MSIR@FIRE:

A Comprehensive Report from 2013 to 2016

Somnath Banerjee[1] · **Monojit Choudhury**[2] · **Kunal Chakma**[3] · **Sudip Kumar Naskar**[1] · **Amitava Das**[4] · **Sivaji Bandyopadhyay**[1] · **Paolo Rosso**[5]

Abstract India is a nation of geographical and cultural diversity where over 1600 dialects are spoken by the people. With the technological advancement, penetration of the Internet and cheaper access to Mobile Data, India has recently seen a sudden growth of Internet users. These Indian Internet users generate contents either in English or in other vernacular Indian languages. In order to develop technological solutions for the contents generated by the Indian users using the Indian languages, the Forum for Information Retrieval Evaluation (FIRE) was established and held for the first time in 2008. Although Indian languages are written using indigenous scripts, often websites and user generated content (such as tweets and blogs) in these Indian languages are written using Roman script due to various socio-cultural and technological reasons. A challenge that search engines face while processing transliterated queries and documents is that of extensive spelling variation. MSIR track was first introduced in 2013 at FIRE and the aim of MSIR was to systematically formalize several research problems that one must solve to tackle the code-mixing in Web search for users of many languages around the world, develop related data sets, test benches and most importantly, build a research community focusing on this important problem that has received very little attention. This document is a comprehensive report on the 4 years of MSIR track evaluated at FIRE between 2013 and 2016.

Keywords Information Retrieval · Indian Languages · Social Media · transliterated search · code-mixed QA

[1]Jadavpur University, India
· [2]Microsoft Research india
· [3]NIT Agartala, India
· [4]IIIT Sriharikota, India
· [5]Universidad Politécnica de Valencia, Spain

1 Introduction

The lingual diversity of the Indian sub-continent is similar to that found in Europe. Geographically, the Indian subcontinent consists of six countries and the total population in this part of the world is about 1.8 billion[1] and more than 25 official languages are used by this population. As per the 2001 Census Report[2] of India, there are over 1600 dialects, 30 languages spoken by more than a million native speakers each in India. Among the major languages in India, Hindi[3] and Bengali[4] rank among the top ten most-spoken languages of the world. According to a study[5], over the past few years, a large volume of Indian language (IL) electronic documents has come into existence with an alarming growth rate from 42 million IL internet users in 2011 to 234 million in 2016 as compared to 175 million English Internet users. According to the report in[6], IL users are expected to grow at a CAGR[7] of 18% approximately to reach 536 million in 2021. Therefore, the need for developing Information Retrieval (IR) systems to deal with this growing repository is unquestionable. The importance of reusable, large-scale standard test collections in Information Access research has been widely recognized. The success of TREC[8], CLEF[9], and NTCIR[10] has clearly established the importance of an evaluation workshop that facilitates research by providing the data and a common forum for comparing models and techniques. Prior to the conceptualization of the Forum for Information Retrieval Evaluation (FIRE) in 2008, there was no other platform for developing Information Retrieval solutions for the Indian Languages. The Forum for Information Retrieval Evaluation (FIRE) was therefore, conceptualized in the Indian context to follow in the footsteps of TREC, CLEF and NTCIR with the following aims:

1. to encourage research in Indian language Information Access technologies by providing reusable large-scale test collections for ILIR experiments
2. to provide a common evaluation infrastructure for comparing the performance of different IR systems
3. to investigate evaluation methods for Information Access techniques and methods for constructing a reusable large-scale data set for ILIR experiments.

Instead of having their own indigenous scripts, websites and user generated content (such as tweets and blogs) in most of the South and South East Asian

[1] https://bit.ly/2Hidzaq
[2] https://bit.ly/2wkG98g
[3] https://bit.ly/1OcHgWT
[4] https://bit.ly/1TaOvnQ
[5] https://bit.ly/2SW3KHf
[6] https://bit.ly/2SW3KHf
[7] https://bit.ly/34g3un9
[8] https://trec.nist.gov
[9] http://www.clef-initiative.eu/
[10] http://ntcir.nii.ac.jp/about/

languages like Bengali, Hindi, etc. are written using Roman script due to various socio-cultural and technological reasons[1]. This process of phonetically representing the words of a language in a non-native script is called transliteration. English being the most popular language of the web, transliteration, especially into the Roman script, is used abundantly on the Web not only for documents, but also for user queries that intend to search for these documents. This situation, where both documents and queries can be in more than one script, and the user expectation could be to retrieve documents across scripts is referred to as Mixed Script Information Retrieval (MSIR).

The MSIR shared task was introduced in 2013 as "Transliterated Search" at FIRE-2013 [34]. Two pilot subtasks on transliterated search were introduced as a part of the FIRE-2013 shared task on MSIR. Subtask-1 was on language identification of the query words and subsequent back transliteration of the Indian language words. The subtask was conducted for three Indian languages - Hindi, Bengali and Gujarati. Subtask-2 was on ad hoc retrieval of Bollywood song lyrics - one of the most common forms of transliterated search that commercial search engines have to tackle. Five teams participated in the shared task.

In FIRE-2014, the scope of subtask-1 was extended to cover three more South Indian languages - Tamil, Kannada and Malayalam. In subtask-2, (a) queries in Devanagari script, and (b) more natural queries with splitting and joining of words, were introduced. More than 15 teams participated in the 2 subtasks [12].

In FIRE-2015, the shared task was renamed from "Transliterated Search" to "Mixed Script Information Retrieval (MSIR)" to align it to the framework proposed by [16]. In FIRE-2015, three subtasks were conducted [37]. Subtask-1 was extended further by including more Indic languages, and transliterated text from all the languages were mixed. Subtask-2 was on searching movie dialogues and reviews along with song lyrics. Mixed script question answering (MSQA) was introduced as subtask-3. A total of 10 teams made 24 submissions for subtask-1 and subtask-2. In spite of a significant number of registrations, no run was received for subtask-3.

In last MSIR track at FIRE-2016, we hosted two subtasks in the MSIR shared task. Subtask-1 was on classifying code-mixed cross-script question; this task was the continuation of last year's subtask-3 [2]. Here Bengali words were written in Roman transliterated Bengali. Subtask-2 was on information retrieval of Hindi-English code-mixed tweets. The objective of subtask-2 was to retrieve the top k tweets from a corpus [11] for a given query consisting of Hind-English terms where the Hindi terms are written in Roman transliterated form.

This report provides the overview of the MSIR track at the Forum for Information Retrieval Conference between 2013 and 2016. Under MSIR, various academic institutions worked with Microsoft Research India, namely IIT Kharagpur, DA-IICT Gandhinagar, Technical University of Valencia, IIIT Sriharikot, Jadavpur University, and NIT Agartala.

We could categorize the tracks in four categories:

– Transliterated Search
– Ad hoc retrieval for Hindi Song Lyrics
– Code-mixed Cross-script Question Answering
– IR on Code-Mixed Hindi-English Tweets

The rest of the paper is organized as follows: Section 2 to Section 4 present the subtasks organized in MSIR track. Section 6 presents the concluding remarks of the report.

2 Transliterated Search

The language identification and transliteration tasks were the part of the MSIR track as '*Transliterated Search*' (TS) except 2016 that was the last time MSIR was organized at FIRE. In 2013, the language identification task was introduced as query word labelling. In this subtask, participation have been observed not only from India but around the world. The participation statistics are given in Table 1. The datasets description along with its availability are given in [12,34,37].

