IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 6, 2021, accepted April 16, 2021, date of publication April 22, 2021, date of current version May 3, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3074950

Arabic Question Answering Systems: Gap Analysis

MARIAM M. BILTAWI[1], SARA TEDMORI[1], AND ARAFAT AWAJAN[1,2]
[1]Computer Science Department, Princess Sumaya University for Technology, Amman 11941, Jordan
[2]Computer Science Department, Mutah University, Karak 61710, Jordan

Corresponding author: Mariam M. Biltawi (maryam@psut.edu.jo)

ABSTRACT Question-answering (QA) systems aim to provide answers for given questions. The answers can be extracted or generated from either unstructured or structured text. Therefore, QA is considered an important field that can be used to evaluate machine text understanding. Arabic is a challenging language for many reasons; although it is spoken by more than 330 million native speakers, research on this language is limited. A few QA systems created for Arabic text are available. They were created to experiment on small datasets, some of which are unavailable. The research on QA systems can be expanded into different components of QA systems, such as question analysis, information retrieval, and answer extraction. The objective of this research is to analyze the QA systems created for Arabic text by reviewing, categorizing, and analyzing the gaps by providing advice to those who would like to work in this field. Six benchmark datasets are available for testing and evaluating Arabic QA systems, and 26 selected Arabic QA systems are analyzed and discussed in this research.

INDEX TERMS Answer extraction, Arabic question answering, information retrieval, question analysis, question answering dataset, question answering system.

I. INTRODUCTION

Question answering (QA) is a benchmark task with significant applications for users. It is a challenging task that can be used to evaluate machine text understanding. QA systems aim to provide an answer for a given question extracted or generated from either unstructured or structured text. Community QA systems, generating question systems, and dialog systems are examples of QA systems. While a general QA system aims to make the machine answer questions, other applications focus on other purposes. For example, a community QA system focuses more on information retrieval rather than on answer extraction [1]. A community QA dataset can be created by collecting questions and answers from forums or websites. A generating question system is the opposite of a QA system; i.e., a generating question system generates questions for given passages by exploiting knowledge bases [2]. A dialog system, on the contrary, aims to respond to any type of text by generating a reply in accordance with the given input [3], [4].

The associate editor coordinating the review of this manuscript and approving it for publication was Chi-Hua Chen.

Natural language understanding is a subfield of natural language processing (NLP) that employs machines to understand human language. QA is one of the most useful ways to evaluate the ability of machines to understand natural language by querying machines and evaluating their answers. Answers are scored to evaluate machine performance. Scoring answers can be considered a problem in itself. Scoring can be adopted in educational centers, specifically in the case of scoring exams, and can save time if it is performed automatically. Creating scoring systems is challenging and depends on the type of question posed. For example, designing a scoring system for multiple-choice questions is considerably easier than designing one for free-form answer questions [5]–[7]. The purpose of each of the abovementioned systems differs, and their datasets also vary in terms of data shape. Each system needs its own dataset to evaluate its performance. Any resource can be built either manually or semiautomatically. When a resource is constructed manually, a set of instructions and crowd workers are needed. By contrast, constructing a resource semiautomatically relies on existing resources and automatic techniques. Automatic techniques can be either rule-based or machine learning (ML)-based; in turn, ML-based techniques can be supervised or unsupervised.

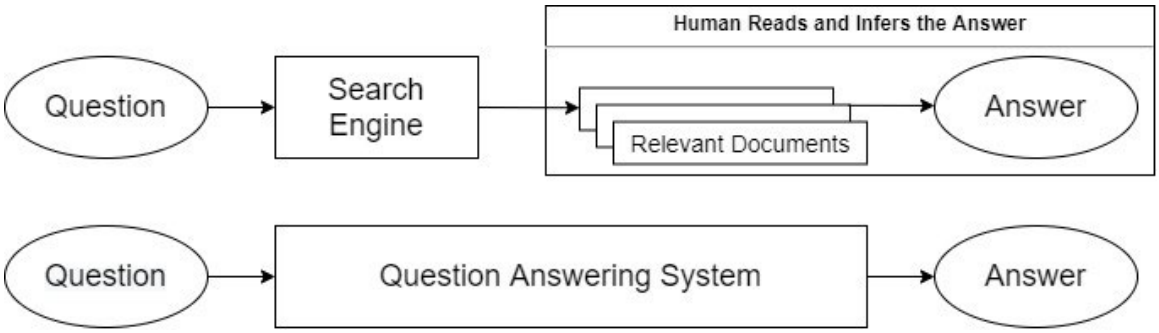

FIGURE 1. Question answering system vs. search engine.

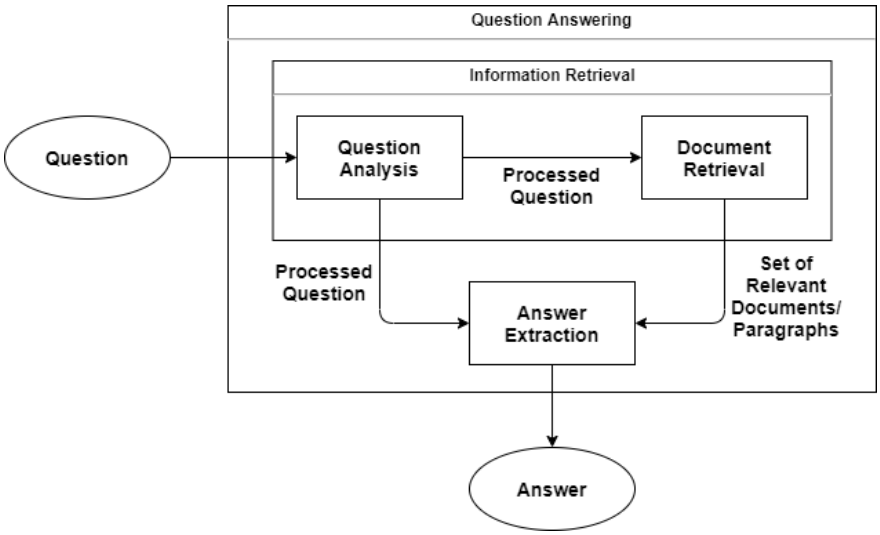

FIGURE 2. Components of QA system.

This research focuses on QA systems. As previously stated, a QA system is a computer science application that aims to provide answers to questions posed by humans. QA systems can be regarded as an extension of search engines. The main difference between QA systems and search engines is that search engines provide a group of relevant documents as a result, thereby leaving the answer extraction task to the users; QA systems provide one answer, thereby saving users navigation time, as shown in Figure 1. QA intersects with different fields in computer science, mainly NLP, information retrieval, human–computer interaction, and artificial intelligence [8].

The vast majority of QA systems share three main components [9]: (1) question analysis, (2) document retrieval, and (3) answer extraction. Question analysis is the first component of QA systems. Their purpose is to analyze a question, and question analysis usually involves one or more of the following tasks: question segmentation, question classification, question formation, answer type detection, keyword extraction, and query expansion (QE). The second component, document retrieval, receives the processed question as an input and aims to retrieve relevant documents, which are in turn passed to the third component. Document retrieval involves one or more of the following tasks: sentence retrieval, short answer retrieval, paragraph retrieval, identification of relevant documents, ranking of relevant documents, and passage retrieval. In addition to receiving the relevant documents from the second component, the last component (namely, answer extraction) receives the processed question from the first component as input and extracts the most relevant answer to the question from the relevant documents. Answer extraction may encompass the following tasks: different NLP techniques, including name entity recognition (NER), answer validation, answer scoring and rating, answer selection, and answer presentation [9]. Figure 2 illustrates the QA system components.

QA systems can be classified in different ways on the basis of different factors, such as (1) domain coverage; (2) retrieval approach or models; (3) supported languages; (4) data source size, heterogeneity, genre, and media; and (4) knowledge base. These factors were discussed by Ray and Shaalan [10] and Mishra and Jain [11]. In terms of the knowledge base, Ray and Shalaan specified that QA systems can be classified in accordance with whether the question is answered using a local corpus or the Web; additionally, Mishra and Jain noted that data structure (i.e., if the data source is structured, semistructured, or unstructured) can be used in the classifi-

TABLE 1. Categories of QA systems.

Factors	Category
Domain Coverage	Open Domain Closed Domain
Knowledge base	Structured Semi-structured Unstructured
Linguistic Analysis level	Syntactic Morphological Semantic Pragmatic/Discourse
Document Retrieval	Rule-based Search Technique Search Engine
Answer Extraction	Rule-based ML-based
Supported Languages	Arabic English . . .
Question Types	Factoid Non-Factoid Hybrid

cation task. Ray and Shaalan added one more factor to the previous factors; this factor is the answer source, i.e., if the QA system is automatic or collaborative. In contrast, Mishra and Jain added two more factors: (1) types of questions supported, such as factoid, list, hypothetical, confirmation, or causal; and (2) the level of analysis, such as morphological, syntactic, semantic, pragmatic, and discourse, performed on questions.

Table 1 illustrates the main factors, along with their classes, that are considered when classifying the papers discussed and surveyed in this research. Each system can be given more than one class. The classes are assigned in accordance with the following factors: (1) domain coverage, (2) question types, (3) knowledge base, (4) linguistic analysis level, (5) document retrieval, (6) answer extraction, and (7) supported languages. Domain coverage, as stated earlier, can be either open or closed. Most open-domain QA systems retrieve the answers to the posed questions from the Web, whereas other QA systems rely on the use of a large dataset of open-domain documents. Closed-domain QA systems restrict the answer to specific domain documents. Knowledge bases, as stated previously, can be structured (such as relational databases), semistructured (such as XML files), or unstructured (such as natural language). The linguistic analysis level differs from the question analysis level discussed earlier because the former encompasses the analysis performed on the question and retrieved documents rather than the question alone. The classes include syntactic, morphological, semantic, and pragmatic/discourse. Document retrieval can be accomplished using either rule-based methods, search techniques, or search engines. Rule-based methods include regular expressions and a simple text-matching technique. Search techniques include the vector space model (VSM). Search engine APIs include Google Search. Depending on the approaches used to extract answers, answer extraction can be classified as rule- or ML-based. The classes in document retrieval and answer extraction differ from those discussed earlier. The need for such classification is to demonstrate the different techniques used in both factors for the surveyed papers. Supported languages specify the type of languages experimented with for the QA system itself.

A question is a natural language sentence, phrase, or even a word used to request information or test someone's knowledge [12]. In [13], the authors classified the questions of English teachers into four categories: (1) convergent and divergent; (2) productive and reproductive; (3) display and referential; and (4) form, content, and purpose. Convergent questions have one right answer, whereas divergent questions may have many answers. Productive questions are known as higher-order questions and are used to ask readers for their own opinion. The types of productive questions are analysis questions (e.g., "Why?," "How?," and "What?"), synthesis questions (e.g., "How many ways can you study for a test?"), and evaluation questions (e.g., "Do you agree with ...?" and "What is your opinion on ...?"). Reproductive questions are known as lower-order questions, such as recall questions (e.g., "What did the character say?"), comprehension questions (e.g., "Why did the character make a phone call?"), and application questions (which ask readers to provide their own experience on the basis of a given story, e.g., "How would you solve ...?"). The answers to display questions are known by the questioner, and such questions are used to test the understanding or knowledge of the respondent (e.g., "Where does the bread come from?" and "Plants or animals?"). The answers to referential questions are unknown to the questioner, and such questions are used to seek information (e.g., "The word X in the passage refers to ...?"). Last, form questions are either "why" or "yes/no" questions, content questions are either fact or opinion questions, and purpose questions are display questions.

Questions in the Arabic language can be asked either (1) by using interrogative words (IWs) or (2) without using them. IWs in the Arabic language can come at the beginning or end of a question. Arabic IWs are divided into two categories: (a) Hamza (أ) and Hal (هل), which are called interrogative particles (IPs) and mainly used for yes/no questions; and (b) "who" (من), "what" (ما، ماذا), "where" (اين), "when" (ايان متى ،), "how" (كيف ، انى), "how much/many" (كم), and "which" (اي) words, which can be used with different types of questions. In this research, we use the term IWs to refer to both IPs and IWs for simplicity. The questions that do not use IWs usually start with verbs, such as "عدد" (which means "list") and "اشرح" (which means "explain") [14].

In accordance with the Text Retrieval Conference (TREC),[1] which is a series of workshops designed to enhance research in information retrieval and is cosponsored by the National Institute of Standards and Technology, eight main types of questions can be used in QA systems: factoid, list,

[1] https://trec.nist.gov/

hypothetical, definition, causal, relationship, confirmation, procedural, description, and opinion. This categorization is adopted for classifying the question-type factor. Factoid questions demand a specific answer to an entity or an event, such as a person's name, organization, and location. Such questions usually start with ''who,'' ''where,'' ''when,'' ''how much/many,'' and ''which.'' The answer to list questions is a list of entities, facts, or events; these questions can be considered factoid questions that are repeated multiple times (e.g., ''What are the brand names of Japanese cars?'' and ''List the names of the cities in Jordan.''). Hypothetical questions are based on assumed facts and seek a general answer for the given question. These questions usually start with ''what if'' or ''what would happen if.'' Another type of question whose answer is not a name entity is a definition question. Definition questions seek a definition for an entity, event, or term and usually start with ''what is.'' Additionally, the answer to a causal question is not an entity. Causal questions require answers that have considerable details and explanations about entities. Relationship questions require defining relationships or comparing two entities. Confirmation questions require answers that are either ''yes'' or ''no.'' Procedural questions comprise a subset of questions that usually start with ''how'' [15]. Description questions differ from definition questions in that the former may give not only the definition of a term but also additional information about it (e.g., ''Can you describe a typical day of yours?'') [16]. Last, opinion questions request opinions about something. The question-type factor in Table 1 classifies the question types into three classes: factoid, nonfactoid, and hybrid. Nonfactoid questions represent one or more of the types of questions, such as list, hypothetical, definition, causal, relationship, confirmation, procedural, description, and opinion. Hybrid questions are a combination of factoid and nonfactoid questions.

The objective of this research is to address the topic of QA by using Arabic text by surveying and categorizing the available work proposed in the field. The remainder of this paper is structured as follows. Section 2 briefly discusses the Arabic language and NLP. Section 3 presents the related work. Section 4 provides the methodology. Section 5 describes the Arabic resource-building attempts for QA systems. Section 6 shows the measures used in evaluating QA systems. Section 7 provides a comprehensive survey of available work in Arabic QA systems. Section 8 presents an overall discussion, and Section 9 is the conclusion.

