Efficient and Effective Query Expansion for Web Search

Claudio Lucchese
University of Venice, Italy
claudio.lucchese@unive.it

Franco Maria Nardini
ISTI-CNR, Pisa, Italy
francomaria.nardini@isti.cnr.it

Raffaele Perego
ISTI-CNR, Pisa, Italy
raffaele.perego@isti.cnr.it

Roberto Trani
ISTI-CNR and University of Pisa, Italy
roberto.trani@isti.cnr.it

Rossano Venturini
University of Pisa, Italy
rossano.venturini@di.unipi.it

ABSTRACT

Query Expansion (QE) techniques expand the user queries with additional terms, e.g., synonyms and acronyms, to enhance the system recall. State-of-the-art solutions employ machine learning methods to select the most suitable terms. However, most of them neglect the cost of processing the expanded queries, thus selecting effective, yet very expensive, terms. The goal of this paper is to enable QE in scenarios with tight time constraints proposing a QE framework based on structured queries and efficiency-aware term selection strategies. In particular, the proposed expansion selection strategies aim at capturing the efficiency and the effectiveness of the expansion candidates, as well as the dependencies among them. We evaluate our proposals by conducting an extensive experimental assessment on real-world search engine data and public TREC data. Results confirm that our approach leads to a remarkable efficiency improvement w.r.t. the state-of-the-art: a reduction of the retrieval time up to 30 times, with only a small loss of effectiveness.

CCS CONCEPTS

• **Information systems → Query reformulation**; *Retrieval effectiveness*; *Retrieval efficiency*;

KEYWORDS

Query Expansion; Effectiveness Efficiency trade off; Term selection;

ACM Reference Format:
Claudio Lucchese, Franco Maria Nardini, Raffaele Perego, Roberto Trani, and Rossano Venturini. 2018. Efficient and Effective Query Expansion for Web Search. In *The 27th ACM International Conference on Information and Knowledge Management (CIKM '18), October 22–26, 2018, Torino, Italy*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3269206.3269305

1 INTRODUCTION

Web documents and Web search queries have a very different nature. While Web documents are written with different purposes and using a huge variety of writing styles, users express their information needs through just a few terms by using their own vocabulary and their expertise of interaction with the search system. The phenomenon, commonly known as the *vocabulary mismatch problem* [5], is largely affected by synonymy and polysemy of natural languages. While synonymy and word inflections may induce the mismatch of relevant documents, polysemy may cause the retrieval of irrelevant results. *Query Expansion* (QE) is a well-known research field investigating techniques to expand the query with additional terms that can better convey the search intent of the user, thus alleviating the vocabulary mismatch problem and improving the retrieval effectiveness [3]. Our work stems from the observation that most of the literature on QE focuses on the effectiveness, partially neglecting the query processing efficiency. Nevertheless, efficiency is a primary concern for Web search companies as it strongly affects the users satisfaction [1]. To enable QE in query processors with strict latency constraints, we explore efficiency-effectiveness trade-offs in selecting expansion terms having maximal impact on the effectiveness and minimal impact on the retrieval efficiency.

Pseudo-Relevance Feedback (PRF) methods are QE techniques that received attention in the past [8, 10, 17]. They exploit the terms distribution in the top-ranked documents initially retrieved to find useful expansions to use for a second retrieval. However, the expansion terms devised by PRF methods have not one-to-one relation with the original query terms, thus forcing the use of the OR operator to combine the expansion terms and the original query terms. It significantly increases the number of matching documents, negatively impacting the query processing time. Ad-hoc *indexing* and *documents optimization* strategies [4, 16] can only partially address such inefficiency issues. Thesaurus-based methods are another family of QE techniques, which expand the query with related concepts chosen from controlled sources [12, 14]. Some studies observe that expansions generated by PRF and thesaurus methods can be noisy or even harm the effectiveness [2, 19], thus supervised learning techniques were proposed to select only the most effective expansions among the candidate ones [2, 9, 10, 18].

