Arabic Query Auto-Completion for Questions at a Restaurant

Abdallah Alabed, Mohammad Faqeh, Ali Mohammed
Computer Engineering Department
Birzeit University
Birzeit, Palestine

***Abstract*— This paper describes an implementation of an Arabic query auto-completion system for customers at a restaurant. Customers frequently ask about menu items, prices, discounts, ingredients, cooking time, etc. Auto-completion is a critical feature in various user interfaces such as web pages, social media sites, and mobile apps. We examine multiple approaches to candidate generation and ranking of completion and the impact of Arabic natural language processing (NLP). We obtained the data directly from Burger King customers, which we can expand to include additional query types and information needs expression methods. We describe our dataset, assess individual system components, and overall system performance. Our aim is to improve the ordering experience by accelerating query entry. We divide the implementation into multiple phases, and we evaluate the results for each phase separately. Our system results were excellent in comparison to other systems, and it has the potential to enhance Arabic QAC. Lastly, we present a casual evaluation of the overall findings and the benefits of using QAC in our ordering system.**

I. INTRODUCTION

In many user interfaces, including search bars on websites, social networking sites [1], mobile applications, and text editors like VS Code and Sublime, auto-completion (AC) has become a widespread feature [2]. By anticipating their information needs and offering query alternatives that appropriately represent those needs, it saves consumers time and effort. In addition to identifying the user's needs, AC helps users create more precise and unambiguous queries for effective information retrieval [3].

A. Query Auto-Completion (QAC)

Modern search engines make extensive use of QAC to improve the search experience. By making suggestions based on the user's input, it shortens the time needed to enter a query. By giving users a list of choices, this functionality also reduces typing errors. Users can choose complete searches based on a few characters in areas like web and eCommerce searches, which accelerates and streamlines the process [4]. Using the user's input prefix, QAC creates a list of completions, and then ranks those suggestions depending on how closely they match the user's information needs [5].

B. Problem Description

Restaurant auto-completion system for patron queries Customers frequently inquire about the menu selections, costs, promotions, ingredients, cooking times, and other topics. It might be quite helpful to have a system that answers consumer requests by translating their incomplete queries into standard ones that can be executed in structured query language (like SQL or SPARQL).

QA is not simple, especially when users have the freedom to express any questions they want in their own words. By recommending particular completions for a given prefix, auto-completion (AC) can be used to reduce the variety of user questions. To quickly find the solution, the chosen completion can then be converted into a structured query language. We hope to address any query a user may possibly make and map it to a standard query, only provide suggestions for normal inquiries.

The customer care portal for the restaurant may benefit greatly from the integration of a system that offers direct responses to customer inquiries. There are a few crucial factors that need be taken into account in such a system:

1) The completion list's suggestions should be sorted so that the desired question is one of the first few.

2) The system should be adaptable, learning from both good ideas that users accept and bad ones that users reject.

3. The QAC process consists of two stages: generation, where potential completions for the partially entered query are identified, and ranking, where the generated completions are ranked using factors like relevance to the prefix entered, user profile, popularity, or learning-based techniques like Neural Language Modeling [6].

II. BACKGROUND AND RELATED WORK

We will go through several QAC and QA-related topics that are pertinent to the system we are building in this part. We'll discuss datasets for QAC, heuristic approaches to QAC, learning- and language-based approaches, Arabic QAC, QA, the connection between quality assurance and information retrieval, the impact of auto-completion on QA, and QA.

A. Query Auto-Completion (QAC) Heuristic Approaches:

1) Probabilistic QAC algorithms: Using the user's input prefix, time, and user profile, these algorithms determine the likelihood of generating a question completion.

2) General QAC Models: Based on the frequency of a query completion in search logs, these models rank query completions according to their historical popularity [7].

3) Time-sensitive QAC Models: These models consider the passage of time when ranking query completions because some questions are more prevalent during particular times.

4) User-centered QAC models: These models prioritize query completions depending on the user's particular context by utilizing the user's search history and interactions.