Table 1 Participation in Transliterated search

Year	2013	2014	2015
Number of teams who made a submission	5	18	9
Number of runs received	25	39	14

2.1 Task Description

Suppose that $q :< w_1 w_2 w_3 \dots w_n >$, is a query is written in Roman script. The words, w_1, w_2, w_3, ..., w_n, could be standard English(en) words or transliterated from another language L={Bengali (bn), Gujarati (gu), Hindi (hi), Kannada (kn), Malayalam (ml), Marathi (mr), Tamil (ta), Telugu (te)}. The task is to label the words as English or L or *Named Entity* depending on whether it is an English word, or a transliterated L-language word [22], or a named-entity. Named Entities(NE) could be sub-categorized as *person*(NE_P), *location* (NE_L), *organization*(NE_O), *abbreviation*(NE_PA,NE_LA,NE_OA), *inflected named entities* and *other*. For instance, the word USA is tagged as NE_LA as the name entity is both a location and an abbreviation. Sometimes, the mixing of languages can occur at the word level. In other words, when two languages are mixed at word level, the root of the word in one language, say L_r, is inflected with a suffix that belongs to another language, say L_s. Such words should be tagged as MIX. A further granular annotation of the mixed tags can be done by identifying the languages L_r and L_s and thereby tagging the word as $MIX\_L_r - L_s$.

In 2013, the labeling task was restricted to queries or very short text fragments. In contrast, in 2014 most of the sentences were acquired from social media posts (public) and blogs. We argued that with a large number of spelling variations and contractions happening over social media, the task of 2014 was more challenging than 2013. In 2013, three language pairs (namely English-Bangla, English-Gujarati, English-Hindi) were used in dataset. However, three more language pairs were added in 2014, amounting to a total of six language pairs: English-Bangla, English-Gujarati, English-Hindi, English-Kannada, English-Malayalam and English-Tamil. The language labeling task of 2015 was differed greatly from the 2013 and 2014. While the previous years' (i.e., 2013, 2014) task required one to identify the language at the word level of a text fragment given the two languages contained in the text (in other words, the language pair was known a priori). However, all the text fragments containing monolingual or code-switched (multilingual) data were mixed in the same file in 2015. Hence, an input text could belong to any of the 9 languages or a combination of any two out of the 9. Therefore, the task in 2015 was not only more challenging task than previous years (i.e., 2013 and 2014 respectively) but also was more appropriate because in real world, a search engine would not know the languages contained in a document to begin with.

Moreover, unlike 2013 and 2014, the back-transliteration of the Indic words in the native scripts was not included in 2015. This decision was made due to the observation that the most successful runs from previous years had used off-the-shelf transliteration APIs (e.g. Google Indic input tool) which beats the purpose of a research shared task.

2.2 Dataset

In 2013, organizers provided 500, 100 and 150 labelled queries as development data for English, Bangla and Gujarati respectively. The development data contained 1056, 298 and 546 distinct word transliteration pairs respectively. Due to the small size of the data, it was recommended to the participants not to use the given data for training participant algorithms, but rather as a development set for tuning model parameters. Further, 500, 100 and 150 unlabelled queries were provided as test data for English, Bangla and Gujarati respectively.

For MSIR 2014, data were collected for all the 6 language pairs (Bangla, Gujarati, Hindi, Malayalam, Tamil, Telugu, mixed with English) from various publicly available sources such as Facebook [11], Geutenberg [12] etc. In 2014, the dataset contains 6 language pairs and the data was collected from various publicly available sources. For the Hindi-English language pair, data was procured from the last year's shared task and newly annotated data from our more recent work . For the three language pairs (namely, Hindi-English, Bangla-English and Gujarati-English) data was procured from MSIR-2013 shared task

[11] www.facebook.com

[12] www.gutenberg.org

and newly annotated data sets [41, 7]. For the remaining three language pairs (Malayalam-English, Tamil-English and Telugu-English), the data was based out of publicly available sources such as Facebook[13], Geutenberg[14], etc. The details of the development and test data are given in Table 2 respectively.

Table 2 TS@MSIR-14: Dataset (Lang2 refers to the Indian language in the English-Indian language pair)

Lang2	Sentences	Tokens	E-tags	L-tags	MIX	O	NEs	Translits
Development								
Bangla	800	20,648	8,786	7,617	0	3,783	462	364
Gujarati	150	937	47	890	0	0	0	890
Hindi	1,230	27,614	11,486	11,989	0	3,371	768	2,420
Malayalam	150	1,914	326	1,139	65	292	92	0
Test								
Bangla	1,000	17,305	7,215	6,392	0	3,236	462	397
Gujarati	1,000	1,078	12	1,050	0	0	16	1,064
Hindi	1,273	32,111	12,434	13,676	0	4,815	1,186	2,542
Kannada	1,000	1,271	280	812	3	138	38	815
Malayalam	1,000	1,473	243	885	37	233	75	885
Tamil	1,000	974	460	399	0	115	0	0

In 2015, like MSIR-2014 newly annotated [7, 41] data for the language pairs were combined with the previous years' training data. The training data set was composed of 2908 utterances and 51,513 tokens. The details of the datasets were given in Table 3 .

Table 3 TS@MSIR-15: Dataset (Lang2 refers to the Indian language in the English-Indian language pair)

	Lang2	Utterances	Tokens	L-tags	Old Data
Development	Bangla	388	9,680	3,551	21,119
	Gujarati	149	937	890	937
	Hindi	294	10,512	4,295	27,619
	Kannada	276	2,746	1,622	0
	Malayalam	150	2,111	1,159	2,111
	Marathi	201	2,703	1,960	0
	Tamil	342	6,000	3,153	0
	Telugu	525	6,815	6,478	0
Test Set	Bangla	193	2,000	1,368	17,770
	Gujarati	31	937	185	1,078
	Hindi	190	2,000	1,601	32,200
	Kannada	103	1,057	598	1,321
	Malayalam	20	231	1,139	1,767
	Marathi	29	627	454	0
	Tamil	25	1,036	543	974
	Telugu	80	1,066	524	0

[13] www.facebook.com

[14] www.gutenberg.org

2.3 Evaluation Metrics

For the first two years (i.e., 2013 and 2014), the following metrics were used for evaluating the subtask. The metrics reflect various degrees of strictness, including the strictest (Exact Query Match Fraction) to the most lenient (Labeling Accuracy) metrics.

$$Exact\ query\ match\ fraction\ (EQMF) = \frac{\#(Quer.\ for\ which\ lang.\ labels\ and\ translits.\ match\ exactly)}{\#(All\ queries)} \quad (1)$$

$$Exact\ transliteration\ pair\ match\ (ETPM) = \frac{\#(Pairs\ for\ which\ translits.\ match\ exactly)}{\#(Pairs\ for\ which\ both\ o/p\ and\ reference\ labels\ are\ L)} \quad (2)$$

The value of this ratio can be treated as a measure of transliteration precision, but the absolute values of the numerator and denominator are also important. Along these lines, transliteration precision, recall and F-score were also computed as below.

$$Transliteration\ precision(TP) = \frac{\#(Correct\ transliterations)}{\#(Generated\ transliterations)} \quad (3)$$

$$Transliteration\ recall(TR) = \frac{\#(Correct\ transliterations)}{\#(Reference\ transliterations)} \quad (4)$$

$$Labelling\ accuracy(LA) = \frac{\#(Correct\ label\ pairs)}{\#(Correct\ label\ pairs) + \#(Incorrect\ label\ pairs)} \quad (5)$$

$$English\ precision(EP) = \frac{\#(E-E\ pairs)}{\#(E-L\ pairs) + \#((E-E\ pairs)} \quad (6)$$

$$English\ recall(ER) = \frac{\#(E-E\ pairs)}{\#(L-E\ pairs) + \#(E-E\ pairs)} \quad (7)$$

$$English\ F-score(EF) = \frac{2 * EP * ER}{EP + ER} \quad (8)$$

Here, an A-B pair refers to a word that is labeled by the system as A, whereas the actual label (i.e., the ground truth) is B. X is a wildcard that stands for any category label. Thus, E-E pair is a word that is of English and also labeled by the system as E, whereas E- X pair consists of all those words which are labeled as English by the system irrespective of the ground truth.

In 2015, the standard precision, recall and f-measure values were employed for evaluation. In addition, the average f-measure and weighted f-measure metrics were used to compare the performance of the teams. As there were some discrepancy in the training data with respect to the X tag, two separate versions of the aforementioned metrics were released: one considering the X tags liberally and the other version where X tags were considered strictly.

