" would be deleted so that the performance may decrease.

3.2 Indexing
Indexing lets the time for retrieval decrease, and we generate the following three files, vocabulary, inverted file, and title field indexing.

Vocabulary After word stemming procedure is done, we build the vocabulary list for all documents. We, first, separate all the terms in documents by space character, and then record all the words in a vocabulary. After removing redundant words, the words in the vocabulary are unique.

Inverted File Once the vocabulary are built, we construct inverted files for further indexing. An inverted file is the sorted list of vocabulary, which each vocabulary having links to the documents containing that vocabulary. Figure 2 illustrates the key concept of the inverted file. Each entity (vocabulary) in the inverted file keeps the value of document frequency (DF) and term frequency (TF) in each documents. Inverted Files also keep the position each vocabulary located, so for each
4. RETRIEVAL MODEL

In this section, we introduce three different conventional retrieval models described in Section 4.1, 4.2, and 4.3 respectively. Nowadays these models are widely used and compared as the baseline in many ad hoc retrieval projects. Furthermore, modern retrieval systems are almost based on them. For example, Section 4.4 describes an Indri retrieval model and inference network retrieval frameworks.

4.1 Vector Space Model (VSM)

Vector space model is one of the most popular retrieval models in the ad hoc retrieval, because of not only its simplicity on implementation but also its performance. The relevance between the query and the document depends on the similarity between their terms. To compare the similarity between the query term and the document terms, we represent each query and document as a vector and the weight of each term is computed as the cosine similarity in the vector space.

\[
R_{VSM}(q, d) = \text{sim}(\vec{q}, \vec{d}) = \frac{\vec{q} \cdot \vec{d}}{|\vec{q}| |\vec{d}|},
\]


4.2 Okapi BM25

Okapi BM25 model was another popular retrieval model in the ad hoc retrieval. First developed in the 1970s and 1980s by Stephen E. Robertson, Karen Sparck Jones [6]. Actually, the name of scoring function is called BM25. However, it usually referred to as “Okapi BM25” since the Okapi IR system implemented at London’s City University, was the first system to implement BM25 function. Unlike VSM, Okapi BM25 model was based on probabilistic retrieval framework. Given a query q, the BM25 score of a document dj is as

\[
R_{BM25}(q, d_j) = \sum_{t_i \in q} \frac{tf_{q, t_i} (k_1 + 1)}{tf_{q, t_i} + k_1 (1 - b + b \frac{dl_j}{avg(dl)})},
\]

where \(tf_{q, t_i}\) is term frequency (TF) of \(t_i\) in the document \(d_j\), \(dl_j\) is the length of the document \(d_j\), and \(avg(dl)\) is the
average document length in the text collection from which
documents are drawn. \( k_1 \) and \( b \) are free parameters. \( idf_t \)
is the inverse document frequency (IDF) weight of the term \( t_i \),
\( N \) is the total number of documents in the collection, and
\( df_t \), document frequency (DF), is the number of documents
containing \( t_i \).

4.3 Language Model
Language model approach for IR attempts to model query
generation process; documents can be ranked by the proba-
bility that a query would be observed as a random sample
from the respective document model. Thus, we train a lan-
guage model for each document \( d_j \) called \( M_{d_j} \). Then, for a
query \( q \), we can rank the documents by following equation:

\[
R_{LM}(q, d_j) = P(q | d_j) = P(d_j) \cdot P(q | d_j) \\
\simeq P(d_j) \cdot P(q | M_{d_j}) \\
\propto P(q | M_{d_j}).
\]  

(7)

We can use maximum likelihood estimation to compute the
query generation probability given document model by uni-
gram.

\[
P(q | M_{d_j}) = \prod_{t_i \in q} P(t_i | M_{d_j}) = \prod_{t_i \in q} \frac{tf_{t_i,j}}{dl_j}, \tag{8}
\]

where \( tf_{t_i,j} \) is the term frequency of term \( t_i \) in document \( d_j \),
and \( dl_j \) is the document length of document \( d_j \).

