SMS Normalisation, Retrieval and Out-of-Domain Detection Approaches for SMS-Based FAQ Retrieval

Deirdre Hogan[1], Johannes Leveling[1], Hongyi Wang[1], Paul Ferguson[2], and Cathal Gurrin[2]

[1] Centre for Next Generation Localisation (CNGL)
School of Computing
Dublin City University (DCU)
Dublin 9, Ireland
[2] CLARITY Research Centre
School of Computing
Dublin City University (DCU)
Dublin 9, Ireland
{dhogan,jleveling,pferguson,hwang,cgurrin}@computing.dcu.ie

Abstract. This paper gives an overview of DCU's participation in the SMS-based FAQ Retrieval task at FIRE 2011. DCU submitted three runs for monolingual English experiments. The approach consisted of first transforming the noisy SMS queries into a normalised, corrected form. The normalised queries were then used to retrieve a ranked list of FAQs by combining the results from three different retrieval methods. Finally, using information from the retrieval results, out-of-domain (OOD) queries were identified and tagged. The results of our best run on the final test set were the highest of all 13 participating teams. Our FIRE submission retrieved 70.2% in-domain query answers correctly and 85.6% identified out-of-domain queries correctly.

1 Introduction

This paper describes the participation of Dublin City University (DCU) in the FIRE 2011 evaluation for the SMS-based FAQ Retrieval Task. The task consisted of retrieving the correct answer to an incoming SMS question from an English FAQ consisting of questions and answers on a variety of different topics from career advice to popular Indian recipes. The incoming queries were written in noisy SMS *"text speak"* and contained many misspellings, abbreviations and grammatical errors. Some SMS queries were out-of-domain and had no corresponding FAQ answer in the collection. Such queries needed to be identified and flagged as an out-of-domain (OOD) result before returning *"NONE"* as an answer string.

Figure 1 gives an overview of the DCU system, which can be broken down into three distinct steps: SMS normalisation, retrieval of ranked results, and identifying out of domain query results.

P. Majumder et al. (Eds.): FIRE 2010 and 2011, LNCS 7536, pp. 184–196, 2013.

The first step involves normalising words in the SMS text so that they more closely resemble the text in the FAQ data set (e.g. with correct and standardised spelling). This is achieved by generating a set of candidate corrections for SMS tokens using rules extracted from a mixture of annotated and unannotated corpora. The most likely token substitution, given the context, is then selected from the set of candidates. This step is detailed in Section 2.

For the second step in the process we experimented with different retrieval engines and approaches (i.e. Lucene, Solr and a simple word overlap metric) to retrieve ranked lists of candidate answers from the FAQ, given the normalised query. The retrieval results were combined to produce a single ranked list of question answer pairs. This step is described in more detail in Section 3.

In a final step, outlined in Section 4, we identified likely out-of-domain (OOD) questions using a filtering mechanism based on a combination of evidence from the results of the retrieval engines. For in-domain (ID) questions, the top answers from the combined list were returned; for OOD questions, *"NONE"* was returned. We present test set results in Section 5, before concluding and giving an outlook on planned future work in Section 6.