2.4 Submissions

In 2013, One team each from five institutes participated in TS shared tasks: TU Valencia (Spain), Microsoft Research India (MSRI), NTNU Norway, Gujarat University (GU) and Indian School of Mines (ISM) Dhanbad [34]. Being the part of organizers, MSIR[14] was not considered as competing team. The best performing team MSIR, employed three classifiers (namely, Naive Bayes, Maximum Entropy and Decision Tree) with combinations of character unigram, bigram, trigram, 4-grams, 5-gram, full word and context switch probability as features. TU's[17] learning algorithm was based on non-linear dimensionality reduction techniques that trained a deep autoencoder to learn the character-gram level mappings among inter/intra script words jointly. GU[21] used syllabification approach that involved transliteration from Roman script to Devanagari script (backward transliteration). NTNU's[28] models were based on Joint Source Channel Model. ISM's[30] approach was lookup based.

Table 4 TS@MSIR-13: Results [34]

		ISM	NTNU	GU-1	GU-2	TUVal-1	TUVal-2	TUVal-3	MSRI-1	MSRI-2	MSRI-3
Hindi	EQMF	0.086	0	0.036	0.002	0.022	0.02	0.006	0.194	0.198	0.186
	ETPM	1584/ 2117	540/ 1829	880/ 1853	316/ 1851	1038/ 2392	1063/ 2392	936/ 2392	1985/ 2414	1985/ 2417	1979/ 2415
	TF	0.685	0.252	0.408	0.147	0.421	0.431	0.38	0.813	0.813	0.81
	LA	0.878	0.803	0.811	0.81	0.954	0.954	0.954	0.982	0.985	0.983
	EF	0.783	0.704	0.713	0.713	0.902	0.902	0.902	0.963	0.969	0.964
	LF	0.915	0.852	0.859	0.858	0.97	0.97	0.97	0.988	0.99	0.989
Bengali	EQMF	-	-	-	-	-	-	-	0.08	0.073	0.067
	ETPM	-	-	-	-	-	-	-	485/ 1009	499/ 1026	490/ 1014
	TF	-	-	-	-	-	-	-	0.471	0.48	0.475
	LA	-	-	-	-	-	-	-	0.961	0.976	0.966
	EF	-	-	-	-	-	-	-	0.369	0.435	0.4
	LF	-	-	-	-	-	-	-	0.98	0.988	0.983
Gujrati	EQMF	-	0	-	-	-	-	-	0.01	0.01	0.01
	ETPM	-	59/ 242	-	-	-	-	-	186/ 360	193/ 371	197/ 370
	TF	-	0.186	-	-	-	-	-	0.491	0.503	0.514
	LA	-	0.699	-	-	-	-	-	0.926	0.95	0.946
	EF	-	0.588	-	-	-	-	-	0.847	0.892	0.883
	LF	-	0.763	-	-	-	-	-	0.951	0.967	0.965

In TS@MSIR'14, five different teams have topped in the different language pairs [12]. JU-NLP-LAB[3], DA-IR[29] and IITP-TS[15] participated in one or two language pairs and topped in the Bangla-English, Gujarati-English and Hindi-English tracks respectively. They fine-tuned their system for those languages and performed very well in the respective language tracks. Two teams (Asterish[33] and BITS-Lipyantaran[27]) used Google transliteration API for Hindi, and they achieved the highest TF scores. The teams which used machine learning on token based and n-gram features have higher labeling accuracy than the teams which only relied on dictionaries and rules. However, team Salazar[39] was a notable exception.

In 2015, All the submissions made by the teams for TS subtask used supervised machine learning techniques with character n-grams and character features to identify the language of the tokens. However, WISC and ISMD[31]

Table 5 TS@MSIR-14: Results [12]

Team	Run-ID	LF	EF	LA	EQMF2
Bangla-English					
BMS-Brainz	1	0.701	0.781	0.776	0.29
IIITH	1	0.833	0.861	0.85	0.383
IITP-TS	1	0.88	0.907	0.886	0.411
IITP-TS	2	0.881	0.907	0.886	0.41
IITP-TS	3	0.861	0.888	0.87	0.379
ISI	1	0.835	0.882	0.862	0.378
JU-NLP-LAB∗	1	0.899	0.92	0.905	0.444
JU-NLP-LAB	2	0.899	0.92	0.905	0.444
Gujarati-English					
BMS-Brainz	1	0.856	0.071	0.746	0.173
DA-IR∗	1	0.981	0.2	0.963	0.847
IIITH	1	0.923	0.145	0.856	0.387
Hindi-English					
asterisk	1	0.782	0.803	0.654	0.126
BITS-Lipyantaran	1	0.835	0.827	0.838	0.205
BITS-Lipyantaran	2	0.82	0.813	0.826	0.177
I1	1	0.806	0.797	0.807	0.195
I1	2	0.756	0.664	0.738	0.165
IIITH	1	0.787	0.794	0.792	0.143
IITP-TS∗	1	0.908	0.899	0.879	0.269
IITP-TS	2	0.907	0.899	0.878	0.265
IITP-TS	3	0.885	0.873	0.857	0.209
ISMD	1	0.895	0.878	0.872	0.269
ISMD	2	0.911	0.901	0.886	0.276
ISMD	3	0.911	0.901	0.886	0.276
Salazar	1	0.883	0.857	0.855	0.231
Sparkplug	1	0.693	0.641	0.599	0.053
Kannada-English					
BMS-Brainz∗	1	0.894	0.681	0.836	0.218
I1	1	0.892	0.757	0.848	0.269
IIITH	1	0.932	0.854	0.9	0.429
Malayalam-English					
BMS-Brainz	1	0.851	0.588	0.785	0.217
IIITH∗	1	0.928	0.86	0.891	0.383
Tamil-English					
BMS-Brainz	1	0.705	0.816	0.799	0.122
IIITH∗	1	0.985	0.986	0.986	0.714

teams not used any character features to train the classifier. TeamZine used word normalization as one of the features, Watchdogs converted the words into vectors using Word2Vec techniques, clustering the vectors using k-means algorithm and then using cluster IDs as the features. Three teams, Watchdogs, JU and JU_NLP[26] have gone beyond using token and character level features, by using contextual information or a sequence tagger.

Table 6 TS@MSIR-15, language identification: Performance of submissions. * indicates the best performing team;IL =Indian Languages [37]

Team	Run ID	F-score En	F-score IL	F-score MIX	F-score NE	F-score X	Token-Acc	Uttr-Acc	Average F-score	Weighted F-score
AmritaCEN	1	0.911	0.651	0.670	0.425	0.702	0.766	0.169	0.683	0.767
Hrothgar*	1	0.874	0.777	0.000	0.433	0.947	0.827	0.264	0.692	0.830
IDRBTIR	1	0.831	0.688	0.570	0.387	0.956	0.775	0.181	0.680	0.767
ISMD	1	0.905	0.603	0.400	0.462	0.961	0.771	0.173	0.615	0.769
JU	1	0.892	0.569	0.014	0.433	0.837	0.755	0.216	0.538	0.750
JU_NLP	1	0.747	0.573	0.670	0.432	0.929	0.715	0.129	0.610	0.700
JU_NLP	2	0.678	0.440	0.000	0.434	0.927	0.629	0.102	0.423	0.596
TeamZine	1	0.900	0.669	0.500	0.434	0.964	0.811	0.230	0.618	0.788
Watchdogs	1	0.698	0.644	0.000	0.410	0.967	0.689	0.858	0.576	0.701
Watchdogs	2	0.851	0.689	0.000	0.410	0.964	0.817	0.235	0.623	0.804
Watchdogs	3	0.840	0.561	0.000	0.397	0.963	0.756	0.197	0.525	0.734
WISC	1	0.721	0.356	0.000	0.249	0.824	0.548	0.240	0.387	0.568
WISC	2	0.721	0.408	0.000	0.249	0.824	0.548	0.240	0.387	0.568
WISC	3	0.722	0.408	0.000	0.249	0.822	0.548	0.240	0.387	0.568

3 Ad hoc retrieval for Hindi Song Lyrics

The subtask, Mixed-script Ad hoc retrieval for Hindi Song Lyrics, was also organized for the consecutive 3 years at FIRE since 2013 until 2015. In 2013, this subtask was introduced as '*Multi-script Ad hoc retrieval for Hindi Song Lyrics*'. Later, in 2014, this task was renamed as '*Mixed-Script Ad hoc Retrieval for Hindi Song Lyrics*'. The participation statistics of this subtask are given in Table 7. The datasets description along with its availability are given in [12,34,37].

