Handling Noisy Queries In Cross Language FAQ Retrieval

Danish Contractor Govind Kothari Tanveer A. Faruquie
L. Venkata Subramaniam Sumit Negi
IBM Research India
Vasant Kunj, Institutional Area
New Delhi, India
{dcontrac,govkotha,ftanveer,lvsubram,sumitneg}@in.ibm.com

Abstract

Recent times have seen a tremendous growth in mobile based data services that allow people to use Short Message Service (SMS) to access these data services. In a multilingual society it is essential that data services that were developed for a specific language be made accessible through other local languages also. In this paper, we present a service that allows a user to query a Frequently-Asked-Questions (FAQ) database built in a local language (Hindi) using Noisy SMS English queries. The inherent noise in the SMS queries, along with the language mismatch makes this a challenging problem. We handle these two problems by formulating the query similarity over FAQ questions as a combinatorial search problem where the search space consists of combinations of dictionary variations of the noisy query and its top-N translations. We demonstrate the effectiveness of our approach on a real-life dataset.

1 Introduction

There has been a tremendous growth in the number of new mobile subscribers in the recent past. Most of these new subscribers are from developing countries where mobile is the primary information device. Even for users familiar with computers and the internet, the mobile provides unmatched portability. This has encouraged the proliferation of information services built around SMS technology. Several applications, traditionally available on Internet, are now being made available on mobile devices using SMS. Examples include SMS short code services. Short codes are numbers where a short message in a predesignated format can be sent to get specific information. For example, to get the closing stock price of a particular share, the user has to send a message IBMSTOCKPR. Other examples are search (Schusteritsch et al., 2005), access to Yellow Page services (Kopparapu et al., 2007), Email [1], Blog [2], FAQ retrieval [3] etc. The SMS-based FAQ retrieval services use human experts to answer SMS questions.

Recent studies have shown that instant messaging is emerging as the preferred mode of communication after speech and email.[4] Millions of users of instant messaging (IM) services and short message service (SMS) generate electronic content in a dialect that does not adhere to conventional grammar, punctuation and spelling standards. Words are intentionally compressed by non-standard spellings, abbreviations and phonetic transliteration are used. Typical question answering systems are built for use with languages which are free from such errors. It is difficult to build an automated question answering system around SMS technology. This is true even for questions whose answers are well documented like in a Frequently-Asked-Questions (FAQ) database. Unlike other automatic question answering systems that focus on searching answers from a given text collection, Q&A archive (Xue et al., 2008) or the Web (Jijkoun et al., 2005), in a FAQ database the questions and answers are already pro-

[1] http://www.sms2email.com/
[2] http://www.letmeparty.com/
[3] http://www.chacha.com/
[4] http://www.whyconverge.com/

Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pages 87–96,
MIT, Massachusetts, USA, 9-11 October 2010.

टायफायड का बुखार कैसे फैलता है?
hwz typhoid fvr z sprd
सेवा उद्योग से आपका तात्पर्य क्या है?
wht du u mean bi srvce indstry
क्या एफआरआरओ कार्यालय में पंजीकरण अनिवार्य है?
is rgstrn at FRRO offce ncsry?

Figure 1: Sample SMS queries with Hindi FAQs

vided by an expert. The main task is then to identify the best matching question to retrieve the relevant answer (Sneiders, 1999) (Song et al., 2007). The high level of noise in SMS queries makes this a difficult problem (Kothari et al., 2009). In a multilingual setting this problem is even more formidable. Natural language FAQ services built for users in one language cannot be accessed in another language. In this paper we present a FAQ-based question answering system over a SMS interface that solves this problem for two languages. We allow the FAQ to be in one language and the SMS query to be in another.

Multi-lingual question answering and information retrieval has been studied in the past (Sekine and Grishman, 2003)(Cimiano et al., 2009). Such systems resort to machine translation so that the search can be performed over a single language space. In the two language setting, it involves building a machine translation system engine and using it such that the question answering system built for a single language can be used.

Typical statistical machine translation systems use large parallel corpora to learn the translation probabilities (Brown et al., 2007). Traditionally such corpora have consisted of news articles and other well written articles. Since the translation systems are not trained on SMS language they perform very poorly when translating noisy SMS language. Parallel corpora comprising noisy sentences in one language and clean sentences in another language are not available and it would be hard to build such large parallel corpora to train a machine translation system. There exists some work to remove noise from SMS (Choudhury et al., 2007) (Byun et al., 2008) (Aw et al., 2006) (Neef et al., 2007) (Kobus et al., 2008). However, all of these techniques require an aligned corpus of SMS and conventional language for training. Such data is extremely hard to create. Unsupervised techniques require huge amounts of SMS data to learn mappings of non-standard words to their corresponding conventional form (Acharyya et al., 2009).

