Neural text generation for query expansion in information retrieval

Vincent Claveau

Neural text generation for query expansion in information retrieval

Vincent Claveau
vincent.claveau@irisa.fr
IRISA-CNRS
Rennes, France

ABSTRACT

Expanding users' query is a well-known way to improve the performance of document retrieval systems. Several approaches have been proposed in the literature, and some of them are considered as yielding state-of-the-art results in Information Retrieval. In this paper, we explore the use of text generation to automatically expand the queries. We rely on a well-known neural generative model, OpenAI's GPT-2, that comes with pre-trained models for English but can also be fine-tuned on specific corpora. Through different experiments and several datasets, we show that text generation is a very effective way to improve the performance of an IR system, with a large margin (+10 %MAP gains), and that it outperforms strong baselines also relying on query expansion (RM3). This conceptually simple approach can easily be implemented on any IR system thanks to the availability of GPT code and models.

KEYWORDS

Information Retrieval, Neural text generation, Neural language models, Query expansion, GPT2, Data-augmentation

1 INTRODUCTION

In the Information Retrieval (IR) traditional setting, a user expresses his information needs with the help of a query. Yet, it is sometimes difficult to match the query with the documents, for instance because of the query vocabulary may differ from the documents. Especially when the query is short, the performance of the system is usually poor, as it is difficult to detect the precise focus of the information need, and the relative importance of the query terms.

Query expansion aims at tackling these problems by transforming the short query into a larger text (or set of words) that makes it easier to match documents from the collection. The main difficulty of query expansion is obviously to add only relevant terms to the initial query. Several techniques have been proposed in the literature, based on linguistic resources (e.g. synonym lists) or based on the documents themselves (e.g. pseudo-relevance feedback).

In this paper, we explore the use of recent text generation models to expand queries. We experimentally demonstrate that the recent advances in neural generation can dramatically improve ad-hoc retrieval, even when dealing with specialized domains. More precisely, through different experiments, we show that:

(1) texts artificially generated from the query can be used for query expansion;
(2) this approach does not only provide new terms to the query, but also a better estimate of their relative weights;
(3) in addition, it also provides a better estimate of the importance (i.e. weight) of original query words;
(4) this approach can also be used on specialized domains.

The paper is structured as follows. After a presentation of the related work (Sect. 2), Section 3 details the different components of our approach. Several experiments are then detailed in Section 4. Last, some concluding remarks are given in Section 5.

2 RELATED WORK

Query expansion is a well-established technique to try to improve the performance of an IR system. Adding new terms to the query is expected to specifically improve recall, yet, since the query is, hopefully, better formulated, it may also improve the top rank results and be beneficial to precision. One might classify the existing automatic approaches based on the resources used to expand the query.

2.1 Expansion with external resources.

One obvious way to expand a query is to add semantically related terms to it (synonyms or sharing other semantic relations like hyponyms, quasi-synonyms, meronyms...). Existing lexical resources can be used to add, for each query term, a list of semantically related terms; yet, one has to deal with different problems: existence of lexical resources for the collection language, or for the specific domain of the collection, choice of the appropriateness of certain relations, need of sense disambiguation for polysemous words... WordNet [20] is among the best-known resources for English (general domain language) and have been used with mitigated results at first [36], but later shown to be effective [6, inter alia].

2.2 Expansion with pseudo-relevance feedback

Another category of studies considers only a small set of documents to help to expand or reformulate the query. To be automatic, they replace the user feedback by the hypothesis that the best ranked documents retrieved with the original query are relevant and may contain useful semantic information [30]. It is interesting to note that in this case, not only semantically relevant terms are extracted, but also distributional/statistical information on them and on the original query terms. In this category, Rocchio, developed in the 60's for vector space model was among the first one popularized [17]. One of the current best known approach is RM3, which was developed in the framework of language model based IR systems [1]. It is often reported to yield the best results in ad-hoc retrieval tasks, even compared with recent neural models [15]. Neural approaches have also been proposed to integrate pseudo-relevance feedback information [14], yet, as it is reported by the authors, the results are still lower than traditional models with query expansion.

2.3 Expansion with collection-based resources.

Distributional thesauri have also been exploited to enrich queries. Since they can be built from the document collection (or from a large corpus with similar characteristics), they are suited to the domain, the vocabulary... Traditional techniques to build these thesauri have obtained good results for query expansion [6]. Neural approaches, that is, word embedding approaches are now widely used to build such semantic resources. In the recent years, static embeddings (word2vec [19], Glove [26] or FastText [3] to name a few) were also used in IR, in particular to enrich the query. Indeed, these trainable dense representations make it easy to find new words that are semantically close to query words.

Even more recently, dynamic word representations obtained with transformer-based architectures, such as BERT [8] or GPT [27], have been proposed. They build a representation for each word according to its context, and this ability have been exploited to obtain competitive results in IR tasks [7, 12, inter alia]. BERT has been also used for query expansion in the framework of a neural IR system [21, 39], for instance based on reranking [24]. While these studies show promising results, it is worth noting that the RM3 method for pseudo-relevance feedback, while simpler, still competes with or even outperforms most of these neural-based models (both Glove-based or equivalent and BERT-based and equivalent), as noted in [21].