B. Learning- and language-based QAC Methodologies:

1) Learning-based QAC Approaches: These methods increase the accuracy of query completions by incorporating user interactions and time-related data. They can do this by making use of the Arabic Ontology [8], lexical syntactic patterns [9], and automatic relationship extraction [10].

2) Neural Language Modeling: To forecast the subsequent word in a sequence [11], neural language models employ recurrent neural networks (RNNs) like the Long Short-Term Memory (LSTM) [12]. For QAC ranking, unnormalized language models may be more effective [13].

C. Arabic QAC:

Although there is little research on Arabic QAC, attempts are being made to increase its effectiveness, which will improve the caliber of Arabic Question Answering (QA) [14].

D. Question Answering (QA):

QA systems respond to user inquiries with clear, concise responses rather than by delivering pertinent materials. Free text, structured SQL queries, and unstructured queries using data from a database can all be used as the foundation for quality assurance. Free text searches can be converted into structured query languages to get precise results from databases.

E. The Impact of Auto-Completion on Question Answering:

With the aid of auto-completion, users can choose suggested inquiries that can be quickly converted into structured query languages, allowing the system to quickly deliver detailed responses. The process of converting user questions into common SQL-translatable queries is made simpler by auto-completion.

F. Datasets:

Commonly utilized in QAC studies are general QAC datasets like the AOL dataset. For our system, we require a dataset with standard questions that have been converted into structured query language, variations of typical student queries, and extra information like the student's level of study and the typical duration of the question. Data was scraped from social media sites to create this dataset, and students provided variations to a Google form to represent how they would phrase each common question.

III. METHODOLOGY

A. Design flow

1) Word-Level Completion: The system starts the word-level completion as soon as the customer types a prefix into the search box. Our word-level method generates the top 5 words that begin with the prefix you entered. The frequency is determined by the query variants we have, but it might also be determined by the user's usage pattern.

2) Next-word Prediction: For each of our five completions, we forecast the subsequent n words. Because the model can identify dependencies between the upcoming word and the prior words, the prediction quality improves as the number of previous words rises. The predictions have lower priority than the user's entered inquiry, and the suggestions are fewer as the customer adds more details to the questions he wishes to ask.

B. Dataset collection

For building the dataset, we followed two approaches: data scrapping from social media and direct data collection from our families.

1) *Data scrapping from social media:* Social media sites have many restaurants pages. Some of these pages contain what does these restaurants offer and sometimes prices of their offers. These pages have many posts with different comments. So, we took some questions from these comments and we wrote it formally.
2) *Direct data collection from our families:* we asked our family members whom go to restaurants to provide us with questions they frequently ask or hear.

For both approaches, the following is a sample of our dataset:

```
1   كم سعر وجبة البرجر؟
2   كم سعر وجبة البرجر بدجاج؟
3   كم سعر وجبة البرجر بلحمة؟
4   كم سعر وجبة البرجر بدون كولا؟
5   كم سعر وجبة البرجر دبل؟
6   كم سعر وجبة البرجر بدون خضراوات؟
7   كم سعرة حرارية في وجبة البرجر؟
8   كم تحتاج وجبة البرجر من الوقت؟
9   هل يتوفر وجبة برجر مصنوعة من اللحمة؟
10  هل يتوفر وجبة برجر مصنوعة من الدجاج ؟
```

Figure 1: Dataset samples

C. QAC prediction

Our model can predict questions from substrings, words, or characters. So, it can predict next words alongside of word-level prediction.

We utilized an LSTM model for this task due to its ability to capture long-term dependencies. This is crucial in question answering, as understanding the context is necessary for accurate responses.

We constructed a model that operates sequentially, taking word sequences as input and generating output sequences. This is the common architecture employed in question answering models.

To represent individual words, we established an embedded layer, converting each word in the vocabulary into a numerical vector. This enables the model to comprehend the meaning of words effectively.

We incorporated an additional LSTM layer into the model, consisting of 128 units, with all units initialized as false by default. This allows the model to learn the weights from scratch and capture long-term dependencies.