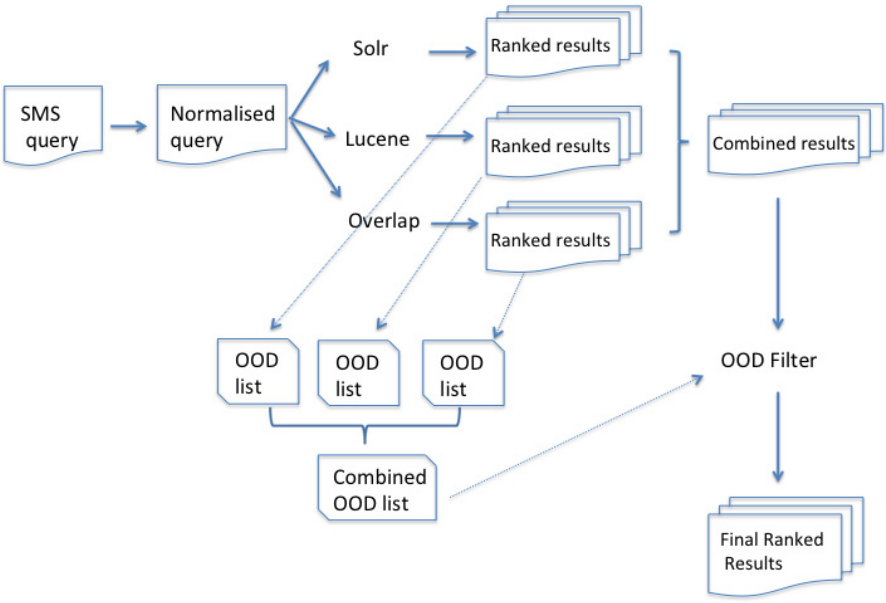

Fig. 1. Data flow diagram of DCU system

2 SMS Normalisation and Correction

The irregular spelling and abbreviations in SMS questions leads to poor retrieval performance due to mismatches between terms in the SMS and terms in the FAQ

text. An SMS normalisation and correction step will thus increase the chance of finding correct answers. Our initial idea to correct SMS messages was to train a statistical MT system, similar to the approach described in [1]. However, *"text speak"* or *"textese"* is productive and often generates new ill-formed or non-standard words which increase the out-of-vocabulary problem of statistical machine translation. Furthermore, training data in large enough quantities to train an accurate machine translation system for SMS correction does not exist and it is time-consuming to construct such data manually. Due to the lack of training data we decided against this approach and implemented a heuristic token substitution approach to correct SMS tokens.

The approach we took involved first carrying out some basic normalisation steps on both SMS queries and FAQ documents (described in Section 2.1). The SMS data then went through a correction step, where candidate corrections for SMS tokens were generated and then the candidate correction with the highest score was chosen. This process is outlined in detail in Section 2.2.

Table 1. Preprocessing steps

Token type	Example	Action	Description
contraction	*"I'll"* → *"I_will"*	expand	rules extracted from PoS-tagged Brown Corpus and annotated SMS corpus
interjection	*"Eeh"* → *"_"*	remove	rules extracted from manual annotation and PoS-tagged Brown Corpus
spelling variant	*"center"* → *"centre"*	normalise	dictionary-based spelling normalisation from AE to BE
acronyms	*"M.Sc."* → *"MSc"*	normalise	words of more than 50% uppercase characters and full stops, extracted from FIRE FAQ data and EN-1M corpus
spelling error	*"Austrailia"* → *"Australia"*	correct	most frequent spelling errors extracted from English Wikipedia
concatenation	*"12ft"* → *"12_ft"*	split	monetary values and measurements

2.1 Pre-processing for Documents and Queries

FAQ documents and SMS questions underwent the same preprocessing steps, which consisted of text normalisation as shown in Table 1. As SMS text can contain non-standard tokens, we adapted tokenisation to allow for digits in a word (e.g. *"2gether"*), and split character sequences of words (typically measurement units) and numbers (e.g. *"12ft"*).

2.2 SMS Correction

We employed three different techniques to generate candidate corrections for SMS tokens. These are described in Section 2.2. One of these techniques involved using correction rules extracted from corpora of hand annotated microtext data. The generation of these corpora is described in the next section.

Generating Training Data for SMS Correction and Normalisation. We created training data to use for automatically correcting SMS queries by manually annotating different microtext corpora (SMS and Tweets). The original text messages and the annotation were aligned on the level of tokens, so that there is a one-to-one correspondence between original token and corrected token. In order to preserve this one-to-one alignment, if necessary one or more tokens were joined together by underscore (e.g. *"I'll"* → *"I_will"*)