2.4 Text generation

In this paper, we propose to use constrained text generation to expand queries. In this approach, the original query is used as a seed (or prompt) for a generative model which will output texts that are, hopefully, related to the query. While text generation with language model is not new, the performance of neural models based on transformers [34] makes this task realistic.

In this paper, we use the Generative Pre-Trained Transformers (GPT) models. These neural models are learned by auto-regression, which means that they are unsupervisedly trained to predict the next token (word) given the previous ones. They are built from stacked transformers (precisely, decoders) that are trained on a large corpus. The second version, GPT-2 [27], contains between twelve layers (smallest model) up to 60 layers (largest model) of transformers with twelve self-attention heads of 64 dimensions. Given that, GPT-2 has 1.5 billion parameters for its largest pre-trained model, released in Nov. 2019. It has been trained on a specially crafted corpus named WebText which contains more than 8M documents from Reddit (i.e. mostly English and general domain language such as discussion on press articles).

A newer version, GPT-3, has been released in July 2020; it is much more larger (175 billion parameters) and outperforms GPT-2 on any tested task. Yet, the access given to this model (through a restricted API), the size of this model (which makes fine-tuning impossible) and the problems arising on how to engineer the prompt to perform the expected generation task, made GPT-2 preferable for this piece of work.

To the best of our knowledge, using generative models to expand queries has not been explored before. Yet, using GPT for data augmentation in other NLP tasks has recently received lots of attention. For instance, generation is exploited in relation extraction tasks [25], or text classification tasks like sentiment classification [13], re-admission prediction and phenotypic classification [2] or fake news detection [5].

3 GENERATED QUERY EXPANSION

In this section, a complete overview of the proposed expansion approach is first given. Additional details about the generative models and their adaptation are given in Section 3.2. The IR systems used in our experiments are presented in Section 3.3.

3.1 Overview of our approach

As it was previously explained, our approach is very simple as it relies on existing tools and techniques. From a query, multiple texts are generated by a GPT-2 model using the query as the seed. Note that the generation process is not deterministic (if some parameters such as top_k and temperature are correctly set; see next sub-section), and thus, even with the same seed, the generated texts are different. The generation of a large number of texts allows to have a large coverage of the vocabulary related to the query and a good estimation of the relative importance of words by their frequency in the generated texts. In the experiments reported below, 100 texts of 512 words (more precisely, tokens) per query are generated, unless specified otherwise (see Section 4.6). These texts are concatenated and considered as the expansion for the query. In our experiments, this new, very large, query is then fed to a simple BM25+ IR system (which are still state-of-the-art models, even compared with pure neural IR systems [33]), but it could obviously be used in any other IR system. Figure 1 presents the whole process; details for each part are given in the following sub-sections. Note that the only task done on-line (at query time) is the generation, which corresponds to the inference step of the neural language models to generate texts. The training or fine-tuning of the model itself is done off-line.

An example of a text generated from a query (query 701 from the GOV2 collection) is presented in Fig. 2. As one can see, the generated text, while completely invented (note the barrel prices), is relevant for the query. It contains many terms, absent from the original query, that are more or less closely related to the information need. More specifically, this generated text provides:

- synonyms and orthographic variants (United States for the query term U.S.),
- meronyms-metonyms (barrel for oil),
- hypernyms (energy for oil),
- more generally any paradigmatic relations (consumer, producer for industry),
- and syntagmatic relations (production for oil).

It is worth noting that such texts also give a valuable information about the relative frequency of each terms (contrary to thesauri or embeddings); this frequency information is an interesting cue to value the importance of a term.