Subsequently, we passed the model output through a hidden layer with 1000 nodes using a dense layer and the Tanh activation function. This layer enables the model to learn complex relationships between inputs and outputs.

To ensure a diverse range of probabilities within a given vocabulary size, we directed the last layer to an output layer with the specific vocabulary size and employed the Softmax activation function. This guarantees the model predicts the most likely word in the vocabulary for a given input sequence.

We trained the model on our dataset for 350 epochs, indicating the number of iterations over the entire training

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_2 (Embedding)     (None, 180, 50)           8850

 lstm_3 (LSTM)               (None, 128)               91648

 dense_4 (Dense)             (None, 1000)              129000

 dense_5 (Dense)             (None, 177)               177177

=================================================================
Total params: 406,675
Trainable params: 406,675
Non-trainable params: 0
_________________________________________________________________
```

Figure 2: Model layers

dataset. The batch size was set to 64, referring to the number of training examples used in each iteration.

The model achieved a perfect accuracy of 100.00% and a loss value of 0.0487. The loss value represents the extent to which the model's prediction deviated from the expected output on a single example. The low loss value indicates highly accurate predictions."As said before, the model is capable of predicting the questions from input prefix. The model searches for the input prefix in beginning of the questions. If it fails, then it searches in different locations: start, middle, end, or any substring. So, it directly suggests full question from the entered prefixes.

D. QAC suggestions ranking

For questions with multiple suggestions, we tend to use the following criteria: frequency ranking and distance ranking.

1) *Frequency ranking:* we took into consideration user selections to make our system adaptive. When the user selects a given autocompletion, its frequency increases. Hence, as long as the frequency increases, its corresponding sentence rank increases. And, this is our first ranking method.