Table 7 Participation in Ad hoc retrieval for Hindi Song Lyrics

Year	2013	2014	2015
Number of teams who made a submission	3	4	5
Number of runs received	8	7	12

3.1 Task Description

In 2013 the task was defined as: given a query in Roman script, the system has to retrieve the top-k documents from a corpus that has documents in mixed script (Roman and Devanagari). The input is a query written in Roman script, which is a transliterated form of a (possibly partial or incorrect) Hindi song title or some part of the lyrics. The output is a ranked list of ten ($k = 10$ here) songs both in Devanagari and Roman scripts, retrieved from a corpus of Hindi film lyrics, where some of the documents are in Devanagari and some in Roman transliterated form.

In 2014, like 2013, the Bollywood song lyrics corpus and song queries were used as the dataset, but two new concepts were introduced this year. First, the queries could also be in Devanagari. Second, Roman queries could have splitting or joining of words. For instance, '*main pal do palka shayar hun*'

(where the words '*pal*' and '*ka*' has been joined incorrectly), or '*madhu ban ki sugandh*' (where the word '*madhuban*' has been incorrectly split incorrectly).

In 2015, the task was changed based on the terminology and concepts defined in [16]. The objective was to retrieve mixed-script documents from a corpus for a given mixed-script query. This year, the documents and queries were written in Hindi language but using either Roman or Devanagari script. Given a query in Roman or Devanagari script, the system has to retrieve the top-k documents from a corpus that contains mixed script (Roman and Devanagari). The input is a query written in Roman (transliterated) or Devanagari script. The output is a ranked list of ten ($k = 10$ here) documents both in Devanagari and Roman scripts, retrieved from a corpus. This year there were three different genres or documents: i) Hindi songs lyrics, ii) movie reviews, and iii) astrology.

3.2 Dataset

In 2013, we first released a development (tuning) data for the IR system – 25 queries, associated relevance judgments (qrels) and the corpus. The queries were Bollywood song lyrics. The corpus consisted of 62, 888 documents which contained song titles and lyrics in Roman (ITRANS or plain format), Devanagari and mixed scripts. The test set also consisted of twenty five queries. On an average, there were 28.38 qrels per query. The mean query length was 4.5 words. The song lyrics documents were created by crawling several popular domains like *dhingana*, *musicmaza* and *hindilyrix*.

In 2014, the development (tuning) data contains 25 Bollywood song lyrics queries, associated relevance judgments (qrels) and the corpus. The corpus consisted of 62, 894 documents which contained song titles and lyrics in Roman (ITRANS or plain format), Devanagari and mixed scripts. The test set consisted of 35 queries in either Roman or Devanagari script. On an average, there were 65.48 qrels per query with average relevant documents per query to be 7.37 and cross-script relevant documents to be 3.26. The mean query length was 4.57 words. The domain of the song lyrics documents were same as 2013.

In 2015, the released development (tuning) data for the IR system – 15 queries, associated relevance judgments (*qrels*) and the corpus. The queries were related to three genres: *i)* Hindi songs lyrics, *ii)* movie reviews, and *iii)* astrology. The corpus consisted of 63, 334 documents in Roman (ITRANS or plain format), Devanagari and mixed scripts. The test set consisted of 25 queries in either Roman or Devanagari script. On an average, there were 47.52 qrels per query with average relevant documents per query to be 5.00 and cross-script[15] relevant documents to be 3.04. The mean query length was 4.04 words. The domain of the song lyrics documents were same as 2013. The movie reviews data was crawled from `http://www.jagran.com/` while astrology data was crawled from `http://astrology.raftaar.in/`.

[15] Those documents which contain duplicate content in both the scripts are ignored.

3.3 Evaluation Metrics

For evaluating the task, we used the well-established IR metrics of normalized Discounted Cumulative Gain (nDCG) [20], Mean Average Precision (MAP) [36] and Mean Reciprocal Rank (MRR) [40]. We used the following process for computing nDCG. The formula used for DCG@p was as follows:

$$DCG@p = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{log_2 i} \tag{9}$$

where p is the rank at which we are computing DCG and rel_i is the graded relevance of the document at rank i. For IDCG@p, we sort the RJs for a particular query in the pool in descending order and take the top-p from the pool, and compute DCG@p for that list (since that is the best possible (ideal) ranking for that query). Then, as usual, we have

$$nDCG@p = \frac{DCG@p}{IDCG@p} \tag{10}$$

nDCG was computed after looking at the first five and the first ten retrieved documents (nDCG@5 and nDCG@10).

For computing MAP, we first compute average precision $AveP$ for every query, where AveP is given by

$$AveP = \frac{\sum_{k=1}^{n}(P(k) \times rel(k))}{\text{No. of relevant documents}} \tag{11}$$

where k is the rank in the sequence of retrieved documents, n is the number of retrieved documents, $P(k)$ is the precision at cut-off k in the list and $rel(k)$ is an indicator function equaling 1 if the item at rank k is a relevant document, zero otherwise. Then,

$$MAP = \frac{\sum_{q=1}^{Q} AveP(q)}{Q} \tag{12}$$

where Q is the number of queries. In our case, we consider relevance judgments 1 and 2 as non-relevant, and 3, 4, and 5 as relevant. MAP was computed after looking at the first ten retrieved documents.

The reciprocal rank of a query response is the multiplicative inverse of the rank of the first correct answer ($rank_i$). MRR is the average of the reciprocal ranks of results for a sample of queries Q

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i} \tag{13}$$

3.4 Submissions

In 2013, three teams NTNU[28], GU[21] and TU-Val[17] submitted 3, 2 and 3 runs respectively. TU-Val performed the best on all the four metrics. TU-Val indexed the songs collection as word 2-grams and used word 2-gram variant of TF-IDF like models for retrieval. GU team employed syllabification approach for transliteration of queries. Later the Hindi song lyrics corpus was indexed and retrieval was performed using the test queries using TF-IDF model. The worst performing team NTNU employed Apache Lucene IR system. Lucene follows the standard IR model with Document parsing, Document Indexing, TF-IDF calculation, query parsing and finally searching/document retrieval. The results are given in Table 8.

Table 8 Ad-hoc@MSIR'13: Results

Metric	NTNU-1	NTNU-2	NTNU-3	GU-1	GU-2	TUVal-1	TUVal-2	TUVal-3	Max	Median
nDCG@5	0.205	0.523	0.561	0.563	0.526	0.767	0.805	0.758	0.805	0.562
nDCG@10	0.207	0.52	0.56	0.562	0.523	0.764	0.8	0.753	0.8	0.561
MAP	0.003	0.152	0.197	0.255	0.216	0.421	0.424	0.356	0.424	0.234
MRR	0.018	0.555	0.593	0.584	0.573	0.775	0.844	0.777	0.844	0.588

In Ad-hoc@MSIR'14, we received 7 runs and we observed that the two runs from BITS-Lipyantran[27] performs best across all the metrics. Table 9 presents the results of the 7 runs received. Using Devanagari as the working script, and mapping both the queries and documents to Devanagari helped BITS-Lipyantran because in the native script their was usually one single correct spelling. Moreover, use of Google Transliteration API and word level n-grams for indexing and matching helped in improving the precision. Team BIT[32] used relevance feedback approach in retrieving the relevant documents from a mixed script documents collection. Another team DCU[13] applied a rule-based normalization on some character sequences of the transliterated words in order to have a single representation in the index for the multiple transliteration alternatives. During the retrieval phase, DCU used prefix matched fuzzy query terms to account for the morphological variations of the transliterated words. It was noted that For all the systems, the performance reasonably well when the scripts of the query and the document were the same.