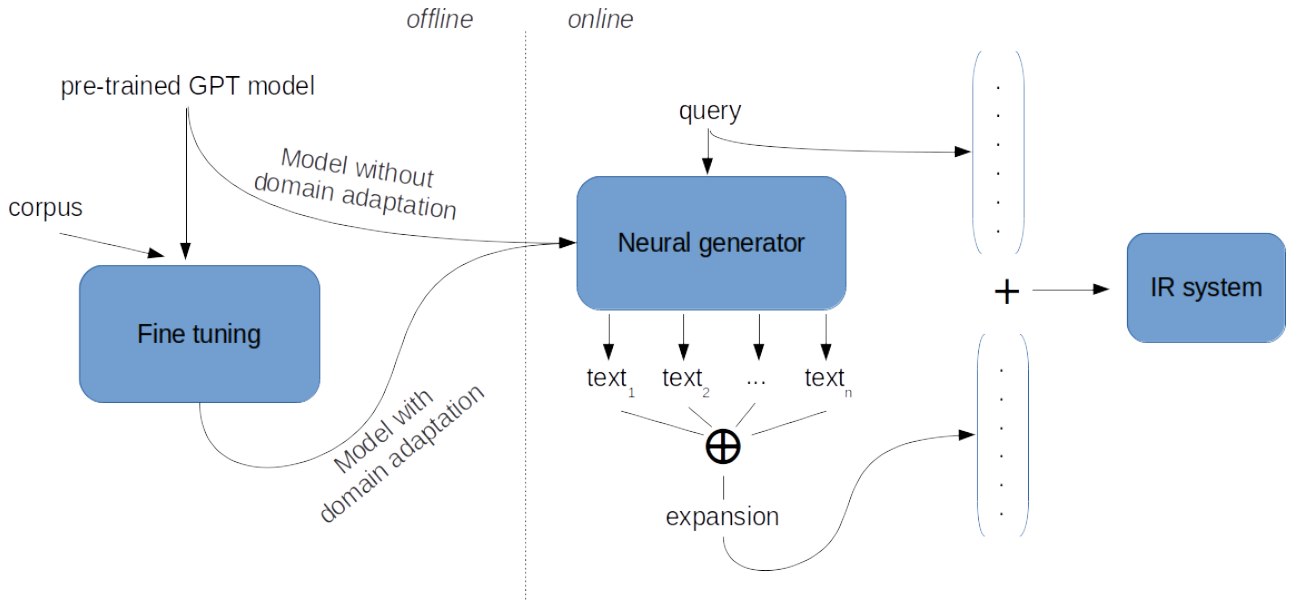

Figure 1: Overview of the proposed query expansion approach based on artificially generated texts.

```
U.S. oil production has been declining steadily for decades and it is not expected to reverse. In fact, some
argue that it may even get worse. The long-term trend is for oil production to decline at a rate of about 1
percent per year. With production of about 8 million barrels per day now, there is no way the United States
can replace its current output.
The U.S. oil boom was a result of an energy revolution in the 1970s that led to increased oil production, and
a significant change in the global oil market. The U.S. now produces about 2.3 million barrels of oil per day,
the highest it has been in over 30 years.
The United States is now the world's largest oil producer and the fourth largest oil exporter.
What happened?
When oil prices peaked in the 1970s, the United States was the world's largest oil producer. But over the
next several decades, the United States' oil production began to decline. The decline was most pronounced in
the 1980s, when the United States began to fall behind other oil producing countries.
The oil price decline in the 1970s was not entirely voluntary. The United States was producing less oil and
exporting more oil than it was consuming. The Federal Reserve controlled the amount of dollars in the Federal
Reserve's reserves, so the United States was not exporting as much oil as it was producing. The decline in
U.S. oil production was a result of the declining price of oil.
The price of oil had declined from $8 per barrel in 1973 to $2.50 per barrel in 1977. In 1979, the price of
oil reached a high of $15.75 per barrel. By 1983, the price of oil had fallen to $4.65 per barrel. By 1986,
the price of oil had fallen to $1.86 per barrel. By the end of the 1980s, the price of oil had fallen to $1.24
per barrel.
The decline in oil prices was a direct result of the energy revolution in the 1970s. The United States was
the world's largest oil producer, but the United States was also the world's largest consumer of oil. When
oil prices fell, so did the cost of producing oil.
```

Figure 2: Example of a document generated with the pre-trained GPT-2 large model from the text seed "U.S. oil industry history" (query 701 from the GOV2 collection)

3.2 Pre-trained models, fine-tuning and parameters

GPT-2 comes with several pre-trained models, having different size in terms of parameters (from 124M to 1.5B). As it was previously said, their training data was news-oriented general domain language. The largest model was used for two of the tested collections (see below). While these all-purpose models are fine for IR collections whose documents are also general domain language, it may not be appropriate for domain-specific IR collections. In the experiment reported in the next section, we use the OHSUMED collection, consisting of medical documents. For this collection, we have fine-tuned the GPT-2 355M model on the documents of the collection in order to adapt the language model to the specific medical syntax and vocabulary. We use Transformers library of

IR model	weighting
BM25+ $w_d(t)$	$\left(\frac{(k1+1)c(t,d)}{k1(1-b+b\cdot dl(d)/avdl)+c(t,d)} + \delta\right) \cdot \log \frac{N+1}{df(t)+0.5}$
BM25+ $w_q(t)$	$\frac{(k3+1)c(t,q)}{k3+c(t,q)}$
	with $k1, k_3, b$ and δ fixed parameters
LM $w_d(t)$	$\log\left(\frac{\mu}{dl(d)+\mu} + \frac{c(t,d)}{(dl(d)+\mu)p(t\|C)}\right)$
LM $w_q(t)$	$c(t,q)$
	$\mu > 0$ a smoothing parameter

Table 1: IR models (weighting functions of terms in the query and the document) for BM25+ [16, 29] and Language modeling with Dirichlet smoothing LM [38]; $c(t,d)$ is the number of occurrences of term t in document d, $df(t)$ is the document frequency of t, $dl(d)$ is the length of document d, $avdl$ is the average document length, C is the collection, N is the number of documents in the collection

HuggingFace. The fine-tuning was stopped after reaching a plateau in terms of perplexity of the model (in practice it corresponds to 250,000 samples processed). Other parameters (batch size, optimizer, learning rate...) were set to their defaults. Although a larger set of medical documents could be used (from Pubmed® for instance), this small fine-tuned model is expected to be more suited to generate useful documents to enrich the query.