Two noteworthy works that address QE efficiency are [11, 18]. Macdonald *et al.* [11] focus on the selection of different query rewritings. Specifically, they aim to predict the execution time of the query rewritings generated by different QE methods and then select the rewriting from the best-on-average source whose predicted cost is below a certain time budget. Zhang *et al.* [18] argue that, within the PRF framework, the time spent to extract the features used by the supervised term selection models has a notable impact on the end-to-end latency. Therefore, they propose a feature selection framework to select only useful, yet inexpensive, features for term selection. To the best of our knowledge, our work is the first that explicitly focuses on efficiency-aware term selection strategies.

CIKM '18, October 22–26, 2018, Torino, Italy

ACM ISBN 978-1-4503-6014-2/18/10…$15.00
https://doi.org/10.1145/3269206.3269305

Table 1: Comparison of retrieval performance of 3,000 queries evaluated on a collection of 51M Web documents.

Query Type	Ret. Time (ms)	Ret. Documents	Recall
OR	361	8,419,221	0.984
AND	36	143,290	0.790
PRF - 3 exp	832	15,939,754	0.990
CNF - 3 exp	174	288,327	0.882

We argue that effective and efficient QE can be obtained using *structured expanded queries* expressed in *Conjunctive Normal Form* (CNF), i.e., a conjunction of disjunctions where each disjunction contains a query token and its synonyms as provided by a given thesaurus. To confirm this hypothesis, in Table 1 we report the average performance of an analysis conducted on 3,000 queries randomly sampled from a Web search engine query log and processed on a collection of 51M Web documents. First, we compare the performance of processing the queries using a *disjunctive* (OR) or a *conjunctive* (AND) logic. As expected, the AND operator provides much faster retrieval (10x speed-up w.r.t. OR) with a limited recall (0.79). Conversely, the OR operator is more demanding, but it guarantees a recall of 0.98 by retrieving one-sixth of the documents on average. Table 1 also reports the second retrieval performance of a PRF method. It adds to the original query the best three terms selected by the Relevance Model [8] on the Wikipedia corpus, as suggested by [17]. PRF, which is a state-of-the-art method exploiting disjunctive query expansions, achieves a recall of 0.99 but it also shows the highest retrieval time (832 ms) due to the tremendous number of documents retrieved (one-third of the index). It confirms the effectiveness of PRF but it also suggests that web-scale search engines can hardly use PRF due to its prohibitive cost. The last row of Table 1 preliminarily shows that processing CNF queries having three expansions is five times faster than PRF and achieves a recall of 0.88. Hence, CNF queries can provide both effective and efficient expanded queries if the expansion terms are carefully selected.

In this paper, we propose a novel QE framework based on a thesaurus to produce effective and efficient CNF queries. It aims at overcoming the PRF weakness highlighted above by exploiting cost-aware supervised models to select effective yet efficient expansion terms. We also introduce a novel pruning strategy to remove expansions that may be harmful, to produce smaller and more efficient CNF queries. It adaptively decides the number of expansion terms to use by considering the cost of adding one or more terms so to trade off the two aspects directly during the expansion.

We assess the performance of the proposed framework on a real-world dataset and on public TREC data. Results show that our framework outperforms a state-of-the-art competitor [2] achieving a remarkable efficiency improvement: up to one order of magnitude reduction of the retrieval time with only a small effectiveness loss.

2 SELECTING QUERY EXPANSIONS

We now introduce our novel QE framework. First, we describe how we built the thesaurus and the tokenization process used to generate the list of candidate expansions. Second, we introduce the metric that the term selection models use to jointly optimize efficiency and effectiveness during the construction of the expanded CNF query. Finally, we describe the methodology used to optimize this trade-off metric during the query expansion process.

Thesaurus. We built the synonyms thesaurus used in this work by exploiting a combination of public resources. Specifically, we employ the synonym dictionary of OpenOffice[1] extended with plurals nouns. The inclusion of entity synonyms has a significant impact on the QE effectiveness, hence we further extend the thesaurus with Wikipedia redirects titles and aliases extracted from the most recent Wikipedia dump[2]. The resulting knowledge base is composed of about 150K dictionary entries and 12M entity entries.

