DCU@FIRE2010: Term Conflation, Blind Relevance Feedback, and Cross-Language IR with Manual and Automatic Query Translation

Johannes Leveling
Centre for Next Generation Localisation
School of Computing
Dublin City University
Dublin 9, Ireland

Debasis Ganguly
Centre for Next Generation Localisation
School of Computing
Dublin City University
Dublin 9, Ireland

Gareth J. F. Jones
Centre for Next Generation Localisation
School of Computing
Dublin City University
Dublin 9, Ireland

ABSTRACT

For the first participation of Dublin City University (DCU) in the FIRE 2010 evaluation campaign, information retrieval (IR) experiments on English, Bengali, Hindi, and Marathi documents were performed to investigate term conflation (different stemming approaches and indexing word prefixes), blind relevance feedback, and manual and automatic query translation. The experiments are based on BM25 and on language modeling (LM) for IR. Results show that term conflation always improves mean average precision (MAP) compared to indexing unprocessed word forms, but different approaches seem to work best for different languages. For example, in monolingual Marathi experiments indexing 5-prefixes outperforms our corpus-based stemmer; in Hindi, the corpus-based stemmer achieves a higher MAP. For Bengali, the LM retrieval model achieves a much higher MAP than BM25 (0.4944 vs. 0.4526). In all experiments using BM25, blind relevance feedback yields considerably higher MAP in comparison to experiments without it. Bilingual IR experiments (English→Bengali and English→Hindi) are based on query translations obtained from native speakers and the Google translate web service. For the automatically translated queries, MAP is slightly (but not significantly) lower compared to experiments with manual query translations. The bilingual English→Bengali (English→Hindi) experiments achieve 81.7%-83.3% (78.0%-80.6%) of the best corresponding monolingual experiments.

Categories and Subject Descriptors

H.3.1 [**INFORMATION STORAGE AND RETRIEVAL**]: Content Analysis and Indexing—*Indexing methods*; H.3.3 [**INFORMATION STORAGE AND RETRIEVAL**]: Information Search and Retrieval—*Query formulation, Relevance feedback, Search process*

1. INTRODUCTION

The Forum for Information Retrieval Evaluation (FIRE) provides document collections, topics, and relevance assessments for information retrieval (IR) experiments on Indian languages. Similar to other IR evaluation initiatives such as TREC[1], NTCIR[2], or CLEF[3], FIRE[4] aims at comparing the retrieval performance of different systems and approaches and at investigating evaluation methods for IR. FIRE started in 2008 with document collections for English, Bengali, Hindi, and Marathi.

This paper describes the participation of Dublin City University (DCU) in the FIRE 2010 evaluation. Monolingual and bilingual IR experiments for English and for the Indian languages Bengali, Hindi, and Marathi have been performed to investigate aspects of term conflation (stemming and prefix indexing), blind relevance feedback (BRF), and manual and automatic query translation.

The rest of this paper is organized as follows: Section 2 introduces related work. Section 3 describes preparations for the retrieval experiments and the experimental setup. The experiments and results for monoliongual and bilingual IR experiments are presented in Section 4. The paper concludes with an outlook on future work in Section 5.

2. RELATED WORK

Larkey, Connell et al. normalize Hindi multi-byte characters using manually crafted rules for the TIDES (Translingual Information Detection, Extraction, and Summarization) surprise language exercise for Hindi IR experiments [10]. More recently, research on information retrieval on Indian languages has been encouraged by the FIRE 2008 evaluation campaign.

Dolamic and Savoy [3] used language modeling (LM) and divergence from randomness (DFR) for Indian language IR in the FIRE 2008 evaluation. Their approach employs light stemming [20], stopword removal based on small stopword lists, and Rocchio-style feedback with $\alpha = \beta = 0.75$.

Xu and Oard [22] apply a Perl Search engine on the FIRE data for English-Hindi CLIR. They employ a stopword list with 275 words for Hindi IR.

McNamee employs n-grams and skipgrams (n-grams with wildcards) as indexing units for IR on English, Bengali, Hindi, and Marathi documents using language modeling (LM) as a retrieval model [13]. He experimented with different but fixed numbers of expansion terms for different indexing

[1] http://trec.nist.gov/

[2] http://research.nii.ac.jp/ntcir/

[3] http://www.clef-campaign.org/

[4] http://www.isical.ac.in/~fire/

methods: 50 feedback terms for words, 150 for 4-grams and 5-grams, and 400 for skip-grams. Additional experiments on the FIRE 2008 data use n-grams on running text, and word truncation (prefixes) [14]. Significant improvements for indexing n-grams compared to the baseline of indexing words are observed. The best effectiveness for Hindi and Bengali is achieved when using 4-grams, highest MAP for Marathi by word-internal 4-grams.

