



TRIE-NLG: trie context augmentation to improve personalized query auto-completion for short and unseen prefixes

Kaushal Kumar Maurya¹ · Maunendra Sankar Desarkar¹ · Manish Gupta² · Puneet Agrawal²

Received: 10 February 2023 / Accepted: 13 July 2023 / Published online: 7 August 2023

© The Author(s), under exclusive licence to Springer Science+Business Media LLC, part of Springer Nature 2023

Abstract

Query auto-completion (QAC) aims at suggesting plausible completions for a given query prefix. Traditionally, QAC systems have leveraged tries curated from historical query logs to suggest most popular completions. In this context, there are two specific scenarios that are difficult to handle for any QAC system: short prefixes (which are inherently ambiguous) and unseen prefixes. Recently, personalized Natural Language Generation (NLG) models have been proposed to leverage previous session queries as context for addressing these two challenges. However, such NLG models suffer from two drawbacks: (1) some of the previous session queries could be noisy and irrelevant to the user intent for the current prefix, and (2) NLG models cannot directly incorporate historical query popularity. This motivates us to propose a novel NLG model for QAC, TRIE-NLG, which jointly leverages popularity signals from trie and personalization signals from previous session queries. We train the TRIE-NLG model by augmenting the prefix with rich context comprising of recent session queries and top trie completions. This simple modeling approach overcomes the limitations of trie-based and NLG-based approaches, and leads to state-of-the-art performance. We evaluate the TRIE-NLG model using two large QAC datasets. On average, our model achieves huge $\sim 57\%$ and $\sim 14\%$ boost in MRR over the popular trie-based lookup and the strong BART-based baseline methods, respectively. We make our code publicly available at <https://github.com/kaushal0494/Trie-NLG>.

Keywords TRIE-NLG · Natural Language Generation · Query Auto Completion · AutoSuggest · Transformers · Pre-trained Models

Responsible editor: Charalampos Tsourakakis.

Extended author information available on the last page of the article

1 Introduction

Query formulation could be time-consuming for naïve users or users with complex information needs. Modern search engines, therefore, have a Query Auto-Completion (QAC) module to assist users in efficiently expressing their information need as a search query. The goal is to help users finish their search task faster by accurately understanding their query intent using the partially-typed prefix. While users type a partial search query (i.e., query prefix), the QAC system recommends a list of relevant complete queries (i.e., query auto-completions or suggestions).

Most of the popular search engines adopt a two-stage approach for QAC: *candidate retrieval* and *candidate ranking* (Cai and De Rijke et al. 2016). A set of prefix-preserving suggestions is retrieved from a pool of complete candidate queries in the candidate retrieval stage.¹ Typically this is supported using a trie that records complete suggestions along with their historical popularity scores computed over a time window. Candidate retrieval could leverage various heuristics like historical candidate popularity, language or region-based affinity, freshness, etc. In the candidate ranking stage, these retrieved queries are ranked based on a larger list of features including popularity, the user's previous search intent, the user's profile, etc. Finally, top-N-ranked candidates are shown to the user.

Although QAC has been studied for many decades, there are two major challenges yet to be solved.

1. **Short Prefixes:** High-quality completions for very short prefixes are the most desirable feature for any QAC system. But short prefixes are likely to have a huge candidate pool from the trie, and most of the QAC models return the most popular completions that may not be relevant.
2. **Unseen Prefixes:** Trie-based systems fail to provide recommendations for prefixes that have never been recorded previously, i.e., not a part of the query log. We refer to such prefixes as *unseen* prefixes.

To overcome these problems, more recently, seq2seq models have gained attention (Dehghani et al. 2017; Mustar et al. 2020; Yin et al. 2020). Besides the current prefix, these neural network-based natural language generation (NLG) models are more powerful because they can also utilize relevant session information to recommend personalized query completions. But even NLG models have the following drawbacks: (1) Unlike trie-based methods, NLG models cannot directly incorporate historical popularity which is a very important signal. (2) With increased levels of multi-tasking, sessions have become heterogeneous, diverse and dynamic. This makes it difficult to focus on session queries relevant to the current prefix (Yin et al. 2020). A few such session examples are presented in Fig. 1. (3) Attention-based NLG methods which attempt to discover relevant session queries by computing similarity with prefix representations suffer when

¹ A small percent of suggestions are not prefix preserving; in this work, we focus on prefix-preserving suggestions only.

Session:	https://www.antr.org/download/ docker multistage check url is valid python learning antlr python docker chmode entrypoint
Prefix:	entrypoint dock
Completion:	entrypoint dockerfile
Session:	music index emory changeling 5e fantasy gachapon list wizard spell list cats grace item dnd 5e philosophers stone taz taz relics justin mcelroy reddit fat griffin mcelroy timer greenflame
Prefix:	green f
Completion:	green flame blade dnd 5e

Fig. 1 Examples of session queries, prefix and completion. Queries are separated by ‘||’

prefixes are too short. Misleading attention leads to poor completions. (4) As the unseen prefixes are typed rarely, corresponding session information may not be very relevant.

NLG models can nicely capture the semantic relationships between existing session queries, prefix and completion. On the other hand, information in tries is like frequency-based high-confidence rules that capture relationships between prefix and completions in a syntactic manner. We hypothesize that jointly leveraging popularity signals from trie, and semantic and personalization signals from previous session queries using an NLG mechanism are essential for effective QAC. Based on this hypothesis, we propose a novel model for QAC, TRIE-NLG, which uses a sequence-to-sequence Transformer architecture. To the best of our knowledge, such joint modeling of NLG techniques with popularity signals from trie for query auto-completion has not been studied in the literature.

Given a prefix, TRIE-NLG first extracts up to top- m most popular completions from the trie. We utilize a trie with around one billion suggestions constructed using 1.5 years of past query logs (Jul 2020–Dec 2021) from Bing. For unseen prefixes, trie lookups lead to no (prefix-preserving) matches. To solve this problem, inspired by Mitra and Craswell (2015), we first index all suffix word n -grams from query logs into a suffix trie along with suffix popularity. We then lookup unseen prefixes against the suffix trie to extract top- m most popular synthetic completions. These m popularity-based completions, either from the main trie or from suffix trie, are augmented as extra context along with session queries and prefixes and passed as input to the seq2seq model. We hope that having additional knowledge from trie-lookup will enable the NLG model to retain/copy good quality completions along with the generation of the novel but relevant suggestions.

Overall, our main contributions are as follows:

- We motivate the need for incorporating both popularity signals from tries and personalization signals from previous session queries for effective QAC, especially for short and unseen prefixes.
- We propose a novel architecture, TRIE-NLG, which consists of a seq2seq Transformer model trained using rich context comprising of recent session queries and top trie completions. To the best of our knowledge, this is the first attempt of trie knowledge augmentation in NLG models for personalized QAC.
- Our proposed model provides state-of-the-art performance on two real prefix-to-query click behaviour QAC datasets from Bing and AOL. We also perform

several analyses including ablation studies to prove the robustness of the proposed model.

The rest of the paper is organized as follows. We discuss related work on traditional, learning-based, and language generation-based approaches for query auto-completion in Sect. 2. Next, we formally define the problem in Sect. 3. In Sect. 4, we present details of our proposed TRIE-NLG modeling approach. We present dataset details, evaluation metrics, baselines, and implementation details in Sect. 5. We present and analyze the results of different baselines and the proposed model in Sect. 6. We conclude with a brief summary in Sect. 7.

2 Related work

In this section, we focus on three threads of related work for Query Auto-Completion (QAC), viz., traditional, learning-based, and language generation-based approaches.