Removal of noise from SMS without the use of parallel data has been studied but the methods used are highly dependent on the language model and the degree of noise present in the SMS (Contractor et al., 2010). These systems are not very effective if the SMSes contain grammatical errors (or the system would require large amounts of training data in the language model to be able to deal with all possible types of noise) in addition to misspellings etc. Thus, the translation of a cleaned SMS, into a second language, will not be very accurate and it would not give good results if such a translated SMS is used to query an FAQ collection.

Token based noise-correction techniques (such as those using edit-distance, LCS etc) cannot be directly applied to handle the noise present in the SMS query. These noise-correction methods return a list of candidate terms for a given noisy token (E.g. 'gud' $->$ 'god','good','guide') . Considering all these candidate terms and their corresponding translations drastically increase the search space for any multi-lingual IR system. Also , naively replacing the noisy token in the SMS query with the top matching candidate term gives poor performance as shown by our experiments. Our algorithm handles these and related issues in an efficient manner.

In this paper we address the challenges arising when building a cross language FAQ-based question answering system over an SMS interface. Our method handles noisy representation of questions in a source language to retrieve answers across target languages. The proposed method does not require hand corrected data or an aligned corpus for explicit SMS normalization to mitigate the effects of noise. It also works well with grammatical noise. To the best of our knowledge we are the first to address issues in noisy SMS based cross-language FAQ retrieval. We propose an efficient algorithm that can handle noise in the form of lexical and semantic corruptions in the source language.

2 Problem formulation

Consider an input SMS S^e in a source language e. We view S^e as a sequence of n tokens $S^e = s_1, s_2, \ldots, s_n$. As explained in the introduction, the input is bound to have misspellings and other lexical and semantic distortions. Also let $\mathcal{Q}^h$ denote the set

of questions in the FAQ corpus of a target language h. Each question $Q^h \in \mathcal{Q}^h$ is also viewed as a sequence of tokens. We want to find the question $\hat{Q^h}$ from the corpus $\mathcal{Q}^h$ that best matches the SMS S^e.

The matching is assisted by a source dictionary $\mathcal{D}^e$ consisting of clean terms in e constructed from a general English dictionary and a domain dictionary of target language $\mathcal{D}^h$ built from all the terms appearing in $\mathcal{Q}^h$. For a token s_i in the SMS input, term t_e in dictionary $\mathcal{D}^e$ and term t_h in dictionary $\mathcal{D}^h$ we define a *cross-lingual similarity measure* $\alpha(t_h, t_e, s_i)$ that measures the extent to which term s_i matches t_h using the clean term t_e. We consider t_h a *cross lingual variant* of s_i if for any t_e the cross language similarity measure $\alpha(t_h, t_e, s_i) > \epsilon$. We denote this as $t_h \sim s_i$.

We define a weight function $\omega(t_h, t_e, s_i)$ using the *cross lingual similarity measure* and the *inverse document frequency* (idf) of t_h in the target language FAQ corpus. We also define a scoring function to assign a score to each question in the corpus $\mathcal{Q}^h$ using the weight function. Consider a question $Q^h \in \mathcal{Q}^h$. For each token s_i, the scoring function chooses the term from Q^h having the maximum weight using possible clean representations of s_i; then the weight of the n chosen terms are summed up to get the score. The score measures how closely the question in FAQ matches the noisy SMS string S^e using the composite weights of individual tokens.

$$\text{Score}(Q^h) = \sum_{i=1}^{n} \max_{t_h \in Q^h, t_e \in \mathcal{D}^e \;\&\; t_h \sim s_i} \omega(t_h, t_e, s_i)$$

Our goal is to efficiently find the question $\hat{Q^h}$ having the maximum score.

3 Noise removal from queries

In order to process the noisy SMS input we first have to map noisy tokens in S^e to the possible correct lexical representations. We use a similarity measure to map the noisy tokens to their clean lexical representations.

3.1 Similarity Measure

For a term $t_e \in \mathcal{D}^e$ and token s_i of the SMS input S^e, the similarity measure $\gamma(t_e, s_i)$ between them is

$$\gamma(t_e, s_i) = \begin{cases} \frac{LCSRatio(t_e, s_i)}{EditDistance_{SMS}(t_e, s_i)} & \text{if } t_e \text{ and } s_i \text{ share same starting character *} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

Where $LCSRatio(t_e, s_i) = \frac{length(LCS(t_e, s_i))}{length(t_e)}$ and $LCS(t_e, s_i)$ is the *Longest common subsequence* between t_e and s_i.
** The intuition behind this measure is that people typically type the first few characters of a word in an SMS correctly. This way we limit the possible variants for a particular noisy token*

The *Longest Common Subsequence Ratio* (LCSRatio) (Melamed et al., 1999) of two strings is the ratio of the length of their LCS and the length of the longer string. Since in the SMS scenario, the dictionary term will always be longer than the SMS token, the denominator of LCSRatio is taken as the length of the dictionary term.