Stemming approaches can be classified into different categories, e.g. by the results produced by the stemmer (light stemming [19] vs. aggressive stemming [11]) or by the resources used (corpus-based [21] vs. dictionary-based [9, 12]).

The most widely used stemming approach for English is the rule-based Porter stemmer [16], which successively applies rules to transform a word form into its base form. The successive removal of affixes means that words with a recursive morphological structure are reduced to their base form, e.g. words such as *'hopelessness'* may be reduced to *'hope'* by removing the suffixes *'ness'* and *'less'*.

Light stemming focuses on removing only few but the most frequent suffixes from word forms. Recently, light stemming has been researched as a less aggressive means to reduce words to their root form. For English, the s-stemmer which removes only the *'-s'*, *'-es'*, and *'-ies'* suffixes from words and other light stemming approaches have been proposed (see, for example, [7] and [20]).

YASS is a clustering-based suffix stripper which has been applied to English, French, and Bengali [12]. YASS identifies clusters of equivalence classes for words by calculating distance measures between strings. This stemmer does not handle morphologic prefixes and relies on several dictionaries which have to be extracted from documents, i.e. all words starting with the same character have to be collected in the same word list in a scan over all documents.

Xu and Croft [21] use a combination of aggressive suffix removal with co-occurrence information from small text windows to identify stemming classes. This technique is corpus-based and requires little knowledge about the document language. The original stemmer was developed for a Spanish document collection [21] and shows an increase in recall for Spanish.

Goldsmith [6] identified suffixes employing a minimum description length (MDL) approach. MDL reflects the heuristic that words should be split into a relatively common root part and a common suffix part. Every instance of a word (token) must be split at the same breakpoint, and the breakpoints are selected so that the number of bits for encoding documents is minimal.

Oard, Levow et al. [15] apply the Linguistica tool by Goldsmith [6] to create a statistical stemmer. Suffix frequencies are computed for a subset of 500,000 words in a document collection. The frequencies of suffixes up to a length of 4 were adjusted by subtracting the frequency of subsumed suffixes. Single-character suffixes were sorted by the ratio between their final position likelihood and their unconditional likelihood. Suffixes were sorted in decreasing order of frequency, choosing a cutoff value where the second derivate of the frequency vs. rank was maximized.

3. EXPERIMENTAL SETUP

The following subsections describe the experimental setup for the participation of DCU in the FIRE 2010 ad hoc retrieval task. Three approaches to term conflation are employed: n-prefixes, corpus-based stemming, and a rule-based stemming.

3.1 Term conflation

3.1.1 N-prefixes

The goal of term conflation is to reduce different derivational or inflectional variants of the same word to a single indexing form to increase performance in IR. Full word forms can be conflated by truncating words after n characters. This approach is inexpensive and language-independent, because it does not rely on additional external language resources. For experiments on the FIRE 2008 document collections and topics, we found that $n = 5$ or $n = 6$ produced the highest MAP for all languages tested (English, Bengali, Hindi, Marathi).

3.1.2 Corpus-based Stemming

A corpus-based, language-independent stemming approach was implemented following a morpheme induction approach described by Dasgupta and Ng [1, 2], which has been evaluated for English and Bengali. On a manually annotated set of Bengali words this approach achieved a substantially higher F-score than Linguistica [6].

For the retrieval experiments described in this paper, the first steps of the morpheme induction were implemented to obtain a stemmer. The additional morpheme induction steps described in [2] mainly test the validity of composite suffix candidates and suffix attachments. The morpheme induction produces a list of candidate suffixes based on a frequency analysis of potential word roots and suffixes. For example, the English word *'hopeful'* is split into the root-suffix pairs *'hop'*+*'eful'*, *'hope'*+*'ful'*, and *'hopef'*+*'ul'*. The middle variant is chosen, because its root and suffix frequency are highest.

In a second step, suffix combinations (forming composite suffixes) are determined via the frequency of potential root forms, which allows for a recursive morphological word structure. A word is stemmed by removing the longest suffix found in the generated suffix lists or by not removing a suffix, otherwise.

The list of candidate suffixes is produced following a method suggested by [8]. In the first step, all words w are analyzed by successively selecting all possible segmentation points splitting them into a potential root form r and a suffix s. Thus, w is the concatenation of r and s. If the potential root form r can also be found in the set of raw word forms (e.g. it is part of the collection vocabulary and the root frequency is higher than 0), s is added to the list of suffix candidates and r is added to the list of root candidates. Candidate suffixes are filtered as follows:

1. As a minor variation of the approach proposed by Dasgupta and Ng [2], suffixes with a frequency less than a given threshold t_f are removed (in this case, $t_f < 5$).