Query tokenization and candidates generation. Several thesaurus entries may appear as query n-grams, hence query tokenization is critical to identify the most appropriate thesaurus entries and their synonyms (e.g., "New York City"). To this end, we employ the simple and effective dynamic algorithm proposed by [6] to find the best tokenization among all the possible ones. Then, we remove the stopwords and, lastly, for each token we extract the list of candidate expansions from the thesaurus.

Tradeoff metric. Inspired by Wang *et al.* [15], our framework employs the *Efficiency-Effectiveness Tradeoff* (EET) metric to jointly evaluate and optimize the trade-off between efficiency (σ) and effectiveness (γ) of an CNF query Q:

$$\mathrm{EET}(Q) = \frac{\gamma(Q) \cdot \sigma(Q)}{\gamma(Q) + \sigma(Q)}$$

We use the *Step + Exponential Decay* function as efficiency metric $\sigma(Q)$. It maps the query execution time $\tau(Q)$ in $[0, 1]$ and demotes the queries whose retrieval time exceeds a given time threshold t, a common practice in the Web search scenario [7], according to an exponential decay function with constant α:

$$\sigma(Q) = \begin{cases} 1 & \text{if } \tau(Q) \leq t \\ \exp(\alpha \cdot (\tau(Q) - t)) & \text{otherwise} \end{cases}$$

Hereinafter, we assume a cascade ranking architecture where QE aims at maximizing the number of relevant documents matched (i.e., Recall), and the later stages aim at optimizing the list-wise metrics (e.g., NDCG or MAP). Therefore, we set $\gamma(Q) = \mathrm{Recall}(Q)$.

Term Selection Models. Our general methodology for efficient and effective CNF query expansion assumes to have for each query token a list of candidate expansions generated by a thesaurus-based method. The two strategies detailed below allow: i) to select up to k expansion terms for the original query, and ii) to exploit the EET metric to choose the best candidate expansions to use.

Static Selection (S2) casts the term selection problem into a ranking problem. Given a query and a list of its candidate expansions, S2 ranks the candidates according to the EET gain that each expansion individually brings to the original query and then selects the top-k terms to create the expanded CNF query. S2 could directly predict the EET gain of each candidate (regression) and then rank the list according to these gains, but a recent study [18] shows that ranking models produce better results than regression ones when applied to term selection. The reason is that ranking models focus on the order of the terms rather than on predicting the exact gain of all of them, which is a more difficult task. Our experiments using regression models, not reported here for brevity, confirm this finding. We build the gold standard for training the ranking

[1] https://www.openoffice.org/lingucomponent/thesaurus.html
[2] https://dumps.wikimedia.org/enwiki/latest/enwiki-latest-pages-articles.xml.bz2

model of S2 by using a set of training queries as follows: for each query Q we generate its candidate expansions, then we represent each query-expansion pair $\langle Q, e \rangle$ through pre-retrieval features and we label it with the EET gain that the expansion term brings to the user query, i.e., $EET(Q \cup e) - EET(Q)$. Differently from [2, 18], we label the expansion terms by using EET and not MAP, thus considering efficiency besides effectiveness. S2 "statically" ranks all the candidates at once, thus it does not consider the dependencies among the selected terms. For instance, S2 may select two close expansions (e.g., "*New York*" and "*New York City*") of the same query token (e.g., "*N Y C*") even if the selection of one of them already covers all the relevant documents matched by the other one.

Sequential Greedy Selection (SGS) captures the term dependencies during the selection process by iteratively selecting one expansion at a time, thus by taking into account the previously selected terms. SGS starts from the original query and the whole set of candidate expansions and performs the following steps: (1) ranks the candidates expansions according to the expected EET gain, (2) selects the best expansion, adds it to the CNF query and removes it from the list of candidates, (3) repeat steps (1-2) until k terms are selected. The gold standard for training SGS is built performing the same three steps above on the training queries by ranking the candidates according to the true EET gains. Then, within the step (1) we gather all the query-expansion training pairs that we represent and we label using the same features and labels used by S2.