The $EditDistance_{SMS}$ (Figure 2) compares the Consonant Skeletons (Prochasson et al., 2007) of the dictionary term and the SMS token. If the Levenshtein distance between consonant skeletons is small then $\gamma(t_e, s_i)$ will be high. The intuition behind using $EditDistance_{SMS}$ can be explained through an example. Consider an SMS token "gud" whose most likely correct form is "good". The longest common subsequence for "good" and "guided" with "gud" is "gd". Hence the two dictionary terms "good" and "guided" have the same LCSRatio of 0.5 w.r.t "gud", but the $EditDistance_{SMS}$ of "good" is 1 which is less than that of "guided", which has $EditDistance_{SMS}$ of 2 w.r.t "gud". As a result the similarity measure between "gud" and "good" will be higher than that of "gud" and "guided". Higher the *LCSRatio* and lower the $EditDistance_{SMS}$, higher will be the similarity measure. Hence, for a given SMS token "byk", the similarity measure of word "bike" is higher than that of "break".

4 Cross lingual similarity

Once we have potential candidates which are the likely disambiguated representations of the noisy

Procedure $EditDistance_{SMS}(t_e, s_i)$
Begin
  return $LevenshteinDistance(CS(s_i), CS(t_e)) + 1$
End

Procedure *CS (t): // Consonant Skeleton Generation*
Begin
  Step 1. remove consecutive repeated characters in t
    // $(fall \rightarrow fal)$
  Step 2. remove all vowels in t
    //$(painting \rightarrow pntng, threat \rightarrow thrt)$
  return t
End

Figure 2: $EditDistance_{SMS}$

term, we map these candidates to appropriate terms in the target language. We use a statistical dictionary to achieve this cross lingual mapping.

4.1 Statistical Dictionary

In order to build a statistical dictionary we use the statistical translation model proposed in (Brown et al., 2007). Under IBM model 2 the translation probability of source language sentence $\bar{e} = \{t_e^1, \ldots, t_e^j, \ldots, t_e^m\}$ and a target language sentence $\bar{h} = \{t_h^1, \ldots, t_h^i, \ldots, t_e^l\}$ is given by

$$Pr(\bar{h}|\bar{e}) = \phi(l|m) \prod_{i=1}^{l} \sum_{j=0}^{m} \tau(t_h^i|t_e^j) a(j|i, m, l). \quad (2)$$

Here the word translation model $\tau(t_h|t_e)$ gives the probability of translating the source term to target term and the alignment model $a(j|i, m, l)$ gives the probability of translating the source term at position i to a target position j. This model is learnt using an aligned parallel corpus.

Given a clean term t_e^i in source language we get all the corresponding terms $\mathcal{T} = \{t_h^1, \ldots, t_h^k, \ldots\}$ from the target language such that word translation probability $\tau(t_h^k|t_e^i) > \varepsilon$. We rank these terms according to the probability given by the word translation model $\tau(t_h|t_e)$ and consider only those target terms that are part of domain dictionary i.e. $t_h^k \in \mathcal{D}^h$.

4.2 Cross lingual similarity measure

For each term s_i in SMS input query, we find all the clean terms t_e in source dictionary $\mathcal{D}^e$ for which similarity measure $\gamma(t_e, s_i) > \phi$. For each of these term t_e, we find the cross lingual similar terms $\mathcal{T}_{t_e}$ using the word translation model. We compute the cross lingual similarity measure between these terms as

$$\alpha(s_i, t_e, t_h) = \gamma(t_e, s_i).\tau(t_h, t_e) \quad (3)$$

The measure selects those terms in target language that have high probability of being translated from a noisy term through one or more valid clean terms.

4.3 Cross lingual similarity weight

We combine the idf and the cross lingual similarity measure to define the cross lingual weight function $\omega(t_h, t_e, s_i)$ as

$$\omega(t_h, t_e, s_i) = \alpha(t_h, t_e, s_i).idf(t_h) \quad (4)$$

By using idf we give preference to terms that are highly discriminative. This is necessary because queries are distinguished from each other using informative words. For example for a given noisy token "bck" if a word translation model produces a translation output "wapas" (as in came back) or "peet" or "qamar" (as in back pain) then idf will weigh "peet" more as it is relatively more discriminative compared to "wapas" which is used frequently.

5 Pruning and matching

In this section we describe our search algorithm and the preprocessing needed to find the best question $\hat{Q}^h$ for a given SMS query.