2. A score is assigned to each suffix by multiplying the suffix frequency and the suffix length in characters. Using suffix length as a scoring factor is motivated by the observation that short, low-frequency suffixes are likely to be erroneous [6].

The suffix candidates are then ranked to obtain the top K suffixes. For the experiments described here, a fixed value of

Table 1: Bengali examples for corpus-based stemming.

Stemming rule	Example
A[khani] → A	[*chhobikhani*]→ [*chhobi*]
A[khana] → A	[*tukrokhana*]→ [*tukro*]
A[bhaabe]→ A *	[*bistaaritobhaabe*]→ [*bistaarito*]
A[chhe] → A	[*bhaaschhe*]→ [*bhaas*]
A[der] → A *	[*gyanider*]→ [*gyan*]
A[goner] → A	[*jonogoner*]→ [*jono*]
A[Taake] → A	[*poribeshTaake*]→ [*poribesh*]
A[Tike] → A	[*monTike*]→ [*mon*]
A[Taay] → A *	[*jaigaTaay*]→ [*jaiga*]
A[Tite] → A	[*jomiTite*]→ [*jomi*]
A[Tiro] → A	[*ghorTiro*]→ [*ghor*]
A[Tuku] → A *	[*sonchoyTuku*]→ [*sonchoy*]
A[gaami] → A	[*kolkatagaami*]→ [*kolkata*]

$K = 50$ was used for all languages tested. Dasgupta and Ng used the same number of suffixes for morpheme induction for English [2]. Considering that about 60 affix removal rules are defined by the Porter stemmer this seems a plausible setting for mildly aggressive stemming.

In a second step, composite suffixes are detected by combining all suffixes in the induced candidate list, e.g. *'lessness'* in *'fearlessness'*. The detection of composite suffixes s_1+s_2, builds on the assumption that a root form r will also combine with part of the suffix (s_1). This property typically does not hold for non-composite suffixes. The morpheme induction method presumes that s_1+s_2 is a composite suffix if s_1+s_2 and s_1 are similar in terms of the words they can combine with. Specifically, s_1+s_2 and s_1 are considered to be similar if their similarity value – which is calculated as shown in Equation 1 – is greater than a threshold t_s (specifically, $t_s > 0.6$ was used).

$$\mathit{similarity}(s_i + s_j, s_i) = P(s_i | s_i + s_j) = \frac{|W^{iji}|}{|W^{ij}|} \qquad (1)$$

where $|W^{iji}|$ is the number of distinct words that combine with both s_i+s_j and s_i, and $|W^{ij}|$ is the number of distinct words that combine with s_i+s_j.

The corpus-based stemmer reads the lists of suffixes and processes words which are longer than a given threshold t_l ($t_l = 3$). All other words remain unstemmed. The stemmer determines the one longest suffix in the suffix lists (if any) and removes it from the word to produce a root form. Some example suffixes which are removed by the corpus-based stemmer for Bengali words are shown in Table 1 (* indicates cases also handled by the rule-based stemmer, which is described in the next subsection). Throughout this paper we have used italicized Romanized phonetic transliteration enclosed within square braces to represent Bengali strings. Stemming rules are specified as A[*suffix*] → A where 'A' denotes a string in Bengali and 'Cond.' represents a condition that has to be satisified before a stemming rule can be applied.

3.1.3 Rule-based Stemming

A simple (but requiring a-priori linguistic knowledge) approach towards stemming is to successively apply rules to a given word form to remove the most common suffixes. The advantages of a rule-based stemmer over a clustering-based approach are that firstly it is much faster and secondly it does not require any pre-processing on the indexed documents, hence making it a suitable candidate for retrieval tasks where the document collection is not large enough for statistical training. A disadvantage is that the stemming rules may have to be created manually and for each language. For the rule-based stemmer employed for Bengali IR, the stemming rules were compiled manually by one of the authors who is a native Bengali speaker.

Table 2: Rules for simple suffixes with transliterated Bengali examples.

Stemming rule	Cond.	Example
A[*i*] → A	$\vert A\vert \geqslant 4$	[*aadhikyai*]→ [*aadhikya*]
A[*o*] → A	$\vert A\vert \geqslant 4$	[*bigyaaniraao*]→ [*bigyaaniraa*]

Table 3: Rules for compound suffixes with transliterated Bengali examples.

