Decoding a Neural Retriever's Latent Space for Query Suggestion

Leonard Adolphs[†] **Michelle Chen Huebscher**[‡] **Christian Buck**[‡]

Sertan Girgin[‡] **Olivier Bachem**[‡] **Massimiliano Ciaramita**[‡] **Thomas Hofmann**[†]

[†]ETH Zürich
ladolphs@inf.ethz.ch

[‡]Google Research

Abstract

Neural retrieval models have superseded classic bag-of-words methods such as BM25 as the retrieval framework of choice. However, neural systems lack the interpretability of bag-of-words models; it is not trivial to connect a query change to a change in the latent space that ultimately determines the retrieval results. To shed light on this embedding space, we learn a "query decoder" that, given a latent representation of a neural search engine, generates the corresponding query. We show that it is possible to decode a meaningful query from its latent representation and, when moving in the right direction in latent space, to decode a query that retrieves the relevant paragraph. In particular, the query decoder can be useful to understand "what should have been asked" to retrieve a particular paragraph from the collection. We employ the query decoder to generate a large synthetic dataset of query reformulations for MSMarco, leading to improved retrieval performance. On this data, we train a pseudo-relevance feedback (PRF) T5 model for the application of query suggestion that outperforms both query reformulation and PRF information retrieval baselines.

1 Introduction

Neural encoder models (Karpukhin et al., 2020; Ni et al., 2021; Izacard et al., 2021) have improved document retrieval in various settings. They have become an essential building block for applications in open-domain question answering (Karpukhin et al., 2020; Lewis et al., 2020b; Izacard and Grave, 2021), open-domain conversational agents (Shuster et al., 2021; Adolphs et al., 2021), and, recently, language modeling (Shuster et al., 2022). Neural encoders embed documents and queries in a shared (or joint) latent space, so that paragraphs can be ranked and retrieved based on their vector similarity with a given query. This constitutes a conceptually powerful approach to discovering semantic similarities between queries and documents that is often found to be more nuanced than simple term frequency statistics typical of classic sparse representations. However, such encoders may come with shortcomings in practice. First, they are prone to domain overfitting, failing to consistently outperform bag-of-words approaches on out-of-domain queries (Thakur et al., 2021). Second, they are notoriously hard to interpret as similarity is no longer controlled by word overlap, but rather by semantic similarities that lack explainability. Third, they may be non-robust as small changes in the query can lead to inexplicably different retrieval results.

In bag-of-words models, it can be straightforward to modify a query to retrieve a given document: e.g., following insights from *relevance feedback* (Rocchio, 1971), by increasing the weight of terms contained in the target document (Adolphs et al., 2022; Huebscher et al., 2022). This approach is not trivially applicable to neural retrieval models as it is unclear how an added term might change the latent code of a query.

In this paper, we look into the missing link connecting latent codes back to actual queries. We thus propose to train a "query decoder", which maps embeddings in the shared query-document space to query strings, inverting the fixed encoder of the neural retriever (cf. Figure 1a). As we will show, such a decoder lets us find queries that are optimized to retrieve a given target document. It deciphers what information is in the latent code of a document and how to phrase a query to retrieve it.

We use this model to explore the latent space of a state-of-the-art neural retrieval model, GTR (Ni et al., 2021). In particular, we leverage the structure of the latent space by traversing from the embedding of a specific query to its human-labeled gold paragraph and use our query decoder to generate reformulation examples from intermediate points along the path as shown in Figure 1b. We find that using this approach, we can generate a large

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 8786 - 8804

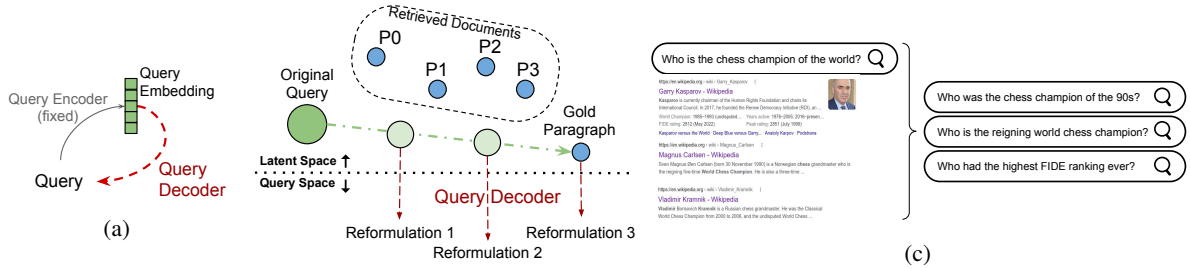


Figure 1: We train a query decoder (QD) model that inverts the shared encoder of a neural retrieval model (a). Then, we leverage the structure of the latent space of a neural retrieval model by traversing from query to gold paragraph embeddings and using our query decoder to generate a dataset of successful query reformulations (b). Finally, we train a pseudo-relevance feedback query suggestion model on this dataset that predicts promising rewrites, given a query and its search results (c).

dataset of query reformulations on MSMarco-train (Nguyen et al., 2016) that improve retrieval performance without needing additional human labeling. We use this dataset to train a pseudo-relevance feedback (PRF) query suggestion model. Here, we fine-tune a T5-large model (Raffel et al., 2020) that uses the original query, together with its top-5 GTR search results, as the input context to predict a query suggestions as depicted in Figure 1c. We show that our model provides fluent, diverse query suggestions with better retrieval performance than various baselines, including a T5 model trained on question editing (Chu et al., 2020), and a PRF query expansion model (Pal et al., 2013).

We make the resources to reproduce the results publicly available[1].

2 Related Work

Neural Retriever Classic retrieval systems such as BM25 (Robertson and Zaragoza, 2009) use term frequency statistics to determine the relevancy of a document for a given query. Recently, neural retrieval models have become more popular and started to outperform classic systems on multiple search tasks. Karpukhin et al. (2020) use a dual-encoder setup based on BERT-base (Devlin et al., 2019), called DPR, to encode query and documents separately and use maximum inner product search (Shrivastava and Li, 2014) to find a match. They use this model to improve recall and answer quality for multiple open-domain question-answer datasets, including OpenQA-NQ (Lee et al., 2019). Ni et al. (2021) show that scaling up the dual encoder architecture improves the retrieval performance. They train a shared dual encoder model, based on T5 (Raffel et al., 2020), in a multi-stage manner, including fine-tuning on MSMarco (Nguyen et al., 2016), and evaluate on the range of retrieval tasks of the BEIR benchmark (Thakur et al., 2021). Izacard et al. (2021) show that one can train an unsupervised dense retriever and be competitive against strong baselines on the BEIR benchmark.

Xiong et al. (2021) propose approximate nearest neighbor negative contrastive learning (ANCE) to learn a dense retrieval system. On top of this dense retriever, Li et al. (2022) consider a pseudo-relevance feedback method. Other than our approach, this method does not provide the user with rephrased queries.

Applications of Neural Retrievers Neural retrieval models have been at the core of recent improvements among a range of different NLP tasks. Lewis et al. (2020b) augment a language generation model, BART (Lewis et al., 2020a), with a DPR neural retriever and evaluate on multiple knowledge-intensive NLP tasks; most notably, they improve over previous models on multiple open-domain QA benchmarks using an abstractive method.

Izacard and Grave (2021) propose the Fusion-in-Decoder method to aggregate a large set of documents from the neural retriever and provide them to the model during answer generation. Their focus is on open-domain QA where they significantly outperform previous models when considering a large set of documents during decoding.

Shuster et al. (2021) use neural retrieval models to improve conversational agents in knowledge-grounded dialogue. They show that the issue of hallucination – i.e., generating factual incorrect knowledge statements – can be significantly re-

[1] https://github.com/leox1v/query_decoder

duced when using a neural-retriever-in-the-loop architecture. Separating the retrieval-augmented knowledge generation and the conversational response generation can further improve the issue of hallucination in knowledge-grounded dialogue and helps fuse modular QA and dialogue models (Adolphs et al., 2021). Recently, retrieval query generation approaches have been proposed to improve open-domain dialogue (Komeili et al., 2021) and language modeling (Shuster et al., 2022).