5.1 Indexing

Our algorithm operates at a token level and its corresponding cross lingual variants. It is therefore necessary to be able to retrieve all questions $Q_{t_h}^h$ that contain a given target language term t_h. To do this efficiently we index the questions in FAQ corpus using *Lucene*[5]. Each question in FAQ is treated as a document. It is tokenized using whitespace as delimiter before indexing.

[5]http://lucene.apache.org/java/docs/

The cross lingual similarity weight calculation requires the *idf* for a given term t_h. We query on this index to determine the number of documents f that contain t_h. The *idf* of each term in $\mathcal{D}^h$ is precomputed and stored in a hashtable with t_h as the key. The cross lingual similarity measure calculation requires the word translation probability for a given term t_e. For every t_e in dictionary D^e, we store $\mathcal{T}_{t_e}$ in a hashmap that contains a list of terms in the target language along with their statistically determined translation probability $\tau(t_h|t_e) > \varepsilon$, where $t_h \in \mathcal{D}^h$.

Since the query and the FAQs use terms from different languages, the computation of IDF becomes a challenge (Pirkola, 1998) (Oard et al., 2007). Prior work uses a bilingual dictionary for translations for calculating the IDF. We on the other hand rely on a statistical dictionary that has translation probabilities. Applying the method suggested in the prior work on a statistical dictionary leads to errors as the translations may themselves be inaccurate.

We therefore calculate IDFs for target language term (translation) and use it in the weight measure calculation. The method suggested by Oard et al (Oard et al., 2007) is more useful in retrieval tasks for multiple documents, while in our case we need to retrieve a specific document (FAQ).

5.2 List Creation

Given an SMS input string S^e, we tokenize it on white space and replace any occurrence of digits to their string based form (e.g. 4get, 2day) to get a series of n tokens $s_1, s_2, \ldots, s_n$. A list L_i^e is created for each token s_i using terms in the monolingual dictionary D^e. The list for a single character SMS token is set to *null* as it is most likely to be a stop word. A term t_e from D^e is included in L_i^e if it satisfies the threshold condition

$$\gamma(t_e, s_i) > \phi \tag{5}$$

The threshold value ϕ is determined experimentally. For every $t_e \in L_i^e$ we retrieve $\mathcal{T}_{t_e}$ and then retrieve the *idf* scores for every $t_h \in \mathcal{T}_{t_e}$. Using the word translation probabilities and the idf score we compute the cross lingual similarity weight to create a new list L_i^h. A term t_h is included in the list only if

$$\tau(t_h|t_e) > 0.1 \tag{6}$$

This probability cut-off is used to prevent poor quality translations from being included in the list.

If more than one term t_e has the same translation t_h, then t_h can occur more than once in a given list. If this happens, then we remove repetitive occurrences of t_h and assign it a weight equal to the maximum weight amongst all occurrences in the list, multiplied by the number of times it occurs. The terms t_h in L_i^h are sorted in decreasing order of their similarity weights. Henceforth, the term "list" implies a sorted list.

For example given a SMS query "hw mch ds it cst to stdy in india" as shown in Fig. 3, for each token we create a list of possible correct dictionary words by dictionary look up. Thus for token "cst" we get dictionary words lik "cost, cast, case, close". For each dictionary word we get a set of possible words in Hindi by looking at statistical translation table. Finally we merged the list obtained to get single list of Hindi words. The final list is ranked according to their similarity weights.

5.3 Search algorithm

Given S^e containing n tokens, we create n sorted lists $L_1^h, L_2^h, \ldots, L_n^h$ containing terms from the domain dictionary and sorted according to their cross lingual weights as explained in the previous section. A naive approach would be to query the index using each term appearing in all L_i^h to build a *Collection set* $\mathcal{C}$ of questions. The best matching question $\hat{Q}^h$ will be contained in this collection. We compute the score of each question in $\mathcal{C}$ using $Score(Q)$ and the question with highest score is treated as $\hat{Q}^h$. However the naive approach suffers from high runtime cost.