```
1   1,كم سعر وجبة البرجر؟
2   2,كم سعر وجبة البرجر بدجاج؟
3   2,كم سعر وجبة البرجر بلحمة؟
4   3,كم سعر وجبة البرجر بدون كولا؟
5   1,كم سعر وجبة البرجر دبل؟
6   1,كم سعر وجبة البرجر بدون خضراوات؟
7   1,كم سعرة حرارية في وجبة البرجر؟
8   1,كم تحتاج وجبة البرجر من الوقت؟
9   1,هل يتوفر وجبة برجر مصنوعة من اللحمة؟
10  1,هل يتوفر وجبة برجر مصنوعة من الدجاج ؟
```

Figure 1: Frequency ranking

2) *Distance ranking:* when we the ranked questions with previous method have frequency of 1 (not selected by user yet), we rank the questions according to Levenshtein distance.

$$\operatorname{lev}(a,b) = \begin{cases} |a| & \text{if } |b| = 0, \\ |b| & \text{if } |a| = 0, \\ \operatorname{lev}\left(\operatorname{tail}(a), \operatorname{tail}(b)\right) & \text{if } a[0] = b[0], \\ 1 + \min \begin{cases} \operatorname{lev}\left(\operatorname{tail}(a), b\right) \\ \operatorname{lev}\left(a, \operatorname{tail}(b)\right) \\ \operatorname{lev}\left(\operatorname{tail}(a), \operatorname{tail}(b)\right) \end{cases} & \text{otherwise} \end{cases}$$

Figure 2: Levenshtein distance

E. QAC Evaluation

An effective QAC system is defined by its ability to provide answers to user questions using the shortest entered prefix. Additionally, the accuracy of the suggested questions holds significant importance. The accuracy of our model is depicted in the graph below.

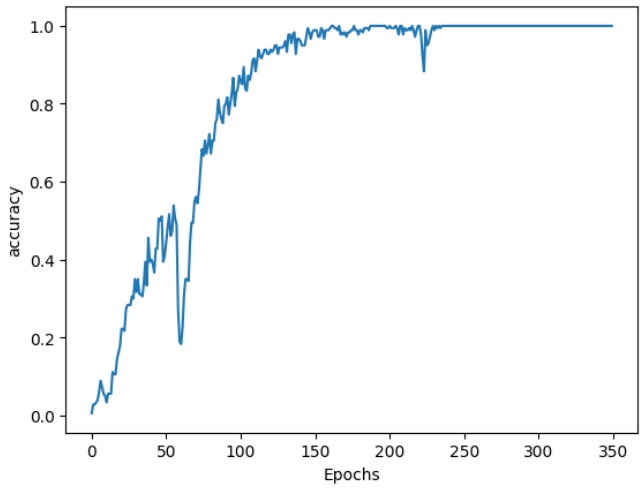

Figure 3: Model accuracy

From the graph we notice at after epoch 250 the accuracy is nearly 1. Hence, our model is the best one of the 350 generated models.

Alongside the accuracy, there is loss measure which indicates how much our model is bad in predicting single example.

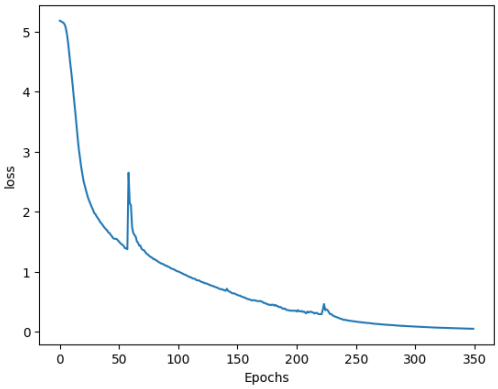

Figure 4: Model loss

From the Fig. 6, it is noticeable that the loss is nearly zero in the epoch number 350.

IV. Results

We have tested many cases from our data set in the search box.

1. Normal search: the prefix is in the beginning of the word. In this case, we searched for the questions that start with "كم". Here are the following top suggestions:

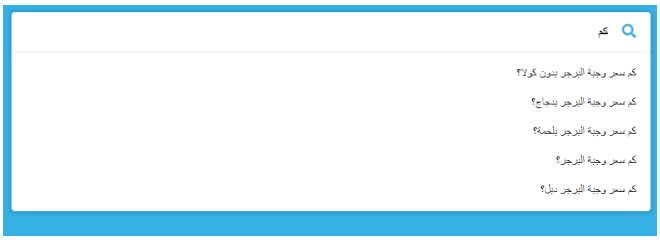

Figure 5: Search results of "كم"

2. Show ranked questions upon user selection: when the user selects one of the suggested questions, its rank increases. For example, if the user selects " كم سعر وجبة البرجر؟". Its rank increases, and it is suggested first. The following figure shows how it worked:

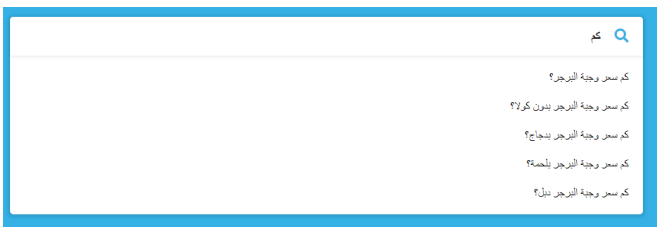

Figure 6: Ranked suggestions

3. Results when the entered prefix doesn't exist in the beginning of the question: in this case, the model searches for all results in all positions of questions.

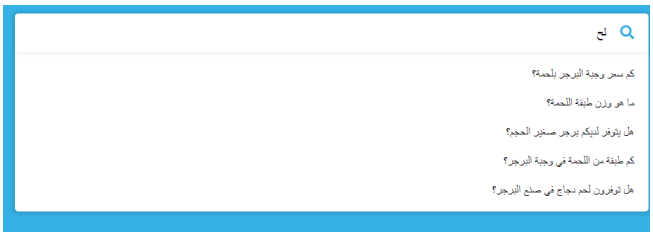

Figure 7: Results for prefixes that doesn't exist in the beginning

4. Search results when the frequency is 1: in this case, we use Levenshtein distance to rank the suggested questions. The following figure demonstrates it:

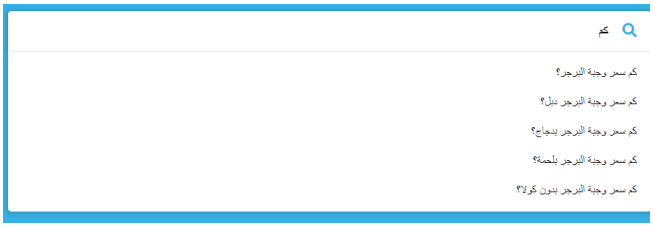

Figure 8: Levenshtein distance

V. CONCLUSION AND FUTURE WORK

