Improving Document Ranking using Query Expansion and Classification Techniques for Mixed Script Information Retrieval

Subham Kumar, Anwesh Sinha Ray, Sabyasachi Kamila, Asif Ekbal, Sriparna Saha, Pushpak Bhattacharyya
Department of Computer Science and Engineering
Indian Institute of Technology Patna
Bihta, Patna-801103 (Bihar)
{shubham.ee12,anwesh.cs12,sabyasachi.pcs16,asif,sriparna,pb}@iitp.ac.in

Abstract

A large amount of user-generated transliterated contents in Roman scripts are available in the Web for the languages that use non-Roman based indigenous scripts. This creates a mixed script space which is mono-lingual or multilingual having more than one script. Information retrieval (IR) in the mixed-script space is challenging as both query and documents can be written in either native or Roman script, or may be in both the scripts. Moreover, due to lack of any standard ways of spelling a word in a non-native script, transliterated contents can be written with different spelling variations. In this paper, we propose the effective techniques for query expansion and query classification for mixed-script IR. The proposed techniques are based on deep learning, word embedding and traditional TF-IDF. We generate our own resources for creating the test-bed for our experiments. Extensive empirical analyses show that our proposed methods achieve significantly better performance (20.44% increase in MRR, 22.43% increase in NDCG@1 & 15.61% increase in MAP) over a state-of-the-art baseline model.

1 Introduction

Information Retrieval (Manning et al., 2008) refers to finding material (usually, in the form of documents) of a semi-structured nature (usually, in text) that satisfies an information need within a large document collections. Nowadays, due to various socio-cultural and technological reasons, often the websites and the user-generated contents in languages such as Arabic, Russian, Indic etc., are written using Roman scripts. Such contents create a mono-lingual or multi-lingual space with more than one script which we refer to as the Mixed-Script space. Information retrieval in the mixed-script space, known as Mixed-Script IR (MSIR), is more challenging because queries can be written in both the native as well as Roman scripts, and these should also be matched to the documents written in both the scripts.

Transliteration (Lopez, 2008) is the process of phonetically describing the words of a given language using a non-native script. For both the web documents and intended search queries to retrieve those documents, transliteration, especially into Roman script, is generally used. Since no standard ways of spelling any word into a non-native script exist, transliterated contents offer extensive spelling variations; typically, we can transliterate a native term into Roman script in many ways (Gupta et al., 2012). For example, the word *khusboo* ("fragrance") can be written in Roman script using different variations such as *kushboo*, *khusbu*, *khushbu* and so on. This type of problem is termed as a non-trivial term matching problem for search engines with the aim to match the native-script or Roman-transliterated query with the documents in multiple scripts after considering the spelling variations. Many single (native) script queries and documents with spelling variations have been studied (French et al., 1997; Zobel and Dart, 1996) as well as transliteration of named entities (NE) in IR (Collier et al., 1998; Wang et al., 2009).

It is important for every IR engine to present users with information that are most relevant to the users' needs. While searching, user has an idea of what s/he wants, but in many instances, due to the variations in query formulations, retrieved results could greatly vary. As a result, understanding the nature of information need behind the queries issued by Web users has become an im-

D S Sharma, R Sangal and A K Singh. Proc. of the 13th Intl. Conference on Natural Language Processing, pages 81–89,
Varanasi, India. December 2016.

portant research problem. Classifying queries into predefined target categories, also known as query classification, is important to improve search relevance. Successfully classifying incoming general user queries to topical categories can bring improvements in both the efficiency and the effectiveness of general web search. Query classification encounters two important challenges: (i) many queries are very short and contain noise. (ii) a query can often have multiple meanings. In our work we propose a query classification technique for improving document ranking in mixed-script IR scenario. The query can be in multiple scripts, and the dataset is of mixed-domain [1]. At first we develop a baseline model for query expansion in line with (Gupta et al., 2014) using a deep learning architecture. We develop the query classification technique based on distributed word representation and traditional TF-IDF weighting scheme. The proposed method is proven to be effective in improving the ranking of relevant documents.We present an extensive empirical analysis of the proposed technique that achieves significantly better performance (20.44% increase in MRR, 22.43% increase in NDCG@1 & 15.61% increase in MAP) over the baseline model. We summarize the contributions of our current work as follows:
(i) proposal of an effective query classification (traditional tf-idf along with word-embedding) technique that improves the performance in a mixed-script IR scenario.
(ii) development of a model of query expansion based on deep learning architecture.
(iii) creation of resources for mixed-script IR.