2.1 Traditional approaches for QAC

Most of the traditional QAC systems leverage *tries* (Hsu and Ottaviano 2013) which store historical co-occurrence statistics of prefix and complete query pairs. The most popular QAC approach using trie lookups is “*Most Popular Completion*” (MPC; Bar-Yossef and Kraus (2011)) which suggests top-N most popular (frequent) queries that start with the given prefix. Mitra and Craswell (2015) extended this approach to generate candidates for rare prefixes using frequently observed query suffixes mined from historical search logs. On similar lines, other methods rely on term co-occurrence (Huang et al. 2003), user click information (Mei et al. 2008), clustering queries (Sadikov et al. 2010), and using word level representations (Bonchi et al. 2012). Some previous studies (Bhatia et al. 2011; Maxwell et al. 2017) also focused on modeling approaches when search logs are not available.

2.2 Learning-based approaches for QAC

Query log-based approaches are usually context-agnostic and suffer from data sparsity issues. It is critical to leverage context for capturing personalized intent and behavior. To cope with these limitations, different sources of knowledge have been exploited in the candidate ranking stage with the learning-to-rank framework (Wu et al. 2010). These additional signals include session information (Bar-Yossef and Kraus 2011; Jiang et al. 2014), user behavior (Hofmann et al. 2014; Mitra et al. 2014), personalization (Cai et al. 2014; Shokouhi 2013) and time/popularity-sensitivity (Shokouhi and Radinsky 2012). Learning methods include LambdaMART (Burgess 2010), logistic regression (Shokouhi 2013), convolutional neural network (CNN; Mitra and Craswell (2015)), deep learning based ranking model (DRM; Zhou et al. (2018)), and eXtreme Multi-Label Ranking (Yadav et al. 2021). These ranking models, however, fail to generate completions for unseen prefixes.

Unlike these, we develop NLG models which capture personalization, learn contextual input representations, and provide completions even for unseen prefixes.

2.3 NLG-based approaches for QAC

Recently, sequence-to-sequence language model-based approaches have also been tried for QAC (Park and Chiba 2017; Wang et al. 2018). Given a prefix and optionally personalization information, these models generate prefix-preserving completions. These models can generate completions for unseen prefixes. Wang et al. (2018) use LSTM (Long Short-Term Memory networks) and GRU (Gated Recurrent Units) based character-level language model to generate completions. Dehghani et al. (2017) proposed GRUs with attention and copy mechanism to incorporate the most prominent part of the previous queries. Mustar et al. (2020) and Yin et al. (2020) proposed Transformer (Vaswani et al. 2017) based models. Yin et al. (2020)'s approach requires additional browsed item information and also needs CTR values as labels to train the model. Moreover, these generation models still fail to generate good completions for short and rare prefixes. Unlike these methods, we encode additional trie context along with session in the NLG model which leads to more meaningful completions for short, rare, and unseen query prefixes. Note that in our case, additional context is obtained from tries that are a part of any QAC system.

3 Problem formulation

Consider a user u whose previous n queries (earliest to latest order) in the current session s are q_1, q_2, \dots, q_n . The user is typing the current query q , where p is the query prefix typed so far. Additionally, there are up to m candidate query completions (top-ranked to low-ranked order) c_1, c_2, \dots, c_m available as additional context e from a trie. We aim to generate top- N query completions conditioned on current query prefix p , additional trie context e , and session information s . Mathematically, the task can be formulated as learning a model with parameters θ such that the probability of generating query q , $P_\theta(q|p; c_1, c_2, \dots, c_m; q_1, q_2, \dots, q_n)$, is maximized. Here we consider the value of N is equal to 8, i.e., the number of auto-completions is 8.

4 Methodology

The proposed TRIE-NLG model extracts a few completions from the trie and augments them as part of the input to an NLG model. For a given prefix, up to top- m completions are extracted as additional context from the trie using MPC. Those prefixes for which completions can be obtained from the MPC are called *Seen* prefixes, while those for which completions are not present are called *Unseen* prefixes. For seen prefixes, we leverage the main trie, and for unseen prefixes, we leverage a new trie called as suffix trie. These suggestions from the main or

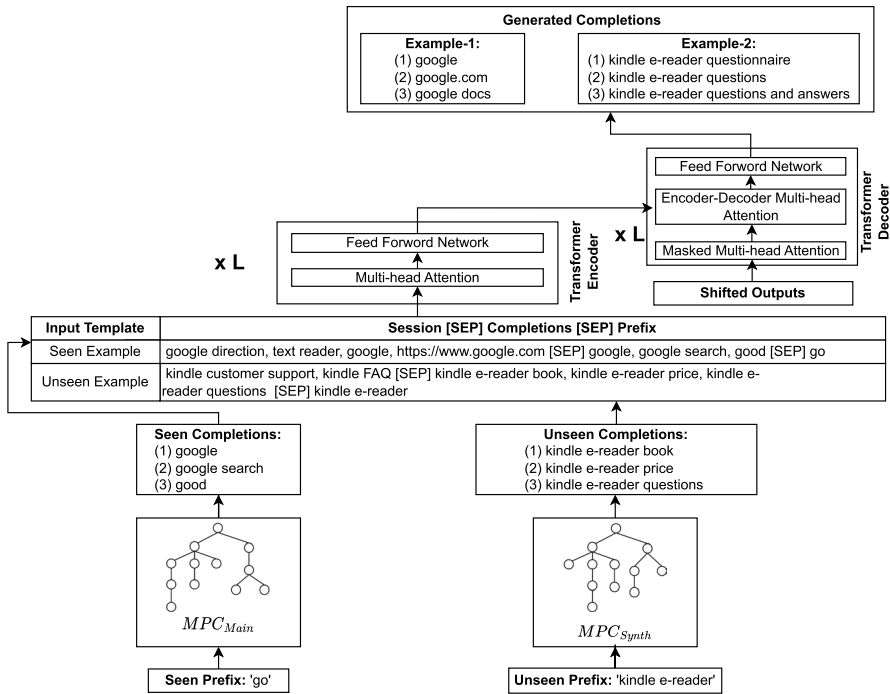


Fig. 2 An overview of the proposed TRIE-NLG model

suffix trie are augmented with previous queries in the session and the current prefix, and passed as input to the NLG model to generate accurate completions. Figure 2 illustrates the architecture of the proposed model. To enable a concrete understanding of the proposed model, we consider two running examples. For simplicity, we consider only the prefix and ground truth completion in the $\langle prefix, completion \rangle$ template. Example-1: $\langle go, google.com \rangle$ and Example-2: $\langle kindle e-reader, kindle e-reader questionnaire \rangle$.

4.1 Trie context extraction (MPC_{Main})

To extend the context associated with seen prefixes, top-ranked completions are extracted from the main trie which has been created using 1.5 years' worth of Bing query logs. Given a prefix p , MPC_{Main} provides up to m completions $\langle c_1, c_2, \dots, c_m \rangle$. In case the prefix is not present in the trie, the lookup will return no responses. For our running example-1, for the prefix go , MPC_{Main} returns three completions: $google, google.com,$ and $good$. However, for running example-2, no completions are obtained from MPC_{Main} for the prefix $kindle e-reader$. The prefix go is referred to as *seen* prefix, while $kindle e-reader$ is referred to as *unseen* prefix.