Table 2. Twenty most frequent corrections in FIRE preview and training questions

Rank	Word	Correction	Frequency	Rank	Word	Correction	Frequency
1	*"d"*	*"the"*	194	11	*"n"*	*"and"*	37
2	*"2"*	*"to"*	147	12	*"gt"*	*"get"*	32
3	*"hw"*	*"how"*	146	13	*"whch"*	*"which"*	29
4	*"r"*	*"are"*	108	14	*"bst"*	*"best"*	24
5	*"wht"*	*"what"*	101	15	*"fr"*	*"for"*	22
6	*"4"*	*"for"*	82	16	*"frm"*	*"from"*	22
7	*"f"*	*"of"*	71	17	*"wt"*	*"what"*	22
8	*"cn"*	*"can"*	63	18	*"wrld"*	*"world"*	21
9	*"wat"*	*"what"*	50	19	*"s"*	*"is"*	20
10	*"whr"*	*"where"*	38	20	*"watz"*	*"what_is"*	19

We manually annotated the following corpora with the corrected, normalised forms:

1. FIRE SMS training questions (1071 questions)
2. FIRE SMS preview questions (456 questions)
3. All SMS messages containing a question mark extracted from the NUS SMS Corpus [2] (3786 questions); the corpus was created at the National University of Singapore and consists of about 10000 SMS messages collected by students[1].
4. Tweets (549 messages) from CAW 2.0 - Content Analysis for the WEB 2.0[2]

Table 2 shows the top twenty corrections in the FIRE SMS QA training data.

[1] `http://www.comp.nus.edu.sg/~rpnlpir/downloads/corpora/smsCorpus/`
[2] `http://caw2.barcelonamedia.org/node/`

Generation and Selection of Correct Tokens. Our SMS correction approach is token-based. First all tokens are pre-processed as outlined in Table 1. Then each token in the SMS query is examined in turn to decide if it remains unchanged. Stopwords, punctuation, numerals, and acronyms are not modified. For each remaining token, a set of correction candidates is generated and the best candidate in the context is selected as a correction. The candidate corrections are generated by combining lists of candidates obtained from the following methods.

Correction rules: The manually corrected SMS questions were employed to extract correction rules and their corresponding frequencies, which are then used to generate the first list of candidate corrections. If applicable correction rules are found for a token, the frequencies of the rules in the annotated data are used to calculate normalised weights for each correction. For example, the token *"2"* can be corrected into *"two"*, *"too"*, or *"to"*, with *"to"* being the most frequent (see Table 2).

Consonant skeletons: An additional set of candidate corrections was created using consonant skeletons. We used two background corpora, the English 1M sentence corpus[3] from the Leipzig Corpora Collection (EN-1M), and the FIRE FAQ corpus used in the SMS QA track (43,871 sentences). Each token in the background corpora is processed to obtain its consonant skeleton [3], a shorter form of the word with all vowels removed (for example, *"rsdnt"* is the consonant skeleton for *"resident"*). The mapping between consonant skeletons and words is used to obtain additional correction candidates for question words that match a consonant skeleton.

Clippings: Finally, candidates are generated by looking up all words in the background corpora that have the same prefix as the question word to identify truncated or clipped words, e.g. *"exam"* → *"examination"* or *"lab"* → *"laboratory"*.

These methods yield lists of replacement candidates, which are merged by adding up their weights (derived from their term frequency in the background corpora). For each of the top twenty candidates, a token score (similar to a probability) is computed based on the so-called 'stupid' backoff [4] for 3-grams with $\alpha = 0.4$ (see Equation 1). We used the background corpora to collect n-gram statistics. The candidate with the maximum product of weight and n-gram score is selected as the token correction. Equation 1 shows the n-gram score where w_a^b is the n-gram ($n = b - a$) of tokens between position a and b and $f(w_a^b)$ is the frequency of the n-gram in the corpus.

$$S(w_i|w_{i-k+1}^{i-1}) = \begin{cases} \frac{f(w_{i-k+1}^{i})}{f(w_{i-k+2}^{i-1})} & \text{if } f(w_{i-k+1}^{i}) > 0 \\ \alpha S(w_i|w_{i-k+2}^{i-1}) & \text{otherwise} \end{cases} \quad (1)$$

[3] http://corpora.uni-leipzig.de/

2.3 Evaluation of SMS Correction

Table 3 shows results for our SMS correction approach applied to the FIRE SMS QA training and preview questions. When testing on preview and training sets, we excluded the correction rules generated from corresponding annotated SMS data. Note that the results for the training data are actually much lower than expected because many of the token correction rules were missing. In contrast, the results for the preview data might be too high because there is an overlap in training and preview data. Note also that correcting to a wrong word form, i.e. an incorrect surface form with the same stem as the correct token, is counted as an error (e.g. correcting to *"resident"* instead of *"residents"*).