The remaining of the paper is organized as follows. In Section 2, we introduce the concept of MSIR in correspondence to our task and show the possible application scenarios and challenges. Section 3 presents different approaches that we developed to improve the document ranking in an ad-hoc retrieval setting of mixed-script IR over the baseline. In Section 4, we present the details of resources that we created for our experiments. Section 5 reports the experimental setup and the results of evaluation with some empirical analysis. Finally, we conclude in Section 6.

[1]Refers to the dataset that contains texts written in more than one domain

2 MSIR: Definition and Challenges

In this section, we present the definition of mixed-script information retrieval (MSIR) with respect to our work.

2.1 Mixed Script IR

The task of IR engine is to rank the documents retrieved from a document pool D, such that documents which are relevant to query q appear at the top of the ranked list. For a *Mixed Script IR* setup, the queries and the documents are all in the same language, i.e., Hindi in our case, but those are written in more than one different scripts, i.e., Roman and Devanagari scripts in our case. The task of IR engine is to search across the scripts.
More specifically, we define our problem statement as: Given a query in Roman or Devanagari script, the system has to retrieve the top-k documents from a corpus that contain mixed scripts (i.e., either Roman or Devanagari or both).
Input: a query written in Roman (transliterated) or Devanagari script
Output: a ranked list of ten (k=10 here) documents both in Devanagari and Roman scripts generated from some given document collections.
As part of our contribution to FIRE 2015 shared task (Sequiera et al., 2015), we had access to the queries which cover three different genres, i.e., Hindi songs lyrics, movie reviews, and astrology. In line with this we prepare 25 queries as a test collection for various information needs. Queries related to lyrical documents express the need to retrieve relevant song lyrics while queries related to movie reviews and astrology are informational in nature (Sequiera et al., 2015).

2.2 Challenges in MSIR

There are mainly two challenges exist in mixed-script information retrieval (MSIR) which are: (a) problems in handling extensive spelling variations in transliterated queries and documents during term matching phase? and (b) problems in identifying, representing and processing a mixed type query, mixed and transliterated documents? The solution for the first problem could be the following: the given text can be transliterated and converted into a common space and finally some standard matching techniques in single-script space can be applied. Otherwise as shown in (Gupta et al., 2014) an abstract orthographic space can be created to represent the words of different scripts.

Here the words can be matched to obtain variants of a query word across scripts.

In mixed query processing, language identification of query words adds another interesting challenge. Query words should be assigned appropriate semantic classes, which denote either native or transliterated scripts. This is challenging as depending upon the context of the query a same word may have different meanings. For example a word *man* can be a representative of the English word *man*, or the transliterated Hindi word of *man* having meaning *mind*, or another transliterated Hindi word *maan* meaning *reputation*. Hence, language identification seems to be an extremely important problem in MSIR setting, especially when multiple languages are involved. Our algorithm for language identification is based on supervised machine learning, which is in line with the system as described in (Gupta et al., 2015).

There are some other problems apart from these basic challenges like how to present information etc. This requires some additional information to be known in advance like whether the user can read all the scripts, or prefer some scripts over the other.