II. ARABIC LANGUAGE AND NLP

A. ARABIC LANGUAGE

Arabic is a Semitic language that is spoken widely, mainly by more than 330 million native speakers. Arabic is the official language of over 22 countries, spreading from Northwest Africa to the Arabian Gulf in the east. Arabic is the language of Islam; thus, Arabic is considered the second language of several Asian countries, such as Indonesia, Pakistan, and Malaysia. The Arabic language has three main forms: Classical Arabic, Modern Standard Arabic, and colloquial Arabic. Classical Arabic is the oldest form of Arabic; Classical Arabic is fully vowelized, represents the language of the Quran and is no longer used as a spoken language. Most Arabic speakers use Modern Standard Arabic either in television news and the media (written and spoken) or as a written language in educational organizations, such as schools and universities; on street and shop signs, in books; and in formal paperwork and documentation. Colloquial Arabic is used in speech and differs by region. The Middle East alone has different forms of colloquial (Jordanian, Lebanese, Palestinian, Syrian, etc.), but the difference between the different forms of Middle Eastern colloquial is insignificant compared with that of North African colloquial [17]. Although Arabic is popular, work on the Arabic language is still limited, especially in QA. To understand the challenges of the Arabic language, the basics of Arabic should be understood.

FIGURE 3. The shape of the letters (ب and غ) pronounced (Ba and Gha, respectively).

Arabic differs from English in many ways. For example, the writing direction of Arabic is from right to left, and the Arabic alphabet contains 28 letters, of which 25 are consonants and 3 are long vowels (ي ، ا و pronounced as Alif, Ya, Waw). The form and shape of each letter change in accordance with its position in the word. For example, Figure 3 demonstrates the shapes of the letters ب and غ (pronounced as Ba and Gha, respectively); the letters are disconnected at the beginning, at the end, and in the middle of words. Three other letters, namely, Ta-Marbouta (ة), Alif-Maqsoura (ى), and Hamza (ء), exist. Hamza (ء) can be isolated, on the Alif (أ), under the Alif (إ), on the Waw (ؤ), and on the Alif-Maqsora (ئـ ، ـئـ ، ـئ). Diacritics, also called short vowels, are symbols used in the Arabic language to differentiate either the meaning or pronunciation of words or to even change the grammatical formulation of words. Diacritics are usually placed above and/or below the letters of words. The Arabic diacritics are Fat-ha (ـَ); Damma (ـُ); Kasra (ـِ); Sokon (ـْ); Tanween-Fateh (ـً); Tanween-Dam (ـٌ); Tanween-Kaser (ـٍ); Madda, which is usually placed above Alif (آ); and gemination mark (ـّ), which is called Shadda in Arabic [18]. Some punctuation marks used in the Arabic language differ from those used in the English language; these

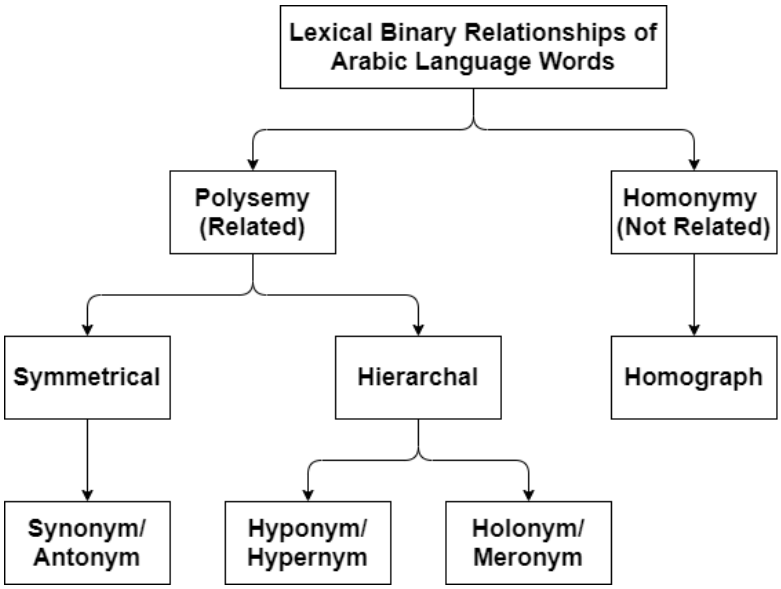

FIGURE 4. Lexical binary semantics of arabic language words.

marks include the question mark (the English question mark ? and Arabic question mark ؟), comma (the English comma, and Arabic comma ،), and semicolon (English semicolon; and Arabic semicolon ؛).

Similar to words in other languages, words in Arabic can belong to one of three main categories: verbs, nouns, and particles. However, these categories can be further classified. Arabic verbs have three main forms: perfect, imperfect, and command. Arabic nouns can be categorized in accordance with different considerations, such as masculine or feminine; definite or indefinite; and singular, plural, or dual. Particles in Arabic comprise pronouns, prepositions, conjunctions, interrogatives, interjections, adjectives, and adverbs. Arabic morphology is rich and complex in derivational and inflectional aspects. The main types of morphemes are concatenative and template-based. Concatenative morphemes can be created by combining the stem with affixes, clitics, or both; this process is also called inflection. Most Arabic words are generated from basic entities, namely, roots. Most Arabic roots consist of three letters, but four- and five-letter roots can also be found. Arabic has three types of affixes, namely, prefix, infix, and postfix. In English, only prefix and postfix are available. Derivation is the process of using roots accompanied by a list of patterns to generate words [14].

The semantic level of any language concerns the meaning of words and their relationships with one another. The main lexical binary relations that may exist among words for different languages consist of polysemy and homonymy, each having subcategories. Polysemy includes symmetrical and hierarchal categories, and homonymy consists of homographs and homophones. The symmetrical category can be either synonyms or antonyms, whereas the hierarchal category can be either hyponyms/hypernyms or holonyms/meronyms. Arabic differs from other languages in that it has no homophones because it is a phonetic language; additional details can be found in [19]. Figure 4 illustrates the main categories of the lexical relationships of Arabic words.

B. ARABIC LANGUAGE CHALLENGES

Arabic is a challenging language. The main challenges include the following:

1. Arabic has diverse forms, but the widely used forms are the different types of colloquial. In addition to these forms, some Web users use English letters to write Arabic; this condition is sometimes called Arabizi [20].
2. Arabic does not have upper- or lowercase letters; thus, proper nouns are difficult to distinguish. It differs from English, which can distinguish proper nouns easily because they start with a capital letter.
3. Arabic has not only prefixes and suffixes added to the beginning and end of words but also infixes added inside words.
4. Arabic has a rich morphology, with one root being a base of different words with different meanings.
5. Arabized words that have no root exist. These words are not Arabic words but are used considerably by Arabic native speakers.
6. In Arabic, the meaning of one word can differ depending on the context, even though the pronunciation of the word can be different. This problem can be overcome by using diacritical marks. The problem is that most people do not use diacritics when writing in Arabic and depend on their knowledge.
7. Particles in Arabic usually come attached to a word itself; hence, a single word in Arabic may represent a sentence.
8. Arabic resources are limited, especially those that are freely available and used as a benchmark. In addition

to resources, Arabic tools are also limited, and most of them are not freely available.

C. ARABIC NLP

To overcome the challenges posed by the Arabic language, several NLP techniques are available. For instance, text preprocessing can encompass tokenization, noise removal, normalization, stemming, and rooting. Another NLP step is text augmentation, which encompasses part-of-speech (POS) tagging, NER, and QE. This subsection presents the main Arabic NLP techniques shared by most of the research papers that have addressed the problem of QA and most of the preprocessing tools of the Arabic language. However, these techniques were implemented mainly in the text analysis component and document retrieval component.

The main NLP step in any language is text preprocessing, which converts an unstructured natural language into a structured or standard form that can be used in later steps. Languages differ from one another in many aspects. The preprocessing also differs, but some general steps can be shared by all languages having a difference in specific details. These general steps are tokenization, noise removal, normalization, stemming, and rooting. Nonetheless, some preprocessing steps are not shared by languages. For example, in the English language, uppercase letters are converted into lowercase letters as a preprocessing step. No such step exists in Arabic because Arabic does not have upper- or lowercase letters.

Tokenization is the first preprocessing step that is shared by all languages; it is the breaking of a sentence, a paragraph, or an entire document into smaller pieces. These pieces can be chunks of multiple words (i.e., phrases), words, or smaller pieces (such as morphemes). Choosing the token type depends on the application itself. Tokenization is usually applied to QA systems at the word level. The second preprocessing step is noise removal, which refers to the cleaning of text. In Arabic text, noise can be HTML tags, extra white spaces, punctuations, non-Arabic text, numbers, diacritics, and stop words. Removing one of the abovementioned seven things depends on the application itself. HTML tags and extra white spaces are considered noise and should always be removed. Punctuations are sometimes important to indicate the end of a sentence, for example, and diacritics are occasionally important to remove the ambiguity of a word. Some non-Arabic words, such as Arabizi words, may not be noise. One way to address this issue is to create a dictionary for Arabizi words that can be replaced with Arabic written words equivalent to Arabizi words. Numbers can be important if the question concerns time. Stop words can also be vital. For example, in the case of QA, IWs are considered stop words that are needed to know the question type. Therefore, they should not be removed for QA application. Researchers should be careful when removing such information and should be aware of application needs.

The third preprocessing step is normalization, which is used to standardize words. Normalization is implemented by (1) removing elongation or TATWEEL (-) in Arabic. For example, the word "princess" in Arabic with elongation is written as (أمـــــيـــــرة) after removing elongation, the word becomes (أميرة). Normalization is also implemented by (2) removing HAMZA (ء) from the two letters (ىء) and (ؤ) to become ALIF-MAQSOORA (ى) and WAW (و), respectively; (3) transforming all forms of ALIF (أ ، آ ، إ) into bare ALIF (ا); and (4) removing the two dots of TA' AL-MARBOOTA (ة) to become HA' (ه) and, sometimes, the two dots of YA' (ي) to become ALIF-MAQSOORA (ى). The fourth step is stemming and rooting. In Arabic, a great difference exists between stemming and rooting. Stemming is the act of removing the affixes from the beginning and the end of a word. Rooting is the process of converting the word into its origin. As stated earlier, Arabic has three types of affixes; thus, removing only the prefix and postfix will not give the origin of the word.

Another important step in NLP is text augmentation. Text augmentation is the process of enriching the original text with information the text did not have previously. Examples of text augmentation are POS tagging, NER, and QE. POS tagging is the process of assigning a grammatical tag for each word in a sentence, paragraph, or document [21]. NER is the process of assigning classes for some text elements, such as the names of an organization, a person, and a location [22]. QE is the process of enriching keywords with either synonyms or different forms of words [23] to reformulate a query. POS tagging and NER can be implemented after tokenization and before other text-processing steps. QE can be implemented after removing stop words. In the QA application, POS tagging, NER, and QE are considered important tasks. For example, NER can identify if the question asked is about a certain thing or a person. If the question does not have any name entities (NEs), then POS tagging may assist in recognizing the question tags and determining what the question is about. QE can help in disambiguating keywords by retrieving their synonyms or by representing the words with their different morphological representations. QE accordingly helps the document retrieval component find relevant documents.

III. RELATED WORK

The literature review has shown that four journal articles and two conference papers focusing on Arabic QA have been published in the QA literature; to the best of the authors' knowledge, the last survey was published in 2017. For example, Ezzeldin and Shaheen [24] reviewed studies that focus on the three main components of QA systems. The authors highlighted the challenges imposed on the Arabic QA and how these challenges can be addressed. The authors presented the available tools that can aid in Arabic QA experiments. Additional information, including a discussion of the main measures needed to evaluate Arabic QA systems, was presented by the same authors in [25]. The authors flagged several freely available datasets created for Arabic QA. The authors organized the presentation of Arabic QA research

TABLE 2. Summary of QA survey papers.

Ref.	Published Year	Publisher	Summary
[24]	2012	ACIT[2]	• Categorize Arabic QA systems according to the main components of QA. • Discuss the challenges faced using Arabic in QA. • List the measurements used to evaluate Arabic QA. • List available tools for processing Arabic text. • List freely available datasets can be used in Arabic QA. • List the future trends.
[25]	2014	Springer	
[10]	2016	IEEE	• Arabic QA development history. • Discuss the challenges faced using Arabic in QA. • Discuss the factors that can be used to categorize Arabic QA. • Discuss the main components to design an Arabic QA. • List the measurements used to evaluate Arabic QA. • List available tools for processing Arabic text. • Address some issues that are missing in Arabic QA.
[26]	2016	ACIT	• Comparative study between 11 Arabic QA systems.
[8]	2016	Springer	
[27]	2016	SAI[3]	• Survey and categorize 13 Arabic QA systems.
[28]	2017	IJAISC[4]	• Brief summary of 5 Arabic QA systems.

efforts in surveys in accordance with the three main components of QA systems: (1) question processing, (2) document processing, and (3) answer processing. The authors also presented future trends in Arabic QA, such as (1) the need for further research on restricted Arabic QA, (2) the use of application-dependent approaches, (3) the use of semantics, (4) the use of deep parsing/reasoning, and (5) the use of logic- and inference-based techniques.

Ray and Shaalan [10] started their survey with a discussion on Arabic QA systems and their development history, followed by a presentation of the challenges faced by Arabic QA systems. Next, the authors presented the factors that determine the categorization of Arabic QA systems; these factors were stated earlier in this paper. The authors also discussed the main components and subtasks needed to design an Arabic QA system, in addition to providing an overview of the main metrics that can be used to evaluate Arabic QA systems and of the available tools that can be used in processing Arabic text. The authors emphasized issues that still require attention in Arabic QA; these issues include the need for additional research on (1) restricted (closed-domain) Arabic QA and (2) nonfactoid Arabic QA. The authors further highlighted the need to (3) develop collaborative Arabic QA systems, (4) exploit semantic Web resources, (5) develop testbeds and standards for evaluation, and (6) use social media data and blogs.

Bakari *et al.* [8], [26] presented a comparative study of 11 Arabic QA systems. The authors discussed these systems, marked the subtasks of the main QA components used in each of these Arabic QA systems, and conducted a comparison in accordance with the following criteria: domain, programming language, the use of WordNet, the use of ontologies, linguistic resources, the approaches used, dataset sources, answer shape, question type, features, and experimental result. The authors also provided the contributions and limitations of each of the 11 Arabic QA systems. Sati *et al.* [27] presented a review of Arabic QA systems and classified them into two main categories: answers generated from raw text and answers generated on the basis of frequently asked questions. Thirteen Arabic QA systems were surveyed and compared in terms of goal, domain coverage, dataset used, experimental results, and limitations. Bouziane *et al.* [28] presented a survey of QA systems that are based mainly on ontology. The authors briefly discussed five Arabic QA systems. Table 2 shows a summary of the surveys discussed above.

This paper differs from the other papers in that it addresses the history of the QA systems created for Arabic from the first system created until 2020 and provides detailed information about each system. The available surveys are outdated. For example, the last two surveys published in 2016 and 2017 covered papers published before 2015. Many things can change in 6 years in terms of available datasets, new technologies, new trends, and research conducted in this field. The main goal is to inform the reader where Arabic QA systems are now, what is missing, and what can be done. This paper also provides a gap analysis comparing Arabic and other language QA systems.