Inspired by the Threshold Algorithm (Fagin et al., 2001) we propose using a pruning algorithm that maintains a much smaller *candidate set* $\mathcal{C}$ of questions that can potentially contain the maximum scoring question. The algorithm is shown in Figure 4. The algorithm works in an iterative manner. In each iteration, it picks the term that has maximum weight among all the terms appearing in the lists $L_1^h, L_2^h, \ldots, L_n^h$. As the lists are sorted in the descending order of the weights, this amounts to picking the maximum weight term amongst the first terms of the n lists. The chosen term t_h is queried to find the set Q_{t_h}. The set Q_{t_h} is added to the candi-

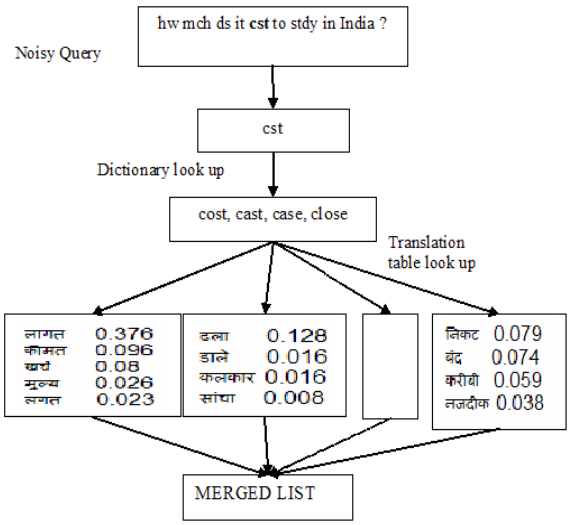

Figure 3: List creation

date set $\mathcal{C}$. For each question $Q \in Q_{t_h}$, we compute its score $Score(Q)$ and keep it along with Q. After this the chosen term t_h is removed from the list and the next iteration is carried out. We stop the iterative process when a thresholding condition is met and focus only on the questions in the candidate set $\mathcal{C}$. The thresholding condition guarantees that the candidate set $\mathcal{C}$ contains the maximum scoring question $\hat{Q}^h$. Next we develop this thresholding condition.

Let us consider the end of an iteration. Suppose Q is a question not included in $\mathcal{C}$. At best, Q will include the current top-most tokens $L_1^h[1], L_2^h[1], \ldots, L_n^h[1]$ from every list. Thus, the upper bound UB on the score of Q is

$$\text{Score}(Q) \leq \sum_{i=0}^{n} \omega(L_i^h[1]).$$

Let Q^* be the question in $\mathcal{C}$ having the maximum score. Notice that if $Q^* \geq \text{UB}$, then it is guaranteed that any question not included in the candidate set $\mathcal{C}$ cannot be the maximum scoring question. Thus, the condition "$Q^* \geq \text{UB}$" serves as the termination criterion. At the end of each iteration, we check if the termination condition is satisfied and if so, we can stop the iterative process. Then, we simply pick the question in $\mathcal{C}$ having the maximum score and return it.

Procedure Search Algorithm
Input: SMS $S = s_1, s_2, \ldots, s_n$
Output: Maximum scoring question $\hat{Q}^h$.
Begin
  $\forall s_i$, construct L_i^e for which $\gamma(s_i, t_e) > \epsilon$
    // L_i lists variants of s_i
  Construct lists $L_1^h, L_2^h, \ldots, L_n^h$ //(see Section 5.2).
    // L_i^h lists cross lingual variants of s_i in decreasing
    //order of weight.
  Candidate list $\mathcal{C} = \emptyset$.
  repeat
    $j^* = \text{argmax}_i \omega(L_i^h[1])$
    $t_h^* = L_{j^*}^h[1]$
      // t_h^* is the term having maximum weight among
      // all terms appearing in the n lists.
    Delete t_h^* from the list $L_{j^*}^h$.
    Retrieve $Q_{t_h^*}$ using the index
      // $Q_{t_h^*}$: the set of all questions in $\mathcal{Q}^h$
      //having the term t_h^*
    For each $Q \in Q_{t_h^*}$
      Compute $\text{Score}(Q)$ and
      add Q with its score into $\mathcal{C}$
    $\text{UB} = \sum_{i=1}^{n} \omega(L_i^h[1])$
    $\widehat{Q} = \text{argmax}_{Q \in \mathcal{C}} \text{Score}(Q)$.
    if $\text{Score}(\widehat{Q}) \geq \text{UB}$, **then**
        // Termination condition satisfied
      Output $\widehat{Q}$ and exit.
  forever
End

Figure 4: Search Algorithm with Pruning

6 Experiments

To evaluate our system we used noisy English SMS queries to query a collection of $10,000$ Hindi FAQs. These FAQs were collected from websites of various government organizations and other online resources. These FAQs are related to railway reservation, railway enquiry, passport application and health related issues. For our experiments we asked 6 human evaluators, proficient in both English and Hindi, to create English SMS queries based on the general topics that our FAQ collection dealt with. We found 60 SMS queries created by the evaluators, had answers in our FAQ collection and we designated these as the in-domain queries. To measure the effectiveness of our system in handling out of domain queries we used a total of 380 SMSes part of which were taken from the NUS corpus (How et al.,

whch metro statn z nr pragati maidan ?
dus metro goes frm airpot 2 new delhi rlway statn?
is dere any special metro pas 4 delhi uni students?
whn is d last train of delhi metro?
whr r d auto stands N delhi?