Stemming rule	Cond.	Example
A[*Taa*] → A	$\vert A\vert \geqslant 4$	[*mukhoshTaa*]→ [*mukhosh*]
A[*Ti*] → A	$\vert A\vert \geqslant 4$	[*phoolTi*]→ [*phool*]
Ax[*taa*] → A	$\vert A\vert \geqslant 4$	[*satarkataa*]→ [*satarka*]
A[*raa*] → A	$\vert A\vert \geqslant 4$	[*konishthoraa*]→ [*konishtho*]
A[*er*] → A	$\vert A\vert \geqslant 4$	[*bhaarater*]→ [*bhaarat*]
A[*der*] → A	$\vert A\vert \geqslant 4$	[*shilpider*]→ [*shilpi*]
A[*ke*] → A	$\vert A\vert \geqslant 5$	[*deshbaashike*]→ [*deshbaashi*]
A[*Taar*] → A	$\vert A\vert \geqslant 5$	[*duniyaaTaar*]→ [*duniyaa*]
A[*Taay*] → A	$\vert A\vert \geqslant 5$	[*dwipTaai*]→ [*dwip*]
A[*taar*] → A	$\vert A\vert \geqslant 5$	[*deshadrohitaar*]→ [*deshadrohi*]
A[*taay*] → A	$\vert A\vert \geqslant 5$	[*maanoshikataay*]→ [*maanoshik*]
A[*shil*] → A	$\vert A\vert \geqslant 5$	[*sthitishil*]→ [*sthiti*]
A[*tuku*] → A	$\vert A\vert \geqslant 5$	*khaadyatuku*]→ [*khaadya*]
A[*debi*] → A	$\vert A\vert \geqslant 5$	[*karunadebi*]→ [*karuna*]
A[*baabu*] → A	$\vert A\vert \geqslant 5$	[*anupambaabu*]→ [*anupam*]
A[*bhaai*] → A	$\vert A\vert \geqslant 5$	[*munnabhaai*]→ [*munna*]
A[*bhaabe*]→ A	$\vert A\vert \geqslant 5$	[*byapakbhaabe*]→ [*byapak*]

The set of rules for stemming Bengali words and examples are shown in Tables 2, 3 and 4. Rule-based stemming is carried out by the following method:

1. Apply the rules as specified in Table 2.

2. Iteratively apply in sequence the rules from Table 3 until none of them can be applied any more.

3. Apply the rules in Table 4 to remove plural inflections.

Table 4: Rules for plural suffixes with transliterated Bengali examples.

Stemming rule	Cond.	Example
A[*gulo*] → A	$\vert A\vert \geqslant 5$	[*dingulo*]→ [*din*]
A[*guli*] → A	$\vert A\vert \geqslant 5$	[*chaakaaguli*]→ [*chaakaa*]
A[*gulote*]→ A	$\vert A\vert \geqslant 5$	[*boigulote*]→ [*boi*]
A[*gulite*] → A	$\vert A\vert \geqslant 5$	[*shahargulite*]→ [*shahar*]

Step 2 is executed iteratively because of composite suffixes in Bengali. For example, the Bengali word *shaharguliteo* (meaning *in the cities too*) comprises of the root word *shahar* and the suffix *guliteo* which in turn is built up of smaller viable Bengali suffixes *gulite* and *o*.

3.2 Blind Relevance Feedback and LM

Blind relevance feedback (or Pseudo-Relevance Feedback) builds on the assumption that the top-ranked documents provide useful information for query expansion (QE) or for query rewriting. Typically, additional terms are extracted from the top ranked documents and all query terms are reweighted.

For the experiments with the BM25 retrieval model, 20 feedback terms were extracted from the top ranked 10 documents. The blind relevance feedback approach employed follows the one described in [18] and [17].

LM assumes that given a relevant document, queries are generated by the explicit generation of important terms and unimportant terms. The important terms are supposed to be drawn at random from the document. The unimportant terms are supposed to be drawn at random from the full collection. So, in addition to the random variables D for the document and the T_i for the query terms, the model associates indicator random variables I_i with each query term denoting if a query term is important ($I_i = 1$) or unimportant ($I_i = 0$).

The document scoring equation in LM is

$$\begin{aligned} & P(D, T_1, \ldots, T_n) \\ = \quad & P(D) \prod_{i=1}^{n} ((1 - \lambda_i) P(T_i) + \lambda_i P(T_i|D)) \end{aligned} \tag{2}$$

where n is the number of query terms and λ_i is $P(I_i = 1)$, i.e. the probability that term T_i is an important term in the query. Similarily, $1 - \lambda_i$ is the probability of T_i being unimportant.