4.2 Synthetic context extraction (MPC_{Synth})

For unseen prefixes, the main trie fails to provide any completions. In such cases, we make use of another trie called the suffix trie which is created by indexing all suffix word n -grams from query logs along with suffix popularity. Since the suffix word n -grams may not be actual queries, we call them synthetic suggestions. Formally, if a query contains n words w_1, \dots, w_n , its substrings $\langle w_i, \dots, w_n \rangle$ for each $i \in \{1, \dots, n\}$ is a suffix of it. These suffixes are organized in another trie called the suffix trie. For a given unseen prefix p_u , if the main trie fails to return any suggested completions, we look up the suffix trie for suggestions. Matching unseen prefixes in suffix trie enables locating queries that contain the current prefix p_u as an intermediate word. The words surrounding the prefix provide additional context that is useful to generate the final query.

For example, given a query “university of west florida”, the suffix trie will store synthetic suggestions like “florida”, “west florida” and “of west florida”. We lookup unseen prefixes against the suffix trie to extract top- m most popular synthetic yet useful completions. Note that these lookups still attempt to match the unseen prefix with prefixes of suffixes indexed in the suffix trie. In this way, we will be able to obtain completions for unseen prefixes which can not be obtained from MPC_{Main} . We refer to this method as MPC_{Synth} . Although this idea is similar to one described by Mitra and Craswell (2015), unlike them, we consider the whole prefix and not only *end-term* of the prefix. If a prefix has multiple words, the last partial word is the end-term. The whole prefix has more meaningful contextual representation than end-term representation which leads to more accurate completions. For running example-2, MPC_{Synth} returns three completions for the unseen prefix *kindle e-reader*: *kindle e-reader book*, *kindle e-reader price*, and *kindle e-reader questions*.

4.3 Context augmentations in NLG

After obtaining trie suggestions, each data point consists of the session information (s), additional trie context (e), prefix (p), and the corresponding completion (q). We consider an Encoder–Decoder-based NLG model that takes the triplet $\langle s, e, p \rangle$ as input and attempts to generate the complete query (q). The input is provided to the model as a text sequence, where each element of the triplet is separated by a special token [SEP]. Trie context, i.e., top- m candidate completions are obtained from MPC_{Main} or MPC_{Synth} . During model training, the input triplet is first fed through the encoder to obtain a contextual representation. These contextual representations are semantic encodings of $\langle s, e, p \rangle$, which is key for the model’s performance, particularly for short and unseen prefixes. Then, this contextual representation is passed through the decoder to generate top- N completions.

Relevant contextual suggestions from tries help the model with additional input that can guide the generation process. As typically the queries in a user session are often correlated in terms of the user’s information need, the session context helps the model in understanding the user’s current requirement. On the other hand, through

the suggestions from the trie which is backed by historical query logs, a global perspective of the prefix and its possible completions preferred by a large user base can be obtained. The model thereby gets to see a local (concerning the user) as well as a global (concerning a large user pool) perspective surrounding the current prefix, and can appropriately utilize these inputs through language models pre-trained on large general-purpose corpora that understand semantic and syntactic aspects of natural language text. We hypothesize that the combination of these input and modeling choices makes the model superior for the target QAC task.

The model is trained to maximize the probability of ground truth token sequence with maximum likelihood estimation (MLE). So, the following loss function is minimized:

$$L = - \sum_{i=1}^D \sum_{t=1}^{|y^i|} \log P_t^i(\hat{y}_t^i | \hat{y}_{0:t-1}^i; p; e; s) \quad (1)$$

where D is the training dataset size, $|y^i|$ is the length of the i -th ground-truth query, \hat{y}_t^i denotes token generated at time step t . P_t^i is the prediction probability distribution at t -th decoding step to generate the next token conditioned on previously generated tokens, prefix, trie context and session. The top- N completions are generated using beam search. For both running examples, both MPC_{Main} and $\text{MPC}_{\text{Synth}}$ failed to produce the correct ground truth completion. However, with context augmentation in NLG, it can be observed that for the prefix *go*, the final completion includes the correct completion *google.com*. Similarly, the prefix *kindle e-reader* has the correct completion *kindle e-reader questionnaire*. This demonstrates the effectiveness of augmenting additional context in the NLG for QAC systems.

5 Datasets and experimental setup

In this section, first, we provide a detailed overview of the dataset along with analyses then we compare the performance of the proposed model with multiple baseline models for the query auto-completion (QAC) task. Finally, We also provide a detailed analysis of the results from multiple perspectives and report ablation studies and case studies.

5.1 Datasets and analysis

In this subsection, we present the dataset details, including data construction steps, and some critical observations. We use two datasets: (1) Bing query log and (2) AOL public query log (Pass et al. 2006). The Bing dataset covers 9.08M users, while the AOL dataset corresponds to 0.50M users.

The raw AOL query log consists of a sequence of queries entered by the users along with time-stamp details. We first pre-process the dataset by lowercasing all the queries, removing duplicate and single character queries, and removing queries with dominating (>50%) number of non-alphanumerics. Following previous

Table 1 Prefix distribution statistics for Bing dataset with prefix character length

Char length	Train			Validation			Test		
	Total	Seen	Unseen	Total	Seen	Unseen	Total	Seen	Unseen
Total	20.40M	17.86M	2.54M	100K	92.43K	7.57K	100K	92.80K	7.20K
[1-5]	9.10M	8.80M	0.30M	40.68K	40.39K	0.29K	40.46K	40.19K	0.27K
[6-10]	4.30M	4.10M	0.20M	21.40K	21.07K	0.33K	21.62K	21.24K	0.38K
10+	7.00M	4.96M	2.04M	37.92K	30.97K	6.95K	37.92K	31.37K	6.55K

‘M’ and ‘K’ indicate that the value is in the order of millions and thousands respectively

Table 2 Prefix distribution statistics for AOL dataset with prefix character length

Char length	Train			Validation			Test		
	Total	Seen	Unseen	Total	Seen	Unseen	Total	Seen	Unseen
Total	3.91M	3.47M	0.44M	100K	88.73K	11.27K	100K	88.69K	11.31K
1-5	1.42M	1.42M	0.00M	35.55K	35.54K	0.01K	36.59K	36.56K	0.03K
6-10	1.15M	1.11M	0.04M	29.53K	28.67K	0.86K	29.51K	28.61K	0.90K
10+	1.34M	0.94M	0.40M	34.92K	24.52K	10.40K	33.90K	23.52K	10.38K

‘M’ and ‘K’ indicate that the value is in the order of millions and thousands respectively

studies (Sordoni et al. 2015; Yadav et al. 2021), we split the sequence of queries into sessions with at least 30 min of idle time between two consecutive queries. We only retain those sessions which have at least two queries. Next, for a given session with queries (in earliest to latest order) $(q_1, q_2, \dots, q_n, q_{n+1})$ we create a triplet $r = \langle (q_1, q_2, \dots, q_n), p_{n+1}, q_{n+1} \rangle$ where p_{n+1} is a sampled prefix of query q_{n+1} . Sampling follows an exponential distribution favoring shorter prefixes. Each such triplet is a data point for modeling where input is all session queries except the last one, additional trie context, and prefix p_{n+1} . Ground-truth output is the last session query q_{n+1} .

Unlike the AOL dataset, where the prefix-to-query information is not explicitly available, and the prefixes are synthetically created by splitting a full query, the Bing dataset consists of real prefixes. Each example of the dataset consists of the user’s session information s , current real prefix p_{n+1} , and real clicked completed query q_{n+1} . The dataset is obtained by considering only those cases where there were at least one past query in the user session, and the user has set their primary language as English.