Table 3. Performance of SMS normalisation on FIRE preview and training data

	Count	Correct	Incorrect
Training sentences	1071	156 (15%)	915 (85%)
Training tokens	8432	6246 (74%)	2186 (26%)
Preview sentences	456	152 (33%)	304 (67%)
Preview tokens	5087	4546 (89%)	541 (21%)

In these tests, stopwords were among the most frequent errors in the SMS normalisation. For example, *"r"* was often replaced with *"are"* instead of *"or"*. However, these errors will not affect retrieval performance when stopwords are removed from the IR query.

3 Retrieval Engines

Before conducting our retrieval experiments, both SMS queries and FAQ documents were preprocessed as described in Section 2.1. The SMS queries then went through a further correction step (Section 2.2).

We experimented with three different retrieval methods, Lucene, Solr and a simple similarity metric (Term overlap) based on the number of overlapping words between query and document, and achieved the best performance by combining the outputs from the different systems.

We report results for experiments based on indexing FAQ questions, FAQ answers, and both questions and answers. The metric used is the in-domain score, calculated as:

$$\frac{count(\text{correct results in first ranked position})}{count(\text{queries with corresponding FAQ answers})} \quad (2)$$

3.1 Experimental Details

Search Engines. We experimented with two full-text search engines, Lucene and Solr, initially because we wanted a comparison of the ease of use of the two engines for this new task. Although Solr uses Lucene as its underlying search engine, we found it difficult to exactly replicate the results from both engines and found that Lucene consistently gave us better results on the training set.

We adapted both Lucene and Solr to use the BM25 ranking function [5] (with parameters $b = 0.75$, $k_1 = 1.2$, and $k_3 = 7$) and experimented with different stopword lists but otherwise used the default settings.

In our submission to the FIRE challenge, the SMART stopwords[4] were used for the Lucene experiments. We have since found much better results on the training set using Lucene's (much smaller) default stopword list and, indeed, get the best results on the training set by using no stopword list at all for Lucene. However, these improvements did not carry over to the Test set.

Table 4 and Table 5 show the results achieved by Solr and Lucene respectively on the in-domain queries of the training set. We display results for three different indexes (questions only, answers only and questions and answers). The numbers in the tables denote the accuracy (fraction of correct in all correct answers) and the absolute number of correct results (in brackets). Indexing the questions gives the best results. This is unsurprising given that the text of the corrected SMS queries is often very similar to the matching FAQ question. In Table 5, for point of comparison, an additional row displays the Lucene score on the question index when the SMART stopword list was used. For all other rows in Table 5, no stopword list was used.

In both tables, results are given for the original, unaltered *"textese"* SMS queries (raw), the automatically corrected queries (auto-correct) and the hand-corrected version of the SMS queries (gold).

Table 4. Training set comparison of **Solr-BM25** in-domain results when indexing questions only, answers only and both questions and answers. Total number of in-domain queries is 701.

	SMS question type		
Indexing	raw	auto-correct	gold
Questions	38.37 (269)	**72.04 (505)**	**72.46 (508)**
Answers	39.08 (274)	66.48 (466)	66.76 (468)
Questions & Answers	**39.66 (278)**	71.89 (504)	72.03 (505)

Term Overlap (Overlap). In addition to Solr and Lucene, we used a simple overlap metric as a baseline, The term overlap uses a text similarity score to rank results based on the number of matching terms in the query and each FAQ question.