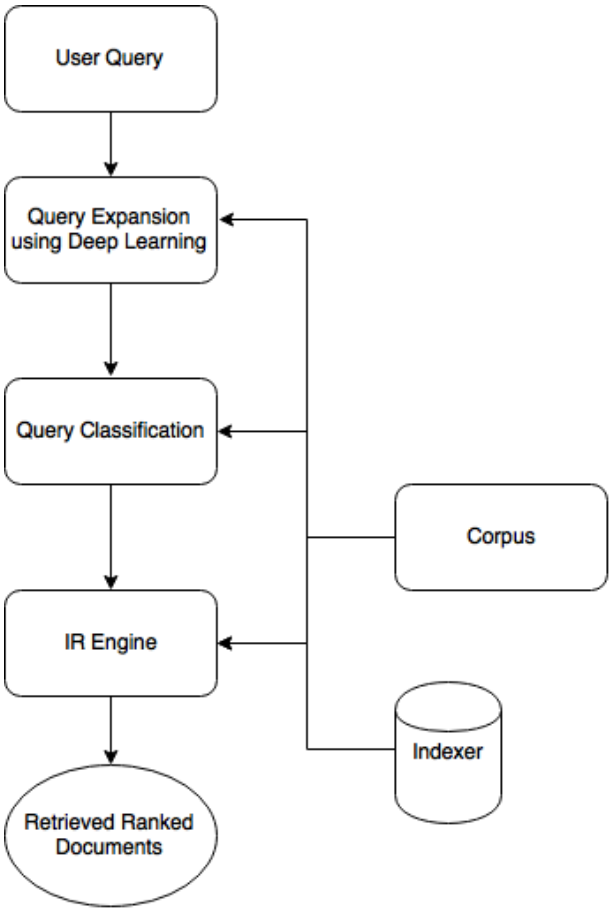

Figure 1: Block Diagram of the system

3 Methodology

Having defined the basic MSIR setting, and establishing its prevalence in Web search through availability of several mixed-script contents in the web, i.e., Indian Astrology, Bollywood movie reviews, Hindi song lyrics etc., we now present the approaches that we develop along with the baseline system. An overall architecture of the system showing the basic components is depicted in Figure 1.

3.1 Baseline System

We develop a baseline model based on deep-learning architecture in line with the prior work as proposed in (Gupta et al., 2014). This is based on *restricted Boltzmann machine*(RBMs) (Hinton, 2010) model. Our baseline, though uses the concept proposed in (Gupta et al., 2014), is developed to handle a bigger dataset comprising of multiple domains instead of only Hindi song lyrics dataset of FIRE 2014, thus extending the research challenges in a MSIR setup. The baseline proposes a principled solution to handle the mixed-script term matching and spelling variation. The mixed-script features are modeled jointly in a deep-learning architecture in such a way that these can be compared in a low-dimensional abstract space where term equivalents are close to each other. The principle behind the formation of common abstract space for mixed-script modeling is that original words and the corresponding transliterated words into a non-native script share the same sound/pronunciation. This process can help in determining possible equivalents of the query term across the script. Finally the expansion of the original query can be done using the thus found equivalents.

3.1.1 Formulation

The phonemes are considered as character-level *topics* in the terms. Terms are the representatives of mixtures of *topics*, with each topic representing a probability distribution over character n-grams. Let us assume that the feature set F = $\{f_1, ..., f_K\}$ contain character grams of scripts s_i for all i $\in$ $\{1, ., r\}$ and |F| = K.

$$t_1 = \bigcup_{i=1....r} w_{1,i}$$

represents a data-point from training data T of language l_1 where $w_{1,i}$ denotes word w written in language l_1 and script s_i. Here r counts the number of jointly modeled scripts. Each data-point is assumed to be a K-dimensional feature vector x where x_k is a measurement of k^{th} feature $f_k \in$ F in data-point t_1. The count for character grams within terms can be measured using Dirichlet multi-nomial distribution[2]. We have considered total N independent draws from a categorical distribution with K different output classes. In the current context, the value of N can be computed as follows:
N = $\sum_{i=1}^{k} x_i$ and $\{f_1, \ldots, f_K\}$ denote K categories. Here x_k indicates the number of times a particular feature f_k appears in the data-point t_1. Here x = $\{x_1, \ldots, x_K\}$ satisfies a multinomial distribution with two parameters N and p. Here p = (p_1, . . . , p_K) and p_k denotes the probability that k^{th} feature obtains the value of x_k. The parameter p can not be directly computed in our case rather it is computed using a conjugate prior distribution.
In the current paper we have proposed a model based on non-linear dimensionality reduction procedures like deep auto-encoder. A deep learning based architecture is proposed after stacking several RBMs together. The bottom-most RBM in our framework is used for modeling the input terms. This can be viewed as a character-level variant of the replicated softmax (RSM) model presented in for documents. In general character n-grams follow Dirichlet multi-nomial distribution. But in the current work RSM has been used to model these as at the time of applying the inference, methods like Gibbs sampling, Dirichlet prior distributions are often marginalized out. We assume that v $\in \{0, 1, , N\}^{\text{K}}$ denotes the visible multinomial units and h $\in \{0, 1\}^{\text{m}}$ denotes a stochastic binary hidden latent unit. Also assume that v is a K-dimensional input vector such as feature vector x for data-point t_1. h denotes the m-dimensional latent feature vector and N = $\sum_{i=1}^{k} x_i$. The energy of the state $\{v, h\}$ is calculated as follows:

$$E(v,h) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_i \sum_j v_i w_{i,j} h_j$$

[2]http://en.wikipedia.org/wiki/Dirichlet-multinomial_distribution

Here the following terms are used: v_i denotes the measurement of x_i, $w_{i,j}$ is the weight matrix entry between i^{th} visible node and j^{th} hidden node, while a_i and b_j are bias terms of visible and hidden layers, respectively. A large number of layers are inserted to develop a deep auto-encoder. A cascaded structure of binary RBMs is created in such a way that output of a lower RBM is passed as an input to the immediate next RBM.

3.1.2 Training

In our proposed model, the visible layer of the bottom-most RBM is the character-level replicated soft-max layer. The feature space F is a combination of character uni- and bi-grams of training data (r = 2). The terms are represented as low dimensional embedding in the abstract space. The hidden layer of the top-most RBM is kept linear for this purpose. A two-phased training procedure is carried out for the developed auto-encoder i) pre-training the layers in a greedy manner and; ii) updation of the weights using backpropagation process.

3.1.3 Finding Equivalents

After the training of the model is finished, the process of equivalent- discovery has started. This is composed of two steps: i) the index of mining lexicon in abstract space is produced and; ii) the equivalents for the query term are generated. The mining-lexicon is a lexicon of reference collection (ideally, mixed-script) . This is later on utilized to extract term equivalents. This is executed after projecting each term in the mining lexicon into an abstract space. For each term, one index entry is stored which contains the corresponding projected term. Finally the query term x_q is also projected into the abstract space h_q and the corresponding cosine similarity with respect to all terms in the index is calculated. For a given query word, x_q, the terms with cosine similarity $> \theta$ are considered as equivalents. Here θ is used as the similarity threshold.
For developing the baseline system, similarity thresholds of 0.95 and 0.98 are considered.

3.2 Query Classification

Query classification helps identifying intents behind each query, which, in turn, provides better retrieval search results.We divide our dataset into 3 broad domains, namely *Music Lyrics*, *Movie Reviews* and *Astrology*.

3.2.1 TF-IDF based Technique

Term frequency (TF)-inverse document frequency (IDF) is a technique to show the importance of a word in a document given a large collection of documents or corpus. TF-IDF technique is widely used in information retrieval or text mining as a weighting factor. In general if a word appears multiple times in a document then TF value would be more. But this is offset by the frequency of the word appearing in the other documents of the corpus. This way the TF-IDF scores of words which appear frequently in general can be reduced.

We create the minion lexicon for each domain through indexing along with the normalized TF-IDF scores of each term in that respective domain.

For each domain d, we obtain normalized term frequency of the term t as follows:

$$TF(t,d) = \frac{tf(t,d)}{\max_t tf(t,d)}$$

Here, $tf(t,d)$: term frequency of the tth term in document d; $max_t tf(t,d)$: frequency of the most frequent term.

For each domain d, we obtain normalized inverse document frequency of the term t

$$IDF(t,d) = \frac{idf(t,d)}{\max_t idf(t,d)}$$

Here, $idf(t,d)$: inverse document frequency of term t in document d; $max_t idf(t,d)$: maximum inverse document frequency obtained throughout the corpus.