Each dataset has two splits: *Seen Dataset* and *Unseen Dataset*. To obtain these splits, we use a trie with around one billion suggestions constructed using 1.5 years of past Bing query logs (Jul 2020 to Dec 2021). For a given prefix, if the trie contains at least one completion then the prefix is called a *Seen Prefix*. Else, it is called an *Unseen Prefix*. The set of all Seen Prefixes (along with other search log attributes) is referred to as *Seen Dataset* and the set of all Unseen Prefixes is called *Unseen Dataset*. Statistics of Bing and AOL datasets shown in Tables 1 and 2 respectively

indicate that there are 12.4% and 11.4% unseen prefixes in the training part of Bing and AOL datasets respectively. The traditional MPC baseline leads to zero suggestions for such prefixes. The AOL and BING datasets each contain three sub-splits, namely train, validation, and test, which are based on temporal information and are applicable to both seen and unseen datasets as well. The timestamps for the train, validation, and test datasets are denoted as t_{train} , $t_{validation}$, and t_{test} , respectively. It is stipulated that t_{train} is the least recent of the three timestamps, and $t_{validation}$ and t_{test} are of increasing temporal proximity, with t_{test} being the most recent of the three.

To analyze the accuracy of various methods for different slices of the dataset, we created three prefix-length buckets: between 1 to 5 characters, 6 to 10 characters, and greater than 10 characters. Prefix distribution statistics with respect to each bucket are also reported in Tables 1 and 2 for Bing and AOL datasets, respectively. We observe that ~45% and ~36% of the prefixes from the training datasets have lengths less than 6 for Bing and AOL, respectively. This indicates the dominance of short prefixes and necessitates the design of better modeling techniques. An approach that provides additional context (as we propose) is promising. We also observe that ~80% and ~91% of the unseen train dataset have prefix lengths 10+ characters for Bing and AOL respectively, which is another reason for them being less popular. The addition of more relevant context from tries may lead to better completions in such scenarios. We also observe that the average number of queries in a session for Bing is 5.2 and for AOL is 2.4. This provides diverse personalization contexts to better judge the applicability and usefulness of the models. Overall, unseen and short prefixes in QAC are frequent and challenging problems.

5.2 Evaluation metrics

We evaluate all the baselines and proposed model with three evaluation metrics. To cover multiple aspects of the evaluation, we use both ranking-oriented metrics (MRR) and metrics to identify the quality of the generated sequence (BLEU and BLEU_{RR}).

1. **Bilingual Evaluation Understudy (BLEU):** It is a popular metric used for multiple NLG tasks. For our experiments, BLEU evaluates the degree of lexical match between the ground-truth complete query and the first ranked generated query.
2. **Mean Reciprocal Rank (MRR):** MRR is one of the most popular metrics for evaluating ranking systems. MRR score is calculated as

$$\text{MRR} = \frac{1}{D_{ts}} \sum_{i=1}^{D_{ts}} \frac{1}{r_i} \quad (2)$$

Here, D_{ts} is the size of the test dataset and r_i is the rank of the ground-truth complete query in the generated rank list for the i^{th} input. If the ground-truth complete query is not in the generated rank list, then r_i is set to ∞ .

3. **BLEU Reciprocal Rank (BLEU_{RR} Yadav et al. (2021))**: It is defined as the reciprocal rank weighted average of BLEU score between the ground-truth query and generated completions.

$$\text{BLEU}_{RR} = \frac{1}{D_{ts}} \sum_{i=1}^{D_{ts}} \frac{\sum_{j=1}^N \frac{1}{j} \text{BLEU}(q, q'_{i,j})}{\sum_{j=1}^N \frac{1}{j}} \quad (3)$$

where q is the ground-truth complete query and $q'_{i,j}$ is the j -th generated completion for the i -th test example.

5.3 Baselines

Our proposed model is based on both NLG and Trie models. Such joint modeling of NLG systems with popularity signals from trie has not been previously explored. To thoroughly evaluate its performance, we have carefully selected ten diverse baselines, including the basic trie-based model (MPC, MPC+SynthMPC) to ranking (GRM) to Deep learning (LSTM, Transformers) to most recent pre-trained NLG models (T5, BART). In light of the superior performance demonstrated by transformer-based models, we have also included multiple strong transformer-based baselines. As our focus is on generation rather than ranking, we have selected more generative baselines for comparison. However, the outputs of our proposed model can be used as features in learning-to-rank and traditional models. The following baselines have been considered for our experimental evaluation:

1. **MPC_{Train}**: This uses the traditional MPC method (Bar-Yossef and Kraus 2011). The candidate rankings are obtained based on the popularity of each query from the historical query log. Here the historical query log is the training data itself.
2. **MPC_{Main}**: In this baseline the completions are obtained using the main trie created using 1.5 years of historical query logs from Bing.
3. **MPC_{Train} + MPC_{Synth}/MPC_{Main} + MPC_{Synth}**: Completions are obtained using MPC_{Train} and MPC_{Synth}/MPC_{Main} for seen and unseen prefixes resp.
4. **GRM**: We first represent a session, prefixes and complete query as a bag-of-word (BOW) vector and then LambdaMART is trained with these features.
5. **Seq2Seq LSTM**: Standard LSTM based sequence-to-sequence model with attention. Input is a prefix and the target is the complete query.
6. **Seq2Seq Transformer**: Standard Seq2Seq Transformer model with architecture similar to T5-base. We train the model from scratch. Input is “session [SEP] prefix” and the target is the complete query.
7. **T5**: Same as Seq2Seq Transformer, except that we *fine-tune* T5-base (Raffel et al. 2020) on the QAC dataset.
8. **BART**: Similar to Mustar et al. (2020), we fine-tune BART-base (Lewis et al. 2019) with QAC dataset. Input and output are the same as that of T5.
9. **BART + Implicit Trie Context (ITC)**: In this modeling, we try to augment trie’s knowledge implicitly. It is a two-step training procedure: (i) BART-base is fine-tuned using a dataset, which consists of session and prefix as input and top

- m suggestions from $\text{MPC}_{\text{Main}}/\text{MPC}_{\text{Synth}}$ as the target. (ii) This training checkpoint is further trained with the QAC dataset, where the input is the session and the prefix and output are the clicked query. In the second step, we freeze the parameters of the first six decoder layers to retain the trie-based knowledge. During inferencing, the model checkpoint obtained from the second stage of training takes prefix and session as input and outputs the query suggestion.
10. **BART + MPC_{Main}**: This baseline augments the trie knowledge explicitly. With each training example, we add up to top- m trie completions as additional context. There are no completions for unseen prefixes. Input is session, prefix, and additional trie context and output is the clicked query.

5.4 Implementation details

The proposed model and all the baselines are implemented in Python. GRM is implemented using learning to rank library² and seq2seq LSTM is implemented using Texar.³ All the transformer-based models are implemented using HuggingFace Library.⁴ All the experiments were conducted on eight A100 Azure cloud GPUs. The batch size is 128; the learning rate is $1e-4$, the scheduler is 'linear,' the number of epochs is 5, and early stopping was enabled. We used the Adam optimizer with a max source length of 200 and a max target length of 32. BART-base has 6 layers and 12 heads. Layer normalization was enabled and the hidden layer dimension is 768. For the generation, the number of beam size (i.e., k) is 8, the maximum sequence length is set to 16, and the repetition penalty⁵ is 0.6. We applied grid search for hyper-parameter tuning on the validation dataset and all the scores are reported on the test dataset. We experimented with 1, 3, 5, and 8 as values of m , the number of suggestions to extract from the trie. Based on the results of the validation dataset, $m = 3$ was selected for running experiments on the test data. We make our code publicly available.⁶

6 Results and discussions

In this section, we present and analyze results of different baselines and the proposed model.

6.1 Overall performance comparison

Tables 3 and 4 summarize the experimental results on the Bing and AOL datasets, respectively. Due to the confidential nature of the Bing dataset, we cannot report the

² <https://github.com/jma127/pyltr>.