Table 9 Ad-hoc@MSIR'14: Results

Team	**Run**	**NDCG@1**	**NDCG@5**	**NDCG@10**	**MAP**	**MRR**	**R@10**	**csR@10**
BIT	1	0.5024	0.3967	0.3612	0.2698	0.5243	0.4343	0.2193
BIT	2	0.6452	0.4918	0.4572	0.3415	0.6271	0.4822	0.1898
BITS-Lipyantran	1	0.75	0.7817	0.6822	0.6263	0.7929	0.6818	0.4144
BITS-Lipyantran*	2	0.7708	0.7954	0.6977	0.6421	0.8171	0.6918	0.443
DCU	1	0.5786	0.5924	0.5626	0.4112	0.6269	0.4943	0.3483
DCU	2	0.4143	0.3933	0.371	0.2063	0.3979	0.2807	0.3035
IIITH	1	0.6429	0.5262	0.5105	0.412	0.673	0.5806	0.3407

In In Ad-hoc@MSIR'15, most of the submitted runs (total 12) handled the mixed-script aspect using some type of transliteration approach and then different matching techniques were used to retrieve documents. BIT-M proposed a system where transliteration module used the relative frequency of letter group mappings and search module used the transliteration module to treat everything in devanagari script.

Table 10 Ad-hoc@MSIR15: results averaged over test queries

Team	NDCG@1	NDCG@5	NDCG@10	MAP	MRR	R@10
AmritaCEN	0.2300	0.2386	0.1913	0.0986	0.2067	0.1308
BIT-M	0.7567	0.6837	0.6790	0.3922	0.5890	0.4735
Watchdogs-1	0.6700	0.5922	0.6057	0.3173	0.4964	0.3962
Watchdogs-2	0.5267	0.5424	0.5631	0.2922	0.3790	0.4435
Watchdogs-3	0.6967	0.6991	0.7160	0.3814	0.5613	0.4921
Watchdogs-4	0.5633	0.5124	0.5173	0.2360	0.3944	0.2932
ISMD-1	0.4133	0.4268	0.4335	0.0928	0.2440	0.1361
ISMD-2	0.4933	0.5277	0.5328	0.1444	0.3180	0.2051
ISMD-3	0.3867	0.4422	0.4489	0.0954	0.2207	0.1418
ISMD-4	0.4967	0.5375	0.5369	0.1507	0.3397	0.2438
QAIIITH-1	0.3433	0.3481	0.3532	0.0705	0.2100	0.1020
QAIIITH-2	0.3767	0.3275	0.3477	0.0561	0.2017	0.1042

Table 11 Ad-hoc@MSIR15: results averaged over test queries in cross-script setting

Team	NDCG@1	NDCG@5	NDCG@10	MAP	MRR	R@10
AmritaCEN	0.1367	0.1182	0.1106	0.0898	0.1533	0.1280
BIT-M	0.3400	0.3350	0.3678	0.2960	0.3904	0.4551
Watchdogs-1	0.4233	0.3264	0.3721	0.2804	0.4164	0.3774
Watchdogs-2	0.1833	0.2681	0.3315	0.2168	0.2757	0.4356
Watchdogs-3	0.3333	0.3964	0.4358	0.3060	0.4233	0.5058
Watchdogs-4	0.2900	0.2684	0.2997	0.2047	0.3244	0.2914
ISMD-1	0.0600	0.0949	0.1048	0.0452	0.0714	0.0721
ISMD-2	0.1767	0.2688	0.2824	0.1335	0.1987	0.2156
ISMD-3	0.0600	0.1098	0.1191	0.0563	0.0848	0.0988
ISMD-4	0.2267	0.3242	0.3375	0.1522	0.2253	0.2769
QAIIITH-1	0.0600	0.0626	0.0689	0.0313	0.0907	0.0582
QAIIITH-2	0.0200	0.0539	0.0673	0.0234	0.0567	0.0661

Watchdogs used Google transliterator to transliterate every Roman script word in the documents and queries to Devanagari word. They submitted 4 runs with these settings: 1. Indexed the individual words using simple analyser in lucene and then fired the query, 2. Indexed using word level 2 to 6 grams and then fired a query, 3. Removed all the vowel signs and spaces from the documents and queries and indexed the character level 2-6 grams of the documents, and 4. Removed the spaces and replaced vowel signs with actual characters in the documents and queries and indexed the character level 2-6 grams of the documents. ISMD also submitted four runs. First two runs were using simple indexing, with and without query expansion. Third and fourth

runs were using block indexing, with and without query expansion. The other teams did not share their approaches. The detailed results are given in Table 10 and Table 11 respectively.

4 Task: Code-mixed Cross-script Question Answering

In 2015, the code-mixed cross-script question answering (CMCS-QA) was introduced as a pilot task at FIRE. In 2016, the task was modified considering the provided short time frame. Although a total of 11 teams registered for the task in 2015, no runs were submitted by the registered participants. However, in 2016, 7 teams participated. In this subtask, participation have been observed from academic as well as industries. The participation statistics are given in Table 12. The datasets description along with its availability are given in [2,37].

Table 12 Participation in code-mixed QA

Year	2015	2016
Number of teams registered	11	15
Number of teams who made a submission	-	7
Number of runs received	-	20

4.1 Task Description

The code-mixed QA task can be defined as: Let, Q = $\{q_1, q_2, \ldots, q_n\}$, be a set of factoid questions associated with a document corpus C in domain D and topic T, written in Romanized Bengali. The document corpus C consists of a set of Romanized Bengali social media messages which could be code-mixed with English (i.e., it also contains English words and phrases). The task is to build a QA system which can output the exact answer, along with the message/posts identification number (msg_ID) and message segment (S_ans) that contains the exact answer. This task deals with factoid questions only. For this subtask,

domain D = {Sports, Tourism}

Mixed Language pair = {Bengali-English}

Although a total of 11 teams registered for the task, no runs were submitted by the registered participants.

Considering the time constraint, in 2016, the code-mixed cross-script question classification was introduced as a part of the developing cross-script QA system. The classification problem can be formulated as:

Task: Let, $Q = \{q_1, q_2, \ldots, q_n\}$ be a set of factoid questions associated with domain D. Each question $q : \langle w_1 w_2 w_3 \ldots w_p \rangle$, is a set of words where p denotes the total number of words in a question. The words, $w_1, w_2, w_3, \ldots, w_p$, could

be English words or transliterated from Bengali in the code mixed scenario. Let $C = \{c_1, c_2, \ldots, c_m\}$ be the set of question classes. Here n and m refer to the total number of questions and question classes respectively.

The objective of this subtask is to classify each given question $q_i \in Q$ into one of the predefined coarse-grained classes $c_j \in C$. For example, the question "*last volvo bus kokhon chare?*" (English gloss: "When does the last volvo bus depart?") should be classified to the class 'TEMPORAL'.

4.2 Dataset

So far the code-mixed cross-script QA research is concerned, the only first dataset was [5]. Later, dataset was prepared for named entity recognition for ciode-mixed QA [6]. The dataset described in [5] includes questions, messages and answers that are based on sports and tourism domains in code-mixed cross-script English-Bengali. The sports domain consists of 10 documents that further include 116 informal posts and 192 questions. The tourism domain also consists of 10 documents that contain 183 informal posts and 314 questions. The training dataset comprises of 330 labelled factoid CMCS questions whereas the testset comprises of 180 data points. The average length of a question is 5.321 in the training dataset while the average length of a question is 6.322 in the testset. The statistics of the dataset is provided in Table 13 and Table 14 that are mentioned below. Question class specific distribution of the datasets is given in Figure 1.