The corresponding normalized TF-IDF score of a term t is obtained by

$$TF - IDF(t,d) = TF(t,d).IDF(t,d)$$

After obtaining the above TF-IDF score for each minion lexicon of each domain, the TF-IDF score of a query for a domain is obtained by summing up the TF-IDF score of every expanded query term obtained from that domain. If a expanded query term is not found in that domain, its TF-IDF score is set to 0. Once the score for a query is obtained for each domain, the query is classified to the domain for which it has the highest score.

3.2.2 Word Embedding based Technique

We use word2vec tool (Mikolov et al., 2013) which efficiently captures the semantic properties of words in the corpus. Word2vec is a group of related models (i.e., CBOW: continuous bag-of-words model and Skip-grams) which can be utilized to generate different word embeddings. The linguistic contexts of words can be generated using some models based on shallow-ed, two-layered neural networks. A word is given as an input to the network and the task is to predict the words which can appear in adjacent positions given an input text. Order of the remaining words is not important.

For each domain d, we merge the contents of all the documents belonging to that domain to obtain its respective document collection. We train Word2Vec using CBOW model for each domain dataset by setting window size to 5, and dimension of the final vectors to 100. For each domain, we create a normalized vector *norm_vec(d)* by summing up all the vectors of all the tokens t of that domain, and dividing resultant vector by the total number of documents *N(d)* in that domain. For the expanded query vectorization, we obtain the vectors *vec(t,d)* of each query term t from the trained model of the domains in which that term is present, sum up the vectors, and divide the final vector by the number of domains in which it has occurred (denoted by *N(t,d)*: number of domains d in which the term appears). The final vector of a query Q is obtained by summing up the individual word vectors of each query term. The processes are described mathematically as follows: $N(t,d)$ = No. of domains where the term t has occurred;
$N(d)$ = No. of documents in the domain d;
$vec(t,d)$= Vector of term t in domain d;
$norm\_vec(d)$ = Normalized vector for domain d;
$norm\_vec(Q)$ = Query vector of a query Q;

$$norm\_vec(d) = \frac{\sum_{t \in d} vec(t,d)}{N(d)}$$

$$norm\_vec(Q) = \sum_{t \in Q} \frac{\sum_d vec(t,d)}{N(t,d)}$$

For classifying the queries into one of the 3 domains, we use two similarity measures (i.e., cosine similarity and Euclidean distance) between the query vector *norm_vec(Q)* and the domain vector *norm_vec(d)*.

4 Dataset

4.1 Resource Creation

We create our own resources for experiments. We develop a web crawler using Python modules - *BeautifulSoup*, *mechanize* to crawl the documents from different web sources. For data creation we had to deal with the following issues: (i) very few web-sites having good amount of transliterated contents of Devanagari documents; (ii) no standard html tag containing actual contents of documents, and so manual intervention was necessary. We crawl total 1,503 documents covering all the 3 domains- *Astrology*, *Bollywood Movie Reviews* and *Bollywood Movie Dialogues*. Following sources were used: Bollywood movie reviews: http://www.jagran.com/, Astrology: http://astrology.raftaar.in/ and Bollywood movie dialogues: http://filmyquotes.com/.
We contributed to the FIRE-2015 Shared Task on MSIR in Subtask 2 problem[3], which is related to the mixed-script Ad-hoc retrieval. In Table 1, we present some statistics for the datasets.

Domain	No. of documents	Script
Astrology	343	Devanagari
Movie Reviews	97	Devanagari
Movie Dialogues	1,063	Mixed

Table 1: Statistics of Crawled Datasets

4.2 Pre-processing

The dataset has to be pre-processed for removing inconsistencies and other typical errors. Some of the examples are shown in Table 2. The errors were manually corrected. Some typical examples are shown in Table 2.