³ <https://github.com/asym1/texar>.

⁴ <https://huggingface.co/>.

⁵ <https://huggingface.co/blog/how-to-generate#appendix>.

⁶ <https://github.com/kaushal0494/Trie-NLG>.

Table 3 Results of the models on Bing dataset

Models	Seen + Unseen Dataset			Seen Dataset			Unseen Dataset		
	Δ MRR	Δ BLEU_RR	Δ BLEU	Δ MRR	Δ BLEU_RR	Δ BLEU	Δ MRR	Δ BLEU_RR	Δ BLEU
MPC_{Train}	-7.90	-19.87	-24.64	0.00	0.00	0.00	-	-	-
MPC_{Main}	-0.11	36.61	26.38	8.49	70.38	63.68	-	-	-
$MPC_{Main} + MPC_{Synth}$	7.78	56.42	47.01	8.49	70.38	63.68	0.00	0.00	0.00
GRM	-1.43	-5.32	-5.51	4.30	19.02	16.58	-	-	-
Seq2Seq LSTM	9.61	39.30	56.78	10.24	44.02	72.20	5.34	16.61	91.45
Seq2Seq Transformer	15.74	54.22	76.16	18.24	55.07	82.46	10.53	24.20	99.29
T5	20.78	61.81	78.70	20.76	70.83	85.13	21.60	27.33	103.98
BART	36.73	73.47	91.09	36.95	84.51	100.47	34.53	29.03	110.35
BART + ITC	31.66	71.43	88.72	31.58	81.97	96.84	33.05	29.12	111.00
BART + MPC_{Main}	54.12	86.77	110.78	56.14	101.35	129.00	34.10	28.04	110.80
TRIE-NLG	56.78	88.26	114.52	56.56	101.99	130.04	59.74	33.02	123.07

The reported scores are percentage (%) improvements over $MPC_{Train} + MPC_{Synth}$ baseline. '-' indicates no completions are retrieved/generated for the model. Here we consider up to 3 completions as additional context from MPC_{Main} or MPC_{Synth} . GRM is a ranking model based on clicked queries. As the Unseen dataset does not have click information, GRM models cannot be built

Table 4 Results of the models on AOL dataset

Models	Seen + Unseen Dataset			Seen Dataset			Unseen Dataset		
	MRR	BLEU_RR	BLEU	MRR	BLEU_RR	BLEU	MRR	BLEU_RR	BLEU
MPC _{Train}	20.6	5.4	15.25	23.2	6.1	19.49	–	–	–
MPC _{Train} + MPC _{Synth}	31.3	12.0	40.10	23.2	6.1	19.49	95.2	58.7	95.66
MPC _{Main}	19.7	9.9	24.44	22.2	11.2	29.75	–	–	–
MPC _{Main} + MPC _{Synth}	30.4	16.6	47.76	22.2	11.2	29.75	95.2	58.7	95.66
GRM	21.8	7.3	20.69	24.6	8.3	23.33	–	–	–
Seq2Seq LSTM	43.9	14.7	51.43	43.6	12.9	49.16	47.1	29.3	69.18
Seq2Seq Trans-former	45.4	16.8	57.50	44.5	14.8	51.79	51.9	32.3	73.62
T5	48.1	17.4	59.63	46.6	15.2	53.42	59.5	3461	77.18
BART	51.9	18.3	61.89	50.3	16.1	55.64	64.7	35.8	79.55
BART + ITC	50.7	18.3	61.55	49.1	16.0	55.29	63.5	35.9	79.24
BART + MPC _{Main}	53.2	18.6	62.48	51.8	16.4	56.58	64.1	35.6	79.21
TRIE-NLG	56.5	19.3	66.63	52.0	16.5	56.56	92.1	41.2	94.62

Here we report exact evaluation scores, unlike the ones for the Bing dataset. We consider up to 3 completions as additional context from MPC_{Main} or MPC_{Synth}. GRM is a ranking model based on clicked queries. As the Unseen dataset does not have click information, GRM models cannot be built

exact values of the metrics. This is common practice in many previous studies (Rosset et al. 2018) as well. Hence, in Table 3 and the rest of the paper we report percentage improvement scores of the models over reference MPC_{Train} + MPC_{Synth} baseline for the Bing dataset. We cannot use MPC_{Train} as a reference for showing percentage improvements as the model does not have any completions for unseen prefixes. For the publicly available AOL dataset, we report exact evaluation scores across all three metrics. We also report percentage improvement scores of the models over reference MPC_{Train} + MPC_{Synth} baseline for the AOL dataset in Table 5. Overall, our proposed TRIE-NLG outperforms all the traditional, ranking, and generative models, across both the datasets (including Seen and Unseen) and all three metrics. Paired t-test shows that TRIE-NLG outperforms the best baseline statistically significantly across both the datasets for each of the three metrics with a p-value less than 0.05.

Note that MPC_{Train} + MPC_{Synth} and MPC_{Main} + MPC_{Synth} have identical results for “unseen” datasets. Similarly, MPC_{Train} and MPC_{Train} + MPC_{Synth} yield same results for “seen” dataset. This is expected because MPC_{Train} and MPC_{Main} provide trie suggestions for seen prefixes; while MPC_{Synth} provides trie suggestions for unseen prefixes

Without MPC_{Synth}, the MPC_{Train} and MPC_{Main} do not have completions for the Unseen dataset. MPC_{Synth} provides completions for the unseen prefixes and boosts the overall model performance. As expected, the generative models provide suggestions for unseen prefixes, unlike ranking and database lookup models. Evaluation scores of Seq2Seq Transformer and pre-trained models (i.e., T5 and

Table 5 Results of the models on AOL dataset. The reported scores are percentage (%) improvements over MPC_{Train} + MPC_{Synth} baseline. ‘-’ indicates no completions are retrieved/generated for the model. Here we consider up to 3 completions as additional context from MPC_{Main} or MPC_{Synth}. GRM is a ranking model based on clicked queries. As the Unseen dataset does not have click information, GRM models cannot be built

Models	Seen + Unseen			Seen Dataset			Unseen Dataset		
	MRR	BLEU_RR	BLEU	MRR	BLEU_RR	BLEU	MRR	BLEU_RR	BLEU
	MPC _{Train}	-34.18	-55.37	-61.97	0.00	0.00	0.00	-	-
MPC _{Main}	-37.06	-18.18	-39.05	-4.31	83.60	52.64	-	-	-
MPC _{Main} +MPC _{Synth}	-2.87	37.19	19.10	-4.31	83.60	52.64	0.00	0.00	0.00
GRM	-30.35	-39.66	-48.40	6.03	36.06	19.70	-	-	-
Seq2Seq LSTM	40.25	21.48	28.25	87.93	111.47	152.23	-50.52	-50.08	-27.68
Seq2Seq Transformer	45.04	38.84	43.39	91.81	142.62	165.72	-45.48	-44.97	-23.03
T5	53.67	43.80	48.70	100.86	149.18	174.08	-37.5	-41.05	-19.31
BART	65.81	51.23	54.33	116.81	163.93	185.47	-32.03	-39.01	-16.84
BART+ITC	61.98	51.23	53.49	111.63	163.93	183.68	-33.29	-38.84	-17.16
BART+MPC _{Main}	69.96	53.71	55.81	123.27	168.85	190.30	-32.66	-39.35	-17.19
TRIE-NLG	80.51	59.50	66.15	124.56	170.49	190.20	-3.25	-29.81	-1.08

BART) indicate that the pre-trained models provide better input representation and perform better. BART + ITC fuses the additional context (top-ranked completions obtained from MPC_{Main}) implicitly with two-step training. However, the results are not promising, indicating that the model's learning is distracted in the two-stage training. However, adding explicit context leads to better performance, as shown in BART + MPC_{Main} model. Eventually, adding context from MPC_{Main} and MPC_{Synth} helps the proposed TRIE-NLG model perform the best.

