# Query Expansion using Large Language Models

Hemang Joshi (202001212),* Aditi Das (202001259),† Mohammad Tejabwala (202001406),‡ and Bhavajna Kallakuri (202003046)§
*Dhirubhai Ambani Institute of Information & Communication Technology, Gandhinagar, Gujarat 382007, India*

Integrating Large Language Models (LLMs) like GPT-3.5 and LLama into search engines is a groundbreaking leap in information retrieval. These models, with a profound grasp of language subtleties, provide human-like responses, surpassing traditional keyword matching. Relevance feedback engages users in refining search outcomes iteratively, improving accuracy. Leveraging datasets like Vaswani and TREC DL 19, our project utilizes LLMs to enhance query expansion, refining results for diverse subtasks

## I. Introduction

Recently, there is a lot of hype of using Large language models in improving various tasks and integrating them with Search Engines to improve user experience. In this paper, we are focusing on improving the information retrieval task by forming a new query with the feedback from the Large Language model. We have used GPT-3.5 and BARD model for this purpose with the default parameters.

## II. Query Expansion

The classical approach to address the vocabulary mismatch problem is query expansion using Pseudo-Relevance Feedback, where the query is expanded using terms from the top-$k$ documents in a feedback set. This feedback set is obtained using a first-pass retrieval, and the expanded query is then used for a second-pass retrieval. While query expansion with PRF often improves recall, its effectiveness hinges on the quality of the first-pass retrieval. Non-relevant results in the feedback set introduce noise and may pull the query off-topic. [1]

So, instead of taking feedback from the first retrieval, we are taking the feedback using several subtasks used in [1] and some of our own to expand the query. This will function as the auto-correct that we generally see in the web browsers. The user makes a spelling mistake and the search engine instead of using the spell-error query, corrects the query and uses it instead. So, in our model of retrieval, we will add several terms to the query and use that query for the retrieval. This could save us 1 retrieval step and further speed up the process of retrieval depending upon the latency of the feedback from the Large Language Model.

*Electronic address: 202001212@daiict.ac.in
†Electronic address: 202001259@daiict.ac.in
‡Electronic address: 202001406@daiict.ac.in
§Electronic address: 202003046@daiict.ac.in

## III. Parameters used for BM25 retrieval

We used $k_1 = 0.625$ and $b = 0.625$ for all the retrieval subtasks. We arrived at these values by applying a sort of Binary Search on the parameters. The range of both the parameters was considered to be $0$ to $20$. We fixed one parameter, and varied the other one, taking it in the direction where we saw an increase in the IR metrics and then we did same for the other parameter.

## IV. Generation subtasks and their prompts

We performed the following subtasks,

- **Keywords**: "Based on the query, generate a bullet-point list of relevant keywords"
- **Entities**: "Based on the query, generate a bullet-point list of relevant entities"
- **CoT-Keywords**: "Based on the query, generate a bullet-point list of relevant keywords. Next to each point, briefly explain why:"
- **CoT-Entities**: "Based on the query, generate a bullet-point list of relevant entities. Next to each point, briefly explain why:"
- **Diverse Keywords**: "Based on the query, generate a bullet-point list of diverse keyword queries that will find relevant documents:"
- **Essay**: "Based on the query, write an essay"
- **News Article**: "Based on the query, write a news article"
- **Summary**: "Based on the query, write an summary"
- **Facts**: "Based on the query, generate a bullet-point list of relevant facts present in relevant documents"
- **Document**: "Based on the query, generate a relevant document"
- **Query Expansion**: "Based on the query, generate an expanded query"
- **Webpage**: "Generate web page content for each query"
- **Closely related terms**: "Generate closely related terms for each query"
- **Alternate Sentences**: "Generate at least 5 alternate sentences for each query"

### V. About the dataset

In this project, we have used the following datasets for our analysis TREC Deep Learning (DL) 19 (judged version).

- **TREC Deep Learning (DL) 19:** Expanding on the foundation of the MS MARCO web queries and documents. It consists of 43 topics for DL-19. Notably, both sets of queries are primarily focused on factoid-based information retrieval. It consists of about 9.3k relevance scores.

### VI. Results

TABLE I: Evaluation Metrics without

| Model | MAP | NDCG@10 | Recall@1k |
|---|---|---|---|
| BM25 | 0.3909 | 0.2247 | 0.7401 |
| BM25 + RM3 | 0.4380 | 0.2349 | 0.7736 |

TABLE II: Evaluation Metrics for Different subtasks - 1 (GPT-3.5)

| Category | MAP | NDCG@10 | Recall@1k |
|---|---|---|---|
| Keywords | 0.4057 (+3.79%) | 0.2306 (+2.63%) | 0.7878 (+6.45%) |
| Entities | 0.4377 (+11.97%) | 0.2549 (+13.44%) | 0.7935 (+7.22%) |
| CoT Keywords | 0.3541 | 0.2215 | 0.7587 (+2.51%) |
| CoT Entities | 0.3439 | 0.2260 (+0.58%) | 0.7532 (+1.77%) |
| Diverse Keywords | 0.3847 | 0.2288 (+1.82%) | 0.7889 (+6.59%) |
| Essay | 0.4267 (+9.16%) | 0.2584 (+15.00%) | 0.7828 (+5.77%) |
| News article | 0.4357 (+11.46%) | 0.2598 (+15.62%) | 0.7979 (+7.81%) |