Pruning Expansion Terms. Expanding all the queries with the same number of terms is surely not the best solution because not all the queries have the same difficulty. To this end, we introduce a pruning phase to identify and prune the terms selected by the term selection strategy that cannot further improve the target EET metric, i.e., have a negative EET gain. In the case of S2, we apply the pruning to all the selected terms, and we remove those that are expected to decrease the trade-off metric. Instead, in the case of SGS, we apply the pruning at the end of each iteration on the term just selected, and we stop the iterative process if it prunes such term. We implement this pruning step as a classification task. There is a clear dependency between the selection and the pruning, hence we build the gold standard for training these classifiers by applying the respective selection models to the training queries, then by gathering all the query-expansion pairs selected by the selection strategy. We represent each pair using the same features of the previous models and label it with a binary value stating if the expansion improves the EET metric of the query or not.

Our QE strategy is made up of two distinct phases: while the term selection strategy aims at identifying the best expansions among the candidates, the pruning phase is in charge of deciding whether the selected terms are eligible to be used or not. There is a strict relation between the selection and the pruning models because the latter focuses only on the terms selected by the former. This separation of roles makes the two aforementioned models more effective than only one regressor that uses the predicted EET gain to select-and-prune the terms. Indeed, the term selection model learns to rank the candidate terms, while the pruning model learns to classify only the selected ones. We conducted experiments proving the advantage of using the selection and pruning models in place of a single all-in-one model, but they are not reported here for brevity.

3 EXPERIMENTS

In this section, we show a detailed experimental assessment of the proposed QE techniques[3]. We first describe the dataset and the external resources used for the assessment, then we present a comprehensive performance analysis of our proposal.

Datasets. We conducted the experimental assessment using two different datasets. We built the first dataset, hereinafter called QED, by randomly sampling 24K unique queries from the MSN query log. This set of queries was used to collect 550K documents by querying a main Web search engine and fetching the content of the top results, that were further checked by a pool of assessors. Thus, we produced a set of positive results for the queries that were used to extend the ClueWeb09-B collection, composed of 51M Web pages. To identify only the queries that may be impacted by QE we discarded the queries showing maximum recall without QE. Then, we selected 13K of the remaining queries and divided them into training, validation and test sets (70%-15%-15%). We used the first set to learn the models employed by the different expansion strategies and the second one to select the best learning parameters according to the target metric. Then, we evaluated all the techniques on the test set and, to assess their generalization ability, we also tested the models trained on QED on the TREC 2009/10/11/12 Web track datasets. We discarded 7 queries returning no relevant results in ClueWeb09-B and 3 generic queries having very high retrieval time even without expansions. At the end of this process, the TREC dataset used for the evaluation is composed of 190 queries.

Index. The two document collections used to evaluate the performance of S2 and SGS were indexed by using the public implementation of *Partitioned Elias-Fano Indexes*[4] [13]. The code was extended to evaluate CNF queries. Furthermore, since some thesaurus entries correspond to n-grams, e.g., "new york", "wall street", etc., we indexed also the posting lists of these n-grams. The retrieval efficiency is assessed by using a single core of a machine equipped with an AMD Opteron 6276 processor and 128 GiB of memory.

Models and Features. At indexing time, for each thesaurus term, we precomputed some collection-based statistics, such as doc frequency, term frequency and co-occurrence frequencies with the other terms. Then, we redefined the pre-retrieval features used by Zhang *et al.* [18] to better capture the terms-relations within the CNF queries, e.g., by considering co-occurrences of the expansion with the sibling terms. We implemented these features by using the previous statistics, and we enriched them with textual features, such as common-prefix length, edit distance, and others. The models were trained with the Gradient Boosting Trees (GBT) implementation available in XGBoost[5]. The term selection models were trained using a pairwise loss function minimizing the number of wrong pairs inside the ranked lists. Conversely, the pruning models were trained using a binary logistic loss to minimize the number of misclassified pruned terms. We tested different supervised models, such as SVM-based or Neural Network-based, and we also tried GBT models with different loss functions. We obtained the best results with the combinations mentioned above, but we do not report here the results obtained with the other models for brevity.