Table 13 CMQA@MSIR'16: dataset

Dataset	Questions(Q)	Total Words	Avg. Words/Q
Trainset	330	1776	5.321
Testset	180	1138	6.322

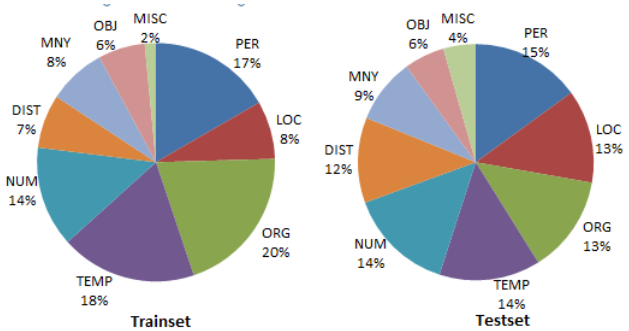

Fig. 1 Distribution of CMQA@MSIR'16 train and test datasets

Table 14 CMQA@MSIR'16: question class statistics

Class	Training	Testing
Person (PER)	55	27
Location (LOC)	26	23
Organization (ORG)	67	24
Temporal(TEMP)	61	25
Numerical(NUM)	45	26
Distance(DIST)	24	21
Money(MNY)	26	16
Object(OBJ)	21	10
Miscellaneous(MISC)	5	8

4.3 Evaluation Measures and Baseline

We employed *accuracy* to evaluate code-mixed cross-script question classification performance.

$$accuracy = \frac{\text{number of correctly classified samples}}{\text{total number of testset samples}}$$

Additionally, we also computed the standard precision, recall and F1-measure to evaluate the class specific performances of the participating systems. The precision, recall and F1-measure of a classifier on a particular class c are defined as follows:

$$precision(P) = \frac{\text{number of samples correctly classified as } c}{\text{number of samples classified as } c}$$

$$recall(R) = \frac{\text{number of samples correctly classified as } c}{\text{total number of samples in class } c}$$

$$F1 - measure = \frac{2.P.R}{P + R}$$

In order to provide a benchmark for the comparison of the submitted systems, a baseline system was also developed using the Bag-of-Words (BoW) which obtained 79.444% accuracy.

4.4 Submissions

Team NLP-NITMZ [25] submitted 3 runs based on the three approaches: i) based on direct feature set; ii) based on direct and dependent feature set and iii) based on Naïve Bayes classifier. A total of 39 rules were identified for the first and second approaches. BITS_PILANI [9] team converted the data in English and extracted the n-grams. Then, they applied three different machine learning classifiers, namely Gaussian Naïve Bayes, Logistics Regression and Random Forest Classifier. BITS_PILANI team jointly ranked second with the team ANUJ . Team ANUJ [35] used term TF-IDF vector as a feature. A number of machine learning algorithms, namely Support Vector Machines

Table 15 CMQA@MSIR'16: teams performance (*denotes late submission)

Team	Run ID	Correct	Incorrect	Accuracy
Baseline	-	143	37	79.440
AmritaCEN	1	145	35	80.556
AmritaCEN	2	133	47	73.889
AMRITA-CEN-NLP	1	143	37	79.444
AMRITA-CEN-NLP	2	132	48	73.333
AMRITA-CEN-NLP	3	132	48	73.333
Anuj	1	139	41	77.222
Anuj	2	146	34	81.111
Anuj	3	141	39	78.333
BITS_PILANI	1	146	34	81.111
BITS_PILANI	2	144	36	80.000
BITS_PILANI	3	131	49	72.778
IINTU	1	147	33	81.667
IINTU	**2**	**150**	**30**	**83.333**
IINTU	3	146	34	81.111
NLP-NITMZ	1	134	46	74.444
NLP-NITMZ	2	134	46	74.444
NLP-NITMZ	3	142	38	78.889
*IIT(ISM)D	1	144	36	80.000
*IIT(ISM)D	2	142	38	78.889
*IIT(ISM)D	3	144	36	80.000

Table 16 CMQA@MSIR'16: class specific performances (NA denotes no identification of a class)

Team	Run ID	PER	LOC	ORG	NUM	TEMP	MONEY	DIST	OBJ	MISC
AmritaCEN	1	0.8214	0.8182	0.5667	0.9286	**1.0000**	0.7742	0.9756	0.5714	NA
AmritaCEN	2	0.7541	0.8095	0.6667	0.8125	**1.0000**	0.4615	0.8649	NA	NA
AMRITA-CEN-NLP	1	0.8000	0.8936	0.6032	0.8525	0.9796	0.7200	0.9500	**0.5882**	NA
AMRITA-CEN-NLP	2	0.7500	0.7273	0.5507	0.8387	0.9434	0.5833	0.9756	0.1818	NA
AMRITA-CEN-NLP	3	0.6939	0.8936	0.5455	0.8125	0.9804	0.6154	0.8333	0.3077	NA
IINTU	1	0.7843	0.8571	0.6333	0.9286	**1.0000**	0.8125	0.9756	0.4615	NA
IINTU	2	0.8077	**0.8980**	0.6552	0.9455	**1.0000**	0.8125	0.9756	0.5333	NA
IINTU	3	0.7600	0.8571	0.5938	0.9455	**1.0000**	0.8571	**0.9767**	0.4615	NA
NLP-NITMZ	1	0.7347	0.8444	0.5667	0.8387	0.9796	0.6154	0.9268	0.2857	0.1429
NLP-NITMZ	2	0.6190	0.8444	0.5667	0.9630	0.8000	0.7333	0.9756	0.4286	0.1429
NLP-NITMZ	3	**0.8571**	0.8163	0.7000	0.8966	0.9583	0.7407	0.9268	0.3333	**0.2000**
Anuj	1	0.7600	0.8936	0.6032	0.8125	0.9804	0.7200	0.8649	0.5333	NA
Anuj	2	0.8163	0.8163	0.5538	**0.9811**	0.9796	0.9677	0.9500	0.5000	NA
Anuj	3	0.8163	0.8936	0.5846	0.8254	**1.0000**	0.7200	0.8947	0.5333	NA
BITS_PILANI	1	0.7297	0.7442	**0.7442**	0.9600	0.9200	0.9412	0.9500	0.5000	**0.2000**
BITS_PILANI	2	0.6753	0.7805	0.7273	0.9455	0.9600	**1.0000**	0.8947	0.4286	NA
BITS_PILANI	3	0.6190	0.7805	0.7179	0.8125	0.8936	0.9333	0.6452	0.5333	NA
*IIT(ISM)D	1	0.7755	0.8936	0.6129	0.8966	0.9412	0.7692	0.9524	**0.5882**	NA
*IIT(ISM)D	2	0.8400	0.8750	0.6780	0.8525	0.9091	0.6667	0.9500	0.1667	NA
*IIT(ISM)D	3	0.8000	0.8936	0.6207	0.8667	**1.0000**	0.6923	0.9500	0.5333	NA
Avg		0.7607	0.8415	0.6245	0.8858	0.9613	0.7568	0.9204	0.4458	NA

(SVM), Logistic Regression (LR), Random Forest (RF) and Gradient Boosting were applied using Grid Search to come up with the best parameters and model.Amrita_CEN [24] submitted two runs based on the two approaches: Bag of Words (BoW) and Long Short Term Memory (LSTM). The Bag-of-words based model achieved better accuracy than LSTM based model. IIT(ISM)D used three different machine learning based classification models - Sequential Minimal Optimization, Naïve Bayes Multi model and Decision Tree FT to annotate the question text. AMRITA-CEN-NLP [19] approached the problem using a Vector Space Model (VSM). A weighted term approach based on

the context was applied to overcome the shortcomings of VSM. The approach employed by the team IINTU [10] was performed the best among all participating teams. A vector representation was proposed for each question which was used as an input to the classifier. They considered the top 2000 most frequently occurring words in the supplied training dataset as features. Three separate classifiers were used, namely Random Forests, One-vs-Rest (OvR) classifier and k-Nearest Neighbour (k-NN) classifier. Then, an ensemble classifier was built using these three classifiers for the classification task. The ensemble classifier took the output label by each of the individual classifiers and provided the majority label as output, otherwise any label was chosen at random as output. Each of the individual classifiers was trained on a subset of the original training dataset, by sampling with replacement.