Figure 5: Sample SMS queries

2005) and the rest from the "out-of-domain" queries created by the human evaluators. Thus the total SMS query data size was 440. Fig 5 shows some of the sample queries.

Our objective was to retrieve the correct Hindi FAQ response given a noisy English SMS query. A given English SMS query was matched against the list of indexed FAQs and the best matching FAQ was returned by the Pruning Algorithm described in Section 5. A score of 1 was assigned if the retrieved answer was indeed the response to the posed SMS query else we assigned a score of 0. In case of out of domain queries a score of 1 was assigned if the output was NULL else we assigned a score of 0.

6.1 Translation System

We used the Moses toolkit (Koehn et al., 2007) to build an English-Hindi statistical machine translation system. The system was trained on a collection of 150,000 English and Hindi parallel sentences sourced from a publishing house. The 150,000 sentences were on a varied range of subjects such as news, literature, history etc. Apart from this the training data also contained an aligned parallel corpus of English and Hindi FAQs. The FAQs were collected from government websites on topics such as health, education, travel services etc.

Since an MT system trained solely on a collection of sentences would not be very accurate in translating questions, we trained the system on an English-Hindi parallel question corpus. As it was difficult to find a large collection of parallel text consisting of questions, we created a small collection of parallel questions using 240 FAQs and multiplied them to create a parallel corpus of 50,000 sentences. This set was added to the training data and this helped familiarize the language model and phrase tables used by the MT systems to questions. Thus in total the MT system was trained on a corpus of 200,000 sentences.

Experiment 1 and 2 form the baseline against which we evaluated our system. For our experiments the lexical translation probabilities generated by Moses toolkit were used to build the word translation model. In Experiment 1 the threshold ϕ described in Equation 5 is set to 1. In Experiment 2 and 3 this is set to 0.5. The Hindi FAQ collection was indexed using Lucene and a domain dictionary D^h was created from the Hindi words in the FAQ collection.

6.2 System Evaluation

We perform three sets of experiments to show how each stage of the algorithm contributes in improving the overall results.

6.2.1 Experiment 1

For Experiment 1 the threshold ϕ in Equation 5 is set to 1 i.e. we consider only those tokens in the query which belong to the dictionary. This setup illustrates the case when no noise handling is done. The results are reported in Figure 6.

6.2.2 Experiment 2

For Experiment 2 the noisy SMS query was cleaned using the following approach. Given a noisy token in the SMS query it's similarity (Equation 1) with each word in the Dictionary is calculated. The noisy token is replaced with the Dictionary word with the maximum similarity score. This gives us a clean English query.

For each token in the cleaned English SMS query, we create a list of possible Hindi translations of the token using the statistical translation table. Each Hindi word was assigned a weight according to Equation 4. The Pruning algorithm in Section 5 was then applied to get the best matching FAQ.

6.2.3 Experiment 3

In this experiment, for each token in the noisy English SMS we obtain a list of possible English variations. For each English variation a corresponding set of Hindi words from the statistical translation table was obtained. Each Hindi word was assigned a weight according to Equation 4. As described in Section 5.2, all Hindi words obtained from English variations of a given SMS token are merged to create

	Experiment 1	Experiment 2	Experiment 3
MRR Score	0.41	0.68	0.83

Table 1: MRR Scores

	F1 Score
Expt 1 (Baseline 1)	0.23
Expt 2 (Baseline 2)	0.68
Expt 3 (Proposed Method)	0.72

Table 2: F1 Measure

a list of Hindi words sorted in terms of their weight. The Pruning algorithm as described in Section 5 was then applied to get the best matching FAQ.

We evaluated our system using two different criteria. We used MRR (Mean reciprocal rank) and the best matching accuracy. Mean reciprocal rank is used to evaluate a system by producing a list of possible responses to a query, ordered by probability of correctness. The reciprocal rank of a query response is the multiplicative inverse of the rank of the first correct answer. The mean reciprocal rank is the average of the reciprocal ranks of results for a sample of queries Q.

$$MRR = 1/|Q| \sum_{i=1}^{Q} 1/rank_i \qquad (7)$$

Best match accuracy can be considered as a special case of MRR where the size of the ranked list is 1. As the SMS based FAQ retrieval system will be used via mobile phones where screen size is a major constraint it is crucial to have the correct result on the top. Hence in our settings the best match accuracy is a more relevant and stricter performance evaluation measure than MRR.

Table 1 compares the MRR scores for all three experiments. Our method reports the highest MRR of 0.83. Figure 6 shows the performance using the strict evaluation criterion of the top result returned being correct.