[3] Source code: https://github.com/hpclab/efficient-query-expansion
[4] https://github.com/ot/ds2i
[5] http://xgboost.readthedocs.io/

Table 2: Performance comparison on the QED dataset.

	Time (ms)	Recall	EET
NoEXP	26	0.652	0.747
StaticRecall [2]	890	0.855	0.537
S2	473	0.851	0.576
SGS	294	0.847	0.649
S2 & pruning	84	0.802	0.823
SGS & pruning	60	0.802	0.853

Table 3: Performance comparison on the TREC dataset.

	Time (ms)	Recall	EET
NoEXP	34	0.891	0.922
StaticRecall [2]	1951	0.964	0.376
S2	656	0.960	0.449
SGS	600	0.959	0.488
S2 & pruning	70	0.937	0.921
SGS & pruning	68	0.942	0.935

Experimental Results. We compare the performance of our term selection strategies with the state-of-the-art model proposed by Cao *et al.* [2]. For a fair comparison all models have been embedded into our framework, which uses the thesaurus described above to tokenize and expand the queries into CNF. All models use the same set of features and select exactly 5 terms, except for the strategies using pruning. We employed the EET metric using the *Step + Exponential Decay* function with step $t = 200$ms and decay factor $\alpha = -0.01$. We choose a step of 200ms because it is a common per-query time budget used by real-world search engines [7].

Table 2 and 3 report the results of this comparison on the QED test set and TREC datasets. The first row refers to the performance of the original queries, properly tokenized, but not expanded. *StaticRecall* refers to the instantiation of [2] into our framework, which uses a ranking model to select the terms according to the Recall metric. This technique boosts the recall but produces very expensive queries - from 30 to 60 times slower than the original ones. It is reflected by the low EET achieved by this method, which is much worse than the one of NoEXP. S2 uses the ranking approach and the EET metric to select the expansion terms. It achieves almost the same effectiveness of StaticRecall but the expanded queries are from 2 to 3 times faster. SGS optimizes the EET metric using our proposed sequential strategy, further improving the previous results and showing the impact of the term dependency among successive selections. This technique produces queries having a recall nearly identical to the one of the previous models and showing an even lower retrieval time - up to 1.6 times - than S2. Despite this, we can see that the queries produced by all these models have an EET lower than not expanding at all. It is due to the fact that the queries requiring more than 200 ms are gradually more and more penalized by the EET metric. Moreover, all the previous techniques are not adaptive and select exactly 5 additional terms for each query. To this end, we also report the performance of the two techniques employing pruning, which limits the number of added terms under the control of the EET metric. The last two rows of the tables show the performance reached by S2 and SGS when employing term pruning. The effectiveness loss - from 0.85 to 0.80 in the worst case - achieved by the two techniques on the QED test set is due to the expensive queries whose expansion is now aggressively reduced by the pruning strategy. A smaller loss - from 0.96 to 0.94 in the worst case - is also observed for TREC data. Nevertheless, the pruning step allows to significantly reduce the query processing time up to 9 times w.r.t. the original models. Moreover, when considering the EET metric, the gains achieved by the SGS & pruning technique are statistically significant - using a t-paired test with $p < 0.05$ - compared to those of all the other methods evaluated on both QED and TREC data. Second, the retrieval time is reduced by more than one order of magnitude with respect to the baseline, i.e., StaticRecall.