Because insufficient data may produce zero probability;
for example, some query terms are unseen term in some
documents such that \( tf_{t_i,j} = 0 \). To avoid zero probability,
the smoothing is necessary, and the (8) can be changed into
(9).

\[
P(q | M_{d_j}) = \prod_{t_i \in q} \Lambda P(t_i | M_{d_j}) + (1 - \lambda)P(t_i | M_c), \tag{9}
\]

\[
P(t_i | M_c) = \frac{cf_{t_i}}{cl}, \tag{10}
\]

where \( \lambda \) is a weighting factor and \( M_c \) is the collection model;
\( cf_{t_i} \) is collection frequency of term \( t_i \) and \( cl \) is collection
length. With the smoothing approach, because \( cf_{t_i} \neq 0 \),
unseen terms won’t lead to zero probability.

4.4 Indri Retrieval Model
Although Language Model is quite powerful in manipu-
lating query generation process, it’s still lack of the ability
to handle various query type, such as "A and B" or "A or B".
In order to adapt such kind of situation, we introduce Indri
retrieval model based on a combination of the language mod-
elling and inference network retrieval frameworks. Language
model is good at modeling document content and query lan-
guage and inference network (also known as a Bayesian net-
work) is a succinct way of defining a joint probability distri-
bution over a collection of random variables. Both models
are found to be very effective for a wide range of IR task.
Therefore, combining the benefits of these two frameworks
may lead to a great successful and powerful unified model,
called Indri.

Indri was first introduced in 2004 \cite{2}. It’s also imple-
mented and applied in TREC Terabyte Track \cite{3}. Fortu-
nately, Lemur toolkit is the state-of-the-art open source
which implements Indri and people can easily access this
complex model by a few commands.

Here are some features of Indri model listed in lemur
toolkit website\footnote{\url{http://www.nlpir.nist.gov/projects/terabyte/}}:

- Easily handles phrases (ordered and unordered)
- Can make use of multiple document representations
- Explicit term weighting
- Robust query language
- Formally well-grounded
- Highly effective
- Can be efficiently implemented

Let’s take a deeper look! Figure 3 illustrates an example
for inference network. Each node in this graph represents
a random variable, with the shaded nodes being "observed"
and the unshaded nodes being "hidden". The edges in the
graph define a set of independence assumptions over the
random variables.

In Indri model, there are six node types as following.

1. Document node \((D)\)
2. Smoothing parameter nodes \((\alpha, \beta)\)
3. Model nodes \((M)\)
4. Representation concept nodes \((r)\)
5. Belief nodes \((q)\)
6. Information need node \((I)\)

We detail the meaning of each node type in the following
paragraph.

1. Document node \((D)\)
The document node is a random variable over document
representations. Indri represents documents as
multisets of binary vectors, where each entry in the
binary vector represents the presence or absence of some
feature of the text. One such vector is ’extracted’ for
every term position within the document. This treats

\footnote{\url{http://www.lemurproject.org/}}
the document as a sample from a Multiple-Bernoulli distribution.

The following illustrates a simple example that considers unigram and bigram features. There are a total of 12 features, with 5 feature vectors extracted, one for each text position. Assume feature order, \( \vec{f} = (A \ B \ C \ AA \ AB \ AC \ BA \ BB \ BC \ CA \ CB \ CC) \) (unigram feature and bigram feature),

\[
D = \begin{pmatrix}
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\]

2. Smoothing parameter nodes \((\alpha, \beta)\)

These nodes (parameter) is used as smoothing parameter for language model. As we know there are plenty of smoothing methods for language model. Therefore, there is more than one set of smoothing parameter nodes for each model node.