The performance of the teams in terms of accuracy is given in Table 15. Table 16 presents the class specific performances in terms of precision, recall and F1-measure. IINTU team performed the best and obtained the highest accuracy of 83.333%. It is prominent from Table 16 that the classification performance on the temporal (TEMP) class was very high for almost all the teams. However, Table 16 suggest that the miscellaneous (MISC) question class was very difficult to identify. Due to very low presence (almost 2%) of MISC class in the training data, most of the teams could not identify the 'MISC' class. The F-score for all the classes are above 85% except for 'OBG', 'ORG' and 'MISC'. It was observed that the deep learning approach did not performed well due to the tiny size of the dataset. Later, [4] showed that incorporating linguistic features with deep learning could enhance the performance of the code-mixed question classification.

5 IR on Code-Mixed Hindi-English Tweets

This subtask was based on the concepts discussed in [16]. Table 17 shows the participation of this subtask. The dataset description along with its availability are given in [2].

Table 17 Participation in IR on code-mixed Hindi-English tweets

Year	2016
Number of teams registered	15
Number of teams who made a submission	7
Number of runs received	13

5.1 Task Description

In this subtask, the objective was to retrieve Code-Mixed Hindi-English tweets from a corpus for code-mixed queries. The Hindi components in both the tweets

and the queries are written in Roman transliterated form. This subtask did not consider cases where both Roman and Devanagari scripts are present. Therefore, the documents in this case are tweets consisting of code-mixed Hindi-English texts where the Hindi terms are in Roman transliterated form. Given a query consisting of Hindi and English terms written in Roman script, the system has to retrieve the top-k documents (i.e., tweets) from a corpus that contains Code-Mixed Hindi-English tweets. The expected output is a ranked list of the top twenty (k=20 here) tweets retrieved from the given corpus.

5.2 Datasets

Initially we released 6,133 code-mixed Hindi-English tweets with 23 queries as the training dataset. Later we released a document collection containing 2,796 code-mixed tweets along with with 12 code-mixed queries as the testset. Query terms are mostly *named entities* with Roman transliterated Hindi words. The average length of the queries in the training set is 3.43 words and in the testset it is 3.25 words. The tweets in the training set cover 10 topics whereas the testset cover 3 topics.

5.3 Evaluation Metric

The retrieval task requires that the retrieved documents at higher ranks be more important than the retrieved documents at lower ranks for a given query and we want our measures to account for that. Therefore, set based evaluation metrics such as Precision, Recall and F-measure are not suitable for this task. Therefore, we used Mean Average Precision (MAP) as the performance evaluation metric. MAP is also referred to as "*average precision* at seen relevant documents". MAP is represented as

$$MAP = \frac{1}{N}\sum_{j=1}^{N}\frac{1}{Q_j}\sum_{i=1}^{Q_j}P(doc_i)$$

where Q_j refers to the number of relevant documents for query j, N indicates the number of queries and $P(doc_i)$ represents precision at the i^{th} relevant document.

5.4 Submissions

The evaluation results of the submitted 13 runs are reported in Table 18. The submitted runs for the retrieval task of Code-Mixed tweets mostly adopted preprocessing of the data and then applying different techniques for retrieving the desired tweets. Team Amrita_CEN [18] removed some Hindi/English stop words to declutter useless words. After that, they have tokenized all the tweets. The cosine distance was used to score the relevance of tweets to the

Table 18 Results for Subtask-2 showing Mean Average Precision

Team	Run ID	MAP
UB	1	0.0217
UB	2	0.016
UB	3	0.0152
Anuj	1	0.0209
Anuj	2	0.0199
Amrita_CEN	1	**0.0377**
NLP-NITMZ	1	0.0203
NITA_NITMZ	1	0.0047
CEN@Amrita	1	0.0315
CEn@Amrita	2	0.0168
IIT(ISM)D	1	0.0021
IIT(ISM)D	2	0.0083
IIT(ISM)D	3	0.0021

query. The highest MAP (0.0377) was achieved by team Amrita@CEN which is still very low. After that, the top 20 tweets based on the scores were retrieved. Team CEN@Amrita [38] used a Vector Space Model based approach. Team UB [23] adopted three different techniques for the retrieval task. First, they used *Named Entity boosts* where the purpose was to boost the documents based on their NE matches from the query, i.e., the query was parsed to extract NEs and each document (tweet) that matched the given NE was provided a small numeric boost. At the second level of boosting, phrase matching was carried out , i.e. documents that more closely matched the input query phrase were ranked higher than those that did not. The UB team used *Synonym Expansion* and *Narrative based weighting* as the second and third techniques. Team NITA_NITMZ [8] performed stop word removal followed by query segmentation and finally merging. Team IIT(ISM)_D considered every tweet as a document and indexed using uniword indexing on Terrier implementation. Query terms were expanded using the soundex coding scheme. Terms with an identical soundex code were selected as candidate query and included in final queries to retrieve the relevent tweets (documents). Further, they used three different retrieval models BM25, DFR and TF-IDF to measure the similarity. However, this team submitted the runs after the deadline.

6 Conclusion

In this report, we elaborated the four subtasks of the MSIR track at the FIRE from 2013 to 2016. For each subtask, the task description, dataset, evaluations metric and submissions are discussed in detail.

Transliterated search subtask was on language labeling of short text fragments and back-transliteration the text fragment in the native script based on the identified language label. This subtask is one of the first steps before one can tackle the general problem of mixed script information retrieval.

Ad-hoc retrieval of Hindi film lyrics, movie reviews and astrology documents are some of the most searched items in India, and thus, are perfect and practical examples of transliterated search.

Due to the rapid growth of multi-lingual contents on the web, existing QA systems faced several challenges. Code-mixing is one such challenge that makes the QA even more complex. Nowadays, the research in this topic is gaining notable attention.

IR on Code-Mixed Hindi-English Tweets subtask is also an ad-hoc retrieval task like Ad-hoc retrieval of Hindi film lyrics. The only difference between these two subtasks is the corpus. The earlier used a corpus of Hindi song lyrics, whereas, the later subtask used code-mixed tweets.

On final note, MSIR track through FIRE platform systematically formalized research problems in code-mixing scenario that one must solve to tackle this prevalent situation in Web search for users of many languages around the world, developed related data sets, tested benches and most importantly, built a research community around this important problem that has previously received very little attention. Undoubtedly, MSIR played a remarkable role in the incredible journey of FIRE.

Acknowledgements Somnath Banerjee and Sudip Kumar Naskar are supported by Media Lab Asia, MeitY, Government of India, under the Visvesvaraya PhD Scheme for Electronics & IT. The work of Paolo Rosso was partially supported by the MISMIS research project PGC2018-096212-B-C31 funded by the Spanish MICINN.

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. Ahmed, U.Z., Bali, K., Choudhury, M., B., S.V.: Challenges in designing input method editors for indian languages: The role of word-origin and context. Advances in Text Input Methods (WTIM 2011) pp. 1–9 (2011)
2. Banerjee, S., Chakma, K., Naskar, S.K., Das, A., Rosso, P., Bandyopadhyay, S., Choudhury, M.: Overview of the mixed script information retrieval (msir) at fire-2016. In: Forum for Information Retrieval Evaluation, pp. 39–49. Springer (2016)
3. Banerjee, S., Kuila, A., Roy, A., Naskar, S.K., Rosso, P., Bandyopadhyay, S.: A hybrid approach for transliterated word-level language identification: Crf with post-processing heuristics. In: Proceedings of the Forum for Information Retrieval Evaluation, pp. 54–59. ACM (2014)
4. Banerjee, S., Naskar, S., Rosso, P., Bandyopadhyay, S.: Code mixed cross script factoid question classification - a deep learning approach. Journal of Intelligent & Fuzzy Systems **34**(5), 2959–2969 (2018)
5. Banerjee, S., Naskar, S.K., Rosso, P., Bandyopadhyay, S.: The First Cross-Script Code-Mixed Question Answering Corpus. Proceedings of the workshop on Modeling, Learning and Mining for Cross/Multilinguality (MultiLingMine 2016), co-located with The 38th European Conference on Information Retrieval (ECIR) (2016)

