IRIT at INEX: Question Answering Task

Liana Ermakova, Josiane Mothe

Institut de Recherche en Informatique de Toulouse
118 Route de Narbonne, 31062 Toulouse Cedex 9, France
liana.ermakova.87@gmail.com, josiane.mothe@irit.fr

Abstract. In this paper we describe an approach to tweet contextualization developed in the context of INEX QA track. The task is to provide a context up to 500 words to a tweet. The summary should be an extract from the Wikipedia. Our approach is based on the index which includes not only lemmas, but also named entities. Sentence retrieval is based on standard TF-IDF measure enriched by named entity recognition, POS weighting and smoothing from local context.

Keywords: Information retrieval, summarization, extraction, contextual information, smoothing, part of speech tagging, named entity.

1 Introduction

This paper describes the approach we developed at IRIT in the framework of the Question answering track (QA@INEX) of INEX (Initiative for the Evaluation of XML Retrieval). In 2011 the track aims at evaluating tweet contextualization in terms of relevance of the retrieved information to tweets and readability of the presented results. There are 132 tweets which include the title and the first sentence of a New York Times articles. The summary should be made of extracted relevant sentences from a local XML dump of English Wikipedia (April 2011), totally 3 217 015 non-empty pages [1].

In the method we developed, firstly we parsed tweets and articles with Stanford CoreNLP[1] and we looked for documents similar to queries. We computed indices for sentences. Then we searched for relevant sentences using standard TF-IDF measure enriched by named entity recognition, part-of-speech weighting and smoothing from local context.

The idea to contextualize short text like microblogs or tweets is quite recent [2]. However, in [2] a tweet is mapped into a set of Wikipedia articles and a summary is not provided. Summaries are either "extracts", if they contain the most important sentences extracted from the original text, or "abstracts", if these sentences are re-written or paraphrased, generating a new text [3]. There exits two general approaches to text summarization, namely statistical methods and linguistic ones. Apparently, the first article on automated summarization was published in 1958 [4]. H. P. Luhn pro-

1 http://nlp.stanford.edu/software/corenlp.shtml

posed order sentences by the number of the most frequent meaningful words. This approach was extended by taking into account sentence position in the text, key word and key phrase occurrence etc. [5] [6]. In case of a subject related summary, the query may be expanded e.g. by synonyms [7]. CORTEX combines such metrics as word frequency, overlap with query terms, entropy the words, shape of text etc. [8]. LexRank underlies DISQ algorithm, where special attention is paid to redirects on the Wikipedia pages [9]. Sentence importance may be computed from text energy matrix [10] [11]. Text corpora provide much useful information on features which should be kept in a summary, how long a text should be etc. [12]. Linguistic methods fall into several categories: (1) rule-based approaches, which may be combined with statistics [13] [12], (2) methods based on genre features, text structure etc. [5] [12] [14] and (3) methods based on syntax analysis [14].

The paper is organized as follows. Firstly we describe our approach. Then evaluation results are provided. Future development description concludes the paper.

2 Method Description

2.1 Preprocessing

We first looked for the documents similar to the queries. For this stage, document retrieval was performed by the Terrier Information Retrieval Platform[2], an open-source search engine developed by the School of Computing Science, University of Glasgow. To this end we transformed queries into to the format accepted by Terrier. We used the default settings for Terrier. We applied the BasicIndexer with the Porter stemmer [15] and the default list of stopwords. Text was converted to lowercase before parsing. There were no limits to the maximum number of tokens indexed for a document. We chose the Ponte and Croft’s language model [16]. During document retrieval words with low IDF were ignored. For query expansion we used Rocchio algorithm with the parameter 0.4 [17]. The number of top-ranked documents to be considered in the pseudo relevance set was equal to 3 and the number of the highest weighted terms to be added to the original query was set to 10. A term was considered to be informative if it was found no less than in two documents[3].

The next stage was parsing of tweets and retrieved texts by Stanford CoreNLP developed by the Stanford Natural Language Processing Group. CoreNLP integrates such tools as POS tagger, named entity recognizer, parser and the co-reference resolution system[4]. It uses the Penn Treebank tag set [18]. In our approach, tweets were transformed into queries with POS tagging and recognized named entities. It allows taking into account different weights for different tokens within a query, e.g. NE are considered to be more important than common nouns; nouns are more significant than verbs; punctuation marks are not valuable, …

[2] http://terrier.org/
[3] http://terrier.org/docs/v2.2.1/properties.html
[4] http://nlp.stanford.edu/software/corenlp.shtml