TABLE III: Evaluation Metrics for Different subtasks - 2 (GPT-3.5)

| Category | MAP | NDCG@10 | Recall@1k |
|---|---|---|---|
| Summary | 0.4305 (+10.13%) | 0.2585 (+15.04%) | 0.7923 (+7.05%) |
| Facts | 0.4186 (+7.09%) | 0.2520 (+12.15%) | 0.8051 (+8.78%) |
| Documents | 0.4513 (+15.45%) | 0.2562 (+14.02%) | 0.8183 (+10.57%) |
| Expanded Query | 0.4053 (+3.68%) | 0.2341 (+4.18%) | 0.7478 (+1.04%) |
| Webpage | 0.4050 (+3.61%) | 0.2492 (+10.90%) | 0.7733 (+4.49%) |
| Closely Related terms | 0.3721 | 0.2073 | 0.7855 (+6.13%) |
| Alternate queries | 0.4401 (+12.59%) | 0.2440 (+8.59%) | 0.8141 (+10.00%) |

TABLE IV: Evaluation Metrics for Different Combinations - 1(GPT-3.5)

| Combination | MAP | NDCG@10 | Recall@1k |
|---|---|---|---|
| Alternate queries + Documents | 0.5195 (32.90%) | 0.2864 (27.46%) | 0.8614 (16.39%) |
| Alternate queries + Entities | 0.4922 (25.91%) | 0.2767 (23.14%) | 0.8242 (11.36%) |
| Alternate queries + News | 0.4850 (24.07%) | 0.2719 (21.00%) | 0.8236 (11.28%) |
| Documents + Entities | 0.5188 (32.72%) | 0.2881 (28.22%) | 0.8525 (15.19%) |
| Documents + News | 0.5053 (29.26%) | 0.2840 (26.39%) | 0.8522 (15.15%) |
| Entities + News | 0.4919 (25.84%) | 0.2851 (26.88%) | 0.8262 (11.63%) |

TABLE V: Evaluation Metrics for Different Combinations - 2(GPT-3.5)

| Combination | MAP | NDCG@10 | Recall@1k |
|---|---|---|---|
| Alternate queries + Documents + Entities | 0.5252 (34.36%) | 0.2914 (29.68%) | 0.8593 (16.11%) |
| Alternate queries + Documents + News | 0.5180 (32.51%) | 0.2856 (27.10%) | 0.8550 (15.52%) |
| Alternate queries + Entities + News | 0.4970 (27.14%) | 0.2766 (30.26%) | 0.8268 (11.71%) |
| Documents + Entities + News | 0.5185 (32.64%) | 0.2928 (30.31%) | 0.8502 (14.88%) |
| Alternate queries + Documents + Entities + News | 0.5231 (33.82%) | 0.2915 (29.73%) | 0.8520 (15.12%) |

TABLE VI: Evaluation Metrics for all subtasks combined (Documents + Facts + News + Summary + Essay + Keywords + Keywords (CoT) + Diverse Keywords + Entities + Entities (CoT) + Web Pages + Closely Related + Expanded Queries + Alternate Sentences) (GPT-3.5)

| MAP | NDCG@10 | Recall@1k |
|---|---|---|
| 0.5120 (30.98%) | 0.2834 (26.12%) | 0.8433 (9.01%) |

TABLE VII: Evaluation Metrics for Different subtasks - 1 (BARD)

| Category | MAP | NDCG@10 | Recall@1k |
|---|---|---|---|
| Keywords | 0.3751 | 0.2219 | 0.7830 (+5.8%) |
| Entities | 0.3461 | 0.2143 | 0.7554 (+2.07%) |
| CoT Keywords | 0.3970 (+1.56%) | 0.2165 | 0.7843 (+5.97%) |
| CoT Entities | 0.4409 (+12.79%) | 0.2383 (+6.05%) | 0.8175 (+10.46%) |
| Diverse Keywords | 0.4002 (+2.38%) | 0.2193 | 0.7657 (+3.46%) |

TABLE VIII: Evaluation Metrics for Different subtasks - 2 (BARD)

| Document | MAP | NDCG@10 | Recall@1k |
|---|---|---|---|
| Essay | 0.4556 (+16.55%) | 0.2691 (+19.76%) | 0.8012 (+8.26%) |
| News article | 0.4500 (+15.12%) | 0.2536 (+12.86%) | 0.8157 (+10.21%) |
| Summary | 0.4584 (+17.27%) | 0.2726 (+21.32%) | 0.8034 (+8.55%) |
| Facts | 0.4619 (+18.16%) | 0.2584 (+15%) | 0.8292 (+12.04%) |
| Documents | 0.4829 (+23.54%) | 0.2598 (+15.62%) | 0.8320 (+12.42%) |