Concerning the generation of texts, for reproducibity purposes, here are the main GPT-2 parameters used (please refer to HuggingFace Transformers[1] and GPT-2 documentations[2]): length=512, temperature=0.5, top_p=0.95, top_k = 40.

3.3 IR Systems

In the experiments reported in the next section, we use two IR models. The first one is BM25+ [16], a variant of BM25 [29]. The parameters k_1, k_3, b and δ were kept at their default value (resp. 1.2, 1000, 0.75, 1). It is implemented as a custom modification of the GENSIM toolkit [28]. The second IR model is Language modeling with Dirichlet smoothing [38] as implemented in Indri[3] [18, 32]. The smoothing parameter μ is set to 2 500 (a usual default value). Both models are regarded as yielding state-of-the-art performance for bag-of-words representation [15]. Their RSV function can be written:

$$RSV(q,d) = \sum_{t \in q} w_q(t) \cdot w_d(t)$$

with $w_q(t)$ the weight of term t in query q and $w_d(t)$ the weight in document d, as illustrated in Tab. 1 (from [16]). For RM3 expansion, we also rely on the Indri implementation; the results reported in the next section corresponds to the best performing parameters tested for each collection (number of documents considered for pseudo-relevance feedback, number of terms kept, mixing parameter λ).

[1]https://huggingface.co/transformers
[2]https://github.com/openai/gpt-2
[3]https://www.lemurproject.org/indri/

4 EXPERIMENTS

This section is dedicated to the experimental validation of the proposed query expansion approach. After a presentation of our experimental settings, we show the results on several collections (Sect. 4.2 and 4.3). We also present additional experiments about the interest of the frequency given by the generated text (Sect. 4.5) and about the influence of the number of generated texts (Sect. 4.6).

4.1 Experimental settings

Four IR collections are used in our experiments: Tipster [10] , Robust [35], GOV2 [4] and OHSUMED [11]. Some basic statistics are given in Tab. 2.

	Tipster	Robust	GOV2	OHSUMED
nb of documents	170,000	528,000	25M	350,000
nb of queries	50	250	150	106
avg size of queries	6.74	2.76	3.15	7.24
language	En	En	En	En
avg nb of relevant doc per query	849	65.5	179	21

Table 2: Statistics on the IR collections used

Tipster was used in TREC-2. The documents are articles from newspaper, patents and specialized press (computer related) in English. The queries are composed of several parts, including the query itself and a narrative detailing the relevance criteria; in the experiments reported below, only the actual query part is used.

The Robust collection consists of 528,000 news articles from Tipster disks 4 and 5; there 250 topics (301-450, 601-700). As for the published work, we use the titles as queries.

GOV2 is a large collection of Web pages crawled from the .gov domain and used in several TREC tracks. In the experiments reported below, 150 queries from TREC 2004-2006 ad-hoc retrieval tasks are used; as for Tipster, only the actual query part is used (i.e. description and narrative fields are not included in the query).

OHSUMED contains bibliographical notices from Medline and queries from the TREC-9 filtering task. Its interest for our experiments is that it deals with a specialized domain, hence it contains a specific vocabulary.

Performance are assessed with standard scores: Precision at different thresholds (P@x), R-precision (R-prec), *Mean Average Precision* (MAP) on 1,000 first retrieved documents. When needed, a paired t-test with $p = 0.05$ is performed to assess the statistical significance of the difference between systems.

4.2 General domain language

Tables 3 and 4 respectively present the results for the general-language collections Tipster and GOV2. For comparison purposes, we indicate the results of BM25+ with and without RM3 expansion, Indri's Language Model (LM) with and without RM3 expansion. Note that the RM3 expansion are strong baselines, as they achieve state-of-the-art performance, even compared with neural techniques (incl. static embedding or contextualized embedding-based expansion) [21, 37]. Moreover, the results reported here are

for the best-performing RM3 parameters: for BM25+, this is 100 terms from the top 30 documents on Tipster and 80 terms on the top 10 documents on GOV2 ; for LM on Tipster, this is 100 terms from the top 20 documents, and 100 terms for the top 10 documents GOV2. The statistical significance is computed by comparing with the BM25+ + RM3 (noted with †) and LM+RM3 baselines (noted with a *).

On both collections, and on every performance measure, expanding the queries with the generated texts brings important gains compared with the system without expansion. Also, our approach outperforms RM3 expansion in almost every situation, and with a large margin on MAP, R-prec and precision on the top-ranked documents (P@5, P@10).

4.3 Specialized language

The same setting is used on the OHSUMED collection. For these medical-oriented IR dataset, we report two versions of our approach: one is using the pre-trained model as before, and one relies on a model fine-tuned on the documents of the collection. The best performing setting for RM3 is 100 terms for the top 15 documents for BM25+ and 80 terms for the top 10 documents for LM. The results are reported in Tab. 5.