IV. METHODOLOGY

Many research papers targeting QA systems for different languages can be found. The first QA system for English was created in 1961 which was a system for Baseball [29]. In the past few decades research on English QA systems seemed to have progressed to the point where that they are capable of answering questions with high accuracy. Yet, work on Arabic language is still in its infancy. In this research the authors seek to answer the following research questions:

What is the current state of research in Arabic QA systems? What experiments have been conducted to date and to what degree of success? What are the gaps? And how can the current Arabic QA systems be improved?

To answer these questions, relevant research papers were collected, reviewed, and studied. The collected research

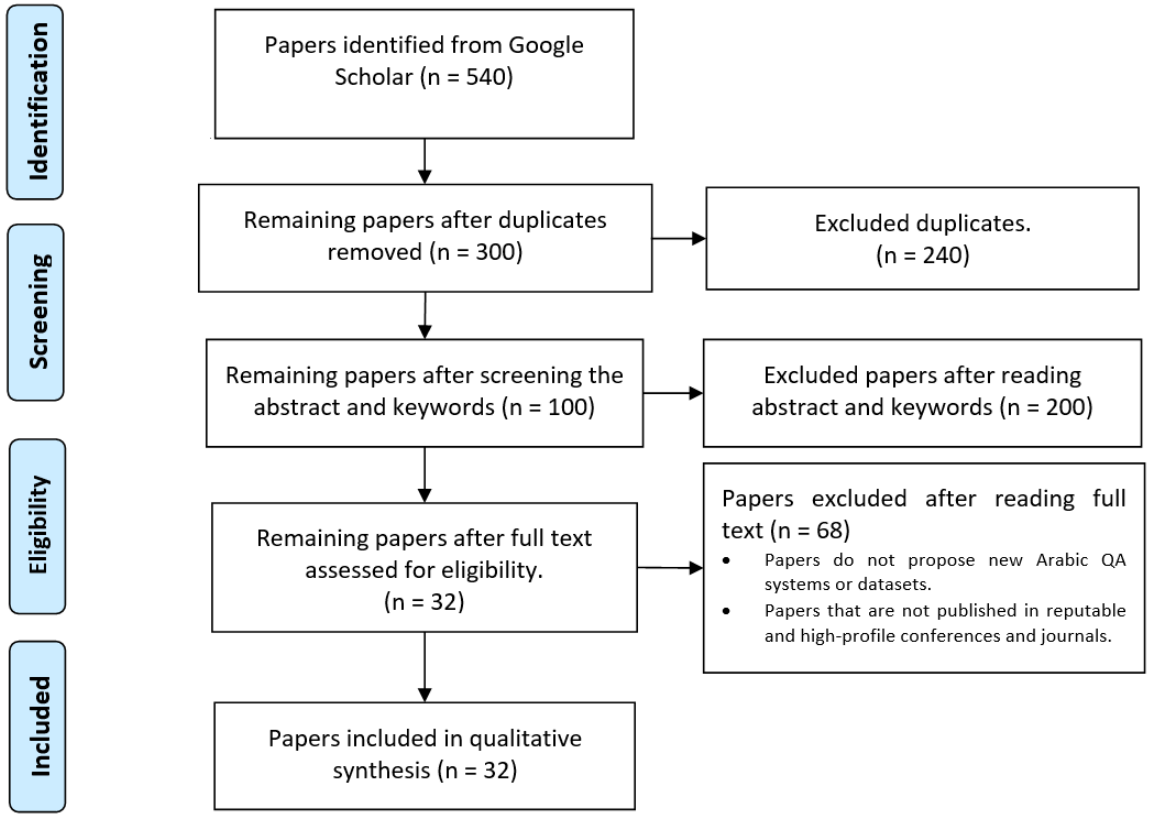

FIGURE 5. Flow diagram showing the papers selection process, using the PRISMA approach.

papers were date from 1993 when the first Arabic QA system was proposed, until the year 2020. Conducting a review of the available research papers is vital to understand the state of Arabic QA systems and to propose avenues of improvement. The following section provides full details of the paper selection criteria.

A. PAPER SELECTION CRITERIA

The papers were collected mainly by searching Google Scholar[5] by using the keywords "Arabic question answering," "question answering systems," "answering Arabic questions," "Arabic question answering resource," "Arabic question answering dataset," and "Arabic question answering corpus." A total of 540 papers were collected, of which only 32 were selected in accordance with the selection criteria detailed next.

Figure 5 demonstrates the paper selection process that followed the PRISMA approach. In the screening process, only papers that included the search keywords "Arabic" and "question answering system" in their title, abstract, or keyword list were selected for inclusion in Arabic QA systems. Papers that contain the keyword "Community Question Answering" were ignored. Only papers that include the keyword "Arabic" and the keywords "dataset," "corpus," or "resource" in their title, abstract, or keyword were selected for inclusion in Arabic QA dataset papers.

In the eligibility process, while considering that the work on Arabic is limited, the list of selected papers was narrowed down to include only papers that proposed new systems or datasets that were published in reputable and high-profile conferences and journals with the exception of some research papers. The goal of these selection criteria was to find and review all research publications that proposed either an Arabic QA system or a resource for the Arabic QA system to provide a comprehensive review of what has been done and what needs to be done.

The final selection of publications, which comprised 32 Arabic QA research papers, was dated between 2001 and 2020. Of the 32 papers, only 6 addressed Arabic QA datasets, and the remaining 26 proposed QA systems. Figures 6 and 7 illustrate the statistics of selected papers in accordance with time and question types, respectively. Most QA systems were proposed in 2014.

The focus of Arabic QA systems was on factoid-type questions with 14 systems, followed by nonfactoid-type questions with 7 systems and hybrid-type questions with 5 systems. However, the number of Arabic QA datasets was distributed equally among the three types of questions.

To categorize Arabic QA systems, we followed the classification factors and the suggested classes for each factor, as shown in Table 1, and neglected the factor "supported

[5] http://scholar.google.com

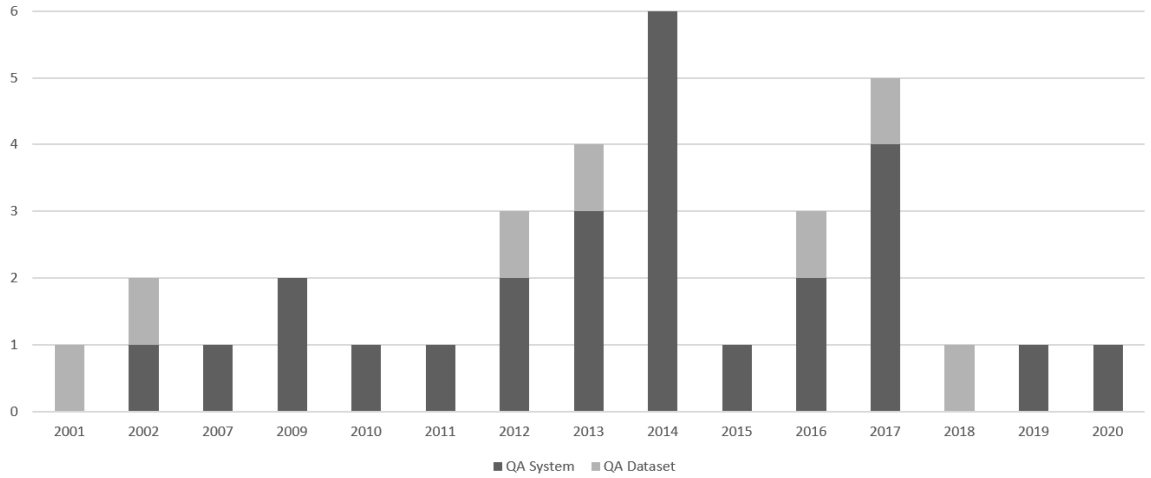

FIGURE 6. Distribution of all selected papers in accordance with publishing date.

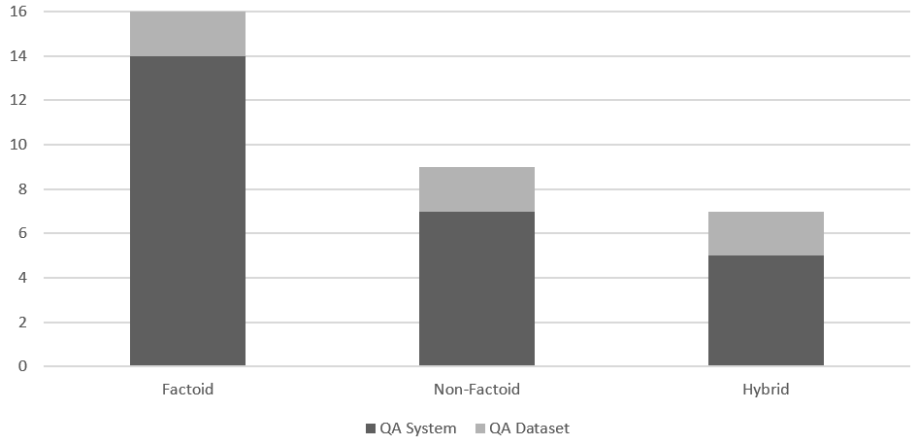

FIGURE 7. Distribution of all selected papers in accordance with question types.

languages" because all the surveyed papers are for Arabic. Table 3 shows the classification of the 26 presented papers. Figure 8 illustrates that 23 systems employ rule-based techniques to extract answers and only 3 systems adopt ML-based techniques for answer extraction.

V. ARABIC RESOURCE BUILDING ATTEMPTS FOR QA SYSTEMS

The first dataset consisted of Arabic questions and was proposed in TREC2001 [56] and TREC2002 [57]. The dataset consisted of 75 questions with no answers, and it addressed the problem of information retrieval rather than QA. Although the QA track was initiated in the Cross-Language Evaluation Forum (CLEF) in 2003, Arabic QA was not introduced into the forum until 2012, particularly in CLEF2012 [58] and CLEF2013 [59]. CLEF addresses multilingual information access research by establishing a conference each year, starting in 2000. CLEF2012 proposed a QA for a machine reading (QA4MRE) task. The main task was to develop an approach that, depending on given text documents, can answer multiple-choice questions. Each multiple-choice question has five choices, and the system chooses one answer among the five. The CLEF2012 conference provided a dataset for this purpose; the dataset became available in Arabic in 2012 for the first time after translation. The dataset was also available in other languages, such as Bulgarian, English, German, Italian, Romanian, and Spanish. The dataset comprised four topics (namely, Alzheimer, AIDS, music and society, and climate change), each having 19278, 8790, 10151, and 15725 documents, respectively. The types of questions involved were purpose, method, causal, which-is-true, and factoid. The factoid question types included location, number, person, list, time, and unknown. The total number of questions was 160. In CLEF2013, the number of questions was increased to 24.

The datasets created in the two conferences, TREC and CLEF, concentrated mainly on text retrieval and multiple-choice questions of different domains, respectively. The answers to the TREC dataset were long and contained no entities; thus, the questions could be considered nonfactoid questions. By contrast, the questions of the CLEF conference datasets consisted of factoid and nonfactoid questions; there-

TABLE 3. Classification of the surveyed research papers.

Time	Year	System	Domain Coverage	Question Type	Knowledge base	Linguistic Analysis Level	Document Retrieval	Answer Extraction
[30]	2002	QARAB	Open	Factoid	Unstructured	Syntax Morphological	Search technique	Rule-based
[31]	2007	ArabiQA	Open	Factoid	Unstructured	Morphological	Search technique	Rule-based
[32]	2009	QASAL	N/A	Factoid	N/A	N/A	Rule-based	Rule-based
[33]	2009	N/A	Open	Hybrid	Unstructured	Syntax	Search Engine	Rule-based
[34]	2010	DefArabicQA	Open	Non-Factoid	Unstructured	Syntax	Search Engine	Rule-based
[35]	2011	QArabPro	Open	Hybrid	Unstructured	Syntax Morphological Semantic	Search technique	Rule-based
[36]	2012	N/A	Closed	Hybrid	Unstructured	Syntax	N/A	Rule-based
[37]	2012	IDRAAQ	Closed	Hybrid	Unstructured	Semantic	Search technique	Rule-based
[38]	2013	ALQASIM	Closed	Hybrid	Unstructured	Syntax Morphological Semantic	Rule-based	Rule-based
[39]	2013	N/A	N/A	Non-Factoid	Unstructured	Syntax Morphological Semantic	Rule-based	Rule-based
[40]	2013	AQuASys	N/A	Factoid	N/A	Syntax Morphological Semantic	Rule-based	Rule-based
[41]	2014	N/A	Closed	Factoid	Unstructured	Syntax Morphological Semantic	Rule-based	Rule-based
[42]	2014	N/A	Open	Non-factoid	Unstructured	Syntax Morphological Semantic	Search Engine	Rule-based
[43]	2014	Al-Bayan	Closed	Factoid	Unstructured	Syntax Morphological	Search technique	Rule-based
[44]	2014	N/A	Open	Factoid	Unstructured	Syntax Morphological Semantic	Search Engine Search technique	Rule-based
[45]	2014	JAWEB	Open	Factoid	Unstructured	Syntax Morphological	Rule-based	Rule-based
[46]	2014	NArQAS	Open	Factoid	Unstructured	Syntax Morphological	Search Engine	Rule-based
[47]	2015	EWAQ	Closed	Non-Factoid	Unstructured	Syntax Morphological Semantic	Search Engine	Rule-based
[48]	2016	N/A	Open	Non-Factoid	Unstructured	Syntax Morphological Semantic	Search technique	Rule-based
[49]	2016	IQAS	Open	Factoid	Unstructured	Syntax	N/A	N/A
[50]	2017	Ontology based AQA	Closed	Non-Factoid	Structured	Syntax Morphological	N/A	Rule-based
[51]	2017	N/A	Open	Factoid	Unstructured	Syntax Morphological Semantic	Search engine Search technique	ML-based
[52]	2017	AlQuAnS	Open	Factoid	Unstructured	Syntax Morphological Semantic	Search engine	Rule-based
[53]	2017	NN-based QA	Open	Factoid	Structured	N/A	Rule-based	ML-based
[54]	2019	SOQAL	Open	Factoid	Unstructured	N/A	Search technique	ML-based
[55]	2020	ASHLK	Closed	Non-Factoid	Unstructured	Syntax Morphological Semantic	N/A	Rule-based

fore, they could be considered hybrid datasets. Another non-factoid dataset, namely, DAWQAS, was proposed by Ismail and Homsi [60]; the dataset was created semiautomatically by the following seven steps. The first step was to retrieve from the Web all titles that consist of the word "why" -- لماذا.. The second step was to clean and extract data; the

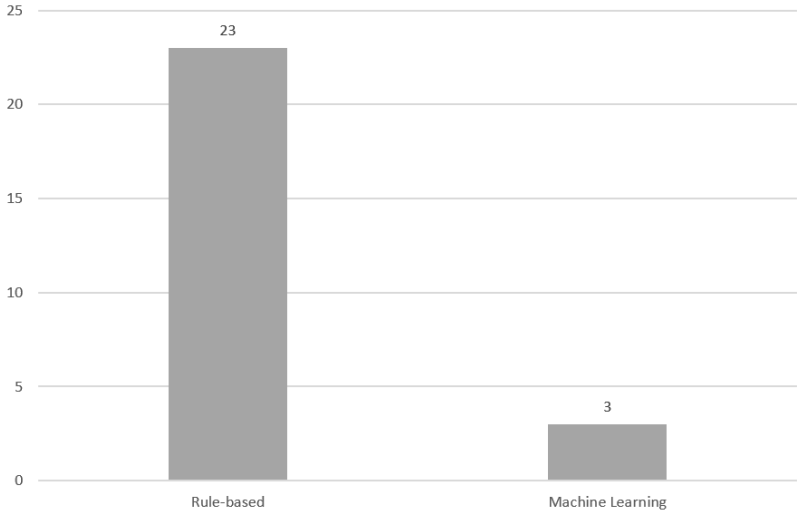

FIGURE 8. Number of arabic QA systems using either ML or Rule-based techniques for answer extraction.

collected data were cleaned of duplicated white spaces, newlines, JavaScript code, and cascading style sheet code. Regular expressions were applied to the cleaned data to extract questions; answers; and other attributes, such as the author of an article, published date, question description and category, and the frequency of discourse maker in each text that has the answer. The third step was to preprocess the text. The fourth step was to recategorize the answers by either merging the classes or replacing them. The fifth step was to compute the probability of the existence of rhetorical relations in the answer text for each sentence in the dataset, in which the answer was assigned a rhetorical relation tag with the highest probability. The sixth step was performed manually by Arabic native speakers to validate and determine the exact position of the answer. In the final step, the dataset files were generated.