The absolute evaluation scores for Unseen AOL data are much higher as compared to Seen AOL data. We observe similar trends for the Bing dataset as well. There could be two possible causes for this: (1) Unseen dataset is ~11% of the original data, and hence it is much smaller compared to the Seen dataset, and (2) the average prefix lengths for Seen AOL, Unseen AOL, Seen Bing, and Unseen Bing, are 8.1, 20.9, 14.5 and 25.9 respectively. In the Unseen dataset, the lengths of the prefixes are longer compared to Seen, which provides more context and the generative models perform better. Most of the baseline models' performance on the Unseen dataset is very poor, but the proposed TRIE-NLG achieves much better performance which shows the promising prospect of our approach. GRM is a ranking model based on clicked queries. As the Unseen dataset does not have click information, GRM models cannot be built.

The MPC_{Main} + MPC_{Synth} model is the best-performing model for the AOL Unseen dataset, and it surpasses the TRIE-NLG by a small margin. However, for the Bing Unseen dataset, the proposed model outperformed all the models. There can be multiple possible reasons behind this observation. For example, (1) **Dataset Timeline:** The AOL dataset is from 2006 while Bing data was collected in 2020-21. Pre-trained NLG models (like BART and T5) have been trained with recent corpus whose vocabulary is expected to be better aligned with recent Bing data rather than AOL. Final suggestions from the model for Unseen AOL data are hence governed by only the partially-aligned language model and without any context from the trie. (2) **Query Log Size:** Bing dataset has 20M queries compared to 4M in AOL dataset. This leads to better synthetic suggestions for Bing, in turn leading to better context augmentation for the Bing TRIE-NLG model. (3) **Prefix and Session Lengths:** The prefix and session length for Bing (4.434 tokens/prefix and 5.619 queries/session) are longer as compared to AOL (3.061 tokens/prefix and 2.530 queries/session). Longer prefixes and sessions lead to better NLG completions for Bing. Recent search interactions for users do involve longer sessions, and the proposed model is expected to do well in such scenarios.

The overall evaluation results indicate that neither trie nor NLG models are effective individually for such a challenging scenario. The proposed hybrid approach that considers the benefits of both worlds (language semantics from NLG and popularity statistics from trie) through a joint modeling technique is a promising approach and can push the QAC research field forward.

Table 6 Performance analysis for short prefixes. For Bing, % improvements over MPC_{Main} + MPC_{Synth} are reported. For AOL, actual scores are reported

Bing Dataset	Prefix Length in [1–5]			Prefix Length in [6–10]		
	Δ MRR	Δ BLEU _{RR}	Δ BLEU	Δ MRR	Δ BLEU _{RR}	Δ BLEU
BART	21.6	–17.6	2.9	38.5	38.4	56.8
BART + MPC _{Main}	52.1	–16.8	10.1	55.2	49.3	72.1
TRIE-NLG	53.2	–15.9	11.4	56.4	50.1	73.6
AOL Dataset	MRR	BLEU _{RR}	BLEU	MRR	BLEU _{RR}	BLEU
BART	41.3	8.0	35.39	52.2	14.3	53.39
BART + MPC _{Main}	41.5	8.0	35.25	54.1	14.7	54.08
TRIE-NLG	42.1	8.1	35.58	55.2	14.9	55.56
Prefix Length 10+						
Bing Dataset	Δ MRR	Δ BLEU _{RR}	Δ BLEU			
BART	49.1	190.0	120.1			
BART + MPC _{Main}	55.6	218.5	145.1			
TRIE-NLG	60.7	221.1	149.5			
Prefix Length 10+						
AOL Dataset	MRR	BLEU _{RR}	BLEU			
BART	63.2	32.8	75.40			
BART + MPC _{Main}	65.1	33.4	76.20			
TRIE-NLG	73.4	35.1	82.79			

Table 7 Results of ablation study using different experimental setups. TRIE-NLG(m) means “TRIE-NLG + Up to Top-m Completions.”

#	Ablation criteria	Bing Dataset			AOL Dataset		
		Δ MRR	Δ BLEU _{RR}	Δ BLEU	MRR	BLEU _{RR}	BLEU
1	No (Trie Context + Session)	-29.5	-1.5	43.1	5.7	5.9	30.9
2	No Trie Context	36.7	73.5	91.1	51.9	18.3	61.9
3	No Session	29.3	47.3	73.7	15.4	9.5	40.6
4	TRIE-NLG(1)	44.8	81.3	104.8	32.3	15.1	54.4
5	TRIE-NLG(5)	50.9	82.9	110.0	42.6	17.3	59.6
6	TRIE-NLG(8)	52.9	83.8	111.4	28.5	13.8	51.5
7	TRIE-NLG(3)	56.8	88.3	114.5	56.5	19.3	66.6

For the Bing dataset, percentage improvements over $MPC_{Main}+MPC_{Synth}$ baseline are reported. For the AOL dataset, actual evaluation scores are reported

6.2 Performance analysis for short prefixes

Table 6 shows the performance of our proposed TRIE-NLG model, and two baselines BART and BART+ MPC_{Main} for different prefix lengths on both the datasets. The evaluation scores are reported for three different buckets based on the character length of the prefix: [1–5], [6–10] and 10+. The evaluation scores in 10+ bucket are higher as compared to the other two. It indicates that as the prefix length increases, the performance of all the models increase. It is aligned with the intuition that model generates more accurate predictions as the prefix becomes longer. The model performance improves across both datasets for short prefixes as the additional context is added, i.e., BART+ MPC_{Main} performs better as compared to BART. Moreover, when synthetic completions are included further, i.e., TRIE-NLG, it outperforms both the baselines even for very short prefixes. This provides evidence that adding additional trie knowledge does help to increase relevant context for short prefixes. A few of the Δ BLEU_{RR} scores for Bing are negative for prefix lengths [1–5]. This implies that the performance of the $MPC_{Main}+MPC_{Synth}$ model is superior to the three models considered, i.e., BART, BART+ MPC_{Main} , and TRIE-NLG. Despite this, the lower negative values for TRIE-NLG demonstrate that its performance is better than the other two models. The reason behind this could be attributed to the fact that (1) the $MPC_{Main}+MPC_{Synth}$ model demonstrates the best performance for the Bing Unseen dataset in terms of unseen prefixes, as discussed in Sect. 6.1, and (2) the Δ BLEU_{RR} metric takes into account both the BLEU and MRR scores. However, other metrics’ results show consistent improvement across all prefix types and both datasets.