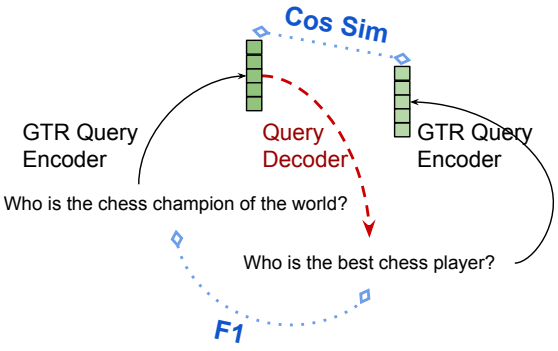
Query Generation Query optimization is a long-standing problem in IR (Lau and Horvitz, 1999; Teevan et al., 2004). Recent work has investigated query refinement with reinforcement learning for Open Domain and Conversational Question Answering (Nogueira and Cho, 2017; Buck et al., 2018; Wu et al., 2021).

The methods presented in this paper are a natural complement to the work of Adolphs et al. (2022), who propose a heuristic approach to generate multi-step query refinements, used to train sequential query generation models for the task of *learning to search*. Their method is also inspired by relevance feedback, but they seek to reach the gold document purely in language space, by brute force exploration. For this purpose, they use specialized search operators to condition the retrieval results as desired. Huebscher et al. (2022) show that, when paired with a hybrid sparse/dense retrieval environment, the search agents trained on this kind of synthetic data combine effective corpus exploration, competitive performance and interpretability.

Web-GPT (Nakano et al., 2021) presents an end-to-end search modeling approach based on human demonstrations, in a similar spirit our work could be seen as way of involving humans-in-loop by proposing better queries.

Fixed-vector decoders Probabilistic decoders mapping from a fixed size vector space to natural language have also been explored in auto-encoder settings. A key challenge in this line of work lies in obtaining decoders that are robust, *i.e.*, they generate natural text for a variety of input vectors. Bowman et al. (2016) proposed using a RNN-based language model in combination with variational autoencoders (VAE) (Kingma and Welling, 2013) which add Gaussian space to the decoder input. Zhao et al. (2018) proposed the use of Adversarial Autoencoder (AAE) (Makhzani et al., 2015) to which Shen et al. (2020) added data denoising by randomly dropping words in the input and the reconstructing the full output.

Recently, RNN-based decoders have been replaced by Transformer-based language models (Vaswani et al., 2017), for example by Montero et al. (2021), Park and Lee (2021) and Li et al. (2020).



Data	F1	Cos Sim
MSMarco	0.750	0.960
NQ	0.886	0.980

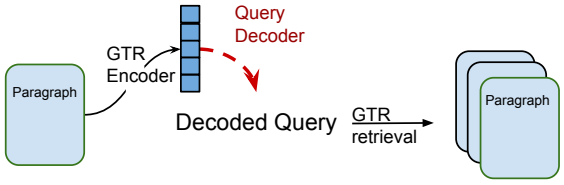
Table 1: Decoding metrics of the Query Decoder (QD) based on the GTR-base neural retrieval model. The F1 score is the F1 word overlap between the original query, of MSMarco or NQ, and the output of the query decoder model when provided with the GTR encoding of the query. The cosine similarity is measured between the re-encoding of the generated query and the encoding of the original query. The figure above depicts the metrics visually with a toy example for clarity.

3 Query Decoder

Training We train a T5 (Raffel et al., 2020), decoder-only model, to (re-)generate a query from its embedding obtained in a neural retrieval model. As training data, we use a subset of 3 million queries of the PAQ dataset (Lewis et al., 2021). We use the GTR-base (Ni et al., 2021), shared-encoder model, to generate the embeddings and use the queries as the targets. The objective of the query decoder learning is to invert the mapping of the *fixed* GTR encoder model, as visually depicted in Figure 1a. More training details of the query decoder are provided in Appendix A.2.1.

Query Reconstruction Evaluation We consider the round-trip consistency as a first step in evaluating the query decoder's effectiveness. A query q is encoded via GTR and then decoded by our decoder to generate q'. We use queries from MS-Marco, and NQ test sets of the BEIR benchmark (Thakur et al., 2021). As a first metric, we compute

the F1 score between the original q and its reconstruction q'. Since word-overlap is imperfect in measuring query drift, we further re-encode q' and compare its latent code with the code for q via their cosine similarity. The results of these evaluations are reported in Table 1, where we also provide an illustrative example of the proposed approach. For both datasets, MSMarco and NQ, the metrics of F1 and cosine similarity are generally high, indicating that the GTR code carries information that allows for close approximate query reconstruction.



Data	Top1	Top3	Top5
MSMarco	0.551	0.737	0.796
NQ	0.721	0.863	0.897

Table 2: Share of gold paragraphs for which we can decode a query that retrieves the given paragraph within its top-k GTR search results. The figure above depicts the metric evaluation visually for clarity.

Paragraph to Query Evaluation Many interesting use cases rely on the ability to generate queries from passages of text (Du et al., 2017; Kumar et al., 2018). As GTR embeds document paragraphs and queries into the the same space, the query decoder can also be used to invert the retrieval process. We thus evaluate the decoder quality by starting from a document paragraph, decoding a query from its embedding and then running the GTR search engine on that query to check if this query retrieves the desired paragraph as a top-ranked result. We test this in an experiment with human-labeled gold paragraphs from MSMarco and NQ, using top-k as the success metric. The results reported in Table 2 are very encouraging in that the desired paragraph is indeed found very often among the topmost GTR search results. Two example paragraph decodings from MSMarco are shown in Table 3; for both decodings, the gold paragraph is retrieved at the top position.

Latent Space Traversal Decoding We have shown that query decoding can reconstruct queries and that it can find retrieval queries for target passages. We now turn to a more concrete practical

Original Query
nebl coin price [Rank: 2]
Decoding from Gold Paragraph
what is the current price of neblio today belo [Rank: 1]
Gold Paragraph
Neblio Price Chart US Dollar (NEBL/USD) Neblio price for today is $16.3125. It has a current circulating supply of 12.8 Million coins and a total volume exchanged of $9,701,465

Original Query
when is champaign il midterm elections [Rank: 3]
Decoding from Gold Paragraph
when is the general election in illinois 2018 [Rank: 1]
Gold Paragraph
Illinois elections, 2018. A general election will be held in the U.S. state of Illinois on November 6, 2018. All of Illinois' executive officers will be up for election as well as all of Illinois' eighteen seats in the United States House of Representatives.

Table 3: Examples of query decodings from the gold paragraph. The rank indicates the retrieval position of the gold paragraph using the corresponding query.

application, namely to automatically generate a data set of query reformulations, from which strategies for interactive retrieval can be learned. In this context, reformulated queries should remain semantically similar to the original query and not overfit to the target passage. They should be somewhat 'in between' the query and the gold passage, as any passage is likely to contain answers to multiple, different questions. This can be operationalized by decoding queries from points along the line connecting the embeddings of the query and its target passage as depicted in Figure 1b.

To validate this idea, we apply it to the MSMarco and NQ retrieval dataset where each query is paired with a human-labeled gold paragraph. In particular, we move in k equidistant increments from the original query embedding $\mathbf{q}$ to the gold paragraph embedding $\mathbf{d}$, i.e.

$$\mathbf{q}_\kappa = \mathbf{q} + \frac{\kappa}{k}(\mathbf{d} - \mathbf{q}) \qquad \kappa = 0, \ldots, k \quad (1)$$

and generate a reformulation at each step.[2] As a sanity check, Figure 3 shows the average retrieval performance of the decoded queries when moving from the original query embedding to the gold paragraph embedding for MSMarco and NQ. For both datasets, the normalized discounted cumulative gain (nDCG) (Järvelin and Kekäläinen, 2002)

[2]We underline that this procedure can be seen as a latent space equivalent of the 'Rocchio Session' process for generating synthetic search sequences of Adolphs et al. (2022).