Two factoid datasets were also proposed for Arabic QA. The first one was proposed by Bakari *et al.* [61]. The corpus consisted of a pair of factoid questions and their answers. The types of factoid questions considered were ''who'' - - من, ''where'' - - اين, ''what'' - هي/هو ما , ''when'' - - متى, and ''how'' (''much/many'') - كم. Building the corpus consisted of four main stages. In stage one, 250 questions were collected from targeted websites covering the categories of health and medicine, discoveries and culture, world news, history and Islam, and sports. In stage two, each question was analyzed by extracting the object of the question (focus), name entities (NEs) (to determine question type), and the terms that will be used to search for the answer (keywords). The question was then reformulated in a declarative form. In stage three, the corpus was built by following four substeps: (1) document search, in which Google was used as the search engine for the given question; (2) web page recovery, in which URLs were retrieved for the given question; (3) text preparation, in which each HTML page was cleaned and converted into a text document; and (4) text classification, in which text was classified manually in accordance with the object and topic of the question. In the final stage, linguistic analysis was performed, as proposed in [46].

The second factoid Arabic QA dataset, namely, TALAA-AFAQ, was proposed by Aouichat and Guessoum [62]. Building this corpus comprised five steps. In the first step, questions, along with their answers, were carefully collected considering grammatical correctness and that each question had only one correct answer. The sources of these question–answer pairs were QA4MRE@CLEF, a set of CLEF and TREC Arabic questions; the Web; and text. The second step was to classify the questions manually into four classes: location, name, quantity, and time. Each of these classes was further divided into fine classes. The third step was answer pattern extraction to provide patterns for QA systems for training. The fourth step was to vocalize questions by assigning them diacritical marks by using the Mishkal: Arabic Text Vocalization tool. The fifth step was feature extraction, in which syntactic and semantic features were extracted from the collected questions by using the AMIRA tool.

Tables 4-6 summarize the datasets discussed in this section; Table 4 lists the nonfactoid Arabic QA datasets, Table 5 lists the hybrid Arabic QA datasets, and Table 6 lists the factoid Arabic QA datasets. These tables compare the listed datasets in terms of domain coverage, size, question types, available annotation, source, and online availability. Creating a dataset is difficult because the dataset should be representative, balanced, and large. The largest nonfactoid Arabic dataset consists of 3205 QA pairs, and the largest factoid Arabic dataset consists of 2002 QA pairs. These sizes are still small compared with the dataset sizes for other languages. For example, WikiMovies [63] (a dataset of different classes of questions) and SimpleQuestions [64] (a dataset of factoid questions), which were created for the English language, consist of 108,442 and 100,000 QA pairs, respectively.

TABLE 4. Nonfactoid arabic QA datasets.

Ref.	Year	Dataset	Domain	Size	Question Types	Annotation	Source	Available online	Comments
[56] [57]	2001-2002	TREC	N/A	75 questions	N/A	N/A	Arabic newswire	Yes	• No answers available • Constructed for text retrieval purposes
[60]	2018	DAWQAS	• Sport • Politics and Economy • Arts and Celebration • Technology and Science • Religion and Philosophy • Nature and Animals • Society and Women • Health and Nutrition	3205 question-answer pairs	Why questions	• Identifies decision maker with the rhetorical relations probability • Question category	Limaza.com Arabic.rt.com Sayidaty.net Mawdoo3.com Ibelieveinsci.com Albayan.ae	Will be available soon	Created Semi-automatically

TABLE 5. Hybrid arabic QA datasets.

Ref.	Year	Dataset	Domain	Size	Question Types	Annotation	Source	Available online	Comments
[58]	2012	QA4MRE	• Alzheimer • Aids • Music and Society • Climate change	• 16 test documents, each topic have 4 documents. • 160 questions, each document have 10 questions. • 800 choices, each question have 5 choices	• Purpose • Method • Causal • Factoid • Which is true	N/A	Several English websites	Yes	• English text translated into Arabic • Multiple choice questions
[59]	2013	QA4MRE	• Alzheimer • Aids • Music and Society • Climate change	• 16 test documents, each topic have 4 documents. • 240 questions, each document have 15 questions. • 1200 choices, each question have 5 choices.	• Purpose • Method • Causal • Factoid • Which is true	N/A	Several English websites	Yes	• English text translated into Arabic • Multiple choice questions

TABLE 6. Factoid arabic QA datasets.

Ref.	Year	Dataset	Domain	Size	Annotation	Source	Available online
[61]	2016	AQA-WebCorp	• Health and medicine • Discoveries and culture • World news • History and Islam • Sport	250 questions	N/A	N/A	No
[62]	2017	TALAA-AFAQ	N/A	2002 Question-Answer pair	N/A	• QA4MRE@CLEF • A set of CLEFF and TREC Arabic questions • The web • Generated from text	Yes

Most of the Arabic QA datasets consist of QA pairs. However, most datasets in other languages created for QA systems after 2015 consist of the question, answer, and related paragraph, evidence, document, or article; additionally, these datasets have large-scale sizes. For instance, webQA [65], which was created for the Chinese language, consists of 42,187 QA pairs with 556k items of related evidence. Related documents or paragraphs can be used to select answers to multiple-choice questions by conducting rules or similarity checks. The existence of the related paragraph, evidence, document, or article in the dataset with the QA pair can also help in creating models for answer extraction or even answer prediction. The input of the model will be both the question and the related document or paragraph, and the model can be trained to output the answer exploiting the related document or paragraph [66].

VI. QA EVALUATION MEASURES

As stated earlier, QA systems comprise three main components: question analysis, document retrieval, and answer extraction. Each component should be evaluated separately, although most systems evaluate only the overall system performance. The QA systems in the literature were evaluated by using multiple measures, such as accuracy, recall, precision, the F-measure, mean average precision (MAP), the mean reciprocal rank (MRR), C@1, the confidence weight score (CWS) to evaluate the confidence of the QA system, the number of answered questions (AQ), and exact match (EM).

Accuracy is the fraction of correctly retrieved documents over the total number of documents. This measure can be used in the case of list questions because the answer of one query may be contained in multiple files. Other measures that can be used to evaluate list questions are recall and precision.

Recall is the fraction of relevant documents that are retrieved over the total number of relevant documents available for a given query. Precision is the fraction of relevant documents retrieved over the total number of documents retrieved for a given query. Recall and precision can be combined to create the weighted harmonic mean, named the F1-measure. The F1-measure is computed for the words of each predicted answer and each golden answer, in which both are treated as bags of words. Thus, the F1-measure is the average overlap between the words of the predicted and golden answers for a given question. Recall, precision, and the F1-measure can also be used to evaluate definition questions, given that such questions do not have a specific answer and depend on the user's satisfaction for the given answer [67]. Overall, accuracy, recall, precision, and the F1-measure can be used to evaluate systems that answer all question types. Formula 1 represents the F1-measure equation.

$$F1 = \frac{2 \times Precision \times Recall}{Precisio + Recall}, \tag{1}$$

MAP is used mainly for IR and ranking but is also used in QA research papers that consider the QA problem as a ranking problem. The retrieved multiple answers are ranked in accordance with their relevance. To compute MAP, the average precision (AP) is first calculated for each query, and then the APs are summed and divided by the total number of queries. MAP assumes that the user is concerned with finding many documents that are relevant. In contrast, the MRR assumes that the user's concern is to find only one relevant document for each query. To compute the MRR, the reciprocal rank (RR) is first computed for each query, and then the mean of the RR is calculated for all queries [68]. Formula 2 represents the MRR equation, where N is the total number of questions and Q_i is question i.

$$MRR = \frac{1}{N} * \sum_{Q_i} \begin{cases} \frac{1}{correct\ answer\ rank}, & if\ correct\ answer\ exists \\ 0, & if\ no\ correct answer exists, \end{cases} \quad (2)$$

The C@1 measure was first introduced in 2009. This measure is used for QA systems that assume one correct answer for each query, and the measure works by evaluating QA systems by giving them an option not to answer the question rather than forcing them to provide an incorrect answer. Therefore, the precision values increase, while the recall remains the same [69]. Formula 3 represents C@1, where q is the total number of questions, q_c is the number of correctly answered questions, and q_u is the number of unanswered questions.

$$c@1 = \frac{1}{q}(q_c + q_u \frac{q_c}{q}), \quad (3)$$

The CWS was introduced in 2012. This measure is used to evaluate systems in accordance with their correct answer rank by giving a high value for those QA systems whose correct answers are at the top of the rank. The following two formulas are for the CWS, where q is the number of questions, i is the position in the ranking, $c(i)$ is the number of correct answers until rank i, j is the answer to the question, and $I(j)$ is either 1 if j is correct or 0 if j is incorrect [69].

$$cws = \frac{1}{q} \sum_{i=1}^{q} \frac{c(i)}{i}, \quad (4)$$

$$c(i) = \sum_{j=1}^{i} I(j), \quad (5)$$

The AQ measure is used to calculate the number of questions that are correctly answered. The following formula represents the AQ measure, where N_s is the number of questions, k is a question that belongs to set s, j is the rank of a passage, and $V_{k,j}$ is a value assigned to the returned passages for question k; $V_{k,j}$ equals 1 if the answer is found in the passage or 0 if not.

$$AQ = \frac{1}{N_s} \sum_{k \in s} max(V_{k,j}), \quad (6)$$

EM [70] assigns 1.0 to the predicted answer that matches the golden answer for a given question and 0 otherwise. Except MAP and the CWS, most of the measures discussed in this section were used by the surveyed papers. MAP can be used to evaluate the second component (document retrieval) and the ranking of the answers in the final component (answer extraction).

VII. ARABIC QA SYSTEMS

Although the introduction of the first QA system can be traced back to 1972, the first QA system for the Arabic language was not proposed until 1993. The Arabic QA System (AQAS) [71] was the first QA system proposed for Arabic. This system is fully knowledge based and was developed by Mohammad *et al.* AQAS comprises (1) a parser that converts a question into a useful query by checking a dictionary and using a morphological step, (2) an interpreter that helps access the knowledge base to retrieve a suitable answer, and (3) a generator that accesses the interpreter to display the answer.

A. PROPOSED QA SYSTEMS FOR ARABIC FACTOID QUESTIONS

The second QA system for the Arabic language, namely, QARAB, was proposed in 2002 by Hammo *et al.* [30]; the system was tested in [72] and targets factoid questions. QARAB analyzes questions within the last component; thus, the system consists of only two modules rather than three as in typical QA systems. The two modules are (1) the information retrieval system (QIRS) and (2) the NLP system. QARAB concentrates on answering questions that include "who" - - من, "when" - - متى, "where" -اين, and "how" (specifically, "how much" or "how many") - - كم IWs. QIRS is based on Salton's VSM, which aims to build an inverted index from the dataset in use with Arabic text extracted from the newspaper Al-Raya, published in Qatar. The NLP system aims to provide a set of tools that can be applied to Arabic in the QA process. Details related to these tools can be found in [73]. The tools include tokenization, type finding or POS tagging, and feature and proper name identification. The features used in the authors' research are gender, number, person, and tense; the proper nouns are a person's name, location, organization, time, and date.

QARAB functions as follows. A question is processed by applying tokenization, POS tagging, and stop-word removal. The type and category of the answer are determined by checking the IW of the question. Questions are treated as bags of words, and the result of the processed questions is passed to QIRS to retrieve the relevant passages by checking the keyword in the processed questions and the inverted index. Finally, the answer is generated by parsing 10 relevant passages retrieved in the previous step to produce an answer of five short strings. Experiments were conducted on 113 Arabic questions by using two approaches in QIRS. Light stemming was used in the first experiment to build the QIRS word index, and QE, which is based on root stemmer to build the index,

was applied in the second experiment. The evaluation was performed manually by four native speakers by examining the answers to the 113 questions. The results showed that QE outperformed the light stemming approach with precision, recall, and the MRR equal to 100%, 79.6%, and 0.718, respectively, in the QE experiment and 97.3%, 97.3%, and 0.860, respectively, in the light stemming experiment.

Kanaan *et al.* [74] presented an Arabic QA system that uses the same method presented in [30] and similarly includes two modules, namely, the IR and NLP systems, and the same steps to process a question and produce the answer, in addition to using the same type of questions. The differences lie in the dataset, experiment, and evaluation. The dataset used by the authors consists of 25 documents collected from the Internet, and only 12 questions were provided by the authors themselves.

Benajiba *et al.* [31] proposed an Arabic QA system, named ArabiQA, to answer factoid questions. It consists of the three QA components. The first component analyzes the question to identify the answer type, extracts the question keywords, and performs NER. The second component is passage retrieval and is implemented using the Java Information Retrieval System (JIRS) after performing shallow stemming on the dataset. JIRS uses the density–distance model (DDM) in two steps to match n-grams between the query and passage. The first step assigns weights to passages that contain the query terms, and the second step gives more weight to similar passages by computing the similarity between the query and passage. The third component, namely, answer extraction, first performs NER on the candidate passages; then, preselection of related NEs and elimination of unrelated NEs are conducted. Finally, the answer is selected in accordance with a set of patterns.