Here again, the GPT-based expansion significantly improves the results of the IR system and outperforms RM3 expansion. Yet, the gains are lower than for the two previous collections. This difference can be explained by the following factors:

(1) the queries are longer more complex and more specific (as can be seen in Tab. 2, few documents are relevant);
(2) the generation model is not sufficiently suited to the documents.

Concerning this latter reason, we can indeed see the interest of fine-tuning the generation model, but better results may be obtained by using a larger set of medical documents, or adopting different fine-tuning parameters (in particular the number of epoch/samples processed, see Sect. 3.2). Unfortunately, defining a priori the best parameters for our IR task is not possible and the cost of the fine-tuning process makes it impossible to test a wide range of possible values.

4.4 Comparison with other expansion approaches

In this section, we position our approach with respect to other expansion approaches that have been proposed in the literature. For this experiment, we use the Robust dataset to make our results comparable with published results; we also re-employ the same evaluation performance scores as in [21]. For this experiment, we use the Pyserini framework[4] which comes with the pre-indexed Robust collection, we thus rely on its BM25 and RM3 implementations. Our GPT2-based expansions have been generated as before with the pre-trained (non fine-tuned) large model. In Table 6 we report the performance of our approach, the results of CEQE and its variants [21], as well as the various baselines proposed in [21], including an expansion based on Glove embeddings (static-embed) [9] and a variant that has its vocabulary limited to terms appearing in the pseudo-relevance feedback documents (static-embed-PRF); see [21] for details. The BM25 parameters are 0.9 and 0.4 as recommended by Pyserini for Robust; in our experiments, the best performing RM3 parameters are 70 terms from the top 10 documents.

There are some slight differences between our baselines (BM25, BM25+RM3) and those of [21], maybe due to differences in the tokenizing and stemming processes of the IR frameworks. Nonetheless, the GPT-based expansion yields the best results, with significant improvement over the BM25+RM3 baseline and several points above the best CEQE configuration.

[4]https://github.com/castorini/pyserini/

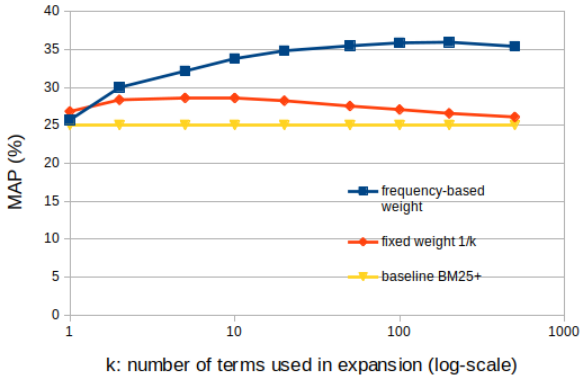

Figure 3: MAP (%) according to size of the expansion (number of terms), terms with fixed weight or weight depending on their frequency in the generated documents; Tipster collection

4.5 Leveraging the term importance in the generated texts

One of the interest of having complete texts that are generated is that we can collect information on the relative importance of words, to the contrary of expanding queries with a thesaurus. To observe the impact of the number of occurrences in the generated texts, we evaluate the effect of keeping the k most frequent terms of the generated texts and either weighting them by their frequency (as done usually by BM25) or by giving a fixed weight ($1/k$). The results for different values of k are presented in Fig. 3. One can observe that adding terms to the query with a fixed weight slightly improves the MAP, but most of the gain is indeed brought by a proper weighting based on the frequency of the term in the generated documents. This is a big advantage of having proper texts, generated and tailored for the query, instead of related terms taken from a thesaurus or computed from an embedding. It also worth noting that the maximum MAP is reached with about 100 terms; this is interesting for a fair comparison with RM3 since this is also the typical numbers of terms yielding the best results in our experiments.

In the next experiment, we also examine how the generated texts can help to re-weight the initial query terms. The idea is that queries are often too short to get relevant information about the relative importance of each query term (often, each of them occurs only once, that is $c(t, q) = 1$). In this experiment, there is no

	MAP	R-Prec	P@5	P@10	P@20	P@100
BM25+	25.06	32.16	95.60	92.60	89.70	73.64
BM25+ + RM3	32.28	36.94	96.20	94.20	90.90	82.96
LM	24.48	31.48	92.40	89.00	85.40	70.70
LM + RM3	31.01	36.38	94.40	93.20	90.60	81.22
BM25+ and expansion	**35.22***†	**39.87***†	**99.60***†	**98.40***†	**98.20***†	**87.84***

Table 3: Performance (%) on the Tipster collection with query expansion; best results in bold, statistical significance over BM25+ + RM3 and LM+RM3 resp. noted † and *

	MAP	R-Prec	P@5	P@10	P@20	P@100
BM25+	25.66	31.25	52.92	49.97	46.52	34.63
BM25+ with RM3	28.16	32.57	54.86	54.13	49.34	39.65
LM	27.96	33.01	56.08	55.20	51.59	37.32
LM with RM3	30.22	34.20	55.00	56.08	53.67	**45.86**
BM25+ and expansion	**34.54***†	**37.76**†	**67.91***†	**63.88***†	**57.94**†	44.30†