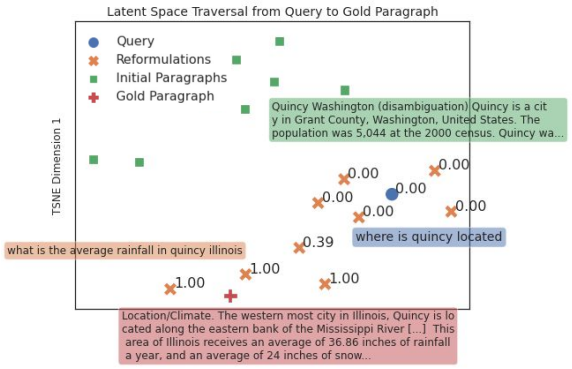


Figure 2: Visualization of the latent-space traversal from query to gold paragraph, using 2D t-SNE. The blue point denotes the embedding of the original query "where is quincy located". The green squares are the embeddings of the retrieved paragraphs for this query. The closest one about "Quincy Washington" is shown in the green text bar. The orange crosses denote the embeddings of the reformulations of the query decoder when moving to the gold paragraph depicted as the red plus. The orange and red text bars show the final reformulation and the gold paragraph text, respectively. The number above the query and reformulations show the nDCG score. As the gold paragraph is describing the climate of Quincy in addition to its location, a reformulation about the "average rainfall in quincy illinois" retrieves the desired paragraph.

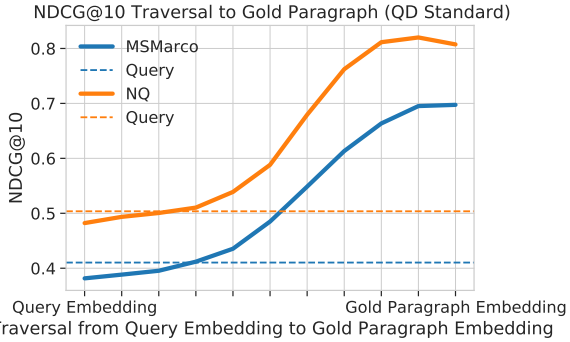


Figure 3: The normalized discounted cumulative gain (nDCG) of the reformulations from the query decoder when moving the input code from the embedding of the query to the embedding of the gold paragraph. Decoding closer to the gold paragraph embedding leads to queries with improved average retrieval performance. The initial decodings nDCG scores are slightly lower than from the original query due to the reconstruction loss of the query decoder.

improves steadily and plateaus, then slightly dips, only when getting close to the gold paragraph embedding. We hypothesize that two effects are at work here that explain this dip: (i) the closer one moves towards the gold passage embedding, the more the query decoder operates out-of-distribution as it is trained on query embeddings. The joint latent space is sparse and likely characterized by distinct regions for queries and passage embeddings, which have different properties (e.g., length or surface structure). (ii) A passage might answer several questions. When decoding from an embedding close to the paragraph, these might start being conflated.

Examples We provide a visual example of the latent traversal approach in Figure 2 where we project the latent space to 2 dimensions using t-SNE (Hinton and Roweis, 2002). The plot shows that for the ambiguous query "where is quincy located" (blue dot), the gold paragraph about the climate of Quincy, Illinois (red plus), is far away from the top-10 retrieved documents (green squares). Traversing the latent space from the query towards the gold paragraph leads to improved reformulations (orange crosses), as is evident from the shrinking distance to the gold paragraph and by the improved nDCG score.

Semantically, the reformulations move to questions about the climate of Quincy, as this is the main topic of the gold paragraph. The full sequence of reformulations is shown in Table 15. Another example is shown in Table 4; similarly, we see that the decoded queries move semantically from the general question of average annual return on the *stock market* to the more specific question of return at the *S&P stock exchange*. Many more examples are provided in Appendix A.5.

4 Query Suggestion Model

Dataset Generation We generate a dataset of query reformulations using the latent space traversal decoding as described in the previous section. In particular, for the 532,761 queries of the MSMarco-train dataset, we leverage GTR's learned latent space structure and move towards the embedding of their gold paragraph. At $k = 20$ intermediate steps on this path, we use our query decoder to generate reformulations.

For more than 80% of the queries, we find at least one optimal reformulation that retrieves the gold paragraph at the top position. In Figure 4, we show histograms of nDCG and the inner product to the gold paragraph for the original query versus the best-found reformulation. The metrics show that the latent space traversal helps us discover good

Original Query	
average yearly return on stock market	[0.00]
Decodings during Traversal	
what is the average annual return on stock market	[0.00]
average return on a stock market year	[0.00]
what is the average annual return on stock market	[0.00]
what is the average return on stock in a year	[0.00]
what is the average return in a stock market	[0.00]
what is the average annual return in stock (s&p)	[0.36]
what is the average return on the stock market (s&p)	[0.36]
what is the average return on the s&p stock exchange at a time	[1.00]
what is the average return in s&p stock at a time	[0.36]
what is the average annual return of the s&p stock exchange (best)	[1.00]
Gold Paragraph	
The S&P 500 gauges the performance of the stocks of the 500 largest, most stable companies in the Stock Exchange. It is often considered the most accurate measure of the stock market as a whole. The current average annual return from 1926, the year of the S&Ps inception, through 2011 is 11.69%. That's a long look back, and most people aren't interested in what happened in the market 80 years ago.	

Table 4: Example of a successful traversal on an MS-Marco query. The nDCG@10 score of each query is provided in the brackets. Here, we traverse the latent space in $k = 10$ equidistant steps from the embedding of the original query "average yearly return on stock market", which results in an nDCG retrieval score of 0, to the embedding of the gold paragraph. The queries decoded from a latent code close to the gold paragraph, focus on the returns of the S&P (as the gold paragraph) and lead to improved retrieval results.

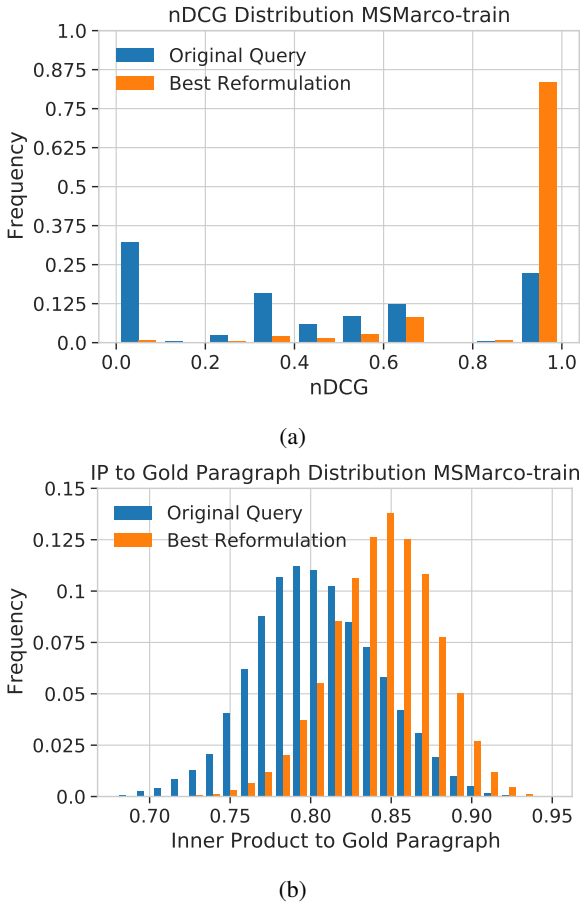


Figure 4: The histogram of nDCG (a) and inner product with the gold paragraph embedding of the original query vs. the best reformulation found with the latent-space traversal approach on MSMarco-train.

query reformulations that lead to massively improved retrieval performance and are closer to the corresponding gold paragraphs in latent space.