Brini *et al.* [32] proposed the QA System for the Arabic Language (QASAL), which aims to answer factoid questions (''who,'' ''when,'' ''where'' and ''how much''). QASAL consists of the three components of QA systems. In the first component, questions are analyzed to identify the answer type, keywords, and question focus by using NER. In the second component, relevant passages are retrieved. In the last component, answers are displayed. This system is implemented using the NooJ linguistic engine. Brini *et al.* [33] updated QASAL to answer definition questions about persons and organizations (''who'' and ''what''). With this update, QASAL is considered the first QA system proposed to answer definition questions for the Arabic language. The update was implemented on the three components. In the first component, the authors added patterns to identify the question focus. In the second component, the authors used the Google Search engine and the Web as data sources. In the last component, the authors used lexical patterns to extract answers.

Bekhti and Al-Harbi [40] proposed an Arabic QA system named AQuASys (Arabic Question-Answering System), which also targets factoid questions. This system consists of three components. The first component, question analysis, aims to identify the answer type by applying a set of rules in accordance with IWs. The question is segmented, and its words are classified into three categories: IWs, keywords of the question, and the verbs of the question. Verbs are given considerable weight in scoring in the last component. The question is augmented with additional keywords in accordance with IWs. This step is equivalent to QE, but the expansion is attained in accordance with the type of IW. The second component differs from the typical document retrieval component of QA systems in that it filters relevant sentences rather than documents or passages by using the original and stemmed forms of the query and documents. The retrieval is performed using a string-matching approach. If at least one keyword or verb is found in the sentence, then the sentence will be retrieved and sent to the next component. For stemming, the authors used the Khoja stemmer. The last component assigns scores to the retrieved sentences and ranks them; the top-ranked sentence represents the accurate answer, and the remaining relevant sentences represent the candidate answers. The scoring is achieved using accuracy-scoring formulas developed by the authors, and the answers are ranked in accordance with their scores. Experiments were conducted using ANERcorp, ANERgazet, and 80 factoid questions related to location, organization, person, date, and number.

Kamal *et al.* [41] proposed an Arabic QA system that uses latent semantic indexing in the last component of the QA system to enhance the performance of answer selection of factoid questions. The proposed system consists of three components. The first component, question analysis, processes the question by applying tokenization; normalization; question classification; keyword extraction, keyword expansion, which gives the user choices to choose from (synonyms, root stemming, or both); and query formulation. The questions are classified in accordance with IWs, while root stemming is performed using the Khoja stemmer. The second component, candidate answer retrieval, consists of two modules. The first module retrieves the relevant passages or documents from an existing corpus, and the second module filters these passages or documents. The second module performs tokenization, normalization, NER, and filtering by using cosine similarity. The last component, answer ranking, ranks the filtered passages or documents by using latent semantic indexing and displays the top five candidate answers. Two experiments were conducted using a different corpus each time. The first corpus consisted of 3000 Fatwa passages collected from different websites, and the second corpus was ANERcorp. The first experiment was performed using 130 questions (25 ''who'' (من), 30 ''when'' (متى), 40 ''what'' (ما), and 40 ''where'' (اين)), and the second experiment comprised 120 questions.

Fareed *et al.* [75] proposed a design for a QA system to answer Arabic factoid questions and experimented with the system by using a small dataset having only 20 questions to conclude that the system produces enhanced results when stemming is used along with two-level QE. One-level QE is conducted by extracting four types of relations (synonyms, supertypes, subtypes, and a definition) from Arabic Word-

Net (AWN) for each word. For each of these extracted relations, another level of extraction is performed, therefore leading to two-level QE. The same authors conducted other experiments on their proposed QA system in [44] by using only one-level QE to avoid the considerable time consumed when applying two-level QE. In the first component, the question is classified in accordance with IWs, and then the question is processed by removing stop words, performing QE, and stemming. The second component retrieves five snippets that may contain the answer by using the Google Search engine, JIRS, and the Khoja stemmer. The last component is answer extraction, which extracts the answer from the returned snippets by searching the snippets for answer types. Experiments were conducted on 56 questions from CLEF and TREC, and the results showed that the proposed system performance indicated improved results by using the Khoja stemmer.

Abdelnasser *et al.* [43] proposed Al-Bayan, a Holy Quran-based QA system, which targets factoid questions. This system retrieves relevant verses from the Quran and then extracts the answer from them along with their interpretation (Tafseer). The question-processing component comprises two main substeps: question preprocessing and classification. Questions are preprocessed using MADA and given POS tags, while question classification is performed using a support vector machine (SVM). The second component of Al-Bayan is IR, which consists of a semantic interpreter constructed from converting text into vectors of Quranic concepts, in which each concept comprises a set of verses and their interpretation, and each entry in the vector is assigned weight by using term frequency–inverse document frequency (tf-idf). The dataset consists of 1217 concepts. IR retrieves relevant verses by applying cosine similarity. The last component is answer extraction and comprises NER and feature extraction. NER was conducted using the LingPipe tool by feeding annotated data. In feature extraction, several features are used to assign each relevant Quranic verse a correctness probability value. These features are the number of matched words between the question and answer, question type, is-a relationship, maximum count of NEs between the question and answer, and minimum distance between the question and answer. Finally, the candidate answers having the highest correctness probability value are returned. Experiments were conducted on the NER module, question classification module, and overall system performance on 59 questions evaluated manually by five Quranic experts.

Kurdi *et al.* [45] proposed an Arabic QA system, named JAWEB, which aims to answer factoid questions (''who,'' ''what,'' ''where,'' and ''when''). JAWEB consists of the three components of QA. The first component tokenizes the question and identifies the answer type, keywords, and additional keywords that represent synonyms generated in accordance with IWs, i.e., QE. Then, stemming is performed on the query. The second component aims to retrieve relevant passages by using pattern matching between the query and dataset. The last component, answer extraction, stems the keywords of the relevant documents. It then performs a similarity check, which determines the number of matching keywords between the stemmed query and relevant passages. The candidate passages are ranked accordingly. An experiment was conducted using a dataset containing 39660 words.

An Arabic QA system that supports recognizing textual entailment, named the New Arabic QA System (NArQAS), was proposed by Bakari *et al.* [46]. The implementation of this system was presented in [61], and additional information about the logical representation of questions was provided in [76]. This system consists of five main phases: text analysis, question analysis, logic representation, textual entailment recognition, and answer generation. However, the first, second, and third phases can be substeps of the first component of the traditional QA system. In text analysis, several substeps are applied. They include cleaning the text, segmenting it into sentences and words (tokenization), NER by using the ArNER tool, identification of syntactic trees by using the Stanford Parser, and POS tagging by using the AlKhalil parser. In question analysis, question type is identified, and keywords are extracted to reformulate the questions. Logic representation is performed to convert the questions into a set of logic predicates. After searching for relevant documents by using the Google Search engine, textual entailments were recognized between the retrieved text and the questions. Finally, the answer is generated by choosing the text with the highest score among the retrieved answers in which the scores are assigned to the answers that have a textual entailment between them and the questions. Two experiments were conducted on AQA-WebCorp, i.e., the first one on 115 questions and the second one on 250 questions.

Neji *et al.* [49] proposed the inference QA system (IQAS), which addresses answering questions about temporal information. IQAS comprises the three main QA components. In the question-processing phase, the question is first classified in accordance with the temporal information it contains and then analyzed by removing stop words and extracting NEs. The document-processing component consists of document and passage extraction. The last component is answer processing. However, the authors provided no information about the techniques used for each component and the experimental results for the overall system performance. Their research mentioned only the experimental results for question classification by using 100 temporal questions and 2000 temporal passages extracted from Wikipedia.

Ahmad *et al.* [51] proposed a Web-based QA system to answer Arabic factoid questions (person, location, organization, and numeric). The system consists of the three components of QA. The first component analyzes the question by removing diacritical marks, tokenization, stemming by using the Khoja stemmer, POS tagging, question focus identification, and question keyword extraction. Then, the question is classified using an SVM, and a two-layer taxonomy is followed. The first layer provides the course-grain class, and the second layer is the fine-grain class. SVM is given two types of features: lexical and syntactic. The lexical features

consist of the unigram, bigram, trigram, IWs, and word shapes of the question, and the syntactic features represent the headword of the question. The second component reformulates the query and passes it to the Google Search engine, which searches the Web and retrieves the most relevant documents. Then, the top 10 documents are searched using the VSM to extract relevant passages. A similarity check is performed between the query and retrieved passages to assign scores for the most similar passages. In the last component, answer extraction, NER and proper noun identification are performed for each candidate passage, and then the features are extracted to be passed to the SVM to rerank the candidate answers. The features used are question keyword, POS tag, headword, stem, question class, and question focus for the question. The length of words, position in the document, POS match, and focus word match are used for the candidate passages. The final answer is the one having the highest rank. Two experiments were conducted. The first experiment was to evaluate the question classifier, and the second experiment was to evaluate the answer ranker. The training set consisted of 1000 translated TREC questions, and the test set consisted of 434 translated TREC questions.

Nabil *et al.* [52] proposed AlQuAnS, an Arabic language QA system that aims to provide a short Arabic answer from the World Wide Web for four types of factoid questions: number (date), location (country), location (city), and human (individual). AlQuAnS comprises two main parts: online and offline parts. The online part starts with preprocessing the questions by using MADAMIRA, in which the text is normalized, stemmed, and given a POS tag, and stop words are removed. Then, the text is passed to the question analysis module, wherein the questions are further processed by using QE and query classification. AWN is used for QE, and the SVM classifier is used for classifying the questions. The query is then passed to the IR module (which consists of two submodules), online search engine (which uses Yahoo API), and passage retrieval (which exploits the semantic interpreter built in the offline part of the proposed system). The semantic interpreter aims to represent the questions and retrieved answers as vectors and then computes the cosine similarity between them. However, the semantic interpreter is built by preprocessing 11,000 Arabic Wikipedia documents by using MADAMIRA, and then a weighted inverted index is built using Lucene. The final module of the online part is answer extraction, which represents the answer-processing component of QA. This module consists of three main phases. The first phase is to construct a table of answer patterns for each question type; the table is built from the output of the IR module. The second phase aims to rank these answer patterns by calculating their precision. The final phase aims to first extract answer patterns and then filter them by using MADAMIRA NER. The final top five results are displayed.

Only two studies have experimented on neural networks (NNs) in answering factoid QA systems. One was conducted by Ahmed *et al.* [53], and the other was conducted by Mozannar *et al.* [54]. The main difference between them lies in the searched dataset; the former searched in a structured dataset, whereas the latter searched in an unstructured dataset. Ahmed *et al.* [53] proposed an Arabic QA system that consists of three components: question analyzer, knowledge retriever, and answer generator. In the first component, the question is represented as a vector by using a bidirectional gated recurrent unit (GRU). In the second component, a search is performed to retrieve relevant facts from a knowledge base. In the last component, the answer is produced in accordance with the retrieved facts and the short-term memory of the recurrent NN. Their system was experimented using a knowledge base created by the authors, and the results showed an accuracy of 53%.

Unlike typical QA systems, the Arabic QA system named SOQAL, proposed by Mozannar *et al.* [54], comprises only two components: document retrieval and a neural reading comprehension (NRC) model that represents the answer extraction component. The component question analysis is not used. This system is similar to the one proposed by Chen *et al.* [77] in that it also skips the question analysis component and uses only two components: document retrieval and a document reader that represents the answer extraction component. The document retrieval of SOQAL is based on n-grams in retrieving relevant documents or paragraphs. The NRC model is based on bidirectional encoder representations from transformers (BERT). The authors conducted experimentation by using two datasets. One was the Arabic Reading Comprehension Dataset, which consisted of 1395 Arabic QA pairs posed by crowd workers on 155 Arabic Wikipedia articles and 48,344 QA pairs posed on 231 articles translated using Google Translate on the Arabic language. These pairs were collected from the Stanford QA Dataset (SQuAD), which was originally created for the English language.

Table 7 illustrates the proposed factoid QA systems in terms of name, question-analysis component, document-retrieval component, answer-extraction component, preprocessing techniques applied to the question and answer, domain coverage, dataset source, searched dataset size, number of questions, answer type experimented, used tools, and results.

B. PROPOSED QA SYSTEMS FOR ARABIC HYBRID QUESTIONS

Akour *et al.* [35] proposed QArabPro, a rule-based QA system for reading Arabic comprehension tests that deals with ''why'' (لماذا) and ''how much/many'' (كم) questions along with factoid questions: ''where'' (اين), ''when'' (متى), ''what/which'' (ما), and ''who'' (من). Their system consists of three main components: question analysis, document/passage retrieval, and answer extraction. In the first component, a question is classified by IW into its type, and tokenization, stemming, POS tagging, and NER are applied. Then, QE is performed on the query terms in the question by checking a small dictionary of synonyms. Subsequently, these query terms are passed to the second component to generate the

TABLE 7. Proposed systems for factoid arabic questions.