Errors	Example	Correct Word
Unicode Characters		-
Punctuation Marks	hai.n, maine;	hain, maine
XML Tags	⟨text⟩	-
Hash Tags	###	-
Underscore	ko_ii	koii
Website Name	MusicMaza.Com	-
Backslash	be\-sadaa	be sadaa

Table 2: Typical Examples of the Dataset

[3]http://fire.irsi.res.in/fire/2015/home

4.3 Dataset Statistics

We use the FIRE 2013 shared task collection on Transliterated Search (Roy et al., 2007) for training the deep auto-encoder (stacked RBMs). The dataset comprises of document collection (D), queryset (Q) and relevance judgments. The collection contains 63,334 documents containing song titles and lyrics in Roman, Devanagari and mixed scripts; Hindu astrology and Bollywood movie reviews in Devanagari. Statistics are shown in Table 3. The query set (Q) contains 25 lyrics search queries covering Bollywood songs; Movie reviews and Hindu astrology in Roman and Devanagari scripts with mean query length of 4.04 words. Queries related to lyrics documents express the need to retrieve relevant song lyrics, while queries related to movie reviews and astrology are informational in nature. The queries are of different difficulty levels: word level joining and splitting, ambiguous short queries, different script queries and mixing of different language keywords. On an average, there are 47.52 qrels per query with average relevant documents per query is set to 5.00. Some example queries are shown in Table 4.

No. of documents	63,334
No. of tokens	1,40,39,002
No. of vocabulary words	1,45,478

Table 3: Corpus Statistics

Sample Queries
tujho nahi lyrics
vicky donor movie reviews
ill effects of rahu kaal

Table 4: Example of Queries

5 Experiments and Results

In this section we describe the experimental results along with necessary analysis for evaluating the effectiveness of the proposed method for retrieval in *mixed-script space*.

5.1 Baseline System

The experimental setup for baseline model is based on a standard ad-hoc retrieval setting. The document collection of three domains is first indexed to create an inverted index. We set the following experimental setups for each domain:

(i) for lyrical retrieval, the sequential information among the terms is crucial for effectiveness evaluation (Gupta et al., 2014), e.g., *love me baby* and *baby love me* are completely different songs. In order to capture the word-ordering we consider word bi-grams as a unit for indexing and retrieval; and (ii) for bollywood movie reviews and astrology, we consider word uni-gram as a unit for indexing and retrieval[4] Every query is enriched with equivalents of the query terms using the deep-learning based query expansion technique (c.f. Section 3.1.3).Suppose a query has n terms in it, and has m equivalents, then the query gets expanded into m^n queries. After the queries are expanded, they are first pre-processed, i.e., converted to word bi-grams for song lyrics and word uni-grams for astrology and movie reviews, and then fed to the retrieval engine to get the top 10 relevant documents.

5.2 Results using Query Classification

Here we present the results using different query classification techniques.

- **Query Classification using TF-IDF**: After expanding the queries with term equivalents (0.98 as the similarity threshold), the document collection is divided into 3 pre-defined domains, and are indexed separately. We consider the same indexing schemes as that of the baseline model (i.e., word bi-gram for song lyrics and word uni-gram for astrology and movie reviews).Every query is classified into one of the 3 domains in which the corresponding TF-IDF score is the highest, and the documents are retrieved from their respective classified domain.

- **Rule-based Query Classification using TF-IDF and Word Embedding**: We compute word embeddings using the process as described in Section 3.2.2. Once the similarity scores of each query for all the three domains are obtained using the word embedding technique as described in Section 3, we propose a rule-based system to classify the queries. We first classify using the traditional TF-IDF based approach. If the score is below a certain threshold level (in this case, it is 0.85), we use the word embedding based approach for query classification. Vector representation of each word in the query is obtained by following the process as mentioned in Section 3.2.2. For Euclidean distance based classification, we classify the queries to the domain for which the distance is minimum and for cosine similarity based classification, we classify the queries to the domain for which the score is the highest.