We filter the dataset to only contain "successful" reformulations to train the reformulation model. Here, we require a reformulation to have an nDCG of 1 (i.e., retrieve the gold paragraph at the top position), to improve the nDCG compared to the original query, and its embedding to have a larger inner product with the gold paragraph than the original query. Using this approach, we generate a dataset of 863,307 successful query rewrites. As the example in Table 4 and Appendix A.5 show, the decoded queries do not always have human-like fluency and for some sequences intent shift occurs when decoding closer to the paragraph. This is one reason we're moving in increments from the original query to the gold paragraph instead of directly decoding it.

Interestingly, however, we find that this noise of the dataset is unspecific enough that it gets smoothed out during model training as described in the following paragraph. More details about this dataset are provided in Sec. A.3.

Model Training We use the reformulation dataset to train two query suggestion model variants. First, we train a model on the plain reformulation examples, from original query to "successful" rewrite. As a second, more powerful approach, we train a model with pseudo-relevance feedback (PRF); here, we provide GTR's top-5 search results for the original query as additional context to the model. Both models are fine-tuned from the T5-large (Raffel et al., 2020) pre-training checkpoint. Consequently, we name the models in the following way:

- **qsT5-plain**: A T5 **q**uery **s**uggestion model trained on the **plain** generated reformulation

examples (no pseudo-relevance feedback) of MSMarco-train, mapping from query to query reformulation.

- **qsT5**: A T5 **q**uery **s**uggestion model trained on the generated reformulation examples of MSMarco-train, where the input is augmented with the content of GTR's top-5 retrieved search results for the original query, mapping from query and search results to query reformulation.

The details and hyperparameters of the model training are provided in Appendix A.2.2.

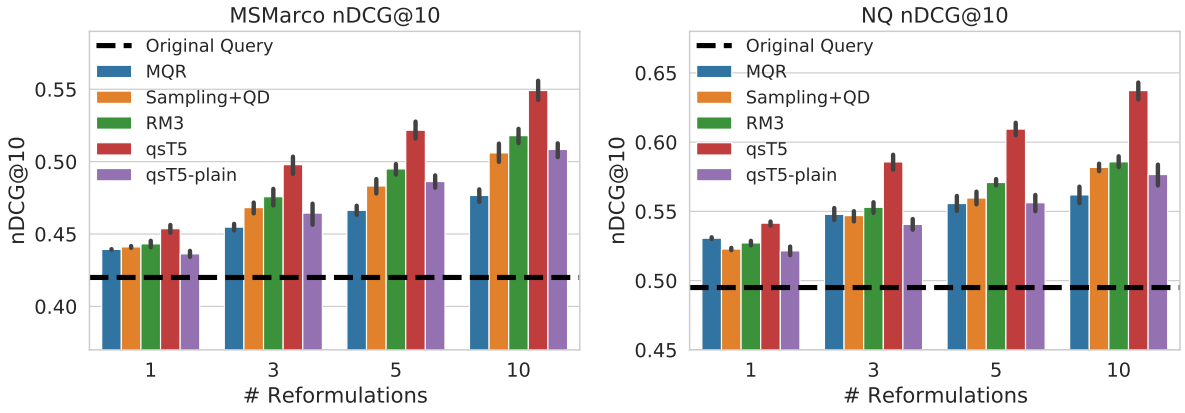
Baseline Models To measure the effectiveness of our query suggestion model, we benchmark it against multiple baselines. The baselines are meant to cover various angles of competitive approaches to query suggestion, namely training on human-generated question-edit histories, a classic RM3 pseudo-relevance feedback query expansion method, and a latent space sampling approach utilizing our query decoder. In the following, we introduce the three baselines in detail.

- **MQR**: We train a T5-large model on the "multi-domain question rewriting" (MQR) (Chu et al., 2020) dataset. This dataset consists of 427,719 human-contributed Stack Exchange question-edit histories, mapping from ill-formed to well-formed. While this is a relatively large training dataset, our synthetic generations dataset is roughly double in size with 863,307 rewrites, yet without any human edits. This baseline captures the effect of turning a query to a well-formed question to improve retrieval performance. It does not use PRF but maps from query to reformulation, as our qsT5-plain model. Training details of this model are provided in Section A.2.3 in the Appendix.
- **RM3**: We employ RM3 (Jaleel et al., 2004) as a strong pseudo-relevance feedback baseline. In particular, we use a query expansion approach that uses the formula described in Eq. 20 of Pal et al. (2013) with $\mu = 2500$ to determine the most relevant terms of the top-5 retrieved documents. Then, each suggestion of the model consists of the original query together with one of the determined relevant terms.
- **Sampling+QD**: To check how much of the retrieval performance gain is due to an ensembling effect in latent space, we compare against a random sampling baseline that includes our query decoder (QD). In particular, we sample a point uniformly at random from an epsilon-ball around the embedding of the original query and use the query decoder to decode that point to a query. This baseline does not use PRF.

Original Query
who created spiritual gangster
MQR
Who created the Spiritual Gangster?
Who created the "spiritual gangster" storyline?
Who created the "spiritual gangster"?
RM3
who created spiritual gangster spiritual
who created spiritual gangster modern
who created spiritual gangster inspired
Sampling+QD
who created gangster a spiritual & egantious
who created spiritual gangster -gangster
who created spiritual gangster
qsT5
who is the founder of spiritual gangsters
who created the spiritual gangster (spiritual yogi)
what is the spiritual gangster movement
qsT5-plain
who are the founders of the gangster spirit band
how many gangsters were formed in white supreme
who was the members of the gangster supremes
Gold Paragraph
About Spiritual Gangster. Spiritual Gangster represents a new generation of yogis seeking balance between the ancient practice of yoga and the modern world. Founded by Vanessa Lee and Ian Lopatin, this newly borne brand calls for high vibration living and radiating love shore-to-shore, person-to-person, heart-to-heart.

Table 5: Examples of the top query suggestions for the different models for the query "who created spiritual gangster". The final row shows the human-labeled gold paragraph.

Evaluation We evaluate the query suggestion models on the MSMarco and NQ test sets. For each example, we generate up to 10 suggestions using nucleus sampling (Holtzman et al., 2020). Our ultimate goal is to provide users with at least one reformulation that better captures their search intent. As we assume the gold paragraph captures the information need of the user, we evaluate if, within a small set of reformulations, there is a query that would lead them closer to that paragraph; i.e., we measure the maximum nDCG@10 of the top-k reformulations and the original query. We provide the results in Figure 5.



(a) (b)

Figure 5: Retrieval metrics of the query suggestion models on the MSMarco (a) and NQ (b) test sets. The dashed line shows the nDCG@10 score of the original query. The bars represent the nDCG@10 of the best 1, 3, 5, and 10 reformulations (including the original query), respectively, of different models. The error bars show the 95 percent confidence interval when doing bootstrap sampling from up to 10 generations of the models. The RM3 and qsT5 models are using pseudo-relevance feedback, i.e., information about the top-retrieved paragraphs.

We see that our qsT5 model significantly outperforms all baselines on both datasets. Notably, it substantially improves upon the RM3 pseudo-relevance feedback baseline; this indicates that our full reformulation approach is more powerful for neural retrievers than a well-established query expansion technique.

The large gap between qsT5 and qsT5-plain validates the importance and usefulness of conditioning on the initial search results.

Successfully rewriting the query to a well-formed variant benefits this task as indicated by the improved nDCG performance of the non-PRF baseline of the T5 model trained on MQR (blue) over the original query (dashed line). The qsT5-plain model outperforms the MQR model when considering multiple reformulations on MSMarco, indicating that in some cases our model learns successful rewriting beyond improving fluency.