Dividing Equation 2 by $\prod_{i=1}^{n}(1 - \lambda_i)P(T_i)$ would not affect the ranking because λ_i and $P(T_i)$ have the same value for each document. In fact, any monotonic transformation of the document ranking function will produce the same ranking of the documents. Because of possible numeric underflow for multiplication of low probabilities, the logarithm of probabilities is used. Therefore, Equation 2 is re-written as

$$\begin{aligned} & P(D_j = d_j, T_1 = t_1, \ldots, T_n = t_n) \\ = \quad & \sum_{i=1}^{n} log(1 + \frac{\lambda_i tf(t_i, d) \sum_t df(t)}{(1 - \lambda_i) df(t_i) \sum_t tf(t, d)}) \end{aligned} \tag{3}$$

Also we can associate non uniform prior probabilities to document relevance (favoring longer documents) resulting in

$$\begin{aligned} & P(D_j = d_j, T_1 = t_1, \ldots, T_n = t_n) \\ = \quad & log(\sum_t tf(t, d)) + \sum_{i=1}^{n} log(1 + \frac{\lambda_i tf(t_i, d) \sum_t df(t)}{(1 - \lambda_i) df(t_i) \sum_t tf(t, d)}) \end{aligned} \tag{4}$$

The queries vectors are formed by considering term frequencies alone (no *idf* and no length normalization). This is referred to as *nnn* weighting scheme in the SMART system.[5]

For *nnn* query weights i.e. $q_k = tf$ and

$$d_k = log(1 + \frac{tf \times (sum\ of\ dfs)}{df \times document\ length} \times \frac{\lambda_k}{1 - \lambda_k}) \tag{5}$$

the dot product of the two vectors is identical to the RHS of Equation 3 and hence is an useful estimation of $P(D_j = d_j, T_1 = t_1, \ldots, T_n = t_n)$ acting as measure for similarity ranking. For a system with the vector data structure implemented in it, reweighting the document vectors and using simple term frequencies as weights of the query vectors gives the LM scores with the dot product operation on vectors.

In computing the dot product based similarity scores either of Equation 3 or 4 can be used. The documents were indexed by using Equation 3 and provision was made for adding the term $log(\sum_t tf(t, d))$ during the retrieval step. This process of adding up the extra term is subsequently referred to as *document length correction* in this paper.

Success of query expansion for the LM relies heavily on assigning appropriate λs for the new query terms. Since the λ_i values are indicative of the importance of the ith query term and since the expansion terms are not chosen directly by the user, an intuitive approach is to set these λ_i to a somewhat lower value and simultaneously giving the λ_i of the original query terms a boost.

We describe the LM approach used in our experiments with the FIRE 2010 document collections below (parameters for this approach are r, p, α, and β).

1. Select the top r documents from the results of the initial retrieval step as feedback documents which are presumed to be relevant. The other $N - r$ documents are assumed not to be relevant where N is the total number of documents to be retrieved.

2. Select p feedback terms from the r documents. Typically, the most frequently occuring terms are selected and added to the query.

3. Choose λ_i for the ith query term q_i as follows:

$$\lambda_i = \begin{cases} \beta & \text{if } q_i \text{ is a term in the original query} \\ \alpha & \text{otherwise} \end{cases} \tag{6}$$

   where $\alpha \le \lambda \le \beta$ (λ is the value assigned to all query terms during the baseline retrieval).

3.3 Query Translation

Several cross-lingual IR experiments were conducted to compare CLIR performance for automtatic and manual query translation. The Bengali and Hindi queries were manually translated from English by native speakers. The Google translate web service[6] has been used for automatic query translation from English to Hindi.

3.4 Processing and Indexing

The FIRE document collection for ad hoc IR contains newspaper articles on various topics including sports, politics, business, and local news. The articles are represented as structured XML documents in TREC format, using UTF-8 encoding.

FIRE topics resemble topics from other retrieval campaigns such as TREC in format and content. They comprise a brief phrase describing the information need (topic title, T), a longer description (topic description, D), and a part with information on how documents are to be assessed for relevance (topic narrative, N). Retrieval queries

[5] ftp://ftp.cs.cornell.edu/pub/smart/

[6] http://translate.google.com/

are typically generated from the title and description fields of topics (TD). For each language, fifty unique topics and the relevance assessments were provided together with the corresponding document collection. For all FIRE topics, relevant documents have been assessed by pooling submissions from systems participating in the FIRE retrieval track.

Not all documents could be indexed properly: some files include invalid XML characters or contain otherwise invalid XML; others contain no valid text at all. These documents have not been indexed at all, but they make up only a small portion of each collection.

The stopword lists used for the experiments described in this paper (stopword lists for English, Bengali, Hindi, and Marathi) originate from different sources. First, special characters (e.g. punctuation marks) were compiled in a list. For example, ‘|’ is used as a end-of-sentence marker in Bengali. A second list is created during indexing, containing the most frequent index terms. Terms occuring in more than half of all documents in the document collection are considered as stopwords.