2.2 Sentence Retrieval

The general idea of the proposed approach is to compute similarity between the query and sentences and to retrieve the most similar passages. To this end we used standard TF-IDF measure. We extended this approach by adding weight coefficients to POS, NE, headers, sentences from abstracts, and definitional sentences. Moreover sentence meaning depends on the context. Therefore we used an algorithm for smoothing from the local context which will be described later. The sentences were sorted by their similarity scores. The sentences with the highest score were added to the summary until the total number of words exceeds 500. In the implemented system there is a possibility to choose one of the following similarity measures: cosine, Dice and Jaccard similarity [17]. We took into account only lexical vocabulary overlap between a query and a document. However it is possible also to consider morphological and spelling variants, synonyms, hyperonyms, …

Different words should not have the same weight, e.g. usually it is better not to take into account stop-words. Our system provides several ways to assign score to words. The first option is to identify stop-words by frequency threshold. The second way is to assign different weights to different parts of speech. One can specify whether vector components should be multiplied by this POS rank, e.g. determiners have zero weight, proper names have the highest weight equal to 1.0, and nouns have greater weight than verbs, adjectives and adverbs. Another option gives a possibility to consider or not IDF.

NE comparison is hypothesized to be very efficient for contextualizing tweets about news. Therefore for each NE in queries we searched corresponding NE in the sentences. If it is found, the whole similarity measure is multiplied by NE coefficient computed by the formula:

$$NE_{COEF} = weight(NE) \times \frac{NE_{common}+1}{NE_{query}+1} \qquad (1)$$

where $weight(NE)$ is floating point parameter given by a user (by default it is equal to 1.0), NE_{common} is the number of NE appearing in both query and sentence, NE_{query} is the number of NE appearing in the query. We used Laplace smoothing to NE by adding one to the numerator and the denominator. The sentence may not contain a NE from the query and it can be still relevant. However, if smoothing is not performed the coefficient will be zero. NE recognition is performed by Stanford CoreNLP. We considered only the exact matches of NE. Synonyms were not identified. However, it may be done later applying WordNet, which includes major NE.

We consider that *Headers*, *labels*, …. should not be taken into account since they are not "good" sentences for summarization. Therefore we assign them lower weights. Stanford parser allows making distinction between auxiliary verbs and main verbs, personal and impersonal verb forms. We assumed that such kinds of sentences do not have personal verbs. One of the settings allows assigning weights to sentences without personal verb forms. By default this parameter is equal to 0. Sentences with personal verb forms have the weight equal to 1.0. It is possible to give smaller weights to sections than to abstracts. By default we assume that sections have the weight equal to

0.8 and for abstracts this parameter is 1.0. We assumed that definitional sentences are extremely important to contextualizing task. Therefore they should have higher weights. We have taken into account only definitions of NE by applying the following linguistic pattern: $< NE >< Be_{pers} >< NounPhrase >$, where Be_{pers} is a personal form of the verb *to be*. Noun phrase recognition is also performed by Stanford parser. We considered only sentences that occurred in abstracts since they contain more general and condensed information and usually include definitions in the first sentence. However, the number of extracted definitions was quite small and therefore we did not use them in our runs.

Since sentences are much smaller than documents, general IR systems provide worse results to sentence retrieval. Moreover, document retrieval systems are based on the assumption that the relevant document is about the query. However this is not enough for sentence retrieval, e.g. in QA systems the sentence containing the answer is much more relevant that the sentence which is about the subject. General approach to document IR is underlined by TF-IDF measure. In contrast, usually the number of each query term in a sentence is no more than one [19]. Traditionally, sentences are smoothed by the entire collection, but there exist another approach namely smoothing from local context [19]. This method assigns the same weight to all sentences from the context. In contrast, we assume that the importance of the context reduces as the distance increases. So, the nearest sentences should produce more effect on the target sentence sense than others. For sentences with the distance greater than k this coefficient is zero. The total of all weights should be equal to one. The system allows taking into account k neighboring sentences with the weights depending on their remoteness from the target sentence. In this case the total target sentence score R_t is a weighted sum of scores of neighboring sentences r_i and the target sentence r_0 itself:

$$R_t = \sum_{i=-k}^{k} w_i \times r_i \qquad (2)$$

$$w_i = \begin{cases} \frac{1-w_t}{k+1} \times \frac{k-|i|}{k}, 0 < |i| \le k \\ w_t, i = 0 \\ 0, |i| > k \end{cases} \qquad (3)$$

$$\sum_{i=-k}^{k} w_i = 1 \qquad (4)$$

where w_t is a target sentence weight set by a user, w_i are weights of the sentences from k context. The weights become smaller as the remoteness increases. If the sentence number in left or right context is less than k, their weights are added to the target sentence weight w_t. This allows keeping the sum equal to one. By default, $k = 1$, target sentence weight is equal to 0.8.