6. Banerjee, S., Naskar, S.K., Rosso, P., Bandyopadhyay, S.: Named entity recognition on code-mixed cross-script social media content. Computación y Sistemas **21**(4), 681–692 (2017)
7. Barman, U., Das, A., Wagner, J., Foster, J.: Code mixing: A challenge for language identification in the language of social media. In: Proceedings of the first workshop on computational approaches to code switching, pp. 13–23 (2014)
8. Bhardwaj, P., Pakray, P., Bajpeyee, V., Taneja, A.: Information Retrieval on Code-Mixed Hindi-English Tweets. In: Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation, Kolkata, India, December 7-10, 2016, CEUR Workshop Proceedings (2016)
9. Bhargava, R., Khandelwal, S., Bhatia, A., Sharmai, Y.: Modeling Classifier for Code Mixed Cross Script Questions. In: Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation, Kolkata, India, December 7-10, 2016, CEUR Workshop Proceedings. CEUR-WS.org (2016)
10. Bhattacharjee, D., Bhattacharya, P.: Ensemble Classifier based approach for Code-Mixed Cross-Script Question Classification. In: Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation, Kolkata, India, December 7-10, 2016, CEUR Workshop Proceedings. CEUR-WS.org (2016)
11. Chakma, K., Das, A.: CMIR: A Corpus for Evaluation of Code Mixed Information Retrieval of Hindi-English Tweets. In: In the 17th International Conference on Intelligent Text Processing and Computational Linguistics (CICLING) (2016)
12. Choudhury, M., Chittaranjan, G., Gupta, P., Das, A.: Overview of fire 2014 track on transliterated search. Proceedings of FIRE pp. 68–89 (2014)
13. Ganguly, D., Pal, S., Jones, G.J.: Dcu@ fire-2014: Fuzzy queries with rule-based normalization for mixed script information retrieval. In: Proceedings of the Forum for Information Retrieval Evaluation, pp. 80–85. ACM (2014)
14. Gella, S., Sharma, J., Bali, K.: Query word labeling and back transliteration for indian languages: Shared task system description. FIRE Working Notes **3** (2013)
15. Gupta, D.K., Kumar, S., Ekbal, A.: Machine learning approach for language identification & transliteration. In: Proceedings of the Forum for Information Retrieval Evaluation, pp. 60–64. ACM (2014)
16. Gupta, P., Bali, K., Banchs, R.E., Choudhury, M., Rosso, P.: Query expansion for mixed-script information retrieval. In: Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval, pp. 677–686. ACM (2014)
17. Gupta, P., Rosso, P., Banchs, R.E.: Encoding transliteration variation through dimensionality reduction: Fire shared task on transliterated search. In: Fifth Forum for Information Retrieval Evaluation (2013)
18. HB, B.G., M, A.K., KP, S.: Distributional Semantic Representation for Information Retrieval. In: Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation, Kolkata, India, December 7-10, 2016, CEUR Workshop Proceedings (2016)
19. HB, B.G., M, A.K., KP, S.: Distributional Semantic Representation for Text Classification. In: Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation, Kolkata, India, December 7-10, 2016, CEUR Workshop Proceedings. CEUR-WS.org (2016)
20. Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of IR techniques. ACM Trans. Inf. Syst. **20**, 422–446 (2002). DOI http://doi.acm.org/10.1145/582415.582418. URL `http://doi.acm.org/10.1145/582415.582418`
21. Joshi, H., Bhatt, A., Patel, H.: Transliterated search using syllabification approach. In: Forum for Information Retrieval Evaluation (2013)
22. King, B., Abney, S.: Labeling the languages of words in mixed-language documents using weakly supervised methods. In: Proceedings of NAACL-HLT, pp. 1110–1119 (2013)
23. Londhe, N., Srihari, R.K.: Exploiting Named Entity Mentions Towards Code Mixed IR: Working Notes for the UB system submission for MSIR@FIRE'16. In: Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation, Kolkata, India, December 7-10, 2016, CEUR Workshop Proceedings (2016)

24. M, A.K., P, S.K.: Amrita-CEN@MSIR-FIRE2016: Code-Mixed Question Classification using BoWs and RNN Embeddings. In: Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation, Kolkata, India, December 7-10, 2016, CEUR Workshop Proceedings. CEUR-WS.org (2016)
25. Majumder, G., Pakray, P.: NLP-NITMZ @ MSIR 2016 System for Code-Mixed Cross-Script Question Classification. In: Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation, Kolkata, India, December 7-10, 2016, CEUR Workshop Proceedings. CEUR-WS.org (2016)
26. Mandal, S., Banerjee, S., Naskar, S.K., Rosso, P., Bandyopadhyay, S.: Adaptive voting in multiple classifier systems for word level language identification. In: FIRE Workshops, pp. 47–50 (2015)
27. Mukherjee, A., Datta, K., Ravi, A.: Mixed-script query labelling using supervised learning and ad hoc retrieval using sub word indexing: Shared task report by bits pilani, hyderabad (2014)
28. Pakray, P., Bhaskar, P.: Transliterated search system for indian languages. In: Preproceedings of the 5th FIRE-2013 Workshop, Forum for Information Retrieval Evaluation (FIRE) (2013)
29. Patel, S., Desai, V.: Liga and syllabification approach for language identification and back transliteration: A shared task report by da-iict. In: Proceedings of the Forum for Information Retrieval Evaluation, pp. 43–47. ACM (2014)
30. Prabhakar, D.K., Pal, S.: Ism@ fire-2013 shared task on transliterated search. In: Post-Proceedings of the 4th and 5th Workshops of the Forum for Information Retrieval Evaluation, p. 17. ACM (2013)
31. Prabhakar, D.K., Pal, S.: Ism@ fire-2015: Mixed script information retrieval. In: FIRE Workshops, pp. 55–58 (2015)
32. Prakash, A., Saha, S.K.: A relevance feedback based approach for mixed script transliterated text search: Shared task report by bit mesra, india (2014)
33. Raj, A., Karfa, S.: A list-searching based approach for language identification in bilingual text: Shared task report by asterisk. In: Working Notes of the Shared Task on Transliterated Search at Forum for Information Retrieval Evaluation FIRE'14 (2014)
34. Roy, R.S., Choudhury, M., Majumder, P., Agarwal, K.: 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, p. 4. ACM (2013)
35. Saini, A.: Code Mixed Cross Script Question Classification. In: Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation, Kolkata, India, December 7-10, 2016, CEUR Workshop Proceedings. CEUR-WS.org (2016)
36. Salton, G., McGill, M.J.: Introduction to Modern Information Retrieval. McGraw-Hill, Inc. (1986)
37. Sequiera, R., Choudhury, M., Gupta, P., Rosso, P., Kumar, S., Banerjee, S., Naskar, S.K., Bandyopadhyay, S., Chittaranjan, G., Das, A., et al.: Overview of fire-2015 shared task on mixed script information retrieval. In: FIRE Workshops, vol. 1587, pp. 19–25 (2015)
38. Singh, S., M, A.K., KP, S.: CEN@Amrita: Information Retrieval on CodeMixed Hindi-English Tweets Using Vector Space Models. In: Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation, Kolkata, India, December 7-10, 2016, CEUR Workshop Proceedings (2016)
39. Sinha, N., Srinivasa, G.: Hindi-english language identification, named entity recognition and back transliteration: Shared task system description. In: Working Notes os Shared Task on Transliterated Search at Forum for Information Retrieval Evaluation FIRE'14 (2014)
40. Voorhees, E.M., Tice, D.M.: The TREC-8 Question Answering Track Evaluation. In: TREC-8, pp. 83–105 (1999)
41. Vyas, Y., Gella, S., Sharma, J., Bali, K., Choudhury, M.: Pos tagging of english-hindi code-mixed social media content. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 974–979 (2014)