The qsT5-plain is mostly on par with sampling randomly around the embedding of the original query and using our query decoder to generate a reformulation; hence, we can speculate that the main benefit of this non-PRF model comes from an ensembling effect of generating suitable reformulations around the neighborhood of the original query. Again, this reinforces the benefit of pseudo-relevance feedback for the application of query suggestion.

Additional plots showing the inner product metric for this experiment and a table summarizing the numbers are provided in Appendix A.1.

Diversity & Fluency To quantitatively highlight the characteristics of the evaluated query suggestion models, we report Self-BLEU (Zhu et al., 2018) and perplexity of a language model as proxies for diversity and fluency, respectively, in Table 6. Self-BLEU is measured between 10 suggestions for a given query and averaged across the dataset, where a low Self-BLEU indicates large diversity between suggestions. For the perplexity evaluation, we employ the T5-base language model trained on C4 (Raffel et al., 2020) and measure the average per-token perplexity of all suggestions for a given dataset; here, we associate lower perplexity with higher fluency of the suggestions.

Table 6 shows that the MQR baseline generates the most fluent queries, but with low diversity compared to our reformulation approaches. The RM3 query expansions score worst in diversity as they always use the original query as a base. Our qsT5 model scores second best in diversity, with a good comparative perplexity, only surpassed by the qsT5 variant without PRF, due to the fact that this model does not focus on the "narrowed-down" topics of the retrieved results. Notably, the perplexity is higher for the NQ dataset than for MSMarco due to the nature of queries in NQ being closer to well-formed questions as opposed to 'search engine queries'.

Examples In Table 5, we cherry-pick a representative example of query suggestions for the different models. This example showcases typical be-

Model	MSMarco S-BLEU	MSMarco PPL	NQ S-BLEU	NQ PPL
Original Query	-	1622.2	-	217.9
MQR	46.1	59.6	61.4	56.8
RM3	74.8	1562.6	88.0	309.5
Sampling+QD	23.7	726.0	26.6	687.5
qsT5	17.8	247.8	18.4	223.2
qsT5-plain	9.2	196.6	7.6	249.8

Table 6: Self-BLEU (Zhu et al., 2018) and Perplexity (PPL) for the query suggestions of the different models on MSMarco and NQ. Self-BLEU is measured between 10 suggestions of a model for a given query and then averaged across the dataset to provide information about the diversity of the suggestions. Perplexity is measured per token based on a T5 language model to provide a relative comparison of fluency of the query suggestions.

haviors of the models. The MQR model is trained on turning ill-formed into well-formed questions. Hence, it usually produces grammatical but low diversity reformulations, especially when the original query is already close to a well-formed question. The relatively high Self-BLEU score amongst its reformulations for a given query, reported in Table 6, supports the argument of limited diversity.

The RM3 model appends the most relevant terms to the original query and therefore has the lowest overall diversity (i.e., highest Self-BLEU). The Sampling+QD model can result in non-grammatical or even nonsensical queries depending on the sampled point in latent space. While the qsT5 model can utilize the top-retrieved search results to form reformulations that are in accordance with the topic of the query (e.g., "yogi" in the example of Table 5), the qsT5-plain needs to rely on its internal world knowledge stored in its parameters. It thus cannot connect "gangster" with "yogi" here.

5 Conclusion

Dual encoders have reset the standard in IR. However, language-based inverted index architectures still hold their ground, especially in out-of-domain evaluations (Thakur et al., 2021). To help further our understanding of the connections, and potential, between the two, we propose a method that relies on a query decoder to map back to language space the latent codes generated by the encoder.

The interplay between latent and language representations, in combination with a simple goal-directed mechanism for traversing the shared query-document space, allows us to generate a large synthetic dataset of query reformulations on which we train a pseudo-relevance feedback query suggestion model that characteristically tries to predict the location of the target document.

Our contribution is twofold: (i) we develop a generic way to generate training data for directional query refinement by traversing the latent space between queries and relevant documents, and (ii) we build a powerful reformulation model that we evaluate on a novel benchmark inspired by the query suggestion task. Suggestions are typically well-formed, diverse and more likely to lead to the right document than competing methods.

6 Limitations

A proper user study would provide a valuable complement to the current evaluation and contribute to a fuller picture. However, this presents significant challenges that are beyond the scope of the current work. For instance, is not trivial to adequately design a meaningful task for human raters conducive to good agreement, e.g., it may be inevitable to second-guess the original query intent in the presence of unexpected interpretations brought to the surface by the suggestions. For the time being, we feel the automatic evaluation proposed here will be more valuable, as it makes direct comparison and reproducibility straightforward.

Secondly, it seems sensible to further evaluate the query suggestions in an end-to-end IR task. Preliminary experiments in this direction using MS Marco proved somewhat inconclusive, while they introduce significant complexity. The data annotations are sparse (one passage per query, by and large) and it is often the case that multiple relevant passages exist for the same query.[3] This makes reranking a crucial but faulty component, opening up a somewhat orthogonal front. The ideal evaluation would rely on a deeper manual analysis for a limited query set, e.g., TREC-style (e.g., cf. Craswell et al. (2020)).

We leave both for future work.

Acknowledgements

We would like to thank Léonard Hussenot for his insightful feedback and the support with the Diversity & Fluency experiments. Furthermore, we thank Wojciech Gajewski, Sascha Rothe, and Lierni Sestorain Saralegui for helpful discussions and feedback throughout the course of the project.

[3] Including near duplicate passages.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Leonard Adolphs, Benjamin Börschinger, Christian Buck, Michelle Chen Huebscher, Massimiliano Ciaramita, Lasse Espeholt, Thomas Hofmann, Yannic Kilcher, Sascha Rothe, Pier Giuseppe Sessa, and Lierni Sestorain. 2022. Boosting search engines with interactive agents. *Transactions on Machine Learning Research*.

Leonard Adolphs, Kurt Shuster, Jack Urbanek, Arthur Szlam, and Jason Weston. 2021. Reason first, then respond: Modular generation for knowledge-infused dialogue. *CoRR*, abs/2111.05204.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, Berlin, Germany. Association for Computational Linguistics.

Christian Buck, Jannis Bulian, Massimiliano Ciaramita, Wojciech Gajewski, Andrea Gesmundo, Neil Houlsby, and Wei Wang. 2018. Ask the right questions: Active question reformulation with reinforcement learning. In *International Conference on Learning Representations*.

Zewei Chu, Mingda Chen, Jing Chen, Miaosen Wang, Kevin Gimpel, Manaal Faruqui, and Xiance Si. 2020. How to ask better questions? a large-scale multi-domain dataset for rewriting ill-formed questions. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7586–7593.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. 2020. Overview of the TREC 2020 deep learning track.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension.

Geoffrey E Hinton and Sam Roweis. 2002. Stochastic neighbor embedding. In *Advances in Neural Information Processing Systems*, volume 15. MIT Press.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Michelle Chen Huebscher, Christian Buck, Massimiliano Ciaramita, and Sascha Rothe. 2022. Zero-shot retrieval with search agents and hybrid environments.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Towards unsupervised dense information retrieval with contrastive learning. *CoRR*, abs/2112.09118.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online. Association for Computational Linguistics.

Nasreen Jaleel, James Allan, W. Croft, Fernando Diaz, Leah Larkey, Xiaoyan Li, Mark Smucker, and Courtney Wade. 2004. Umass at trec 2004: Novelty and hard.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated Gain-Based Evaluation of IR Techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Mojtaba Komeili, Kurt Shuster, and Jason Weston. 2021. Internet-augmented dialogue generation. *CoRR*, abs/2107.07566.

Vishwajeet Kumar, Kireeti Boorla, Yogesh Meena, Ganesh Ramakrishnan, and Yuan-Fang Li. 2018. Automating reading comprehension by generating question and answer pairs. *CoRR*, abs/1803.03664.