[4] `ftp://ftp.cs.cornell.edu/pub/smart/`

Table 5. Training set comparison of **Lucene-BM25** in-domain results when indexing questions only, answers only and both questions and answers. Total number of in-domain queries is 701.

	SMS question type		
Indexing	raw	auto-correct	gold
Questions (SMART)	50.07 (351)	77.46 (543)	78.17 (548)
Questions	**57.06 (400)**	**80.88 (567)**	**80.03 (561)**
Answers	15.41 (108)	22.97 (161)	22.82 (160)
Questions & Answers	43.08 (302)	72.33 (507)	72.75 (510)

Table 6. Training set comparison of **Term overlap** in-domain results when indexing questions only, answers only and both questions and answers. Total number of in-domain queries is 701.

	SMS question type		
Indexing	raw	auto-correct	gold
Questions	**46.22 (324)**	**72.18 (506)**	**76.61 (509)**
Answers	5.56 (39)	7.99 (56)	8.13 (57)
Questions & Answers	2.11 (148)	3.22 (226)	3.30 (231)

The similarity score between two texts is calculated as a normalised score based on the number of words the two texts have in common. The (F1) score (Equation 3) is between 0 and 1 and is scaled based on the length of the strings.

$$\text{F1} = Sim(text_1, text_2) = \frac{2 * precision * recall}{precision + recall} \tag{3}$$

where

$$precision = \frac{count(\text{overlapping terms})}{count(\text{terms in } text_1)} \tag{4}$$

and

$$recall = \frac{count(\text{overlapping terms})}{count(\text{terms in } text_2)} \tag{5}$$

Software for finding the overlaps between two strings and calculating this text similarity score can be downloaded from CPAN[5]. Table 6 displays in-domain results for the overlap metric on the training set.

[5] `http://search.cpan.org/~tpederse/Text-Similarity/`

Combining Results. The best overall results on the training set (and subsequently on the test set) were achieved by combining the results from the three different retrieval methods. The result sets were combined using a mechanism whereby each search result x is associated with a score $S_{combined}(x)$ which is a weighted sum of the individual normalised scores from each retrieval engine (see Equation 6).

$$S_{combined}(x) = w_s * S_{Solr}(x) + w_l * S_{Lucene}(x) + w_o * S_{Overlap}(x) \quad (6)$$

The weights were determined by manual fine-tuning on the training set. The final weight settings chosen were: $w_l = 0.6$ (Lucene), $w_o = 0.3$ (Overlap), $w_s = 0.1$ (Solr).

Training set in-domain scores for the combined results, indexing questions only, are displayed in Table 7. For ease of comparison we repeat in Table 7 the results achieved by the individual retrieval mechanisms. Combining results gives a marginal increase in score for the automatically corrected queries, and leads to a slight drop for raw and gold queries. Although we used the combined results in our FIRE submission, we now conclude that the benefits of combining are small and using Lucene as the sole retrieval mechanism would simplify the system considerably while still delivering comparably good performance.

Table 7. Training set comparison of in-domain results for three different retrieval engines, indexing questions only. Total number of in-domain queries is 701. The numbers in brackets correspond to the number of correct results.

	SMS question type		
Retrieval method	raw	auto-correct	gold
Lucene	57.06 (400)	80.88 (567)	80.03 (561)
Solr	38.37 (269)	72.04 (505)	72.46 (508)
Overlap	46.22 (324)	72.18 (506)	76.61 (509)
Combined	**57.20 (401)**	**81.46 (571)**	**80.88 (567)**

4 Filtering Out-of-Domain Queries

For each retrieval method we produced a list of SMS queries which were predicted to be out-of-domain.