Table 4: Performance (%) on the GOV2 collection with query expansion; best results in bold, statistical significance over BM25+ + RM3 and LM+RM3 resp. noted † and *

	MAP	R-Prec	P@5	P@10	P@20	P@100
BM25+	18.27	19.94	31.88	26.04	20.50	9.48
BM25+ + RM3	21.44	22.72	32.67	28.70	23.73	10.82
LM	17.61	20.35	29.31	24.06	19.21	9.28
LM + RM3	20.80	22.54	30.89	26.83	22.18	10.51
BM25+ and expansion (no fine-tuning)	21.60	23.75	33.47*	27.62	22.92	11.16
BM25+ and expansion (fine-tuning)	**23.07***†	**24.65***†	**34.65***†	**29.41***	**24.31**	**11.42**

Table 5: Performance (%) on the OHSUMED collection with query expansion; best results in bold, statistical significance over BM25+ + RM3 and LM+RM3 resp. noted † and *

Model	P@20	nDCG@20	MAP	Recall@100	Recall@1000
BM25	36.57	41.93	25.74	41.65	69.33
BM25 + RM3	39.98	45.17	30.69	46.10‡	75.88‡
Static-Embed	36.75	42.85	26.15	42.17	71.25
Static-Embed-PRF	37.81	44.00	27.03	43.24	72.31
CEQE-Centroid	39.22	44.62	30.19‡	45.93‡	76.53†‡
CEQE-MulPool	38.47	43.60	28.45‡	45.17‡	74.35‡
CEQE-MaxPool	40.40‡	45.87	30.86‡	46.51‡	76.89†‡
CEQE-MaxPool(fine-tuned)	39.86‡	45.28	30.71‡	46.47‡	76.26‡
BM25	35.88	41.93	25.15	40.80	69.26
BM25 + RM3	39.52	44.89	29.85	45.97	76.97
BM25 + GPT	**41.71***	**48.32***	**30.96***	**47.48***	**79.85***

Table 6: Performance (%) on Robust; first rows are results from [21] († and ‡ indicate statistical significance over BM25 + RM3 and Static-Embed-PRF, respectively), last rows are our results with the Pyserini framework (* indicates statistical significance over BM25 + RM3)

query expansion, since only the initial query terms are kept, but we use their frequency in the generated texts to compute the BM25+ weight w_q. The results are reported in Tab. 7 and compared with the usual weighting (i.e. BM25 weight with the frequency from the original query). It appears that there is indeed a small improvement of the MAP (+2% absolute gain), that is more noticeable at high document-cutoff values. These two experiments demonstrate the

	MAP	R-Prec	P@5	P@10	P@20	P@100
BM25+	25.06	32.16	95.60	92.60	89.70	73.64
BM25+ with re-weighting	27.12	33.22	96.80	94.20	92.70	77.12

Table 7: Performance (%) on Tipster with re-weighting query words

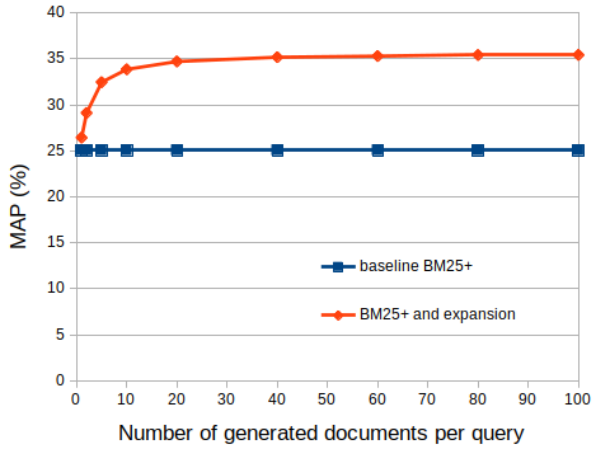

Figure 4: MAP (%) vs. number of generated texts (average over 5 runs; max length of texts = 512 tokens); Tipster collection

usefulness of dealing with full texts and not only word-to-word similarity since the texts provide relevant frequency information.

4.6 Number of generated texts

Text generation with large neural models has a non negligible computing cost: with our settings, for one query, about 40 texts (of 512 tokens) are simultaneously generated in about 5 seconds on one Tesla V100 GPU card. Thus, it is interesting to see how many generated texts are necessary and more generally what is the influence of the size of the expansion on the IR performance. Of course, the size of the generated texts (can be set as a parameter of the generation process) is also to be considered.

In Figure 4, the MAP obtained for up to 100 generated texts per query is presented. For each number n of texts, 5 runs are performed (ie. 5 sets of n texts are generated for each query, and the 5 MAP are averaged). One can observe that a plateau is rapidly reached at around 20 texts per query; it represents about 10,000 words. It means that good performance can be yielded with a limited time and computing cost.