Ref.	Year	System	Question analysis			Document retrieval	Answer extraction	Preprocessing of Q and A	Domain Coverage	Dataset Source	Searched Dataset Size	Num of Q	Answer Type	Used tools	Result
			Query Expansion	Query Formulation	Other										
[30]	2002	QARAB	Root Stemmer	N/A	Identify answer type according to IW	• Passage Retrieval • VSM	• Display 5 answers • Passages having more keyword matches	• Tokenization • POS tagging • Stop-words removal	News	Al-Raya newspaper	N/A	113 question	• Person • Date, Time • Location • Number, Quantity	N/A	R:97.3 P:97.3 MRR: 86
[31]	2007	ArabiQA	N/A	N/A	N/A	• Passage Retrieval • DDM	Display one answer according to NER and a set of patterns	• Stemming • NER	Open	Wikipedia	11000 Arabic Wikipedia documents	200 questions	• Person • Location • Organization • Misc	JIRS	P: 83.3
[32]	2009	QASAL	N/A	N/A	Identify answer type and question focus	Passage Retrieval according to regular expression	Display more than one answer according to a set of patterns	• POS tag • Stemming	N/A	N/A	N/A	N/A	• Person • Temporal expressions • Location or Organization • Numeric expressions	NooJ linguistic engine	N/A
[40]	2013	AQuASys	QE according to IW	N/A	Identify answer type according to IW and a set of patterns	• Sentence retrieval • String matching approach • Ranking	• Displays more than one sentence as answers • Ranking	Stemming	N/A	ANERcorp	150,000 tagged tokens	80 questions	• Person • Time • Location • Object • Numeric expressions	Khoja stemmer	R: 97.5 P: 66.25 F1: 78.89
[41]	2014	N/A	• Using Arabic WordNet • Using root stemming	One query	Identify answer type according to IW	• Passage retrieval • String matching • Ranking	• Display 5 answers • Ranking	• Tokenization • Normalization • Punctuation removal • Diacritics removal • NER	• Religious • And other	Fatwa websites	• 3,000 passages from Islamic websites • 11,000 documents of ANERcorp	• 130 question for Fatwa • 120 question for ANERcorp	• Person • Time • Location • Other	Khoja stemmer	R: 96.6
[43]	2014	Al-Bayan	N/A	One query	Identify answer type using SVM	• Passage retrieval • VSM	• Display more than one answer • Ranking	• POS tagging • NER	Quran and Tafseer	Quran 2 Tafseer books	N/A	59 questions	• Creation • Entity • Physical • Location • Number • Description	• MADA • LingPipe	A: 85.4
[44]	2014	N/A	Using Arabic WordNet	Formulates several queries	Identify answer type According to the IW	• Document retrieval • Google search engine • Passage retrieval • DDM	• Display more than one answer • ranking	• Stop-word removal • Stemming	Open	N/A	Web	56 questions	• Person • Location • Time • Quantity • Thing	• Khoja stemmer • JIRS	A: 32.24 MRR: 20.14 AQ: 68.62
[45]	2014	JAWEB	According to IW	N/A	Identify answer type According to the IW	• Passage retrieval • String matching approach	• Display more than one answer • Passages having more keyword matches	• Tokenization • Stemming • Stop-words removal	Open	N/A	39,660 words	N/A	• Person • Location • Time • Object • Quantity	Khoja stemmer	R:100 P:80
[46]	2014	NArQAS	N/A	N/A	N/A	• Document retrieval • Google search engine	• Display more than one answer • Ranking according to the existence of TE	• Tokenization • NER • POS tagging	Open	Question from AQA-WebCorp	Web	250 question	• Person • Location • Organization	• Alkhalil parser • Stanford parser • ArNER tool	A: 89 C@1: 89
[49]	2016	IQAS	N/A	N/A	N/A	• Document retrieval • Passage retrieval	N/A	• Stop-word removal • NER	Open	Wikipedia	2000 passages	100 temporal questions	• Time-related • Event-related • Temporal-order • Entity-related	N/A	N/A
[51]	2017	N/A	N/A	One query	Identify answer type using SVM	• Document retrieval • Google search engine • VSM	SVM is used to select one answer	• Tokenization • Stop-word removal • Diacritics removal • POS tagging • NER • Stemming	Open	Questions translated from TREC	Web	434	• Person • Location • Organization • Numeric	Khoja stemmer	MRR: 57.7
[52]	2017	AlQuAnS	Using Arabic WordNet	N/A	SVM to classify questions	• Document retrieval • Yahoo search engine	Displays 5 answers according to NER and a set of patterns	• Normalization • Stemming • POS tagging • Stop-words removal • NER	Open	CLEF TREC	Wikipedia	200 questions	• Date • Country • City • Human	• MADAMIRA • Lucene	A: 22.20 MRR: 8.16 AQ: 47.66
[53]	2017	NN-based QA	N/A	N/A	Question is transformed into vector using bidirectional RNN and GRU	Knowledge retrieval	Produce answer using RNN	N/A	Open	Wikipedia	Knowledge-base	200 questions used for testing	N/A	N/A	A:53%
[54]	2019	SOQAL	N/A	N/A	N/A	• Tokenize • Stemming • Stop-word removal • Document retrieval • n-gram based retrieval	• Bert-based • Answer ranking	N/A	Open	Wikipedia	664768 Wikipedia articles	• 1,395 questions on 155 Wikipedia articles • 48,344 questions on 231 articles translated from SQuAD	N/A	N/A	F1: 42.5 EM: 20.7

query and retrieve the candidate passage. The IR system in the second component is constructed on the basis of Salton's VSM. Documents are processed by tokenization, stemming, POS tagging, stop-word removal, and term weighting by using an RDBMS. Finally, candidate passages are passed to the answer extraction component, wherein the answer is selected from multiple candidate passages after applying a set of rules on each of them. Each question type has a set of pre-assigned rules and weighting criteria that are applied accordingly. An experiment was conducted on a dataset collected

from Wikipedia with 75 reading comprehensive tests and 335 questions.

Trigui *et al.* [36] participated in CLEF2012 and proposed a QA system that aims to answer multiple-choice questions in accordance with given documents. Their system consists of four components. The first component is question analysis, which removes stop words. The second component searches for relevant passages, and the third component aligns these passages with the choices of the question. The last component is the answer selection, which chooses the correct choice for an answer depending on its existence in the retrieved passage. If the passage does not contain an answer, then inference rules are applied to reach an answer. When no answer is determined, the question is left unanswered.

Abouenour *et al.* [37] proposed a QA system as a part of their participation in CLEF2012. The system is named Information and Data Reasoning for Answering Arabic Questions (IDRAAQ). It aims to answer multiple-choice questions in accordance with given documents. IDRAAQ comprises the three QA components. The first component, question analysis and classification, aims to extract keywords and identify the answer type. The second component, passage retrieval, differs from the traditional document-retrieval component of QA systems in that it consists of a level that aims to apply QE on the keywords extracted from the original question. Two types of QE are applied. They are inflectional using the AlKhalil system and resource-based using AWN to extract synsets having relations, such as synonyms, hypernyms, hyponyms, and SUMO-AWN, with the keywords. The second level in the second component is responsible for filtering and reranking relevant passages and is implemented using JIRS. Reranking is performed using distance n-gram density. The last component, answer validation, aims to choose the correct answer in accordance with the retrieved candidate answers.

Ezzeldin *et al.* [38] presented an Arabic QA system as part of their participation in CLEF2013. It was named Arabic Language QA Selection in Machines (ALQASIM), which aims to answer multiple-choice questions in accordance with given documents. ALQASIM differs from traditional QA systems because it mimics humans who start by reading text to answer questions. ALQASIM consists of three components: preparing the dataset, analyzing the question-and-answer choices, and selecting the answer. The first component is document analysis, which aims to analyze documents by using stemming, POS tagging, and stop-word removal and by building an inverted index from the remaining stems. Each stem in the inverted index is expanded using AWN and given a weight in accordance with its POS tag and frequency. The second component aims at locating the questions and answer choices. This component differs from the traditional question-analysis component of QA systems in that it analyzes not only the questions but also the answer choices for each question and then searches for three snippets for each question in the inverted index. The analysis includes stemming and stop-word removal, and the search is conducted by calculating scores and retrieving the top three ones in accordance with the number of similar keywords, their weight, and the distance between them. The last component is answer selection, which aims to choose the correct choice among the given options by summing the scores for a question and each of its answer choices, computing the distance between the question and each of its answers, and subtracting the distance from the summation. The highest score is chosen as a correct answer. If two best choices exist, the question is left unanswered.

Table 8 illustrates the proposed hybrid QA systems in terms of name, question-analysis component, document-retrieval component, answer-extraction component, preprocessing techniques applied to the question and answer, domain coverage, dataset source, dataset size, number of questions, question type, tools used, and results.

C. PROPOSED QA SYSTEMS FOR ARABIC NONFACTOID QUESTIONS

The first attempt to handle ''why'' (لماذا) and ''how to'' (كيف) questions in Arabic was in 2010 by [78] and [79]. Their research focused mainly on eight rhetorical relations: causal, evidence, explanation, purpose, interpretation, base, result, and antithesis. These relations were hypothesized among small units of text by dividing the text into small units on the basis of the existence of cue phrases and punctuation marks. An experiment was conducted on selected text of 150–350 words each, given to 15 people who were asked to extract ''why'' and ''how to'' questions with their answers. A total of 98 questions with their answers were assigned by them, and these questions were tested on their system. The results showed that 54 of the questions were correctly answered. An updated version of this approach, implemented using long texts, was proposed by Sadek and Mezaiane [80]; the authors presented an Arabic text parser employed using a QA approach to answer ''why'' (لماذا) and ''how to'' (كيف) Arabic questions; the authors focused on 10 rhetorical relations: result, interpretation, sequence, elaboration, contrast, background, reason, evaluation, certainty, and view. The text parser has two main components: a relation recognizer and tree builder. Therefore, the text parser accepts sentences tagged with intrasentential relations as an input. This input, along with the question, is tokenized, normalized, and stemmed, and the stop words are removed. Then, the relation recognizer produces the hypothesized relation between adjacent sentences and discovers long-distance relations to produce relations. The VSM is used to assign the appropriate relation as an answer by computing similarities between the question and possible relations. The answers are ranked in accordance with the similarity value. An experiment was conducted on text selected from a contemporary Arabic corpus with 870–2138 words each. The text was given to five native speakers to extract ''why'' and ''how to'' questions and their answers. A total of 90 questions and their answers were collected. These questions were tested on their system, and 61 questions were correctly answered with a 0.62 MRR.

TABLE 8. Proposed QA systems for arabic hybrid questions.

Ref.	Year	System	Question analysis		Document retrieval	Answer extraction	Preprocessing of Q and A	Domain Coverage	Dataset Source	Searches Dataset Size	Num of Q	Q-Type	Used tools	Result
			Query Expansion	Question Classification										
[33]	2009	N/A	N/A	N/A	•Document retrieval •Google search engine	Display more than answers according to lexical patterns	•POS tag •Stemming	Open	The web	N/A	43 definition questions	•Factoid •Definition: person and organization	NooJ linguistic engine	R:100 P:94
[35]	2011	QArabPro	Using a small dictionary of synonyms	According to the IW	•Document retrieval •VSM	Answer is extracted after applying a set of rules and weighting criteria	•Tokenization •Root extraction •POS tagging •NER	Open	Wikipedia	N/A	•75 reading comprehensive tests •335 questions	•Factoid •What/ •Which •Why	N/A	R:86 P: 93 F1: 89
[36]	2012	N/A	N/A	N/A	Passage retrieval	Display one answer according to inference rules	Stop-words removal	•Alzheimer •Aids •Music and Society •Climate change	QA4MRE	N/A	•16 test documents, each topic have 4 documents. •160 questions, each document have 10 questions. •800 choices, each question have 5 choices	•Purpose •Method •Causal •Factoid •Which is true	N/A	A: 19 C@1: 19
[37]	2012	IDRAAQ	N/A	N/A	•Inflectional QE and QE on using AWN •Passage retrieval •Using DDM	Display one answer	Stop-word removal	•Alzheimer •Aids •Music and Society •Climate change	QA4MRE	N/A	•16 test documents, each topic have 4 documents. •160 questions, each document have 10 questions. •800 choices, each question have 5 choices	•Purpose •Method •Causal •Factoid •Which is true	N/A	A: 13 C@1: 21
[38]	2013	ALQASIM	N/A	N/A	•QE using AWN •Number of similar keywords, their weight and the distance between them •Document retrieval	•Display one answer •Score of summing the location score and subtracting it from the distance value	•Stemming •POS tagging •Stop-word removal	•Alzheimer •Aids •Music and Society •Climate change	QA4MRE	N/A	•16 test documents, each topic have 4 documents. •240 questions, each document have 15 questions. •1200 choices, each question have 5 choices.	•Purpose •Method •Causal •Factoid •Which is true	N/A	A: 31 C@1: 36

Another attempt to answer Arabic ''why'' questions based on rhetorical structure theory (RST) relations was proposed by Azmi and AlShenaifi [42]. The system has four components: question analysis, document preprocessing, document/passage retrieval, and answer extraction. The first two components share the same NLP techniques for processing the input and dataset: tokenization, normalization, stop-word removal, root extraction by using the Khoja stemmer [81], query formulation and generation, and QE by checking AWN. In the third component, the Lemur[6] module is used to retrieve a list of candidate passages from the given dataset. In the fourth component, the given question and candidate passages are used as a ''bag of words,'' in addition to RST relations. In this research, the authors focused on five rhetorical relations: base - قاعدة, interpretation - - تفسير, explanation - - تفضيل, justification - - تعليل, and result - - نتيجة. These relations are extracted on the basis of cue phrases that exist in the text. Each of the relations is given different priorities to ease the selection of the relevant answer among the retrieved passages. The justification relation is given the highest priority, the base relation is given the lowest priority, and the rest of the relations are given medium priority. For example, if two candidate answers with two different relations exist, then the passage having a relation with higher priority is selected as an answer; if both passages have the same relation or relations having the same priority, then the answer is selected randomly from the two passages. An experiment was conducted on a dataset of 700 documents selected from Open-Source Arabic Corpora (OSAC), and 100 questions with their answers were formulated by a native speaker. These questions were tested on the proposed system, and 71 questions were answered correctly, with recall and precision equal to 71% and 78%, respectively. A comparison study was conducted between this system and a baseline approach, which is a general method used to answer all types of questions, in [82]. Both tested methods have the same four components, but the difference lies in the third component, which was tested first by using a baseline method and then by using the RST-based method proposed in [42]. The baseline method used three criteria, namely, keyword frequency, expanded keyword frequency, and tf-idf, to rank the candidate answer. These criteria were normalized by dividing their values by the maximum score and combined for each answer to compute the global score for that answer. The answer having the highest global score was selected as an answer for the given question. The same baseline experiment conducted in [42] was applied to compare the baseline and RST-based methods. The results showed that the RST-based method outperformed the baseline method. The same authors [83] experimented with their RST-based method by using 110 ''why'' questions on the same dataset. The test was to evaluate the performance of the RST-based method when one of the NLP techniques of the first and second components was skipped each time. The best results were achieved when the stop words were not removed.

Table 9 demonstrates a comparison in terms of rhetorical relations, QE, stop-word removal, answer selection criteria,

[6]The Lemur Project, available at: http://www.lemurproject.org

TABLE 9. Comparison of rhetorical based papers.

Ref.	Year	Rhetorical relations	Query expansion method	Stop words removed	Answer selecting criteria	Dataset size	Dataset source	Question types	Implementation Language	Results
[78]	2010	Base – قاعدة Interpretation - تفسير Explanation - تفضيل Result - نتيجة Casual - سببية Evidence - اثبات Purpose - غاية Antithesis – استدراك	Stemming	Yes	Answers are ranked according to similarity values between the question and the candidate answers	98 questions Selected text of 150-350 words each	Arabic news websites	Why How to	JAVA	R: 55%
[42]	2014	Base - قاعدة Interpretation - تفسير Explanation - تفضيل Result - نتيجة Justification - تعليل	Stemming AWN	Yes	According to the rank of the document and the priority of the rhetorical relation	100 Questions 700 documents	OSAC	Why	JAVA	A: 71% R: 71% P: 78% C@1: 77.4%
[80]	2016	Interpretation - تفسير Result - نتيجة Reason - سبب Elaboration - تفصيل Background Contrast Evaluation Certainty Sequence View	Stemming	Yes	Answers are ranked according to similarity values between the question and the candidate relation	90 questions Selected text of 870-2138 words each	Contemporary Arabic corpus	Why How to	JAVA	R: 68% MRR: 0.62
[83]	2017	Base - قاعدة Interpretation - تفسير Explanation - تفضيل Result - نتيجة Justification – تعليل	Stemming AWN	No	According to the rank of the document and the priority of the rhetorical relation	110 Questions 700 documents	OSAC	Why	JAVA	R: 72.7% P: 79.2% C@1: 78.7%

dataset size, dataset source, question types, implementation language, and results among the papers that attempt to answer ''why'' questions.