TABLE IX: Evaluation Metrics for Different Combinations - 1 (BARD)

| Combination | MAP | NDCG@10 | Recall@1k |
|---|---|---|---|
| Documents + Facts | 0.5218 (+33.49%) | 0.2845 (+26.61%) | 0.8569 (+15.78%) |
| Documents + News | 0.4955 (+26.76%) | 0.2695 (+19.94%) | 0.8321 (+12.43%) |
| Documents + Summary | 0.5229 (+33.77%) | 0.2867 (+27.59%) | 0.8431 (+13.92%) |
| News + Facts | 0.5128 (+31.18%) | 0.2897 (+28.93%) | 0.8499 (+14.84%) |
| Summary + Facts | 0.5140 (+31.49%) | 0.2878 (+28.08%) | 0.8467 (+14.40%) |
| News + Summary | 0.5153 (+31.82%) | 0.2874 (+27.9%) | 0.8413 (+13.67%) |

TABLE X: Evaluation Metrics for Different Combinations - 2 (BARD)

| Combination | MAP | NDCG@10 | Recall@1k |
|---|---|---|---|
| Documents + Facts + News | 0.5188 (+32.72%) | 0.2814 (+25.23%) | 0.8497 (+14.81%) |
| Documents + Facts + Summary | 0.5299 (+35.56%) | 0.2920 (+29.95%) | 0.8558 (+15.63%) |
| Documents + News + Summary | 0.5153 (+31.82%) | 0.2800 (+24.61%) | 0.8409 (+13.62%) |
| Summary + Facts + News | 0.5237 (+33.97%) | 0.2938 (+30.75%) | 0.8507 (+14.94%) |
| Document + Summary + Facts + News | 0.5255 (+34.43%) | 0.2860 (+27.28%) | 0.8523 (+15.16%) |

TABLE XI: Evaluation Metrics for all subtasks combined (Documents + Facts + News + Summary + Essay + Keywords + Keywords (CoT) + Diverse Keywords + Entities + Entities (CoT) + Web Pages + Closely Related + Expanded Queries + Alternate Sentences) (BARD)

| Retrieval Model | MAP | NDCG@10 | Recall@1k |
|---|---|---|---|
| BM25 | 0.5521 (+41.24%) | 0.2989 (+33.02%) | 0.8745 (+18.16%) |
| BM25 + RM3 | 0.5258 (+20.05%) | 0.2878 (+22.52%) | 0.8377 (+8.29%) |

After analysing all our results, we observed that combinations of 2 and 3 are having more gain in the metrics when compared to that of all the subtasks combined. The reason for this could be that in all subtasks combination, due to the large length of the query, noise might be added and leading to a decrease in metrics. We also did another experiment, in which we combined the feedback from both the LLMs as the LLMs have different training data and different parameters, we thought of combining them to observe the change in metrics. So for this, from BARD, we took the subtask having the best MAP which was "Document", and the subtask which was having the best Recall@1k, "Facts" and from GPT-3.5 "Document" subtask was having the best MAP and Recall@1k value so we took the subtask which has the next best Recall@1k, which was "Alternate queries". The results are better than all of our previous experiments.

TABLE XII: Combination of best subtasks of BARD and GPT-3.5 compared with RM3

| Method | MAP | NDCG@10 | Recall@1k |
|---|---|---|---|
| Document [BARD] + Facts [BARD] + Alternate queries [GPT] + Documents [GPT] | 0.5580 (+42.75%) | 0.3049 (+35.69%) | 0.8672 (+17.17%) |
| RM3 | 0.4380 (+12.04%) | 0.2349 (4.5%) | 0.7736 (4.5%) |

## VII. Conclusion

In conclusion, we observed that the subtasks generating content of large length such as Documents, Essay, Facts and Summary are having higher gain value (in regard to the metrics) in both GPT-3.5 and BARD. Also as Trec-dl 19 uses passages underneath, it could be the reason why "Document" subtask has the best MAP and Recall@1k values. The reason could also be that they are able to add more words within the context of the query and the combination of 2 or 3 subtasks results in better metric gain compared with combination of all of the subtasks. If this system was deployed to a web browser to answer in real-time then this result of combination could help in better retrieval of the information. We also believe that certain combinations that are providing better results than the others are due to the fact the text generated by the subtasks are different from each other and that could result in better refined query. Further the threshold of using LLMs in query expansion can only be found out when we train an LLM with the dataset and compare our results with that. Overall, LLM beats the standard expansion method RM3 by a huge margin.

## References

[1] Iain Mackie, Shubham Chatterjee, and Jeffrey Dalton. Generative relevance feedback with large language models. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, July 2023. doi:10.1145/3539618.3591992.