Tessa Lau and Eric Horvitz. 1999. Patterns of Search: Analyzing and Modeling Web Query Rrefinement. In *Proceedings of the 7th International on User Modeling*.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020a. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020b. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Patrick Lewis, Yuxiang Wu, Linqing Liu, Pasquale Minervini, Heinrich Küttler, Aleksandra Piktus, Pontus Stenetorp, and Sebastian Riedel. 2021. PAQ: 65 million probably-asked questions and what you can do with them. *Transactions of the Association for Computational Linguistics*, 9:1098–1115.

Chunyuan Li, Xiang Gao, Yuan Li, Baolin Peng, Xiujun Li, Yizhe Zhang, and Jianfeng Gao. 2020. Optimus: Organizing sentences via pre-trained modeling of a latent space. *arXiv preprint arXiv:2004.04092*.

Hang Li, Shengyao Zhuang, Ahmed Mourad, Xueguang Ma, Jimmy Lin, and Guido Zuccon. 2022. Improving query representations for dense retrieval with pseudo relevance feedback: A reproducibility study. In *Advances in Information Retrieval*, pages 599–612, Cham. Springer International Publishing.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. 2015. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*.

Ivan Montero, Nikolaos Pappas, and Noah A. Smith. 2021. Sentence bottleneck autoencoders from transformer language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1822–1831, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2021. WebGPT: Browser-assisted question-answering with human feedback. *CoRR*, abs/2112.09332.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernández Ábrego, Ji Ma, Vincent Y. Zhao, Yi Luan, Keith B. Hall, Ming-Wei Chang, and Yinfei Yang. 2021. Large dual encoders are generalizable retrievers. *CoRR*, abs/2112.07899.

Rodrigo Nogueira and Kyunghyun Cho. 2017. Task-oriented query reformulation with reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 574–583, Copenhagen, Denmark. Association for Computational Linguistics.

Dipasree Pal, Mandar Mitra, and Kalyankumar Datta. 2013. Query expansion using term distribution and term association.

Seongmin Park and Jihwa Lee. 2021. Finetuning pretrained transformers into variational autoencoders. *arXiv preprint arXiv:2108.02446*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389.

J. J. Rocchio. 1971. Relevance feedback in information retrieval. In G. Salton, editor, *The Smart retrieval system - experiments in automatic document processing*, pages 313–323. Englewood Cliffs, NJ: Prentice-Hall.

Tianxiao Shen, Jonas Mueller, Dr.Regina Barzilay, and Tommi Jaakkola. 2020. Educating text autoencoders: Latent representation guidance via denoising. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 8719–8729. PMLR.

Anshumali Shrivastava and Ping Li. 2014. Asymmetric lsh (alsh) for sublinear time maximum inner product search (mips). In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Kurt Shuster, Mojtaba Komeili, Leonard Adolphs, Stephen Roller, Arthur Szlam, and Jason Weston. 2022. Language models that seek for knowledge: Modular search & generation for dialogue and prompt completion.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jaime Teevan, Christine Alvarado, Mark S. Ackerman, and David R. Karger. 2004. The Perfect Search Engine is Not Enough: A Study of Orienteering Behavior in Directed Search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogenous benchmark for zero-shot evaluation of information retrieval models. *CoRR*, abs/2104.08663.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zeqiu Wu, Yi Luan, Hannah Rashkin, David Reitter, Hannaneh Hajishirzi, Mari Ostendorf, and Gaurav Singh Tomar. 2021. CONQRR: Conversational query rewriting for retrieval with reinforcement learning. arxiv.org:2112.08558.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations*.

Junbo Zhao, Yoon Kim, Kelly Zhang, Alexander Rush, and Yann LeCun. 2018. Adversarially regularized autoencoders. In *International conference on machine learning*, pages 5902–5911. PMLR.

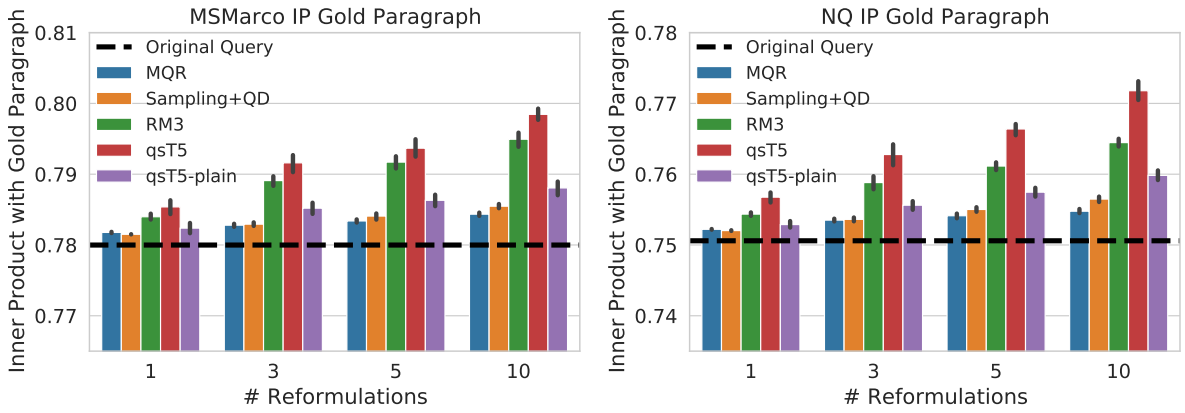
Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *SIGIR*, pages 1097–1100.

A Appendix

A.1 Additional Results of the Query Suggestion Model

Model	MSMarco				NQ			
	1	3	5	10	1	3	5	10
Original Query	.420	-	-	-	.495	-	-	-
MQR	.439 .001	.454 .005	.464 .005	.477 .004	.531 .001	.548 .008	.557 .009	.571 .005
Sampling+QD	.440 .001	.469 .005	.484 .007	.506 .013	.522 .001	.548 .005	.561 .006	.580 .005
RM3	.445 .003	.472 .016	.495 .009	.522 .011	.526 .003	.552 .012	.571 .006	.589 .012
qsT5	.455 .002	.496 .010	.519 .010	.554 .011	.541 .003	.582 .011	.615 .008	.637 .009
qsT5-plain	.440 .005	.470 .007	.488 .005	.508 .008	.520 .006	.543 .006	.553 .010	.577 .013

Table 7: Retrieval metric nDCG@10 of the query suggestion models on the MSMarco and NQ test sets. The numbers represent the nDCG@10 of the best 1, 3, 5, and 10 reformulations (including the original query), respectively, of different models. The small number indicates the standard deviation when doing bootstrap sampling from up to 10 generations of the models.



(a) (b)

Figure 6: Inner product of the best reformulation with the gold paragraph for the various query suggestion models on the MSMarco (a) and NQ (b) test sets. The dashed line shows the inner product of the original query with the gold paragraph. The bars represent the inner product with the gold paragraph of the best 1, 3, 5, and 10 reformulations (including the original query), respectively, of different models. The error bars show the 95 percent confidence interval when sampling repeatedly from up to 10 generations of the models. The RM3 and qsT5 models are using pseudo-relevance feedback, i.e., information about the top-retrieved paragraphs, while the rest is only mapping from query to query.

A.2 Training and Model Details

A.2.1 Query Decoder Model

We initialize the query decoder from the Tensorflow (Abadi et al., 2015) T5-base (Raffel et al., 2020) checkpoint for conditional language generation of the Huggingface transformers library (Wolf et al., 2020). We train and use only the decoder of the 220M parameter model with 12 layers, 12 heads, and a hidden state dimension of 768. We train the model on a random sample of 3M queries from the PAQ dataset (Lewis et al., 2021) with the hyperparameters provided in Table 8 for 130 hours on 16 Cloud TPU v3. Given the size of the model and the associated cost of training, we do not do an exhaustive grid search over the parameters but only sweep over four different values for the learning rate (1e-4, 3e-4, 5e-4, 1e-3). We determine the best model according to sequence classification accuracy on a held-out dev set.