5.3 Results and Analysis

We evaluate the effectiveness of the proposed methods, referred to as **Deep-TF-IDF** and **Deep-Rule-based** and compare with the baseline system which we refer to as **Deep-baseline**. For a fair comparison between the baseline and our proposed methods, we develop the *baseline from scratch*, and evaluate using the cleaned dataset keeping the parameter θ to be 0.98. The retrieval performance of the proposed systems are computed using three evaluation measures, mean average precision (MAP), mean reciprocal rank (MRR) and normalized discounted cumulative gain (NDCG@1). The optimal value θ is set by conducting experiments for a range of θ values starting from 0.90 to 0.99 with a step size of 0.01 and finding the value of θ which yields the highest MAP, MRR and NDCG@1 values. For a particular query, the ranked-list which is a collection of top 10 documents is first created. The ranking model used in the current work is a parameter free divergence from randomness (unsupervised DFR) as described in (Amati, 2006). This model is suitable for short-queries. The obtained average results over Q measured in terms of MAP, MRR and NDCG@1 are illustrated in Table 5. We choose θ for **Deep-baseline** following the process as described in (Gupta et al., 2014). The proposed Deep-Rule-based technique attains the high MRR score. This shows its utility in fetching the first relevant document at very high ranks which is required in case of web search. Moreover Deep-Rule-based technique also attains the highest value of MAP compared to all other techniques. High NDCG@1 describes its ability to fetch useful and the most relevant documents at the first position.

Results of query classification using TF-IDF technique are presented in Table 6. This shows the TF-IDF scores of each query for each domain.

The actual music queries are from 51 to 60, actual movie queries are from 61 to 67 and astrology queries are from 68 to 75. From the above experiments, we come to the conclusion that out of

[4]Since the queries are informational in nature.

Method	NDCG@1	MRR	MAP	θ
Deep	0.6633	0.5613	0.3338	0.95
Deep	0.7133	0.6144	0.3587	0.98
Deep-TF-IDF	**0.7633**	**0.6600**	**0.3347**	0.98
Deep-Rule-Based	**0.8733**	**0.7400**	**0.4147**	0.98

Table 5: Results of Retrieval Performance Measured by MAP, MRR & NDCG@1

Query No.	Music	Movie	Astrology
51	158.93	0.0	4.21
52	461.02	3.97	0.0
53	33.53	0.0	0.0
54	473.83	0.0	846.14
55	389.19	0.0	175.04
56	306.56	31.14	21.60
57	101.15	61.44	74.01
58	525.55	81.94	273.98
59	981.11	84.30	138.07
60	188.33	3.97	4.21
61	1.58	33.73	0.0
62	13.52	7.95	21.33
63	2.88	0.0	4.21
64	0.30	3.97	0.0
65	1.37	29.75	0.0
66	1.29	25.77	0.0
67	12.71	23.85	14.85
68	34.53	3.97	366.12
69	146.84	25.06	1054.97
70	33.07	12.79	125.16
71	6.11	0.0	140.84
72	208.50	15.02	209.04
73	4.41	13.45	645.70
74	4.11	6.72	149.76
75	249.27	10.09	343.47

Table 6: TF-IDF Score of Queries(X10^{-4})

25 queries, there are three queries (i.e., 54,62,63) that are incorrectly classified into astrology domain.Results of query classification using word embedding technique are presented in Table 7 and Table 8 for music queries, and movie and astrology queries, respectively. We use Euclidean distance to classify music queries as this metric gives weight-age to long documents. In our case, out of total 63,334 documents, Hindi song lyrics domain has 62,894 documents. For movie and astrology queries, we use cosine similarity which penalizes long documents by normalizing the vector length.

From the above results, it is evident that the

Query No.	Music	Movie	Astrology
51	267.89	1393.75	540.52
52	414.86	1412.81	593.60
53	272.40	1398.45	542.50
54	597.42	1390.06	702.43
55	398.54	1410.33	665.43
56	260.86	1397.78	542.05
57	279.56	1394.62	559.91
58	2491.55	2771.60	2556.23
59	4112.95	4249.52	4245.69
60	331.28	1399.07	613.57

Table 7: Euclidean Distance Measures between Queries and Domains: Music, Movie and Astrology

Query No.	Music	Movie	Astrology
61	0.22	0.30	0.27
62	0.07	0.19	0.12
63	0.11	0.135	0.132
64	0.10	-0.17	-0.18
65	0.27	0.13	0.03
66	0.18	0.25	0.09
67	0.14	0.34	0.21
68	0.08	0.23	0.44
69	0.35	0.24	0.36
70	0.171	0.178	0.24
71	0.09	0.11	0.06
72	0.15	0.28	0.31
73	0.15	0.17	0.33
74	0.07	0.30	0.29
75	0.16	0.44	0.50

Table 8: Cosine Similarity between Queries Domains: Music, Movie and Astrology

queries which are incorrectly classified by simple TF-IDF based technique (i.e., 54,62,63) are now correctly classified using word embedding based technique.