Another example of an Arabic QA system created to answer ''why'' questions but with answers based on entailment relations was proposed by AL-Khawaldeh [47]; it was named the Entailment-based Why Arabic QA (EWAQ) system. The system consists of the three components of QA. The first component analyzes the question by removing stop words, stemming, and applying QE for each word constituting the question by using AWN. The second component retrieves relevant passages by using a search engine, and the last component reranks them in accordance with their entailment relation and by applying a cosine similarity measure. In each of the top five candidate passages, passages consisting of multiple sentences are split into single sentences. The same reranking step is applied in accordance with the entailment relation and by applying similarity measures. The sentence having the highest score is the answer. Experiments were conducted using 250 questions distributed equally among computers, religion, science, politics, and history.

Ahmad and P [48] proposed an Arabic QA system dedicated to answering ''why'' (لماذا) and ''how'' (كيف) questions. It consists of four components, namely, question analysis, QE, passage retrieval, and answer extraction, of which question analysis and expansion represent the steps in the first component of traditional QA systems. The proposed system starts in its first component to classify questions by IW. Then, the questions are tokenized, and stop words are removed. Subsequently, NEs and noun phrases are recognized by using a chunker. Afterward, the question focus that may represent a word or the noun phrase in the questions is identified. In the second component, AWN is exploited to extract synonyms for the verbs and adjectives of the questions for QE. The second component is implemented using a tf-idf VSM to retrieve relevant documents. In the last component, some patterns are added to the question keywords in accordance with the question type. For the question-type reason, the added patterns are ''because'' (لان، لانه) and ''due to'' (لذلك، لهذا، بسبب); for the question-type manner, the added patterns are ''by'' (بواسطة، عن طريق) and ''using'' (باستخدام). The retrieved top documents are divided into passages, and a check for question focus is performed, giving 1 as a weight for each passage containing the question focus and 0 for those that do not contain the question focus. Cosine similarity is computed between the query and each sentence to compute the total similarity of each passage by summing the cosine similarities of constituting sentences and the weight for each passage. The passage having the highest total is considered the answer for the question. Experiments were conducted on 500 documents from Arabic Wikipedia and 80 questions containing 40 ''how'' questions and 40 ''why'' questions.

The second attempt to create an Arabic QA system for answering definition questions was accomplished by Trigu *et al.* [34]; they proposed DefArabicQA, a QA system that answers Arabic definition questions about a person or an organization (i.e., ''who is'' (من هو، من هي) and ''what is'' (ماهو ، ما هي)). The system consists of four components: question analysis, passage retrieval, definition extraction, and definition ranking. In traditional QA systems, definition extraction and ranking can represent a component, namely, answer extraction. The question analysis component identifies the question topic and answer type. The answer type is identified by IW; question topic, which constitutes the question query, is used by the Web search engine in the passage retrieval component to retrieve relative snippets. The definition extraction component consists of substeps. The first substep identifies and extracts definitions from the retrieved snippets by checking lexical patterns that surround the question topic of the snippets. In the second substep, candidate snippets are filtered using heuristic rules. The definition-ranking component calculates a global score for each candidate answer after computing three scores for each of the pattern weights, snippet positions, and word frequencies. Two experiments were conducted on DefArabicQA by using 50 organization questions, and the results were evaluated manually by an Arabic native speaker. The system yielded five answers. The first experiment was conducted using the Google Search engine as the Web resource, and the second experiment included the Google Search engine and Arabic Wikipedia as the Web resources. The second experiment outperformed the first one.

The first and only attempt to create an Arabic QA system for answering ''yes/no'' questions was proposed by Bdour and Gharaibeh [39]. It comprises the three components of QA. The first component performs tokenization, stop-word removal, POS tagging, QE, and logical representation of the resulting query. QE is performed to augment the query with the synonyms and antonyms of verbs for the verbal phrases and of the predicate for the nominal phrases. The second component aims to retrieve relevant passages by using an indexing scheme. The last component splits relevant paragraphs into sentences and eliminates each sentence that does not contain the noun (المبتدا) for the nominal query or the subject (الفاعل) for the verbal query. For the resulting sentences, the component determines the existence of other keywords in the sentences, ranks them in accordance with their indexes in the sentences, checks for the negation particles in the candidate sentence, compares the query with the candidate sentence, and answers the question with ''yes/no'' accordingly. Two experiments were conducted using 20 Arabic documents and 100 ''yes/no'' questions. A document retrieval technique was implemented in the first experiment, whereas a passage retrieval technique was used in the second experiment. By using the document retrieval technique, five relevant documents were retrieved and split into paragraphs, and then the top five relevant passages were retrieved. By using the passage retrieval technique, the 20 documents were split into paragraphs, and the top five relevant passages were retrieved. The results showed that the passage retrieval technique outperformed document retrieval.

Other nonfactoid Arabic QA systems have been created on the basis of ontologies. For instance, Albarghothi *et al.* [50] introduced a system that consists of three main components. The first component is question processing, in which the question is normalized, tokenized, and tagged with POS, and stop words are removed. The second component is ontology mapping, which differs from the regular component of QA systems, i.e., document retrieval. In document retrieval, relevant documents are retrieved; in the new component, the system maps the ontology with the question keywords to retrieve the answer by using SPARQL. The last component is answer processing, which uses the SPARQL query language to retrieve the final answer or answers. Experiments were conducted on 100 questions, and an ontology was built by the authors by using the Protégé tool. The ontology contained 200 instances and 1260 triples of subject, predicate, and object in the pathology domain. The QA system was evaluated and updated 5 times until the results were the best at the fifth experiment.

Abdi *et al.* [55] proposed the QA system in Al-Hadith using linguistic knowledge (ASHLK). ASHLK comprises three components that are different from the usual QA system components: preprocessing of Hadith text and user input, using a graph-based ranking model, and answer generation. The preprocessing is performed by first applying sentence segmentation where the boundaries of sentences are indicated (!, ?, or .) and paragraphs consist of several sentences and end with a newline. The next step in preprocessing was cleaning, in which stop words, punctuations, diacritics, and Sanad were removed. Then, stemming is performed. The final step in preprocessing is QE by using the Dice similarity measure. In the second component, using a graph-based raking model, the objectives are to retrieve candidate answers among the available sentences and to rank them. Answer retrieval is achieved through a comprehensive similarity step. The final component, answer generation, aims to display answers by comparing the candidate answers and choosing those less similar to others. The last two components are equivalent to the final component of the usual QA systems, answer extraction. Experimentation was conducted on a dataset taken from Sahih al-Bukhari.

Table 10 illustrates the proposed nonfactoid QA systems in terms of name, question-analysis component, document-retrieval component, answer-extraction component, preprocessing techniques applied to the question and answer, domain coverage, dataset source, dataset size, number of questions, question type, tools used, and results.

D. OVERVIEW OF APPROACHES

This section highlights the current state of research in Arabic QA systems. Most of the presented Arabic QA systems experimented with open-domain factoid questions with answers extracted from unstructured text. Only two systems

TABLE 10. Proposed QA systems for arabic nonfactoid questions.

Ref.	Year	System	Question analysis		Document retrieval	Answer extraction	Preprocessing of Q and A	Domain Coverage	Dataset Source	Searched Dataset Size	Num of Q	Q-Type	Used tools	Result
			Query Expansion	Question Classification										
[34]	2010	DefArabicQA	N/A	According to the IW	• Passage retrieval • Google search engine	• Checking lexical patterns • Using heuristic rules • Display 5 answers	N/A	Open	The web and Wikipedia using search engine	N/A	50 question	Definition: person and organization	N/A	MRR: 81 AQ: 64
[39]	2013	N/A	Augment the query with synonyms and antonyms	N/A	• Passage retrieval • Similarity	• Display one answer • Rank and display one answer	• Tokenization • Punctuation removal • Stop-word removal • POS tagging	N/A	N/A	20 documents	100 questions	Yes/No questions	N/A	A: 85
[42]	2014	N/A	AWN	N/A	Passage retrieval	• Display one passage as an answer	• Tokenization • Normalization • Stop-word removal • Root extraction	Open	OSAC	700 documents	100 questions	Why questions	• Khoja stemmer • Lemur module	A:71 R: 71 P:78 C@1: 77.4
[47]	2015	EWAQ	AWN	N/A	• Passage retrieval • Different search engines	• Display five answers • Re-ranking	• Stemming • Stop-word removal	• Computer • Religion • Science • Politic • History	N/A	N/A	250 questions	Why	N/A	A: 68.53
[48]	2016	N/A	AWN	According to the IW	VSM	• Display one answer • Cosine similarity	• Tokenization • Stop-word removal • NER	Open	Wikipedia	500 documents	80 questions	Why and how	N/A	R: 57 P: 64 F1: 60
[50]	2017	Ontology based AQA	N/A	N/A	N/A	• Display more than one answer • SPARQL	• Normalization • Stemming • POS tagging • Stop-words removal	Pathology	N/A	N/A	100 questions	• What are the main symptoms of …? • What causes the disease? • What are the diseases which infect …?	Protégé tool	R: 93 P: 81 F1: 86
[55]	2020	ASHLK	Dice Similarity measure	N/A	N/A	• Different similarity measures • Ranking	• Stop-words removal • Punctuation removal • Diacritics removal • Sanad removal • Stemming	Hadith	Sahih al-Bukhari	4000 Hadiths	3825 questions	Wh-questions	N/A	R: 63.87 P: 83.47 F1: 72.37

experimented with extracting answers from a structured knowledge base.

Document retrieval approaches can be categorized into three categories: rule-based approaches, search techniques, and search engines. The main rule-based approach adapted by the Arabic QA systems was the string-matching algorithm. The main search technique used by the Arabic QA systems was the VSM. However, the search engines used by the presented systems were the Google and Yahoo search engines.

The approaches of answer extraction are categorized into rule-based and ML-based approaches, and most of the Arabic QA systems present adapted rule-based answer extraction methods, such as (1) applying a set of patterns or rules to select or extract the answer; and (2) ranking paragraphs, sentences or NEs that are considered answers. Only 3 of the 26 Arabic QA systems adapted ML-based approaches, such as SVM and NN, for answer extraction. Since 2015, the focus of answer extraction in other languages has become building models that are based on NNs, such as BIDAF [83] and FastQA [84]. These models can be used in the answer extraction component of QA systems.

E. SYSTEMS AND TOOLS

Different tools were used by the Arabic QA systems presented in this paper. Some of these tools, such as the JIRS, Lucene, and Lemur modules, were used specifically in the document retrieval component. However, these tools are language independent and do not consider specific language characteristics.

Other language-independent tools used in the presented papers for text processing were the LingPipe, Protégé tool, and NooJ linguistic engine. LingPipe is a multilingual tool kit that is used for finding name entities, such as people names, location, or organization. The Protégé tool [83] is a domain-independent open-source tool used in creating ontologies and managing terminologies [84]. The NooJ linguistic engine [85] allows users to process datasets with many texts in real time. NooJ contains dictionaries for different languages, such as Arabic, English, Armenian, French, Chinese, Spanish, Danish, Hungarian, and Italian.

The remaining tools used in the presented papers are Arabic language dependent, such as the Khoja stemmer, the AlKhalil parser, the ArNER tool, MADA, MADAMIRA, and the Stanford parser. The Khoja stemmer [86] is a freely available tool for stemming. The AlKhalil parser [87]yy, [88] is an open-source morphological parser that can parse diacritized, undiacritized, or even partially diacritized text. The ArNER tool is Arabic NER, but no information is available about this tool. MADAMIRA, the new version of MADA, performs different linguistic tasks, such as tokenization, morphological disambiguation, POS tagging, NER, phrase chunking, and lemmatization. The Stanford parser is also open source and can be used to parse sentences in various languages, including English, Chinese, Bulgarian, Italian, Portuguese, and German. The Arabic language is based on the Penn Arabic Treebank. Stanford also has other open-source packages, such as the Stanford POS tagger.

VIII. DISCUSSION AND MAIN FINDINGS

Before 2015, the research in Arabic QA was focused mainly on factoid questions, and after 2015, the focus was still on the factoid question. More research should be focused on nonfactoid questions. The second observation is that before 2015,

the document retrieval component was focused on using rule-based techniques, but after 2015, the focus was changed to using either search engines (such as the Google API) or search techniques (such as the VSM) because most of the research papers were based on either Web search or document search to retrieve relevant documents. Another observation is that before 2015, the answer extraction was based mainly on rule-based techniques; however, after 2015, there were only three attempts to use ML-based answer extraction techniques. Next is a detailed discussion on each QA dataset presented and QA system for the Arabic language.

The number of proposed QA datasets for the Arabic language reached six; two of these datasets were created for nonfactoid questions, two for factoid questions, and the remaining two for hybrid questions, which are a combination of factoid and nonfactoid questions. With reference to Tables 4-6, the largest dataset is DAWQAS, which comprises 3205 QA pairs created for nonfactoid questions. DAWQAS is said to be freely available online soon. To the best of the authors' knowledge, no published research paper has used either the DAWQAS or the TALAA-AFAQ dataset. The CLEF dataset consists of 240 QA pairs and is the next smallest dataset; TREC, the smallest dataset, contains only 75 questions. Published papers have used only the TREC, CLEF, and AQA-WebCorp datasets. The Arabic datasets presented in the tables can be considered small compared with the datasets created for the same purpose in other languages [84]. For example, SimpleQuestions [64] and WikiMovies [63], QA datasets created for the English language, contain 108,442 and 100,000 QA pairs, respectively, as stated previously.

A total of 26 Arabic QA systems were proposed between 2002 and 2020; most of these systems have the three discussed components. Some QA systems did not have the first component, others did not have the second component, and some decomposed one of the components into two parts to represent more than three components. The last component, answer extraction, is the most important, given that the goal of QA systems is to answer questions by extracting answers. Fourteen of the 26 systems were dedicated to answering only factoid questions. The remaining 12 systems are distributed as follows. Five were hybrid QA systems that were dedicated to answering factoid questions plus one of the nonfactoid questions: (1) definition questions; (2) ''why'' questions; and (3) purpose, method, and causal questions, which are true questions. Seven were dedicated to answering only one type of nonfactoid question: (1) ''yes/no'' questions, (2) definition questions, (3) ''why'' questions, (4) ''why'' and ''how'' questions, (5) list and causal questions, and (6) Wh-questions.

As stated earlier, question analysis is the first QA system component, which aims to analyze the question to gain an enhanced understanding of the question and, in turn, aid in answer extraction. The main objective of this step is to analyze the semantic and syntactic components of the question, in which these components can be used for information retrieval and answer extraction. Several techniques can be used in question analysis. They include (1) QE, (2) question classification and/or domain classification, and (3) NLP techniques, i.e., NER, tokenization, segmentation, POS tagging, and parsing. These techniques can be used to produce the main outputs of the question-analysis component by (1) identifying the question type, expected answer type, question domain, and question focus; (2) reformulating queries from the given question; and (3) generating answer patterns. Some of the techniques, such as parsing, can be passed to the document retrieval component to be further used to locate answers from the retrieved documents or passages.