Parameter	Value
Number Decoder Layers	12
Number Heads	12
Head Dimension	64
Embedding Dimension	768
MLP Dimension	2048
Batch Size	64
Dropout Rate	0.0
Base Learning Rate	0.0005
Warm-up Steps	100
Optimizer	AdamW
Weight Decay Rate	0.01
Finetuning steps on PAQ	5M

Table 8: Training Parameters for the T5-base Query Decoder model

A.2.2 qsT5 Query Suggestion Model

We initialize our qsT5 and qsT5-plain models from the T5-large checkpoint and train them with the parameters provided in Table 9 for ∼24 hours on 16 Cloud TPU v3 each. We prefix the query with the keyword "Query: " and each passage, for the qsT5 model, with the keyword "Paragraph: ". For this model we rely on standard hyperparameters and, given the cost of training, do not further grid search for better values.

Parameter	Value
Number Encoder Layers	24
Number Decoder Layers	24
Number Heads	16
Head Dimension	64
Embedding Dimension	1024
MLP Dimension	2816
Batch Size	128
Dropout Rate	0.1
Base Learning Rate	0.001
Warm-up Steps	1000
Optimizer	AdaFactor
Input Token Length	1024
Output Token Length	32
Finetuning steps (qsT5)	40k
Finetuning steps (qsT5-plain)	31k

Table 9: Training Parameters for the T5-large qsT5 and qsT5-plain Query Suggestion Models

A.2.3 MQR Query Suggestion Model

As with our qsT5 models, we initialize the MQR query reformulation model from the T5-large checkpoint. We train the model with the parameters provided in Table 10 for ∼3h on 16 Cloud TPU v3, and choose the best checkpoint according to sequence accuracy on the dev set. Training quickly overfits after about

2k step (∼16min). We increased the batch size to 2048 and found the adding dropout didn't help but did no further hyperparameter tuning.

Parameter	Value
Number Encoder Layers	24
Number Decoder Layers	24
Number Heads	16
Head Dimension	64
Embedding Dimension	1024
MLP Dimension	2816
Batch Size	2048
Dropout Rate	0.0
Fixed Learning Rate	0.001
Optimizer	AdaFactor
Input Token Length	32
Output Token Length	32
Finetuning steps	1800

Table 10: Training Parameters for the T5-large MQR Query Reformulation Model

A.3 Dataset Details

We experiment with a few different thresholds on what constitutes a successful reformulation for our generated dataset. We achieve the best results in terms of sequence classification accuracy of a held-out dev set of reformulations for the dataset described in the main body of the paper. Our final dataset of successful reformulations of MSMarco-train queries contains 863,207 successful query rewrites that are split among a training, development, and test set as reported in Table 11.

Split	Number of Examples
Train	768,372
Dev	86,478
Test	8,457
Total	863,307

Table 11: Successful Reformulation Dataset Details

A.4 Additional Reformulation Examples

Type	Query	nDCG@10
Original Query	shopko kennewick address	0.63
MQR	Is the address for the Shopko in Kennewick correct?	1.00
	What is this shopko in Kennewick, NY address?	1.00
	Is there a Shopko in Kennewick with the following address?	1.00
RM3	shopko kennewick address 867	0.50
	shopko kennewick address store	1.00
	shopko kennewick address 5500	0.50
Sampling+QD	who is the localization of koieko shopphan	0.00
	shopk kennewickko address in shopington	0.63
	shopko kennewick is located in which address	0.63
qsT5	what is the address of shopko located in kennewick washington	1.00
	what is the location of shopko in kennewick washington	1.00
	where is shopko store in kennewick washington	1.00
qsT5-plain	what time is open shopko at 867 kennewick	1.00
	what is the main store at shopko kanetown	0.00
	when does store for shopko located in north kansas	0.00
Gold Paragraph	Information about possible store closing and store hours for: ShopKo in Kennewick, Washington, 99336 Address: 867 North Columbia Center Blvd | Phone: (509) 736-0884 | Type: Store, Department Store, Retail More information:	

Table 12: Examples of the top query suggestions for the MSMarco query "shopko kennewick address". The final row shows the human-labeled gold paragraph and the nDCG@10 retrieval score is provided for each query. In this example, we see that the qsT5 model can leverage the PRF to successfully generate queries that focus on Kennewick in *Washington*. For the non-PRF models, this is not possible.

Type	Query	nDCG@10
Original Query	definition of a surge	0.50
MQR	What is the definition of a surge?	0.43
	What is the definition of a surge?	0.43
	What is a surge?	0.63
RM3	definition of a surge surge	0.33
	definition of a surge sudden	0.43
	definition of a surge increase	0.30
Sampling+QD	what is the definition of a surge	0.50
	a a surge to a begin in the surge definition	0.43
	what is the definition of a surge	0.50
qsT5	what is the definition of a surge in the sea	0.50
	what is the meaning of surge in a wave	0.63
	what is the definition of a surge in a sound	1.00
qsT5-plain	what is the swiveling of a rolling motion	0.00
	what is the meaning of a surge in the emotional side of a big wave	0.43
	the surge of a wave or ayurvedic chorus	0.00
Gold Paragraph	surge. 1. a strong, wavelike forward movement, rush, or sweep: the surge of the crowd. 2. a sudden, strong rush or burst: a surge of energy. 3. a strong, swelling, wavelike volume or body of something. 4. the rolling swell of the sea. 5. a swelling wave; billow. 6. the swelling and rolling sea. 7. a. a sudden rush or burst of electric current or voltage. b. a violent oscillatory disturbance.	

Table 13: Examples of the top query suggestions for the MSMarco query "definition of a surge". The final row shows the human-labeled gold paragraph and the nDCG@10 retrieval score is provided for each query. Here, we see that the qsT5 model provides more useful additions to the query than the simple rephrasing of the MQR model. It adds topical terms like "sea", "wave", or "sound" with which it is possible to obtain improved retrieval performance.

Type	Query	nDCG@10
Original Query	what aircraft can you fly with a ppl	0.29
MQR	What aircraft can you fly with a passenger?	0.00
	What aircraft can you fly with passengers?	0.00
	Which aircraft can you fly with a passenger?	0.00
RM3	what aircraft can you fly with a ppl pilot	0.36
	what aircraft can you fly with a ppl license	0.32
	what aircraft can you fly with a ppl private	0.43
Sampling+QD	what aircraft can you fly with a ppl	0.29
	what ships did you fly a learn park and can server (pplalo	0.00
	a able a flying wings able islands with a pistol pl	0.00
qsT5	what kind of airplane can you fly with a ppl	0.00
	what type of aircraft are you able to fly a private pilot	1.00
	what does a personal pilot (a pvl) mean	0.00
qsT5-plain	what are the types of aircraft that can be used for private pilots	1.00
	what kind of licenses are required to have a private pilot	0.32
	how many pilots are required to take a commercial flight	0.00
Gold Paragraph	The types of aircraft one may fly depends on what they are certified for, e.g. Airplane Single Engine land. See Categories and Classes for a list. Usually, a newly minted private pilot is certified to fly all planes in the generic category single engine piston, often abbreviated SEP. This includes the Cessna 172, PA-28, Diamond DA40, Robin DR400 and similar planes. More complex types, like those with retractable undercarriage or variable pitch props, require additional training and licensing.	

Table 14: Examples of the top query suggestions for the MSMarco query "what aircraft can you fly with a ppl"[4]. The final row shows the human-labeled gold paragraph and the nDCG@10 retrieval score is provided for each query. In this example only the two qsT5 models are able to add the term "private pilot" to the query that leads to good retrieval performance. Notably, the RM3 PRF baseline ranks these two terms separately as important and adds them to their suggestions.