6 Conclusion

Although Mixed-Script IR (MSIR) is a very important and prevalent problem, it has attained very little attention. In this work we have shown how query classification aids in document retrieval. Our baseline is developed using a deep learning architecture. We develop query classification using TF-IDF and word embedding based models that improve the document ranking of the baseline. We have presented extensive empirical analy-

ses of the proposed techniques for ad-hoc retrieval settings of Hindi songs lyrics, Astrology and Bollywood Movie reviews, where the proposed methods achieve significantly better results (20.44% increase in MRR, 22.43% increase in NDCG@1 & 15.61% increase in MAP) over the baseline.
In future we would like to investigate a more general setup of MSIR such as Mixed-Script Multilingual IR (MS-MLIR).

References

Giambattista Amati. 2006. Frequentist and bayesian approach to information retrieval. In *Proceedings of the 28th European Conference on Advances in Information Retrieval*, ECIR'06, pages 13–24, Berlin, Heidelberg. Springer-Verlag.

Nigel Collier, Hideki Hirakawa, and Akira Kumano. 1998. Machine translation vs. dictionary term translation: A comparison for english-japanese news article alignment. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, ACL '98, pages 263–267, Stroudsburg, PA, USA. Association for Computational Linguistics.

James C. French, Allison L. Powell, and Eric Schulman. 1997. Applications of approximate word matching in information retrieval. In *Proceedings of the Sixth International Conference on Information and Knowledge Management*, CIKM '97, pages 9–15, New York, NY, USA. ACM.

Kanika Gupta, Monojit Choudhury, and Kalika Bali. 2012. Mining hindi-english transliteration pairs from online hindi lyrics. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012), Istanbul, Turkey, May 23-25, 2012*, pages 2459–2465.

Parth Gupta, Kalika Bali, Rafael E. Banchs, Monojit Choudhury, and Paolo Rosso. 2014. Query expansion for mixed-script information retrieval. In *The 37th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '14, Gold Coast , QLD, Australia - July 06 - 11, 2014*, pages 677–686.

Deepak Kumar Gupta, Shubham Kumar, and Asif Ekbal. 2015. Machine learning approach for language identification & transliteration. In *Proceedings of the Forum for Information Retrieval Evaluation*, FIRE '14, pages 60–64, New York, NY, USA. ACM.

Geoffrey Hinton. 2010. A practical guide to training restricted boltzmann machines. *Momentum*, 9(1):926.

Adam Lopez. 2008. Statistical machine translation. *ACM Comput. Surv.*, 40(3):8:1–8:49, August.

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

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *arXiv preprint arXiv:1301.3781*.

Rishiraj Saha Roy, Monojit Choudhury, Prasenjit Majumder, and Komal Agarwal. 2007. Overview of the fire 2013 track on transliterated search. In *Post-Proceedings of the 4th and 5th Workshops of the Forum for Information Retrieval Evaluation*, FIRE '12 & '13, pages 4:1–4:7, New York, NY, USA. ACM.

Royal Sequiera, Monojit Choudhury, Parth Gupta, Paolo Rosso, Shubham Kumar, Somnath Banerjee, Sudip Kumar Naskar, Sivaji Bandyopadhyay, Gokul Chittaranjan, Amitava Das, and Kunal Chakma. 2015. Overview of FIRE-2015 shared task on mixed script information retrieval. In *FIRE Workshops*, volume 1587 of *CEUR Workshop Proceedings*, pages 19–25. CEUR-WS.org.

Xuerui Wang, Andrei Z. Broder, Evgeniy Gabrilovich, Vanja Josifovski, and Bo Pang. 2009. Cross-language query classification using web search for exogenous knowledge. In *Proceedings of the Second International Conference on Web Search and Web Data Mining, WSDM 2009, Barcelona, Spain, February 9-11, 2009*, pages 74–83.

Justin Zobel and Philip Dart. 1996. Phonetic string matching: Lessons from information retrieval. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '96, pages 166–172, New York, NY, USA. ACM.