4 CONCLUSION

We addressed the query expansion problem for Web search, a scenario characterized by tight time constraints. Our framework expands the initial query into a CNF query by extracting the candidate expansions from a thesaurus. It trades-off efficiency and effectiveness using the EET metric to perform term selection and pruning. We assessed the proposed models on a real-world dataset and on TREC data. Results show that our proposed QE framework remarkably outperforms the state-of-the-art in terms of efficiency, generating queries that are up to 30 times faster to process at the cost of only a small loss of retrieval effectiveness.

Acknowledgements. This paper is partially supported by the BIG-DATAGRAPES project (grant agreement 780751) that received funding from the European Union's Horizon 2020 research and innovation programme under the Information and Communication Technologies programme.

REFERENCES

[1] Ioannis Arapakis, Xiao Bai, and B Barla Cambazoglu. 2014. Impact of response latency on user behavior in web search. In *SIGIR 2014*.
[2] Guihong Cao, Jian-Yun Nie, Jianfeng Gao, and Stephen Robertson. 2008. Selecting good expansion terms for pseudo-relevance feedback. In *SIGIR (2008)*.
[3] Claudio Carpineto and Giovanni Romano. 2012. A survey of automatic query expansion in information retrieval. *Comput. Surveys* (2012).
[4] Marc-Allen Cartright, James Allan, Victor Lavrenko, and Andrew McGregor. 2010. Fast query expansion using approximations of relevance models. In *CIKM (2010)*.
[5] George W. Furnas, Thomas K. Landauer, Louis M. Gomez, and Susan T. Dumais. 1987. The vocabulary problem in human-system communication. *CACM* (1987).
[6] Matthias Hagen, Martin Potthast, Benno Stein, and Christof Braeutigam. 2010. The power of naive query segmentation. In *SIGIR (2010)*.
[7] Myeongjae Jeon, Saehoon Kim, Seung-won Hwang, Yuxiong He, Sameh Elnikety, Alan L Cox, and Scott Rixner. 2014. Predictive parallelization: Taming tail latencies in web search. In *SIGIR (2014)*.
[8] Victor Lavrenko and W Bruce Croft. 2001. Relevance based language models. In *SIGIR (2001)*.
[9] Chia-Jung Lee, Ruey-Cheng Chen, Shao-Hang Kao, and Pu-Jen Cheng. 2009. A term dependency-based approach for query terms ranking. In *CIKM (2009)*.
[10] Yuanhua Lv, ChengXiang Zhai, and Wan Chen. 2011. A boosting approach to improving pseudo-relevance feedback. In *SIGIR (2011)*.
[11] Craig Macdonald, Nicola Tonellotto, and Iadh Ounis. 2017. Efficient and Effective Selective Query Rewriting with Efficiency Predictions. In *SIGIR (2017)*.
[12] Rila Mandala, Tokunaga Takenobu, and Tanaka Hozumi. 1998. The use of WordNet in information retrieval. *Usage of WordNet in Natural Language Processing Systems* (1998).
[13] Giuseppe Ottaviano and Rossano Venturini. 2014. Partitioned elias-fano indexes. In *SIGIR (2014)*.
[14] Ellen M Voorhees. 1994. Query expansion using lexical-semantic relations. In *SIGIR (1994)*.
[15] Lidan Wang, Jimmy Lin, and Donald Metzler. 2010. Learning to efficiently rank. In *SIGIR (2010)*.
[16] Hao Wu and Hui Fang. 2013. An incremental approach to efficient pseudo-relevance feedback. In *SIGIR (2013)*.
[17] Yang Xu, Gareth JF Jones, and Bin Wang. 2009. Query dependent pseudo-relevance feedback based on wikipedia. In *SIGIR (2009)*.
[18] Zhiwei Zhang, Qifan Wang, Luo Si, and Jianfeng Gao. 2016. Learning for Efficient Supervised Query Expansion via Two-stage Feature Selection. In *SIGIR (2016)*.
[19] Le Zhao and Jamie Callan. 2012. Automatic term mismatch diagnosis for selective query expansion. In *SIGIR (2012)*.