3 Evaluation

For the first run we used default settings (default), namely: NE were considered with a coefficient 1.0; abstract had weight equal to 1.0, sections had score 0.8; headers, labels, … were not taken into account; we removed stop-words; cosine similarity was applied; POS were ranked; each term frequency was multiplied by IDF. In the second run we changed the similarity measure to Dice similarity (07_2_07_1_dice). The section weight was reduced to 0.7. The context was extended to two sentences in each direction and the target sentence weight was equal to 0.7. For NE we kept the weight equal to 1.0. In the third run we applied Jaccard similarity measure (05_2_07_1_jac) and we set the weight to sections equal to 0.5.

Evaluation was performed manually by conference organizers [1]. Passages were judged as relevant or not without context. The summaries submitted by participants were compared to each other, to the baseline summary made of sentences (baselinesum) and to the key terms (baselinemwt). The baseline system was based on Indri index without stop word list and stemming (language model). Part of speech tagging was performed by TreeTagger. Summarization algorithm was TermWatch [1].

Since the task was to provide the context to the tweets and therefore found passages should be somehow similar to the original New York Times articles, firstly, obtained results were compared with them. Then, the overlap with relevant passages evaluated manually was computed. N-gram distribution of summaries, namely unigram distribution, bigram distribution and bigram distribution with two word gap, was compared with those from relevant passages and New York Times articles [1]. So, the comparison with two different relevant collections was performed.

In order to evaluate the informative level of summaries the simple log difference was used, since it is less sensitive to smoothing on the given collection than the Kullback-Leibler divergence [1]. **Table 1** presents the comparison of baseline systems and the submitted runs with regards to New York Times articles. All three runs are ranked higher than baseline systems. The best result is given by 05_2_07_1_jac.

Table 1. Log difference to New York Times articles

Ranking	Unigram	Bigram	With 2-gap	Average	Run
0.104925	0.0447	0.076644	0.104925	0.076629	05_2_07_1_jac
0.104933	0.044728	0.076659	0.104933	0.076646	07_2_07_1_dice
0.104937	0.044739	0.076668	0.104937	0.076653	default
0.10646	0.046049	0.078101	0.10646	0.078084	Baselinesum
0.10766	0.047508	0.079385	0.10766	0.079387	Baselinemwt

Table 2 provides comparison referring to the pool of relevant sentences. According to these evaluations, all submitted runs are more relevant than baselines. However, the best results were provided by the run with the default settings. We think that the opposite evaluation results obtained for NYT and the pool of relevant passages from the Wikipedia may be explained by the different language models of these collections. The pool of the relevant sentences from the Wikipedia contained 103 889 tokens, which gave a vocabulary of 19 037 words, and the original news articles with a vocabulary of 26 481 words contained 154 355 tokens [1]. So, the average word frequency differs for 9%. Moreover, these two corpora have different genres and conse-

quently different structure. In our approach NE matching was extremely important and therefore we preferred to select sentences with proper nouns, but not pronouns and other type of references (e.g. American President instead of Barack Obama). In a news article authors try not to repeat themselves and they substitute NE by other words. Since relevant passages were selected without context, the majority of them tended to contain NE. Thus, there exist two main explanations of the opposite ranks: different language models of the collections and the pool peculiarities.

Table 2. Log difference with the set of relevant passages

Ranking	Unigram	Bigram	With 2-gap	Average	Run
0.105506	0.048639	0.07867	0.105506	0.078697	default
0.105747	0.048781	0.078857	0.105747	0.07889	07_2_07_1_dice
0.106195	0.049083	0.079249	0.106195	0.079277	05_2_07_1_jac
0.114346	0.053691	0.085915	0.114346	0.085881	Baselinesum
0.117854	0.055786	0.088604	0.117854	0.088701	Baselinemwt

The readability evaluation was also performed manually. Assessors should indicate if a passage contained one of the following drawbacks: syntactical problems (e.g. bad segmentation), unresolved anaphora, redundant information (that is to say, the information is already mentioned) or the passage is meaningless in the given context (trash). The total score was the average normalized number of words in valid passages [1]. Though the system showed the best results according the relevance judgment, it was worse than the baseline in terms of readability. The major drawback was unresolved anaphora. Trash passages refer not only to readability, but also to relevance. Therefore relevance improvement and sentence reordering may solve this problem.