3. Model nodes \((M)\)

The model nodes represent so-called feature language models. Unlike transition LM, in this framework there may be more than one model node in the network. In the example above, there are three model nodes \((M_{\text{title}}, M_{\text{body}}, M_{h1})\), corresponding to different representations of the same document. For example, the title model corresponds to the pseudo-document made up of all of the title text from the original document, whereas the body model corresponds to all of the text within the main body of the original document. This allows models to be estimated across various document representations and evidence from the different representations to be combined.

4. Representation concept nodes \((r)\)

In order to represent the document in well format, Indri uses binary random variable to indicate that whether the feature exists or not. These random variable are called Representation concept nodes.

Some examples query events associated with these nodes are:

- "the term dog occurred"
- "the term Mike appeared within a title"
- "the exact phrase 'white house' occurred"

Using Representation concept nodes, we are able to distinguish between the event "the term dog occurred in title" and "the term dog occurred in body."

5. Belief nodes \((q)\)

To makes the frame work powerful, Indri introduces "Belief nodes." Belief nodes are binary random variable which represents different probabilities with the network. For example, belief "combine(A B)" means the co-occurrence probability of A and B. Each belief node corresponds to a different conditional probability table described in Indri Query Language Quick Reference\(^3\), allowing beliefs to be combined in a number of ways.

\(^3\)http://ciir.cs.umass.edu/~metzler/indriquerylang.html

This belief functionality are dynamic and can be added to the network according to the structure of the query. Therefore, the network topology/structure changes for every query. This makes the framework very powerful, as it allows a wide range of scoring functions, all based on different query formulations.

6. Information need node \((I)\)

Finally, the information need node is simply a belief node that combines all of the evidence within the network into a single belief. This belief is used as the basis for ranking documents. That is, documents are ranked according to \(P(I = 1|D, \alpha, \beta)\) (I means Information need node).

To easily understand the functionality of inference network in Indri model, we give three three examples to illustrate it. Each example contains structured Indri queries and the concept graph of inference network.

- **Example 1:**
  Query: #combine(#1(abraham lincoln) gettysburg)
  Figure 4 shows the graph of inference network.

- **Example 2:**
  Query: #weight(2.0 #or(#1(north korea) iraq) 1.0 policy)
  Figure 5 shows the graph of inference network.

\[
b_{\#\text{combine}} = \prod_i b_i^{\frac{1}{n}}
\]  

\[
b_{\#\text{weight}} = \prod_{i=1}^n b_i^{w_i}
\]  

**Figure 4: Inference network: Example 1**

- **Example 3:**
  Query: #weight(2.0 #or(#1(north korea) iraq) 1.0 policy)
  Figure 5 shows the graph of inference network.

\[
b_{\#\text{combine}} = \prod_i b_i^{\frac{1}{n}}
\]  

\[
b_{\#\text{weight}} = \prod_{i=1}^n b_i^{w_i}
\]

**Figure 5: Inference network: Example 2**
\[ W = \sum_{i=1}^{n} w_i \]  

**Figure 5: Inference network: Example 2**

- Example 3:
  Query:
  \#combine( \#uw8( hurricane wind ).(title) damage )
  Figure 6 shows the graph of inference network.

  This example shows the power of Indri model. Query retrieves documents from two different language models. "hurricane wind" use \( M_{\text{title}} \) which is the title part of document and "damage" use \( M \) which is the original model.

**Figure 6: Inference network: Example 3**

5. QUERY PREPROCESSING

5.1 Word Stemming

As document preprocessing, we apply word stemming to the query so that the terms can be matched by the vocabulary.

5.2 Stop Word Removal

Because stop words may affect the relevance score computation, stop word removal for query processing can be considered. We use two different stop word lists, the automatically generated list and the manual list.

6. QUERY LANGUAGE GENERATION

We generate the query language after preprocessing, and we use prior score for each document and different weights for the terms in the query, which are described as below.