At this paper, we investigated and discussed the issue of Arabic QAC at restaurants from various angles. Additionally, we looked at the impact of QAC on QA and how it can increase QA efficiency by lowering the volume of queries while enhancing their quality, a critical issue for many apps that interact with users. Our software answered queries about burger joints. With the right datasets and NLP tools, we do think the approach can be easily applied to other domains or applications. We'll make changes to our system so it may be applied to other projects. Additionally, we'll work to expand our dataset and make it accessible to other researchers with an interest in Arabic QAC.

REFERENCES

[1] F. Cai and M. de Rijke, "A survey of query auto completion in information retrieval," *Foundations and Trends® in Information Retrieval*, vol. 10, no. 4, pp. 273–363, 2016.

[2] L. Li, H. Deng, A. Dong, Y. Chang, H. Zha, and R. Baeza-Yates, "Analyzing user's sequential behavior in query auto-completion via markov processes," in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, (New York, NY, USA), p. 123–132, Association for Computing Machinery, 2015.

[3] S. Whiting and J. Jose, "Recent and robust query auto-completion," pp. 971–982, 04 2014.

[4] Z. Bar-Yossef and N. Kraus, "Context-sensitive query autocompletion," in *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, (New York, NY, USA), p. 107–116, Association for Computing Machinery, 2011.

[5] A. Sethy, S. Chen, E. Arisoy, and B. Ramabhadran, "Unnormalized exponential and neural network language models," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5416–5420, 2015.

[6] A. R. A. Qureshi and M. A. Akcayol, "Long short-term memory-based query auto-completion," in *2021 8th International Conference on Electrical and Electronics Engineering (ICEEE)*, pp. 259–266, 2021.

[7] L. Qiu, H. Zhou, Y. Qu, W. Zhang, S. Li, S. Rong, D. Ru, L. Qian, K. Tu, and Y. Yu, "QA4IE: A question answering based framework for information extraction," *CoRR*, vol. abs/1804.03396, 2018.

[8] O. Obeid, N. Zalmout, S. Khalifa, D. Taji, M. Oudah, B. Alhafni, G. Inoue, F. Eryani, A. Erdmann, and N. Habash, "CAMeL tools: An open-source python toolkit for Arabic natural language processing," in *Proceedings of the 12th Language Resources and Evaluation Conference*, (Marseille, France), pp. 7022–7032, European Language Resources Association, May 2020.

[9] M. Jarrar, "The arabic ontology - an arabic wordnet with ontologically clean content," *Applied Ontology Journal*, vol. 16, no. 1, pp. 1–26, 2021.

[10] N. Loukil, K. Haddar, and A. Ben Hamadou, "A syntactic lexicon for arabic verbs.," 01 2010.

[11] M. G. Al Zamil and Q. Al-Radaideh, "Automatic extraction of ontological relations from arabic text," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 26, p. 462–472, dec 2014.

[12] Y. Li, J. Amelot, X. Zhou, S. Bengio, and S. Si, "Auto completion of user interface layout design using transformer-based tree decoders," 2020.

[13] S. Abiteboul, Y. Amsterdamer, T. Milo, and P. Senellart, "Autocompletion learning for xml," 05 2012.

[14] K. Arkoudas and M. Yahya, "Auto-completion for question answering systems at bloomberg," in *the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018* (K. Collins-Thompson, Q. Mei, B. D. Davison, Y. Liu, and E. Yilmaz, eds.), pp. 1351–1352, ACM, 2018.

[15] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Computation*, vol. 1, no. 2, pp. 270–280, 1989.