The third source for stopwords is Jacques Savoy's web page on multilingual resources for IR at the University of Neuchâtel[7]. These stopword lists have been generated following an approach to obtain a general stopword lists for general text [4, 19], in which the N most frequent words are extracted from the document collection, numbers are removed from the list, and the resulting stopword list is manually extended with additional word forms. The resulting stopword lists contain 571 words for English (the SMART stopword list), 119 for Bengali, 163 for Hindi, and 98 for Marathi. For the LM experiments, the stopword list for Bengali provided on the FIRE web site was emplyed (384 stopwords).

Unnormalized text encoded with UTF-8 may use different multi-byte character encodings for the same character. For example, the character *é* in the Spanish name *San José* may be encoded as a single byte (for *é*), as the byte sequence for e + ´ or as the byte sequence for ´ + e. For the experiments described in this paper, encoded text was normalized by following the guidelines for canonical decomposition followed by canonical composition from the International Components for Unicode (ICU) implementing the standard normalization forms described in the Unicode Standard Annex #15 - Unicode Normalization Forms[8]. These normalization steps guarantee a fixed order of characters where multiple variants are allowed.

In addition, text was processed by applying the following normalization rules.

1. Internal word space is removed (e.g. characters 200c and 200d)
2. 'Chandra bindu' and 'anusvara' are mapped to 'anusvara'.
3. 'Chandra' followed by a vowel is mapped to the corresponding vowel.
4. 'Virama' is removed from the text.
5. Combinations of 'nukta' and a consonant are replaced by the corresponding consonant character.
6. Long vowels are mapped to the corresponding short form.
7. Some character sequences visually similar to a single glyph are mapped to a single character (e.g. letter A + sign O, letter A + sign AA + sign E, letter A + sign E + sign AA are mapped to the letter O).
8. Accents (which are typically part of transcribed foreign names) are removed.
9. Digit symbols in Bengali and Devanagari are mapped to Arabic numeric literals, because the FIRE data contains both forms.

These rules serve as a means to normalize orthographic variants.

4. RETRIEVAL EXPERIMENTS AND RESULTS

For the experiments described in this paper, the Lucene IR toolkit was employed.[9] Lucene does not (yet) include state-of-the-art IR models or blind relevance feedback. Support for the BM25 model [18, 17] and for the corresponding blind relevance feedback approach was implemented for Lucene by one of the authors. Runs with BM25 used 10 feedback documents and 20 feedback terms.

Additional experiments for the Bengali monolingual adhoc track were performed using LM implemented within the SMART system by one of the authors [5]. The LM runs used 35 feedback documents and terms. The LM experiments used different seetings for λ: $\lambda = 0.3$ for the baseline run and $\lambda = 0.25$ for the run including blind relevance feedback.

The following parameters were varied in our FIRE 2010 experiments:

- the source and target language for topics/queries (EN: English, MR: Marathi, BN: Bengali, HI: Hindi);
- the translation method for bilingual experiments (Nat: manual translation by native speaker, GT: the Google translate web service);
- the type of word processing before indexing (PS: Porter stemming, P5: 5-prefixes, CS: corpus-based stemming, RS: rule-based stemming, NP: no processing, raw word forms);
- the retrieval model (BM25: Okapi BM25, LM: Language modeling); and
- the use of blind relevance feedback (Y: yes, N: no).

The results shown in Table 5 include the number of relevant and retrieved documents (rel_ret), mean average precision (MAP), geometric MAP (GMAP), and precision at 10 and 20 documents (P@10 and P@20, respectively).

4.1 Results for Monolingual Experiments

Results for the monolingual IR experiments on the FIRE document collections are shown in Table 5 (italics indicate additional experiments). Both stemming approaches, the

[7] `http://members.unine.ch/jacques.savoy/clef/index.html`

[8] `http://www.unicode.org/unicode/reports/tr15/`

[9] `http://lucene.apache.org/`

Table 5: Results for monolingual and bilingual FIRE 2010 experiments.