6.1 PageRank for Prior Scores

Considering the links of documents, we apply the famous approach, PageRank [4], to compute the scores as prior. PageRank gives higher scores to the important pages, which are linked by more pages. Therefore, the score of document \( d_j, s(j) \), can be computed by

\[ s(j) = \alpha \frac{1}{N} + (1 - \alpha) \sum_{d_k \to d_j} \frac{1}{|L(d_k)|} s(k), \]  

where \( s(j) \) is the score of document \( d_j \), which depends on the number of documents linking to \( d_j \). \( d_k \) is the document with a link directing to \( d_j \), and \( |L(d_k)| \) is the number of the out-links from document \( d_k \). \( \alpha \) is the damping factor, and \( 0 \leq \alpha \leq 1 \). The idea is that the document with more in-links from documents with higher scores would be given higher score. Therefore, the final scores are just used as prior because the scores are not related to the query.

6.2 Term Weight Computation

Because different terms have different significance, we assign different weights for each term in the query. To decrease the weight of the insignificant term, we compute inverse document frequency (IDF) as the significance measure, which is described in (6). We remain the same term in the query to involve term frequency (TF) in the query language. Also, there are different fields in the query documents, such as title, description, and narrative, and the term in the description should be more insignificant than the term in the title, so we assign different weights to the terms from different fields. Section 4.4 introduce the retrieval model can record if the terms occurs in the title from corpus. Considering the terms occurring in the title are more important, if some terms are in the titles of query and document, they should obtain higher weights.

The weights of terms in the query are listed in Table 1. Note that the terms occurring in the title of query and document can obtain the weight \((1 + w_c) \cdot \text{idf}\), and that \( 0 \leq w_c, w_d, w_n \leq 1 \).

7. PSEUDO-RELEVANCE FEEDBACK (PRF)

In many retrieval systems, PRF is performed as the second-pass after original result is derived, and it has been proved in many researches that PRF always has significantly improvement on the retrieval result. The theorem of PRF is that documents with relatively higher relevance scores must
have higher probability to be relevant documents. Thus, we can make use of these documents to add extra information to the original retrieval system. However, how to select the pseudo-relevant documents as positive examples is still a major problem in PRF. In the following, two different selection methods, hard selection and soft selection, are introduced.

### 7.1 Hard Selection

The easiest way to select the documents as pseudo-relevant documents for PRF is hard selection. That is, feedback number $K$ is set previously and top $K$ documents are selected as pseudo-relevant documents according to their relevance scores. The advantage of this selection criterion is its simplicity on implementation and the acceptable performance. However, the power of PRF is based on the precision to pseudo-relevant documents. It means that if the precision of selected documents is higher enough, the improvement to the original result would be much more and vice versa. Therefore, the result is highly depended on the chance when we set a fixed feedback number $K$.

### 7.2 Soft Selection

Because the difficulties of queries are different, we shouldn’t use the same number of pseudo-relevant documents for PRF. We consider different numbers of pseudo-relevant documents for different queries. The idea is that the relevance score would significantly decrease when seeing from the relevant document to the irrelevant document. The number of pseudo-relevant documents, $K_{\delta}$, is chosen according to first-pass retrieved results.

$$K_{\delta} = \min \{ \arg \min \delta (S(d_i) > \delta), 10 \},$$

where $S(d_i)$ is the relevance score of document $d_i$ from the first-pass result, $\delta$ is a threshold to decide where is the boundary between the relevant document and the irrelevant document, and we also set the number of pseudo-relevant documents is less or equal to 10.

## 8. EXPERIMENT

### 8.1 Experimental setup

The corpus used in this project is a small portion of WT10G, a TREC Web Corpus, which contains 317,915 documents. Each document is a page of a website and the format is an usual html webpage which provides information such as title, released date, all out-links from this page and the contents of this website. 30 query topics, including 20 for training and 10 for testing, are used to evaluated the performance of all mentioned approaches in this report. Each query topic contains three parts of contents: Title, Description and Narrative. Title

<table>
<thead>
<tr>
<th>Field</th>
<th>Weight</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>$idf$</td>
<td>also in the title of document</td>
</tr>
<tr>
<td>Title</td>
<td>$w_d \cdot idf$</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>$w_d \cdot idf$</td>
<td></td>
</tr>
<tr>
<td>Narrative</td>
<td>$w_n \cdot idf$</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Weights of terms in the query.

is a short sentence describing the requirement for this topic while Description is a more detail description which contains about 10 words. And Narration illustrate clearly what are relevant and what are irrelevant to this topic.