4 Conclusion

In this article we describe a method to tweet contextualization on the basis of the local Wikipedia dump. Firstly, we looked for relevant Wikipedia pages using the search engine Terrier. Secondly, the input tweets and the found documents were parsed by Stanford CoreNLP. After that, a new index for sentences was constructed. It includes not only stems but also NE. Then we searched for relevant sentences. To this end similarity between the query and sentences was computed using an extended TF-IDF measure. We enhance the basic approach by adding weight coefficients to POS, NE, headers, sentences from abstracts, and definitional sentences. Moreover, the algorithm for smoothing from local context is provided. We assume that the importance of the context depends on the remoteness from the target sentence. So, the nearest sentences should produce more effect on the target sentence sense than others. Remote sentences (with the distance greater than k) should not be taken into account. The sentences with the highest score are added to the summary until the total number of words exceeds 500.

Relevance evaluation provides evidence that the approach is better than the baselines underlined by language model. All runs are closer to the original New York Times articles and contain more relevant passages. The run with default settings is the most relevant. However, the run based on Jaccard coefficient and reduced weight for sections gave results more similar to the original New York Times articles. This can be explained by different language models and by the features of the pool of the relevant passages. In terms of relevance the developed system was the first among 11 systems, but in terms of readability it was only the third [1].

Future work includes solving anaphora problems, sentence ordering, additional features selection and applying different similarity measure, e.g. expanded by synonyms and relations from WordNet. This should increase relevance as well as readability.

5 References

1. SanJuan, E., Moriceau, V., Tannier, X., Bellot, P., Mothe, J.: Overview of the INEX 2011 Question Answering Track (QA@INEX). In Focused Retrieval of Content and Structure, 10th International Workshop of the Initiative for the Evaluation of XML Retrieval (INEX 2011) (2011)
2. E. Meij, W.: Adding Seman¬tics to Microblog Posts. Proceedings of the fifth ACM international conference on Web search and data mining (2012)
3. Vivaldi, J., Cunha, I., Ramırez, J.: The REG summarization system at QA@INEX track 2010. (2010)
4. Luhn, H.: The automatic creation of literature abstracts. IBM Journal of Research and Development, 159-165 (April 1958)
5. Seki, Y.: Automatic Summarization Focusing on Document Genre and Text Structure. ACM SIGIR Forum 39(1), 65-67 (2005)
6. Erkan, G., Radev, D.: LexRank: Graph-based Lexical Centrality as Salience in Text Summarization. Journal Of Artificial Intelligence Research 22, 457-479 (2004)
7. Soriano-Morales, E.-P., Medina-Urrea, A., Sierra, G., Mendez-Cruz, C.-F.: The GIL-UNAM-3 summarizer: an experiment in the track QA@INEX'10. (2010)
8. Torres-Moreno, J.-M., Gagnon, M.: The Cortex automatic summarization system at the QA@INEX track 2010. (2010)
9. Cabrera-Diego, L., Molina, A., Sierra, G.: A Dynamic Indexing Summarizer at the QA@INEX 2011 track. INEX 2011 Workshop Pre-proceedings, 154-159 (2011)
10. Linhares, A., Velazquez, P.: Using Textual Energy (Enertex) at QA@INEX track 2010. (2010)
11. Torres-Moreno, J.-M., Velazquez-Morales, P., Gagnon, M.: The Cortex and Enertex summarization systems at the QA@INEX track 2011., 196-205 (2011)

12. Lin, C.-Y., Hovy, E.: Identifying Topics by Position. Proceedings of the fifth conference on Applied natural language processing, 283-290 (1997)
13. Lin, C.-Y.: Assembly of Topic Extraction Modules in SUMMARIST. In AAAI Spring Symposium on Intelligent Text Summarisation (1998)
14. Barzilay, R., McKeown, K., Elhadad, M.: Information fusion in the context of multi-document summarization. ACL '99 Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics, 550-557 (1999)
15. Porter, M.: An algorithm for suffix stripping. In : Readings in information retrieval. Morgan Kaufmann Publishers Inc., San Francisco (1997)
16. Ponte, J., Croft, W.: A language modeling approach to information retrieval. Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval (1998)
17. Manning, C., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press (2008)
18. Marcus, M., Santorini, B., Marcinkiewicz, M.: Building a large annotated corpus of English: the Penn Treebank. Computational Linguistics 19(2) (1993)
19. Murdock, V.: Aspects of Sentence Retrieval. Dissertation (2006)