Run	Parameters					Results				
ID	lang.	transl.	index	retrieval	BRF	rel_ret	MAP	GMAP	P@10	P@20
FE_S1	EN	-	PS	BM25	N	650	0.4647	0.3609	0.4320	0.3300
FE_S1TD_QE20	EN	-	PS	BM25	Y	652	**0.4846**	0.3542	0.4380	0.3550
FE_P5TD_QE20	EN	-	P5	BM25	Y	652	0.4765	0.3525	0.4420	0.3580
FM_B0	MR	-	NO	BM25	N	550	0.2357	0.0120	0.2720	0.2210
FM_P5TD_QE20	MR	-	P5	BM25	Y	614	**0.3411**	0.0230	0.3280	0.2840
FM_S2TD_QE20	MR	-	CS	BM25	Y	601	0.2910	0.0131	0.3020	0.2540
FB_B0	BN	-	NO	BM25	N	489	0.3858	0.2914	0.3300	0.2530
FB_P5TD_QE20	BN	-	P5	BM25	Y	502	0.4526	0.3717	0.3800	0.2670
FB_S2TD_QE20	BN	-	CS	BM25	Y	499	0.4000	0.3269	0.3420	0.2560
LM_baseline_.25	BN	-	RS	LM	N	500	*0.4860*	0.4160	0.3980	0.2930
LM_feedback_.25	BN	-	RS	LM	Y	502	**0.4944**	0.4274	0.3920	0.2910
LM_baseline_.3	BN	-	RS	LM	N	500	0.4902	0.4223	0.4080	0.2940
LM_feedback_.3	BN	-	RS	LM	Y	503	*0.4981*	0.4385	0.3940	0.2910
FH_B0	HI	-	NO	BM25	N	846	0.3896	0.2272	0.4520	0.3250
FH_P5TD_QE20	HI	-	P5	BM25	Y	895	0.4447	0.2957	0.4440	0.3650
FH_S2TD_QE20	HI	-	CS	BM25	Y	902	**0.4677**	0.3299	0.4800	0.3950
FBmt_P5TD_QE20	EN→BN	Nat	P5	BM25	Y	451	0.3700	0.2818	0.3413	0.2478
FBmt_S2TD_QE20	EN→BN	Nat	CS	BM25	Y	447	**0.3771**	0.2785	0.3130	0.2348
FHan_P5TD_QE20	EN→HI	Nat	P5	BM25	Y	725	**0.3771**	0.2516	0.4480	0.3610
FHan_S2TD_QE20	EN→HI	Nat	CS	BM25	Y	724	0.3747	0.2442	0.4440	0.3510
FHgt_P5TD_QE20	EN→HI	GT	P5	BM25	Y	724	0.3647	0.2402	0.4360	0.3470
FHgt_S2TD_QE20	EN→HI	GT	CS	BM25	Y	722	0.3684	0.2361	0.4320	0.3450

corpus-based stemmer and the rule-based stemmer, significantly outperform the baseline for the corresponding language. However, for Marathi retrieval experiments, indexing 5-prefixes yields the highest MAP. Query expansion additionally increases IR performance in these cases. For the run LM_feedback_.25 we used $(\alpha, \lambda, \beta) = (0.05, 0.25, 0.3)$ whereas for the other one we used $(\alpha, \lambda, \beta) = (0.05, 0.3, 0.35)$. Using query expansion in pseudo-relevance feedback on LM baseline retrieval, with the particular choice of assignment of the λs proved useful.

4.2 Results for Bilingual Experiments

Two sets of bilingual IR experiments have been performed (English→Bengali and English→Hindi). Experiments using manual translation of FIRE topics for Bengali→English achieved 81.7%-94.3% of the MAP for the corresponding monolingual experiments. Manual query translation for English→Hindi shows 87.0%-92.9% of the MAP for the corresponding monolingual experiments. In comparison, query translation by the Google translate web service shows a slightly (but not significantly) lower MAP and achieves 85.6%-89.8% of the MAP for the best monolingual Hindi run.

5. CONCLUSIONS AND OUTLOOK

This paper described experiments at FIRE 2010, evaluating stemming, blind relevance feedback and query translation.

In combination with blind relevance feedback, both stemming approaches performed significantly better than the baseline of indexing words in all tested languages (English, Bengali, Hindi, and Marathi). An obvious improvement of the corpus-based stemming approach will be to determine the cut-off point for the suffixes dynamically. In all experiments and for all languages, a fixed set of 50 suffixes was used. For morphologically rich languages such as Bengali (with many different and composite morphological suffixes), this number is probably too low. The rule-based stemmer needs some further improvement because the inflections in a language like Bengali can not only be suffixial but also prefixial. Moreover two Bengali words often get merged together to form a compound word making it a challenging task to obtain compound constituents for indexing. We would like to explore on these language-specific issues in the coming years of participation at FIRE. Additional ideas for future research include indexing at the phrase level for common phrases, extend the stopword list with common phrases, expanding queries with hypernyms derived from thesaural information and devising appropriate λs for them, and combining the EM feedback with query expansion on baseline LM retrieval, would be experimented upon.

6. ACKNOWLEDGMENTS

This material is based upon works supported by the Science Foundation Ireland under Grant No. 07/CE/I1142. as part of the Centre for Next Generation Localisation (CNGL) project. Special thanks to Ankit Srivastava and Sudip Naskar for translating the queries.

7. REFERENCES

[1] Sajib Dasgupta and Vincent Ng. Unsupervised morphological parsing of Bengali. *Language Resources & Evaluation*, 40:311–330, 2006.