In this project, mean average precision (MAP) and precision at 10 (P@10) are two main measurements used to measure the experimental results. In the following, we show the experimental results that have been mention in this report. Furthermore, each experiment is conducted in two different types of indexing, one with stemming and another without stemming when preprocessing is conducted.

### 8.2 Conventional Retrieval Models

In this section, different types of conventional retrieval models are compared. The experimental results are shown in Table 2. Generally, the retrieval results of language model are better than that of vector space model and okapi model in all cases. Moreover, processing stemming seems helpful for all models. Therefore, the best result of conventional retrieval models can be found on language model with stemming.

In addition, we also found that training topics are slightly harder than testing topics. That is, the MAP and P@10 of training topics are always lower than that of testing topics, but the reason may be insufficient amount of testing topics.

### 8.3 Indri Retrieval Model

#### 8.3.1 Basic Indri Model

As mentioned in Section 4.4, Indri model considers only the content in the web page, but also the topic of the web page. In basic Indri model, each query term with equal weight $w_d = w_n = 1$ and different title weight $w_t = 0.5$ is set to the title term. The result of basic Indri model is shown in Table 3. Compared to the previous result of language model, it is easy to find that the MAP increases in almost all cases except for the training topics without stemming. However, the P@10 increases only in testing topics without stemming; however, the basic Indri model is only another type of language, and the effectiveness of Indri model will be seen in the following experiments.

#### 8.3.2 Stop Word List

In the previous experiments, the query used for retrieval consists of all existing terms in the given query document, including title, description, and narrative. However, some query terms are either harmful or useless to our retrieval system described in Section 5.2.

Here two set of stop word list are applied to reform the query. The experimental results are shown in Table 3. It is clearly that removing top 100 stop words are harmful to the retrieval result in all cases while removing manul selected stop words has significant improvement except for the P@10 in testing topics with stemming. This result imply that if we could reformulate the query carefully, the result of retrieval would be much better.

#### 8.3.3 Term Weight

In this section, different weights are set to different query terms according to their idf values. $w_t = 0.3$, $w_d = w_n = 0.7$ are set to decrease the weights of the terms in less important fields. The value of idf is widely used for differentiating the importance of a single index term. The larger the idf
Table 2: Comparison of conventional retrieval models. (%)

<table>
<thead>
<tr>
<th>Approach</th>
<th>Training Topics</th>
<th>Testing Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>W/O Stem</td>
<td>W/ Stem</td>
</tr>
<tr>
<td></td>
<td>MAP, P@10</td>
<td>MAP, P@10</td>
</tr>
<tr>
<td>Vector Space</td>
<td>29.11, 27.50</td>
<td>30.98, 27.50</td>
</tr>
<tr>
<td>Okapi</td>
<td>24.91, 23.50</td>
<td>24.50, 24.50</td>
</tr>
<tr>
<td>Language Model</td>
<td>30.52, 29.50</td>
<td>31.11, 29.00</td>
</tr>
</tbody>
</table>

Table 3: Indri retrieval models with different refined methods. (%)