Solr. In order to generate the list of OOD queries from Solr, we used the same approach that was used by [6] for determining the number of relevant documents to use for query expansion. This approach produces a score based on the inverse document frequency (*idf*) component of BM25 for each query. This essentially disregards the term frequency and document length components which, since the queries are reasonably short, tend to be less important:

$$score(q,d) = \sum_{t \epsilon q} log\left(\frac{N - df_t + 0.5}{df_t + 0.5}\right) \quad (7)$$

Using this approach we can calculate the maximum possible score for any document as the sum of the *idf* scores for all of the query terms: any document containing all the query terms will have this maximum score. We then use a threshold to determine if a query should be considered as OOD. Here we choose to add a query to the OOD list if its score is below 70% of the maximum score.

Lucene. TiMBL [7] implements a memory-based learning approach and supports different machine learning algorithms. For the experiments described in this paper, the IB1 approach (similar to k-nearest-neighbours approach) was employed to train a classifier distinguishing between OOD queries and ID questions. The features for the training instances include query performance estimates, result set size, and document score. The query performance estimates used are: Average Inverse Collection Term Frequency (AvICTF) [8], Simplified Query Clarity Score (SCS) [9], and an estimate derived from the similarity score between collection and query (SumSCQ, AvSCQ, MaxSCQ) [10]. In addition the (unnormalised) BM25 document scores [5] for the top five documents were employed as features.

This classifier achieved 78% accuracy (835 out of 1071 correctly classified instances) on the FAQ SMS training data. Table 8 shows true positives (TP), false negatives (FN) etc. per class, using leave-one-out validation.

Table 8. Scores per question class

Class	TP	FP	TN	FN	F-Score
ID	459	111	376	125	0.80
OOD	376	125	459	111	0.76

Term Overlap. In this approach, the list of OOD queries was predicted based on the number of terms in each incoming query and the number of overlapping terms between incoming query and the highest ranked question from the FAQ (For example, if the incoming query consists of more than one term and has only one term in common with the highest ranked FAQ question, then classify the query as out-of-domain). The heuristic algorithm was fine-tuned on the training set and optimised to maximise both out-of-domain and in-domain accuracy.

Combining OOD Results. Based on experiments on the training set data, we found that combining the OOD lists through simple majority voting led to the best results. Table 9 displays the best in-domain and out-of-domain results achieved on the training set. The OOD score is calculated as:

$$\frac{count(\text{OOD queries correctly identified})}{count(\text{all OOD queries})} \quad (8)$$

Table 9. Training set - 3 final run configurations. ID and OOD results after applying OOD filtering for auto-corrected queries.

		SMS query type								
		raw			auto			gold		
Retrieval	Index	ID	OOD	all	ID	OOD	all	ID	OOD	all
Combined	Q	52.06	**69.18**	**57.98**	**73.32**	**69.19**	**71.90**	**73.03**	**69.19**	**71.71**
Lucene	Q	**52.07**	61.62	55.37	72.61	**69.19**	71.43	71.90	**69.19**	70.96
Lucene	Q+A+QA	**52.07**	62.43	55.65	72.61	69.19	71.43	71.89	**69.19**	70.96

The weights used for combining different retrieval results are as follows: Weights for *Combined* (combining Lucene, Solr and Overlap results): $w_l = 0.5$ (Lucene), $w_s = 0.2$ (Solr), $w_o = 0.3$ (Overlap). Weights for *Lucene Q/A/QA* (combining Lucene results on different indexes): Q: 0.7, A: 0.1, QA: 0.2.

5 Test Set Retrieval Experiments and Results

Table 10. Results on the test set for the normalised queries, as submitted to the FIRE evaluation

Run	Index	Retrieval	ID Correct	OOD Correct	all	MRR
1	Q	Combined	**0.70 (494/704)**	**0.86 (2311/2701)**	**82.38**	**0.896**
2	Q	Lucene	0.67 (472/704)	0.86 (2310/2701)	81.70	0.865
3	Q+A+QA	Lucene	0.68 (477/704)	**0.86 (2311/2701)**	81.88	0.873

Table 11. Latest results on the test set. ID and OOD Results after applying the OOD filtering.