[2] Sajib Dasgupta and Vincent Ng. High-performance, language-independent morphological segmentation. In Candace L. Sidner, Tanja Schultz, Matthew Stone, and ChengXiang Zhai, editors, *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, (NAACL HLT 2007), April 22–27, 2007*, pages 155–163, Rochester, NY, USA, 2007. ACL.

[3] Ljiljana Dolamic and Jacques Savoy. UniNE at FIRE 2008: Hindi, Bengali, and Marathi IR. In *Working Notes of the Forum for Information Retrieval Evaluation, December 12–14, 2008*, Kolkata, India, 2008.

[4] Christopher Fox. *Lexical analysis and stoplists*, pages 102–130. Prentice-Hall, NJ, USA, 1992.

[5] Debasis Ganguly. Implementing a language modeling framework for information retrieval. Master's thesis, Indian Statistical Institute, India, 2008.

[6] John Goldsmith. Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, 27:153–198, 2001.

[7] Donna Harman. How effective is suffixing? *Journal of the American Society for Information Science*, 42(1):7–15, 1991.

[8] Samarth Keshava and Emily Pitler. A simpler, intuitive approach to morpheme induction. In *PASCAL Challenge Workshop on Unsupervised Segmentation of Words Into Morphemes - MorphoChallenge 2005, April 12, 2006*, Venice, Italy, 2006.

[9] Robert Krovetz. Viewing morphology as an inference process. In Robert Korfhage, Edie Rasmussen, and Peter Willett, editors, *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 191–202, Pittsburg, USA, 1993. ACM.

[10] Leah S. Larkey, Margaret E. Connell, and Nasreen Abduljaleel. Hindi CLIR in thirty days. *ACM Transactions on Asian Language Information Processing*, 2(2):130–142, 2003.

[11] Julie Beth Lovins. Development of a stemming algorithm. *Mechanical translation and computation*, 11(1-2):22–31, 1968.

[12] Prasenjit Majumder, Mandar Mitra, Swapan K. Parui, Gobinda Kole, Pabitra Mitra, and Kalyankumar Datta. YASS: Yet another suffix stripper. *ACM transactions on information systems (TOIS)*, 25(4):18:1–20, 2007.

[13] Paul McNamee. N-gram tokenization for Indian language text retrieval. In *Working Notes of the Forum for Information Retrieval Evaluation, December 12–14, 2008*, Kolkata, India, 2008.

[14] Paul McNamee, Charles Nicholas, and James Mayfield. Addressing morphological variation in alphabetic languages. In James Allan, Javed A. Aslam, Mark Sanderson, ChengXiang Zhai, and Justin Zobel, editors, *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009, July 19–23, 2009*, pages 75–82, Boston, MA, USA, 2009. ACM.

[15] Douglas W. Oard, Gina-Anne Levow, and Clara I. Cabezas. CLEF experiments at Maryland: Statistical stemming and backoff translation. In Carol Peters, editor, *Cross-Language Information Retrieval and Evaluation, Workshop of Cross-Language Evaluation Forum, CLEF 2000, Lisbon, Portugal, September 21–22, 2000, Revised Papers*, volume 2069 of *Lecture Notes in Computer Science (LNCS)*, Berlin, 2001. Springer.

[16] Martin F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[17] Stephen E. Robertson, Steve Walker, and Micheline Beaulieu. Okapi at TREC-7: Automatic ad hoc, filtering, VLC and interactive track. In Donna K. Harman, editor, *The Seventh Text REtrieval Conference (TREC-7)*, NIST Special Publication 500-242, pages 253–264, Gaithersburg, MD, USA, 1998. National Institute of Standards and Technology (NIST).

[18] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline M. Hancock-Beaulieu, and Mike Gatford. Okapi at TREC-3. In Donna K. Harman, editor, *Overview of the Third Text Retrieval Conference (TREC-3)*, pages 109–126, Gaithersburg, MD, USA, 1995. National Institute of Standards and Technology (NIST).

[19] Jacques Savoy. A stemming procedure and stopword list for general French corpora. *Journal of the American Society for Information Science*, 50(10):944–952, 1999.

[20] Jacques Savoy. Light stemming approaches for the French, Portuguese, German and Hungarian languages. In Hisham Haddad, editor, *Proceedings of the 2006 ACM Symposium on Applied Computing (SAC), Dijon, France, April 23–27, 2006*, pages 1031–1035. ACM, 2006.

[21] Jinxi Xu and Bruce Croft. Corpus-based stemming using co-occurence of word variants. *ACM transactions on information systems*, 16(1):61–81, 1998.

[22] Tan Xu and Douglas W. Oard. FIRE-2008 at Maryland: English-Hindi CLIR. In *Working Notes of the Forum for Information Retrieval Evaluation, December 12–14, 2008*, Kolkata, India, 2008.