5 CONCLUSIVE REMARKS AND FORESEEN WORK

Neural approaches are increasingly used in IR, with mitigated results, especially when compared with "traditional" bag-of-word approaches [15, 33, 37]. Here, the neural part is successfully used outside of a "traditional" IR system (but note that it could be used with any IR systems, since it simply enriches the query). The expansion approach presented in this paper is simple and easy to implement (thanks to the availability of the GPT models and code) while offering impressive gains. The same approach could be used with other IR systems (neural or not), other approaches to enrich the query, and more sophisticated post-processing (such as re-ranking techniques).

Lot of parameters could be further optimized, especially on the GPT model side (to influence the "creativity" of the text generation), and the fine-tuning capabilities should also be explored more thoroughly (influence of bigger specialized corpus if available, precise mix between pre-trained and fine-tuning, etc.). The recent availability of GPT-3[5] makes it possible to even get greater gains thanks to the alleged high quality of its outputs, but necessitates to change the fine-tuning paradigm (used for OHSUMED here) to a prompt engineering paradigm. Last, let us note that the generation time of the artificial texts and the necessary GPU power may appear as a problem for some industrial contexts. Yet, these costs are not untractable (see Sect. 4.6) and can be dealt with one GPU card and a few seconds of additional processing time. Moreover, model reduction techniques, such as distillation [31], or TPUs could further reduce this generation time and its computational cost.

This whole approach also offers many research avenues: in this work, we have used text generation as a way to perform data augmentation on the query side, but it could also be used to augment the representation of the documents (even if in practice, the cost is still prohibitive on large collection, as seen with the doc2query and docTTTTTquery models [22, 23]). All machine-learning (neural or not) approaches based on pseudo-relevance feedback to train their model could instead use similar text generation with the advantage that they would not be limited by the number of potential relevant documents in the shortlist. And of course, similar data-augmentation strategy could be used for other tasks than document retrieval.

More fundamentally, the recent improvements of text generation also question the relevance of the document retrieval task. Indeed, it is possible to envision systems that will be able to generate one unique document answering the user's information need, similarly to question-answering. If the generative model is trained on the document collection, the generated document will serve as a summary (which is one of the popular applications of GPT-x models) of the relevant documents. Yet, the current limitations of the models tested in this paper make them far from being suited for this ultimate task: the generated documents do deal with the subject of the query, and thus use a relevant vocabulary, but do not provide accurate, factual information (as seen in the Example in Fig. 2 about the price of oil barrels).

[5] https://github.com/openai/gpt-3

REFERENCES

[1] Nasreen Abdul-jaleel, James Allan, W. Bruce Croft, O Diaz, Leah Larkey, Xiaoyan Li, Mark D. Smucker, and Courtney Wade. 2004. UMass at TREC 2004: Novelty and HARD. In *In Proceedings of TREC-13*.

[2] Ali Amin-Nejad, Julia Ive, and Sumithra Velupillai. 2020. Exploring Transformer Text Generation for Medical Dataset Augmentation. In *Proceedings of the 12th Language Resources and Evaluation Conference*. European Language Resources Association, Marseille, France, 4699–4708. https://www.aclweb.org/anthology/2020.lrec-1.578

[3] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. *arXiv preprint arXiv:1607.04606* (2016).

[4] Charles L.A. Clarke and Ian Soboroff. 2005. The TREC 2005 Terabyte Track. In *Proceedings of the Fourteenth Text REtrieval Conference (TREC 2005)*. Gaithersburg, MD, USA.

[5] Vincent Claveau. 2020. Detecting fake news in tweets from text and propagation graph: IRISA's participation to the FakeNews task at MediaEval 2020. In *MediaEval Benchmarking Initiative for Multimedia Evaluation (MediaEval 2020)*. online, United States. https://hal.archives-ouvertes.fr/hal-03116027

[6] Vincent Claveau and Ewa Kijak. 2016. Direct vs. indirect evaluation of distributional thesauri. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. The COLING 2016 Organizing Committee, Osaka, Japan, 1837–1848. https://www.aclweb.org/anthology/C16-1173

[7] Zhuyun Dai and Jamie Callan. 2019. Deeper Text Understanding for IR with Contextual Neural Language Modeling. *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Jul 2019). https://doi.org/10.1145/3331184.3331303

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. (2019). arXiv:1810.04805 [cs.CL]

[9] F. Diaz, B. Mitra, and N. Craswell. 2016. Query expansion with locally-trained word embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

[10] Donna Harman. 1995. Overview of the Second Text Retrieval Conference (TREC-2). *Information Processing and Management* 31, 3 (1995), 271–289.