6.3 Ablation study

Table 7 shows ablation results. We observe similar trends for both AOL and Bing. In setups 1 to 3, we have removed session and/or external contexts. The model performs worst when both the information are removed (setup 1). Modeling with the

Table 8 Number of examples where t trie suggestions were retained in the TRIE-NLG generated completions for *Seen Test Datasets*

t	0 (%)	1 (%)	2 (%)	3 (%)
Bing Seen Test Dataset	19.0	26.4	30.5	23.6
AOL Seen Test Dataset	43.1	36.1	17.2	3.2

Table 9 Percentage of times the candidate suggestion from trie was copied to [1-8]th positions ('Pos') as output by TRIE-NLG for Bing Seen Test Dataset

Rank↓/Pos→	1	2	3	4	5	6	7	8	None
1	37.60	10.17	3.92	2.10	1.43	1.15	0.94	0.97	41.69
2	18.50	19.51	7.86	3.81	2.45	1.81	1.53	1.511	41.69
3	9.98	12.50	11.82	7.04	4.49	2.74	1.88	1.67	47.83

'None' indicates the candidate suggestion was not a part of TRIE-NLG output. Results are shown for Seen Test Data when $m=3$. 'Rank' indicates the rank of candidate suggestion from trie

Table 10 Percentage of times the candidate suggestion from trie was copied to [1-8]th positions ('Pos') as output by TRIE-NLG for AOL Seen Test Dataset

Rank↓/Pos→	1	2	3	4	5	6	7	8	None
1	25.92	6.81	3.43	2.22	1.57	1.23	1.03	1.12	56.62
2	7.41	5.47	3.26	2.23	1.65	1.29	1.11	1.11	56.62
3	3.97	3.46	2.53	1.97	1.54	1.21	1.04	1.01	83.24

'None' indicates the candidate suggestion was not a part of TRIE-NLG output. Results are shown for Seen Test Data when $m=3$. 'Rank' indicates the rank of candidate suggestion from trie

only session (setup 2) performs better than a model that uses only trie context (setup 3), which shows the importance of the user's previous search query log. However, setups 4 to 7 that use both information perform even better, indicating the importance of both types of context. Setups 4 to 7 differ from each other in the number of candidate query completions that are used as additional trie context. It is observed that the use of a single top-ranked candidate query results in worse performance, which may be attributed to an inadequate context. Furthermore, incorporating more than three top-ranked queries also results in poor performance. This can be due to two possible factors: (1) the model may become overwhelmed and unable to effectively distinguish relevant information from the trie context in the presence of too many suggestions in the input, or (2) the trie context may become too long, hindering the model's ability to effectively utilize session signals. Overall, TRIE-NLG with top-3 trie candidate completions (i.e., $m = 3$) in the input performs the best.

Table 11 Runtime of different test dataset splits

Models	Bing Test Dataset			AOL Test Dataset		
	Seen	Unseen	Total	Seen	Unseen	Total
BART	11.25	11.75	11.29	11.69	12.83	11.82
BART+ MPC _{Main}	12.28	12.16	12.27	12.17	13.05	12.27
TRIE-NLG	12.34	12.25	12.34	12.29	13.20	12.40

Values are in millisecond(ms)/record for 8 auto-complete generations.

Example-1: Seen Short Prefix		Example-2: UnSeen Prefix	
<p>Session: kysportsradio kysportsradio cincinnati reds cincinnati reds espn sports espn sports ebth ebth.com ebth.com cnn news cnn news politico news</p> <p>Prefix: p</p> <p>Correct Query: politico</p>		<p>Session: hurricane resistant hurricane lines houston crap houston crap plan hurricane climate</p> <p>Prefix: houston climate actio</p> <p>Correct Query: houston climate action plan</p>	
<p>Suggestions (MPC_{Main}):</p> <ol style="list-style-type: none"> 1. pinterest 2. paypal 3. pittsburgh penguins 4. pandora 5. prime video 6. paypal login account 7. pennlive 8. pogo official site 	<p>Suggestions (Trie-NLG):</p> <ol style="list-style-type: none"> 1. politico 2. profootballtalk 3. politico news 4. pittsburgh pirates 5. pogo official site 6. page tour 7. philadelphia inquirer 8. pennlive 	<p>Suggestions (MPC_{Main}):</p> <p>None</p> <p>Suggestions (MPC_{Synth}):</p> <ol style="list-style-type: none"> 1. houston climate action policy 	<p>Suggestions (Trie-NLG):</p> <ol style="list-style-type: none"> 1. houston climate action plan 2. houston climate action policy 3. houston climate action play 4. houston climate plan action 5. houston climate action plan tx 6. houston climate action program 7. houston climate action plan plan 8. houstonclimate action plan

Fig. 3 Generated query completions for seen (short) and unseen prefixes

6.4 Trie completion retention analysis

Further, we analyze *how many* candidate queries from trie context are generated as completions by the proposed TRIE-NLG model, and *what position* they appear in. For simplicity, we only consider up to 3 candidate queries and seen test datasets. In the ideal scenario, the best performing model should retain good candidate queries of MPC_{Main} into the recommended completion list as well as generate new completions. Table 8 shows that ~19% and ~43% examples do not retain any completions for Bing and AOL datasets, respectively. At the same time, ~23% of the Bing examples retain all the input candidate queries. For the AOL dataset, only 3% of examples have all the input candidates; this value is very low because MPC_{Main} is created with only Bing historical search log but used for generating completions for AOL prefixes. So the trie-recommended completions may not be very relevant and hence not considered by TRIE-NLG for the AOL dataset.

Tables 9 and 10 provide the position distribution of each trie candidate in the TRIE-NLG output for Bing and AOL datasets respectively. In more than 40% of examples, the top-ranked candidate query from the trie doesn't appear in the final

generated output. On the other hand, for 37.6% examples, the top trie candidate is also the top suggestion from TRIE-NLG for the Bing dataset. This also indicates that the model does not blindly copy the trie candidates as outputs. Instead, it learns to determine the candidate's goodness or fit for the specific input and performs the generation accordingly.

6.5 Runtime analysis

Table 11 shows execution runtimes for the three models on an A100 Nvidia GPUs. Trie lookups are very cheap compared to BART-based suggestion generation. Hence, our method TRIE-NLG has almost similar runtimes compared to a standard BART model.

6.6 Case studies

Figure 3 shows two examples of suggestions for a short seen prefix and an unseen prefix respectively. In the first example, the prefix 'p' is very short and MPC_{Main} is unable to understand the context and recommends more general/popular completions. Whereas the proposed TRIE-NLG model learned the personalized context and recommended correct and more relevant completions. The model also considers recommendations from MPC_{Main} as additional context. For instance, 'pogo official site' is present in MPC_{Main} and recommended by TRIE-NLG, although there is no relevant context in the session. In Example-2, there is no query recommendation from MPC_{Main} for the unseen prefix, but MPC_{Synth} has one recommendation and that acts as additional context for TRIE-NLG. TRIE-NLG generates more relevant completions and the top-ranked completion is correct. In summary, we can conclude that the additional trie context is useful for the generative model and helps TRIE-NLG to generate more accurate and relevant query completions.

7 Conclusion

We proposed TRIE-NLG model for personalized QAC. It is based on context augmentation in the NLG model where the additional context is obtained from the main trie or the synthetic trie. To the best of our knowledge, this is the first study to use the trie context in NLG models for QAC. We primarily focused on solving the problem of short and unseen prefixes. The model was evaluated on a prepared AOL QAC dataset and a real prefix-to-click QAC dataset from Bing. The proposed model outperformed all the baselines while specifically improving the performance for short and unseen prefixes.

8 Future work and limitations

8.1 Future work

Not all session queries are relevant to the current user prefix. Irrelevant session queries leads to noisy training data. In the future, we plan to explore modeling techniques that can select and encode only the relevant queries as personalized contexts for TRIE-NLG. Additionally, we will explore *on-the-fly* models like RAG (Lewis et al. 2020), which can provide additional (better) content or completions instead of trie completions to boost the performance. Finally, we will explore a transfer learning approach to extend this to multilingual QAC system.

8.2 Limitations

The proposed TRIE-NLG model operates in a two-step process, comprising the extraction of auto-completions from a trie and augmentation in the NLG model. As a consequence of this approach, the model exhibits slightly higher latency compared to the standard NLG model due to trie lookup. However, the trie lookup time is notably low. It is crucial to highlight that the proposed model is built upon a pre-trained NLG model (BART), which renders it susceptible to displaying unexpected outcomes inherited from the pre-training phase (Gehman et al. 2020). Such outcomes may include toxicity, bias, hallucination, misinformation, and other similar issues.