We also experimented with different values of the threshold for Score(Q) (Section 5.3). The ROC curve for various threshold is shown in Figure 7. The result for both in-domain and out-of-domain queries for the three experiments are shown in Figure 6 for Score(Q) = 8. The F1 Score for experiments 1, 2 and 3 are shown in Table 2.

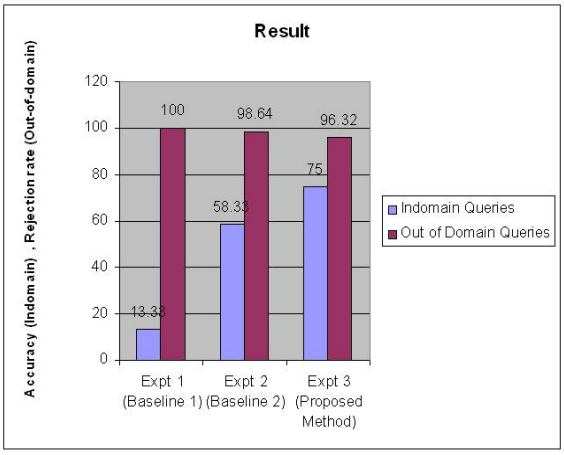

Figure 6: Comparison of results

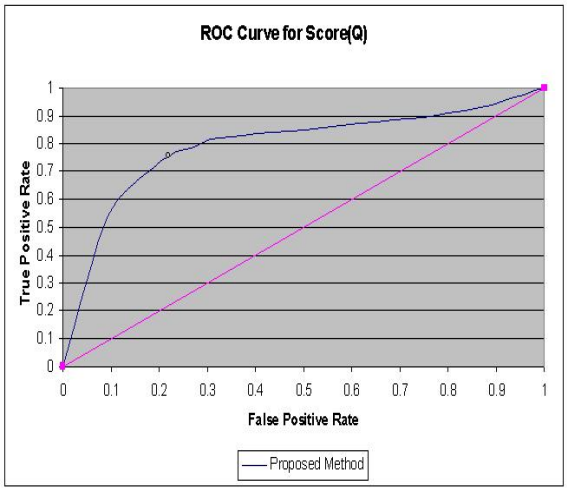

Figure 7: ROC Curve for $Score(Q)$

6.3 Measuring noise level in SMS queries

In order to quantify the level of noise in the collected SMS data, we built a character-level language model(LM) using the questions in the FAQ data-set (vocabulary size is 70) and computed the perplexity of the language model on the noisy and the cleaned SMS test-set. The perplexity of the LM on a corpus gives an indication of the average number of bits needed per n-gram to encode the corpus. Noise re-

		Cleaned SMS	Noisy SMS
English FAQ collection	bigram	16.64	55.19
	trigram	9.75	69.41

Table 3: Perplexity for Cleaned and Noisy SMS

sults in the introduction of many previously unseen n-grams in the corpus. Higher number of bits are needed to encode these improbable n-grams which results in increased perplexity. From Table 3 we can see the difference in perplexity for noisy and clean SMS data for the English FAQ data-set. Large perplexity values for the SMS dataset indicates a high level of noise.

For each noisy SMS query e.g. "hw 2 prvnt typhd" we manually created a clean SMS query "how to prevent typhoid". A character level language model using the questions in the clean English FAQ dataset was created to quantify the level of noise in our SMS dataset. We computed the perplexity of the language model on clean and noisy SMS queries.

7 Conclusion

There has been a tremendous increase in information access services using SMS based interfaces. However, these services are limited to a single language and fail to scale for multilingual QA needs. The ability to query a FAQ database in a language other than the one for which it was developed is of great practical significance in multilingual societies. Automatic cross-lingual QA over SMS is challenging because of inherent noise in the query and the lack of cross language resources for noisy processing. In this paper we present a cross-language FAQ retrieval system that handles the inherent noise in source language to retrieve FAQs in a target language. Our system does not require an end-to-end machine translation system and can be implemented using a simple dictionary which can be static or constructed statistically using a moderate sized parallel corpus. This side steps the problem of building full fledged translation systems but still enabling the system to be scaled across multiple languages quickly. We present an efficient algorithm to search and match the best question in the large FAQ corpus of target language for a noisy input question. We have demonstrated the effectiveness of our approach on a real life FAQ corpus.

References

Sreangsu Acharyya, Sumit Negi, L Venkata Subramaniam, Shourya Roy. 2009. Language independent unsupervised learning of short message service dialect. *International Journal on Document Analysis and Recognition*, pp. 175-184.

Aiti Aw, Min Zhang, Juan Xiao, and Jian Su. 2006. A phrase-based statistical model for SMS text normalization. *In Proceedings of COLING-ACL*, pp. 33-40.