[11] William Hersh, Chris Buckley, T. J. Leone, and David Hickam. 1994. OHSUMED: An Interactive Retrieval Evaluation and New Large Test Collection for Research. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Dublin, Ireland) *(SIGIR '94)*. Springer-Verlag New York, Inc., New York, NY, USA, 192–201. http://dl.acm.org/citation.cfm?id=188490.188557

[12] Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, Jimmy Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (Eds.). ACM, 39–48. https://doi.org/10.1145/3397271.3401075

[13] Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data Augmentation using Pre-trained Transformer Models. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*. Association for Computational Linguistics, Suzhou, China, 18–26. https://www.aclweb.org/anthology/2020.lifelongnlp-1.3

[14] Canjia Li, Yingfei Sun, Ben He, Le Wang, Kai Hui, Andrew Yates, Le Sun, and Jungang Xu. 2018. NPRF: A Neural Pseudo Relevance Feedback Framework for Ad-hoc Information Retrieval. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 4482–4491. https://doi.org/10.18653/v1/D18-1478

[15] Jimmy Lin. 2018. The Neural Hype and Comparisons Against Weak Baselines. *SIGIR Forum* 52, 2 (2018), 40–51. https://doi.org/10.1145/3308774.3308781

[16] Yuanhua Lv and ChengXiang Zhai. 2011. Lower-bounding Term Frequency Normalization. In *Proc. of the 20th ACM International Conference on Information and Knowledge Management* (Glasgow, Scotland, UK) *(CIKM '11)*. ACM, New York, NY, USA, 7–16. https://doi.org/10.1145/2063576.2063584

[17] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, USA.

[18] D. Metzler and W.B. Croft. 2004. Combining the Language Model and Inference Network Approaches to Retrieval. *Information Processing and Management Special Issue on Bayesian Networks and Information Retrieval* 40, 5 (2004), 735–750.

[19] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger (Eds.). 3111–3119. http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality

[20] George A. Miller. 1990. WordNet: An On-Line Lexical Database. *International Journal of Lexicography* 3, 4 (1990).

[21] Shahrzad Naseri, Jeffrey Dalton, Andrew Yates, and James Allan. 2021. CEQE: Contextualized Embeddings for Query Expansion. In *Proceedings of European Conference in Information Retrieval ECIR*. Lucca, IT (virtual event).

[22] Rodrigo Nogueira. 2019. From doc2query to docTTTTTquery. In *An MS MARCO passage retrieval task micro-publication*.

[23] Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document Expansion by Query Prediction. (2019). arXiv:1904.08375 [cs.IR]

[24] Ramith Padaki, Zhuyun Dai, and Jamie Callan. 2020. Rethinking Query Expansion for BERT Reranking. In *Advances in Information Retrieval*, Joemon M. Jose, Emine Yilmaz, João Magalhães, Pablo Castells, Nicola Ferro, Mário J. Silva, and Flávio Martins (Eds.). Springer International Publishing, Cham, 297–304.

[25] Yannis Papanikolaou and Andrea Pierleoni. 2020. DARE: Data Augmented Relation Extraction with GPT-2. arXiv:2004.13845 [cs.CL]

[26] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*. 1532–1543. http://www.aclweb.org/anthology/D14-1162

[27] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. *OpenAI Blog* (2019).

[28] Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, 45–50. http://is.muni.cz/publication/884893/en.

[29] Stephen E. Robertson, Steve Walker, and Micheline Hancock-Beaulieu. 1998. Okapi at TREC-7: Automatic Ad Hoc, Filtering, VLC and Interactive. In *Proc. of the 7th Text Retrieval Conference, TREC-7*. 199–210.

[30] Ian Ruthven and Mounia Lalmas. 2003. A survey on the use of relevance feedback for information access systems. *Knowledge Eng. Review* 18, 2 (2003), 95–145.

[31] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. (2020). arXiv:1910.01108 [cs.CL]

[32] T. Strohman, D. Metzler, H. Turtle, and W.B. Croft. 2005. *Indri: A language-model based search engine for complex queries (extended version)*. Technical Report. CIIR.

[33] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A Heterogenous Benchmark for Zero-shot Evaluation of Information Retrieval Models. arXiv:2104.08663 [cs.IR]

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.). Curran Associates, Inc., 5998–6008. http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf

[35] Ellen Voorhees. 2004. Overview of the TREC 2004 Robust Track. In *Proceedings of the Thirteenth Text REtrieval Conference, TREC 2004*.

[36] Ellen M. Voorhees. 1994. Query Expansion Using Lexical-semantic Relations. In *Proc. of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Dublin, Ireland) *(SIGIR '94)*. Springer-Verlag New York, Inc., New York, NY, USA, 61–69. http://dl.acm.org/citation.cfm?id=188490.188508

[37] Wei Yang, Kuang Lu, Peilin Yang, and Jimmy Lin. 2019. Critically Examining the "Neural Hype": Weak Baselines and the Additivity of Effectiveness Gains from Neural Ranking Models. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, Benjamin Piwowarski, Max Chevalier, Éric Gaussier, Yoelle Maarek, Jian-Yun Nie, and Falk Scholer (Eds.). ACM, 1129–1132. https://doi.org/10.1145/3331184.3331340

[38] C. Zhai and J. D. Lafferty. 2001. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proc. of the SIGIR conference*. 334–342.

[39] Zhi Zheng, Kai Hui, Ben He, Xianpei Han, Le Sun, and Andrew Yates. 2020. BERT-QE: Contextualized Query Expansion for Document Re-ranking. In *Findings of the Association for Computational Linguistics: EMNLP 2020*. Association for Computational Linguistics, Online, 4718–4728. https://doi.org/10.18653/v1/2020.findings-emnlp.424