Author contributions All authors contributed to the design of the project. Specifically, Conceptualization, Methodology, Data preparation: [KKM]; Writing—original draft preparation: [KKM, MG, MSD]; Writing—review and editing: [everyone]; Funding acquisition, Resources, Supervision: [MG, MSD, PA].

Funding This research was partially funded by Microsoft Academic Partnership Grant 2022.

Availability of data and materials We make our code available as supplementary information. It is in line with our claims in the paper and can be trusted to reproduce the results. We use one public dataset (AOL) and a Bing dataset. Since Bing data is proprietary, it cannot be made publicly available.

Code availability We make our code publicly available and it is attached as supplementary data.

Declarations

Conflicts of interest The authors have no competing interests to declare relevant to this article's content.

Consent for publication All the authors of this work have been informed of the submission, and everyone supports it.

References

Bar-Yossef Z, Kraus N (2011) Context-sensitive query auto-completion. In: Proceedings of the 20th international conference on world wide web (pp. 107–116)


- Bhatia S, Majumdar D, Mitra P (2011) Query suggestions in the absence of query logs. In: Proceedings of the 34th international ACM SIGIR conference on research and development in information retrieval (pp. 795–804)
- Bonchi F, Perego R, Silvestri F, Vahabi H, Venturini R (2012) Efficient query recommendations in the long tail via center-piece subgraphs. In: Proceedings of the 35th international ACM SIGIR conference on research and development in information retrieval (pp. 345–354)
- Burges CJ (2010) From ranknet to lambdamart: an overview. *Learning* 11:23–581
- Cai F, De Rijke M (2016) Query auto completion in information retrieval. Universiteit van Amsterdam [Host]
- Cai F, Liang S, De Rijke, M (2014) Time-sensitive personalized query auto-completion. In: Proceedings of the 23rd ACM international conference on conference on information and knowledge management (pp. 1599–1608)
- Dehghani M, Rothe S, Alfonseca E, Fleury P (2017) Learning to attend, copy, and generate for session-based query suggestion. Proceedings of the 2017 ACM on conference on information and knowledge management (pp. 1747–1756)
- Gehman S, Gururangan S, Sap M, Choi Y, Smith NA (2020) Real Toxicity Prompts: evaluating neural toxic degeneration in language models. Findings of the association for computational linguistics: Emnlp 2020 (pp. 3356–3369). Online Association for Computational Linguistics
- Hofmann K, Mitra B, Radlinski F, Shokouhi M (2014) An eye-tracking study of user interactions with query auto completion. Proceedings of the 23rd ACM international conference on conference on information and knowledge management (pp. 549–558)
- Hsu B-J, Ottaviano G (2013) Space-efficient data structures for top-k completion. In: Proceedings of the 22nd international conference on world wide web (pp. 583–594)
- Huang C-K, Chien L-F, Oyang Y-J (2003) Relevant term suggestion in interactive web search based on contextual information in query session logs. *J Am Soc Inf Sci Technol* 54(7):638–649
- Jiang J-Y, Ke Y-Y, Chien P-Y, Cheng P-J (2014) Learning user reformulation behavior for query auto-completion. In: Proceedings of the 37th international ACM SIGIR conference on research & development in information retrieval (pp. 445–454)
- Lewis M, Liu Y, Goyal N, Ghazvininejad M, Mohamed A, Levy O, Zettlemoyer, L (2020) 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 (pp. 7871–7880)
- Lewis P, Perez E, Piktus A, Petroni F, Karpukhin V, Goyal N et al (2020) Retrieval-augmented generation for knowledge-intensive NLP tasks. *Adv Neural Inf Process Syst* 33:9459–9474
- Maxwell D, Bailey P, Hawking D (2017) Large-scale generative query autocompletion. In: Proceedings of the 22nd australasian document computing symposium (pp. 1–8)
- Mei Q, Zhou D, Church K (2008) Query suggestion using hitting time. In: Proceedings of the 17th acm conference on information and knowledge management (pp. 469–478)
- Mitra B, Craswell N (2015) Query auto-completion for rare prefixes. In: Proceedings of the 24th acm international on conference on information and knowledge management (pp. 1755–1758)
- Mitra B, Shokouhi M, Radlinski F, Hofmann K (2014) On user interactions with query auto-completion. In: Proceedings of the 37th international ACM SIGIR conference on research & development in information retrieval (pp. 1055–1058)
- Mustar A, Lamprier S, Piwowarski B (2020) Using bert and bart for query suggestion. Circle
- Park DH, Chiba R (2017) A neural language model for query auto-completion. In: Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval (pp. 1189–1192)
- Pass G, Chowdhury A, Torgeson C (2006) A picture of search. In: Proceedings of the 1st international conference on scalable information systems (pp. 1–es)
- Raffel C, Shazeer N, Roberts A, Lee K, Narang S, Matena M et al (2020) Exploring the limits of transfer learning with a unified text-to-text transformer. *J Mach Learn Res* 21(140):1–67
- Rosset C, Jose D, Ghosh G, Mitra B, Tiwary S (2018) Optimizing query evaluations using reinforcement learning for web search. In: The 41st international ACM SIGIR conference on research & development in information retrieval (pp. 1193–1196)
- Sadikov E, Madhavan J, Wang L, Halevy A (2010) Clustering query refinements by user intent. In: Proceedings of the 19th international conference on world wide web (pp. 841–850)
- Shokouhi M (2013) Learning to personalize query auto-completion. In: Proceedings of the 36th international ACM SIGIR conference on research and development in information retrieval (pp. 103–112)

- Shokouhi M, Radinsky K (2012) Time-sensitive query auto-completion. In: Proceedings of the 35th international ACM SIGIR conference on research and development in information retrieval (pp. 601–610)
- Sordoni A, Bengio Y, Vahabi H, Lioma C, Grue Simonsen J, Nie J-Y (2015) A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In: Proceedings of the 24th ACM international on conference on information and knowledge management (pp. 553–562)
- Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al (2017) Attention is all you need. *Adv Neural Inf Process Syst*. Vol 30
- Wang P-W, Zhang H, Mohan V, Dhillon IS, Kolter JZ (2018) Realtime query completion via deep language models. *ecom@ sigir*
- Wu Q, Burges CJ, Svore KM, Gao J (2010) Adapting boosting for information retrieval measures. *Inf Retrieval* 13(3):254–270
- Yadav N, Sen R, Hill DN, Mazumdar A, Dhillon IS (2021) Session-aware query auto-completion using extreme multi-label ranking. In: Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining (pp. 3835–3844)
- Yin D, Tan J, Zhang Z, Deng H, Huang S, Chen J (2020) Learning to generate personalized query auto-completions via a multi-view multi-task attentive approach. In: Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining (pp. 2998–3007)
- Zhou G, Zhu X, Song C, Fan Y, Zhu H, Ma X, et al (2018) Deep interest network for click-through rate prediction. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining (pp. 1059–1068)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.

Authors and Affiliations

Kaushal Kumar Maurya¹  · Maunendra Sankar Desarkar¹ · Manish Gupta² · Puneet Agrawal²

✉ Kaushal Kumar Maurya
cs18resch11003@iith.ac.in

Maunendra Sankar Desarkar
maunendra@cse.iith.ac.in

Manish Gupta
gmanish@microsoft.com

Puneet Agrawal
punagr@microsoft.com

¹ IIT Hyderabad, Hyderabad, Telangana 502284, India

² Microsoft India, Hyderabad, Telangana 500032, India