		SMS query type								
		raw			auto			gold		
Retrieval	Index	ID	OOD	all	ID	OOD	all	ID	OOD	all
Combined	Q	**63.35**	**84.93**	**80.47**	**70.57**	84.80	**81.94**	**70.31**	**84.86**	**81.85**
Lucene	Q	62.78	**84.93**	80.35	70.31	**84.89**	81.88	70.03	**84.86**	81.79
Lucene	Q+A+QA	62.78	**84.93**	80.35	70.31	**84.89**	81.79	70.03	**84.86**	81.79

The DCU team submitted three runs for the English monolingual task. Table 10 details the results for the three DCU runs. For the first run, only the question text from the FAQ was indexed. The final ranked results list was produced by combining the individual ranked lists from the three different retrieval approaches as described in Section 3. For the second and third run, only the

Lucene search engine was used. In the second run, FAQ question text was indexed whereas in the final run, both question and answer text from the FAQ was indexed.

Table 11 gives our results when no stopword list was used with the Lucene retrieval. Unlike on the training set, where results improved considerably, the combined results for the test set are slightly lower than previously. However, results for using only the Lucene search engine are much the same whether or not stoplists are used.

6 Conclusion and Future Work

Our submission achieved the best performance in the official results of the FIRE 2011 SMS-based FAQ monolingual English retrieval task.

We found that the best retrieval results for an individual method are obtained when using Lucene-BM25 scoring. While the combination of approaches for retrieval and OOD detection increases the number of correct results marginally, we conclude that the benefits of combining are small and using Lucene as the sole retrieval mechanism simplifies the system considerably while still delivering comparably good performance.

As part of future work, we want to simplify the system further by using a single OOD detection approach, rather than combining OOD lists generated from different retrieval methods.

Acknowledgments. This research is supported in part by the Science Foundation Ireland (Grant 07/CE/I1142) as part of the Centre for Next Generation Localisation (CNGL) project.

References

1. Aw, A., Zhang, M., Xiao, J., Su, J.: A phrase-based statistical model for SMS text normalization. In: COLING/ACL 2006, pp. 33–40 (2006)
2. How, Y., Kan, M.Y.: Optimizing predictive text entry for short message service on mobile phones. In: Human Computer Interfaces International (HCII 2005). Lawrence Erlbaum (2005)
3. Kothari, G., Negi, S., Faruquie, T.A., Chakaravarthy, V.T., Subramaniam, L.V.: SMS based interface for FAQ retrieval. In: ACL/IJNLP 2009, pp. 852–860 (2009)
4. Brants, T., Popat, A.C., Xu, P., Och, F.J., Dean, J.: Large language models in machine translation. In: EMNLP-CoNLL, pp. 858–867. ACL (2007)
5. Robertson, S.E., Walker, S., Jones, S., Beaulieu, M.M.H., Gatford, M.: Okapi at TREC-3. In: Harman, D.K. (ed.) TREC-3, pp. 109–126. NIST, Gaithersburg (1995)
6. Ferguson, P., O'Hare, N., Lanagan, J., Smeaton, A.F., Phelan, O., McCarthy, K., Smyth, B.: CLARITY at the TREC 2011 Microblog Track. In: Text Retrieval Conference, TREC (2011)
7. Daelemans, W., Zavrel, J., van der Sloot, K., van den Bosch, A.: TiMBL: Tilburg memory based learner, version 6.2, reference guide. Technical Report 09-01, ILK (2004)

8. Kwok, K.L.: A new method of weighting query terms for ad-hoc retrieval. In: SIGIR 1996, pp. 187–195. ACM, New York (1996)
9. He, B., Ounis, I.: Inferring query performance using pre-retrieval predictors. In: Apostolico, A., Melucci, M. (eds.) SPIRE 2004. LNCS, vol. 3246, pp. 43–54. Springer, Heidelberg (2004)
10. Zhao, Y., Scholer, F., Tsegay, Y.: Effective pre-retrieval query performance prediction using similarity and variability evidence. In: Macdonald, C., Ounis, I., Plachouras, V., Ruthven, I., White, R.W. (eds.) ECIR 2008. LNCS, vol. 4956, pp. 52–64. Springer, Heidelberg (2008)