Peter F. Brown, Vincent J.Della Pietra, Stephen A. Della Pietra, Robert. L. Mercer 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation *Computational Linguistics*, pp. 263-311.

Jeunghyun Byun, Seung-Wook Lee, Young-In Song, Hae-Chang Rim. 2008. Two Phase Model for SMS Text Messages Refinement. *AAAI Workshop on Enhanced Messaging*.

Monojit Choudhury, Rahul Saraf, Vijit Jain, Animesh Mukherjee, Sudeshna Sarkar, Anupam Basu. 2007. Investigation and modeling of the structure of texting language. *International Journal on Document Analysis and Recognition*, pp. 157-174.

Philipp Cimiano, Antje Schultz, Sergej Sizov, Philipp Sorg, Steffen Staab. 2009. Explicit versus latent concept models for cross-language information retrieval. *In Proceeding of IJCAI*, pp. 1513-1518.

Danish Contractor, Tanveer A. Faruquie, L. Venkata Subramaniam. 2010. Unsupervised cleansing of noisy text. *In Proceeding of COLING 2010: Posters*, pp. 189-196.

R. Fagin, A. Lotem, and M. Naor. 2001. Optimal aggregation algorithms for middleware. *In Proceedings of the 20th ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pp. 102-113.

Yijue How and Min-Yen Kan. 2005. Optimizing predictive text entry for short message service on mobile phones. *In M. J. Smith and G. Salvendy (Eds.) Proc. of Human Computer Interfaces International,Lawrence Erlbaum Associates*

Valentin Jijkoun and Maarten de Rijke. 2005. Retrieving answers from frequently asked questions pages on the web. *In Proceedings of the Tenth ACM Conference on Information and Knowledge Management,CIKM*, pp. 76-83.

Catherine Kobus, Francois Yvon and Grraldine Damnati. 2008. Normalizing SMS: Are two metaphors better than one? *In Proceedings of COLING*, pp. 441-448.

Philipp Koehn, Hieu Hoang, Alexandra Birch Mayne, Christopher Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, Evan Herbst 2007. Moses: Open source toolkit for statistical machine translation. *Annual Meeting of the Association for Computation Linguistics (ACL), Demonstration Session* .

Sunil Kumar Kopparapu, Akhilesh Srivastava and Arun Pande. 2007. SMS based Natural Language Interface

to Yellow Pages Directory. *In Proceedings of the 4th international conference on mobile technology, applications, and systems and the 1st international symposium on Computer human interaction in mobile technology*, pp. 558-563 .

Govind Kothari, Sumit Negi, Tanveer Faruquie, Venkat Chakravarthy and L V Subramaniam 2009. SMS based Interface for FAQ Retrieval. *Annual Meeting of the Association for Computation Linguistics (ACL)*.

I. D. Melamed. 1999. Bitext maps and alignment via pattern recognition. *Computational Linguistics*, pp. 107-130.

Guimier de Neef, Emilie, Arnaud Debeurme, and Jungyeul Park. 2007. TILT correcteur de SMS : Evaluation et bilan quantitatif. *In Actes de TALN*, pp. 123-132.

Douglas W. Oard, Funda Ertunc. 2002. Translation-Based Indexing for Cross-Language Retrieval *In Proceedings of the ECIR*, pp. 324-333.

A. Pirkola 1998. The Effects of Query Structure and Dictionary Setups in Dictionary-Based Cross-Language Information Retrieval *SIGIR '98: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* , pp. 55-63.

E. Prochasson, C. Viard-Gaudin, and E. Morin. 2007. Language models for handwritten short message services. *In Proceedings of the 9th International Conference on Document Analysis and Recognition*, pp. 83-87.

Rudy Schusteritsch, Shailendra Rao, Kerry Rodden. 2005. Mobile Search with Text Messages: Designing the User Experience for Google SMS. *In Proceedings of ACM SIGCHI*, pp. 1777-1780.

Satoshi Sekine, Ralph Grishman. 2003. Hindi-English cross-lingual question-answering system. *ACM Transactions on Asian Language Information Processing*, pp. 181-192.

E. Sneiders. 1999. Automated FAQ Answering: Continued Experience with Shallow Language Understanding Question Answering Systems. *Papers from the 1999 AAAI Fall Symposium*. Technical Report FS-99-02, AAAI Press, pp. 97-107.

W. Song, M. Feng, N. Gu, and L. Wenyin. 2007. Question similarity calculation for FAQ answering. *In Proceeding of SKG 07*, pp. 298-301.

X. Xue, J. Jeon, and W.B Croft. 2008. Retrieval Models for Question and Answer Archives. *In Proceedings of SIGIR*, pp. 475-482.