<table>
<thead>
<tr>
<th>Approach</th>
<th>Training Topics</th>
<th>Testing Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>W/O Stem</td>
<td>W/ Stem</td>
</tr>
<tr>
<td></td>
<td>MAP, P@10</td>
<td>MAP, P@10</td>
</tr>
<tr>
<td>Basic Indri Model</td>
<td>30.13, 31.25</td>
<td>31.25, 27.50</td>
</tr>
<tr>
<td>+ top100 stop word</td>
<td>28.39, 29.50</td>
<td>29.92, 25.50</td>
</tr>
<tr>
<td>+ manual stop word</td>
<td>31.66, 34.88</td>
<td>33.90, 30.00</td>
</tr>
<tr>
<td>+ weight</td>
<td>31.83, 33.90</td>
<td>33.90, 30.00</td>
</tr>
<tr>
<td>+ PageRank</td>
<td>29.24, 30.85</td>
<td>28.50, 30.00</td>
</tr>
</tbody>
</table>

Figure 7: The MAP of hard selection under different pseudo-relevance feedback numbers.

Figure 8: The P@10 of hard selection under different pseudo-relevance feedback numbers.

value always means the more importance of the query term; hence, the query weighted by idf value has another meaning of reformulating the origin query. The result is compared to that of basic Indri model in Section 8.3.1. By observing the results, it is clearly that all measurements increase except for the MAP in testing topics with stemming, so assigning different weights for query terms is useful, considering different importance for different query terms.

8.3.4 PageRank

In some retrieval systems, PageRank score of a document is involved to be a prior score to derive the final relevance score as described in Section 6.1. The result of PageRank used in Indri retrieval model is shown in Table 3, too. Although it decreases MAP in almost all measurements, P@10 increase a little when stemming is involved. PageRank can involve the documents with more in-links but not including the query terms. Therefore, in the following experiments, PageRank will be combined with others.

8.3.5 Overall Results

In this experiment, we applying all the above refined methods altogether in Indri model. That is, Indri model is conducted with manual stop word removal, different weights, and PageRank. The result shown in Table 4 implies that all the methods proposed above have additive effects to each other. The result is listed as row (d) in Table 4, we can see the performance of MAP is the best for all topics, and the performance of P@10 for all topics in row (c) is the best.

In addition, we also apply PRF, which are described in Section 8.4, with hard selection or soft selection to the results. The performance for all topics can be improved by PRF with hard or soft selection, and the soft selection performs better. However, the improvement can be shown in training topics but not in testing topics, as we have mentioned before, testing topics include too less queries, only 10 queries, to judge the actual performance of different retrieval methods. Thus, according to all topics, our proposed approaches are all useful.

8.4 Pseudo-Relevance Feedback

After deriving the first-pass results, PRF is conducted in the following experiments to see the sensibility of feedback number of documents. As mention in , different PRF selection criteria are performed.
<table>
<thead>
<tr>
<th>Approach</th>
<th>Training Topics</th>
<th>Testing Topics</th>
<th>All Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAP</td>
<td>RI</td>
<td>P@10</td>
</tr>
<tr>
<td>(a)</td>
<td>31.25</td>
<td>-</td>
<td>27.50</td>
</tr>
<tr>
<td>(b)</td>
<td>34.88</td>
<td>11.62</td>
<td>30.00</td>
</tr>
<tr>
<td>(c)</td>
<td>36.80</td>
<td>17.76</td>
<td>32.00</td>
</tr>
<tr>
<td>(d)</td>
<td>36.88</td>
<td>18.02</td>
<td>30.00</td>
</tr>
<tr>
<td>(e)</td>
<td><strong>38.43</strong></td>
<td><strong>22.98</strong></td>
<td><strong>34.50</strong></td>
</tr>
<tr>
<td>(f)</td>
<td>37.98</td>
<td>21.54</td>
<td>34.00</td>
</tr>
</tbody>
</table>

(a): Indri retrieval model  
(b): (a) + manual stop word  
(c): (b) + weight (w_t = 0.3, w_d = 0.5, w_n = 0.5)  
(d): (c) + PageRank (w_p = 0.7)  
(e): (d) + PRF-hard (K = 9)  
(f): (d) + PRF-soft (δ = 0.1)

Table 4: Indri language retrieval models with combination method(%)
show that all approaches are useful to construct the better retrieval model.

10. REFERENCES