A.5 Additional Traversal Examples

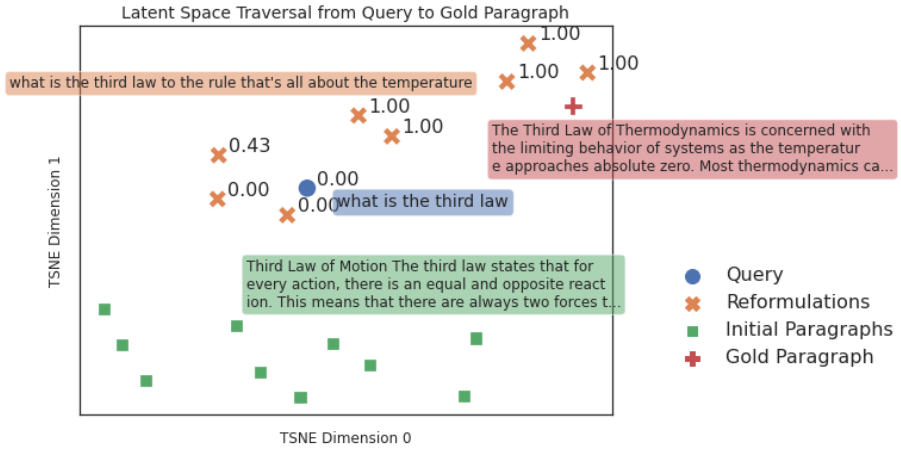


Figure 7: Visualization of the latent-space traversal from query to gold paragraph, using 2D t-SNE. The blue point denotes the embedding of the original query "what is the third law". The green squares are the embeddings of the retrieved paragraphs for this query. The closest one about the "Third Law of Motion" is shown in the green text bar. The orange "x's" denote the embeddings of the reformulations of the query decoder when moving to the gold paragraph depicted as the red plus. The orange and red text bars show the final reformulation and the gold paragraph text, respectively.

[4]The abbreviation PPL stands for private pilot license.

Type	Query	nDCG@10
Original Query	where is quincy located	0.00
Reformulations	where is quincy located in miami	0.00
	where is quincy located in a state	0.00
	where is the city of quincy located	0.00
	where is the state of quincy located	0.00
	where is the city of quincy located in illinois	0.39
	where is the climate of quincy located in illinois	1.00
	where does the water come from in quincy illinois	1.00
	what is the average rainfall in quincy illinois	1.00
Gold Paragraph	Location/Climate. The western most city in Illinois, Quincy is located along the eastern bank of the Mississippi River atop 90 foot limestone bluffs which overlook a wide expanse of the river and a natural harbor. Residents enjoy a moderate, four season climate where the sun shines nearly 68% of the time. This area of Illinois receives an average of 36.86 inches of rainfall a year, and an average of 24 inches of snowfall. The average winter temperature is 28 degrees and the average summer temperature is 79 degrees.	

Table 15: Example of a traversal on an MSMarco query. Here, we traverse the latent space in equidistant steps from the embedding of the original query "where is quincy located", which results in an nDCG@10 retrieval score of 0, to the embedding of the gold paragraph. The queries decoded from a latent code close to the gold paragraph are concerned with the climate of Quincy, Illinois, which is the focus of the gold paragraph.

Type	Query	nDCG@10
Original Query	weather year round in new york city	0.00
Reformulations	what is the weather year in new york city	0.33
	what is the weather in new york city each year	0.00
	what is the weather in new york city	0.00
	what is the hottest season in new york city	0.39
	when is the coldest month in new york	0.30
	when is the hottest month in new york	1.00
Gold Paragraph	New York: Annual Weather Averages. July is the hottest month in New York with an average temperature of 25°C (76°F) and the coldest is January at 2°C (35°F) with the most daily sunshine hours at 11 in July. The wettest month is May with an average of 114mm of rain.	

Table 16: Example of a traversal on an MSMarco query. Here, we traverse the latent space in equidistant steps from the embedding of the original query "weather year round in new york city", which results in an nDCG@10 retrieval score of 0, to the embedding of the gold paragraph. The queries decoded from a latent code close to the gold paragraph are more specific, asking about the hottest and coldest month in New York.

Type	Query	nDCG@10
Original Query	what year declaration of independence	0.00
Reformulations	when was the declaration of independence year	0.00
	when was the declaration of independence year	0.00
	what year is the declaration of independence	0.00
	when was the declaration of independence	0.00
	what is the declaration of independence in the year	0.33
	what is the declaration of independence in most of the person	0.00
	what is the declaration of independence in most of them	0.00
	when was the declaration of independence (most important)	1.00
	when was the declaration of independence (dst) on the most important document	1.00
	when was the declaration of independence (most important)	1.00
Gold Paragraph	The Declaration of Independence is, of course, one of the country's most important documents, adopted at the Second Continental Congress on July 4, 1776. The text and purpose of the Declaration would likely be recognizable to those who have applied for U.S. citizenship, since questions about the document appear on the naturalization test.	

Table 17: Example of a traversal on an MSMarco query. Here, we traverse the latent space in equidistant steps from the embedding of the original query "what year declaration of independence", which results in an nDCG@10 retrieval score of 0, to the embedding of the gold paragraph. The queries decoded from a latent code close to the gold paragraph, include the keyword "important" that seems to be helpful in retrieving this particular gold paragraph.

Type	Query	nDCG@10
Original Query	how many us dollars are currently in circulation	0.00
Reformulations	how many dollars are currently in circulation	0.00
	how many dollars are in circulation in us	0.00
	how many dollars are in circulation in the us	0.00
	how many dollars are in circulation in the us	0.00
	how many dollars are in circulation in the us	0.00
	what are the dollars that are in circulation us	0.00
	what are the sets of dollars in the us	0.00
	what are the denominations of dollars around us	0.50
	what are the denominations of $ 100 were there	1.00
	what are the denominations of $ more than that were used in the us	1.00
Gold Paragraph	Large denominations of United States currency. Large denominations of United States currency greater than $100 were circulated by the United States Treasury until 1969. Since then, the U.S. dollar has only been issued in seven denominations: $1, $2, $5, $10, $20, $50, and $100.	

Table 18: Example of a traversal on an MSMarco query. Here, we traverse the latent space in equidistant steps from the embedding of the original query "how many us dollars are currently in circulation", which results in an nDCG@10 retrieval score of 0, to the embedding of the gold paragraph. The decoded queries drift away from the original questions of dollars in circulation to the denominations of dollar bills. The reason of this drift becomes obvious by looking at the gold paragraph that explains the denominations of the US Dollar but not the money currently in circulation. This is one of many examples where the labeled gold paragraph is not correct and hence the decoded queries drift away from the original query.

Type	Query	nDCG@10
Original Query	are tesla electric cars	0.00
Reformulations	are tesla electric cars no are they electric car	0.00
	is tesla electric cars a car or they are electric	0.00
	tesla electric cars are they or electric car	0.00
	tesla electric cars are they being or electric	0.00
	tesla cars are electric to be or electric cars	0.00
	tesla electric cars are off of what kind of cars	0.00
	tesla electric cars are a bolton on which time	0.00
	tesla electric cars a mile off what speed are they	0.00
	737 el taton speedway was flying away on how many cars	1.00
	737 bolts on a ta speedway how many miles off airport	1.00
Gold Paragraph	A Qantas Boeing 737 aircraft and a Tesla electric car race on the nearly 2 mile runway at Avalon Airport. The Tesla was hard to catch off the start. Both travelled neck and neck as the 737 reached take-off speed of 161 mph and the Tesla hit 155 mph. USA TODAY.	

Table 19: Example of a traversal on an MSMarco query. Here, we traverse the latent space in equidistant steps from the embedding of the original query "are tesla electric cars", which results in an nDCG@10 retrieval score of 0, to the embedding of the gold paragraph. In this example the final decoded query perfectly solves the retrieval problem (nDCG of 1) even though being semantically very strange. These queries are an interesting example as their embedding is close to the embedding of the gold paragraph, even though they have little non-stopword overlap with the gold paragraph (in the second to last query, only "737" overlaps exactly).