For example, Watson [85], an open-domain QA system developed by IBM, used several techniques to process the given question [86]. To identify the focus and lexical answer type (LAT) of the question, a set of rules was applied. The rules for detecting LATs are unreliable and may produce false positive results; hence, the system used a logistic regression classifier to estimate the confidence of the rules. This classifier was trained on manually annotated questions. In addition to estimating the confidence of the rules for detecting LATs, Watson could adjust the answer type by learning from previous questions and answers. Watson also used the English Slot Grammar parser and a predicate–argument structure generator for linguistic analysis for the question and text [87]; the obtained results were further used by other components within the Watson QA system. Another work [88] focused on proposing a system based on an NN and conditional random fields to process a question and identify the question type, domain, answer type, and NER. The system starts with preprocessing the question by (1) removing punctuations, elongations, and useless spaces; (2) using tokenization; and (3) using POS tagging. None of the 26 proposed Arabic QA systems used domain classification, NNs or any ML algorithm other than SVM to classify questions or identify answer type in the question-processing component. Parsing was used experimentally only in one paper [80]. A more sophisticated and intelligent component should be created to analyze questions for the Arabic language.

QE is used to improve the performance of information retrieval; in QA systems, QE is implemented by reformulating the original question and can be classified into three categories: manual, automatic, and interactive QE [23]. QE can be implemented using various techniques, such as (1) using external resources for QE (i.e., resource-based QE), including using WordNet to extract synonyms for the keywords formulating the query [23]; and (2) inflectional-based (morphological-based) expansion, including using different forms for the same keyword of the question [89] or implementing root stemming. Tables 7, 8, and 10 demonstrate that 15 of the 26 proposed Arabic QA systems performed QE; most of these systems performed QE in the question analysis component, while 2 performed QE in the document retrieval component. Twelve of the 15 proposed systems adopted resource-based QE, only 1 adopted inflectional QE, and 2 adopted resource-based and inflectional QE. The advantage of having QE in the question analysis component

is that enriching the question is less time consuming than enriching the document itself.

Answer type and patterns can be identified by using question classification, which in turn can be implemented using either rule- or ML-based techniques. Rule-based techniques can be implemented using NLP techniques and are based on checking IWs and defining patterns to apply pattern matching. The tables show that 12 of the 26 proposed Arabic QA systems adopted question classification, only 3 of them used ML-based classification, and the remaining 9 used rule-based classification. Table 7 indicates the factoid systems, and the tested question types were location, identified by 11 systems; person, identified by 10 systems; time, identified by 8 systems; numeric, identified by 6 systems; and organization, identified by 4 systems. Formulating a query from the given question can be accomplished using several techniques, such as QE, NLP, and identifying the answer type. The NLP techniques performed to create a query can be tokenization; identifying verb and noun phrases; POS tagging; NER; normalization; stemming; keyword extraction; determining question focus; and removing stop words, punctuations, and diacritics. Keyword extraction can be conducted by removing stop words and IWs; thus, the remaining words are the keywords. Question focus can be determined via NER or by identifying the noun phrase. Query reformulation creating multiple queries from the given question was adopted by only one proposed system in 2014 [44]. The tables also show the preprocessing techniques implemented on the question and answer by using the proposed Arabic QA systems. Three of the 26 systems did not specify if they used any preprocessing, but most of the systems implemented it; for example, stop-word removal was implemented by 16 systems, stemming was implemented by 12 systems, POS tagging and tokenization were implemented by 9 systems, normalization was implemented by 4 systems, diacritic removal was implemented by 3 systems, punctuation removal and root extraction were implemented by 2 systems, and Sanad removal was implemented by 1 system. Another important NLP technique is NER, which is implemented mainly by factoid QA systems. As illustrated in the tables, this technique was adopted by only 1 QA system of the hybrid systems, 1 of the nonfactoid systems and 7 of the 14 factoid systems.

In the document retrieval component, the proposed systems adopted document retrieval, passage retrieval, or both. One of the proposed systems adopted sentence retrieval from documents [40]. The aim of using document retrieval is to retrieve relevant documents to be used in the next component, whereas that of passage retrieval is to retrieve relevant passages from different documents to be used in the next component. Document retrieval can be implemented by retrieving documents from the Web by using a search engine API or by creating a local dataset from multiple documents. Passage retrieval can also be performed by using a search engine or by creating a local dataset from a set of passages extracted from documents. A passage is a paragraph separated by a new line in a document. Two proposed factoid Arabic QA systems [44], [49] adopted document and passage retrieval, while [44] mentioned that document retrieval was implemented using the Google Search engine and passage retrieval was implemented using the DDM. The IQAS [49] did not specify the techniques adopted for document and passage retrieval. Four factoid QA systems adopted document retrieval, and 6 factoid QA systems adopted passage retrieval, as illustrated in Table 7; three of the hybrid QA systems adopted document retrieval, and 5 adopted passage retrieval, as illustrated in Table 8; and only one nonfactoid QA system adopted document retrieval, and 4 adopted passage retrieval, as illustrated in Table 10. One factoid QA system [53] did not use the document retrieval component because the answer was extracted from a knowledge base. Two nonfactoid QA systems [50], [55] also did not use the document retrieval component because one system used ontology, while the other used a knowledge base, to retrieve answers. Most of the QA systems used search engines to implement document retrieval and created a local dataset for passage retrieval. Other systems used a simple matching technique to retrieve relevant documents or passages from local datasets.

In the answer extraction component, the surveyed papers, as illustrated in the tables, concentrated mainly on ranking, thus displaying the top-ranked paragraphs or sentences that contain the answer. Ten proposed Arabic QA systems displayed one answer, of which three chose one answer among multiple choices [36]–[38], three displayed top-ranked paragraphs [51], [48], [42], one used a set of patterns and NER to give one correct answer [31], one displayed ''yes'' or ''no'' as an answer for ''yes/no'' questions [39], one extracted answers from a knowledge base [53], and one extracted an answer from unstructured text by using an NN [54]. Nine proposed Arabic QA systems displayed one or more answers in accordance with either a set of patterns or the top-ranked paragraphs or sentences [32], [33], [40], [43]–[46], [50], [55]. Five proposed systems displayed five answers, which represented the top-ranked paragraphs or sentences [30], [41], [52], [34], [47]. Last, two of the proposed systems did not specify the number of answers displayed [49], [35]. Only three proposed systems adopted ML-based techniques to extract answers; two of them used an NN [53], [54] to extract answers, and one used an SVM [51] to rank answers.

The tables demonstrate that 15 proposed Arabic QA systems, including 10 factoid systems, 2 hybrid systems, and 3 nonfactoid systems, were open domain. The number of systems that experimented with closed-domain coverage was 8, including 2 factoid systems, 3 hybrid systems, and 3 nonfactoid systems. Three proposed systems did not specify the domain coverage. The maximum number of experimental QA pairs among all the systems was 49,739 [54]. The minimum number of experimental QA pairs among all the proposed systems was 43 [33]. The maximum number of experimental QA pairs among the factoid QA systems was 49,739, which was obtained by SOQAL [54], and the minimum number was 56 [44]. The maximum number of experimental QA

pairs among the hybrid QA systems was 335 [35], and the minimum number was 43 [33]. The maximum number of experimental QA pairs among the nonfactoid QA systems was 3825 [55], and the minimum number was 50 [34].

With reference to the tables, the evaluation measures used by the proposed Arabic QA systems were accuracy, used by 11 systems; precision, used by 10 systems; recall, used by 10 systems; the F1-measure, used by 6 systems; the MRR, used by 5 systems; C@1, used by 5 systems; the AQ, used by 3 systems; and EM, used by 1 system.

In relation to the performance of the systems, the highest accuracy among all the systems was 89%, which was reached by the factoid QA system NArQAS [46]; this system experimented with 250 open-domain QA pairs and did not implement any QE or question classification. Some Arabic QA systems, such as NArQAS [46], have reported an accuracy that surpasses the accuracy of other language QA systems; for example, English [90] yielded an accuracy of 71.2% for the SimpleQuestions dataset, which contains 100,000 QA pairs. Another QA system [91] dedicated to the Chinese language resulted in an accuracy of 60.1% on the Microsoft open domain question answering dataset, which contains 230,324 QA pairs. Such high accuracy cannot be confirmed without details related to the dataset structure, such as its size and the number of unique keywords.

The highest recall measure among all the systems was 100%, which was achieved by two QA systems. The first was the open-domain factoid QA system JAWEB [45], which did not specify the number of QA pairs experimented on but implemented QE and question classification. The second was a hybrid QA system [33] that experimented with 43 open-domain QA pairs without QE and question classification. While the English QA system presented in [90] yielded a recall of 93.7% for the SimpleQuestions dataset, again for the same reasons, the reported high recall cannot be confirmed.

Among all the systems, the factoid QA system QARAB reached the highest precision (97.3%) and the highest MRR (86%) [30]; the system experimented with 113 open-domain QA pairs by adopting QE and question classification. The highest F1-measure (89%) was reached by the hybrid QA system QArabPro [35], which experimented with 335 open-domain QA pairs by adopting QE and question classification. The highest C@1 (89%) was reached by the open-domain factoid QA system NArQAS [46], which experimented with 250 QA pairs without using QE or question classification. The highest AQ (68.62%) was reached by a factoid QA system [44] that experimented with 56 open-domain QA pairs by adopting QE and question classification.

The tables show that among all the systems, the hybrid QA system IDRAAQ [37] achieved the lowest accuracy (13%); this system experimented with 160 closed-domain QA pairs by adopting only QE. The lowest recall (57%) and the lowest precision (64%) were reached by an open-domain nonfactoid QA system [48] that experimented with 80 QA pairs by adopting QE and question classification. The lowest MRR (56%) was reached by a factoid QA system [44] that experimented with 56 open-domain QA pairs by adopting QE and question classification. The lowest F1-measure (42.5%) was reached by the factoid QA system SOQAL [54], which experimented with 49,739 open-domain QA pairs with no QE and question classification. The lowest C@1 (19%) was reached by a hybrid QA system [36] that experimented with 160 closed-domain QA pairs without using QE or question classification. The lowest AQ (47.66%) was reached by the factoid QA system AlQuAnS [52], which experimented with 200 open-domain QA pairs by adopting QE and question classification.

Among the 14 factoid QA systems, NArQAS reached the maximum accuracy and C@1 measures (89%) on [46] while experimenting with 250 open-domain QA pairs. The maximum precision and MRR measures (97.3% and 86%, respectively) were reached by QARAB [30], which experimented with 113 open-domain QA pairs. The maximum recall (100%) was reached by JAWEB [45], the maximum F1-measure (78.89%) was reached by AQuASys [40], and the maximum AQ measure (68.62%) was reached in [44]. The minimum accuracy, AQ, and MRR measures (22.2%, 47.66%, and 8.16%, respectively) were reached by AlQuAnS [52], which experimented with 200 open-domain QA pairs. The minimum precision reached 66.25% on AQuASys [40], which experimented with 80 QA pairs. The minimum recall (97.3%) was reached by QARAB [30]; the minimum F1-measure (42.5%) was reached by SOQAL [54], which experimented with 49,739 open-domain QA pairs; and the minimum C@1 (89%) was reached by NArQAS [46].

Among the five hybrid QA systems, ALQASIM reached the maximum accuracy and C@1 measures (31% and 36%, respectively) [38]; the system experimented with 240 closed-domain QA pairs. The maximum recall and precision (100% and 94%, respectively) were reached in [33], which experimented with only 43 open-domain QA pairs. The minimum accuracy (13%) was reached by IDRAAQ [37], which experimented with 160 open-domain QA pairs. The minimum C@1 (19%) was reached by [36], which experimented with 160 open-domain QA pairs. The minimum recall and precision (86% and 93%, respectively) were reached by QArabPro [35], which experimented with 335 open-domain QA pairs. The F1-measure was computed by only one hybrid QA system, QArabPro, which reached an F-1 measure of 89%. The MRR and AQ were not computed by any of the proposed hybrid QA systems.

Among the seven nonfactoid QA systems, [39] reached the maximum accuracy (85%); the system in this study experimented with 100 QA pairs. The maximum recall and F1-measure (93% and 86%, respectively) were reached by [50], which experimented with 100 QA pairs. The maximum precision (83.47%) was reached by ASHLK [55], which experimented with 3825 closed-domain QA pairs. The minimum accuracy (68.53%) was reached by EWAQ [47],

which experimented with 250 closed-domain QA pairs. The minimum recall, precision, and F1-measure (57%, 64%, and 60%, respectively) were reached by [48], which experimented with 80 open-domain QA pairs. Only DefArabicQA [34] computed the MRR and AQ, which reached 81% and 64%, respectively. C@1, which was computed by only one system [42], reached 77.4%.

The tables illustrate that the accuracy of hybrid-type QA systems was generally lower than that of factoid and non-factoid QA systems. When QA systems concentrate on one type of question, the accuracy increases; i.e., an analysis of one type of question differs from another. Therefore, using question classification and then adapting the analysis and answering schemes related to each type of question are important.

Most of the proposed QA systems did not use a benchmark dataset. Only 5 out of 26 used available datasets, while the remaining systems created their own dataset or used a translated version of the TREC dataset. Most of the datasets used have small sizes, ranging from 43 to 3825. Only one of the QA systems experimented on a large dataset, which had reached a size of 49,344. Nonetheless, most QA pairs were machine translated using Google Translate, and only 1,395 QA pairs were manually created, thus making the dataset a noisy dataset.

The answer extraction component can be enhanced by using, for example, deep learning techniques, which are considered the most popular techniques for answer extraction or generation. One can experimentally use available models created for other languages on the Arabic language and then enhance these models. The availability of word embeddings, such as Aravec [92], fastText [93], AraBert [94], and even ELMO, for the Arabic language is an advantage in creating and experimenting with such models. However, the lack of large datasets for Arabic QA becomes a disadvantage; thus, an effort should be made to create large datasets for Arabic QA.

Another important element in the proposed Arabic QA systems is that three of them had the option not to answer a question. This is a great option and can be helpful when document retrieval does not return a relevant document or when the relevant document does not contain the answer.

Among the three components of QA systems, the main concentration was on the retrieval process, which encompasses question analysis and document retrieval. Minimal effort was given to the answer extraction component, and none of the QA components were given the right effort and evaluation.

Overall, none of the proposed systems are available online, and the 26 proposed Arabic QA systems cannot be evaluated by comparison with one another because each system worked on different datasets with diverse sizes and did not use a benchmark dataset. Nonetheless, we can summarize the gaps that should be filled for those who would like to work in QA in the Arabic language, as shown as follows:

- Large datasets that are either domain specific or open domain should be created and made available for all.
- Benchmark datasets should be experimented with.
- Ontologies and domain-specific datasets should be used with closed-domain QA systems.
- Linked open data with open-domain QA systems should be used.
- Other tools for processing Arabic text that were not tested in Arabic QA systems, such as Farasa [95], can be used for text segmentation, lemmatization, POS tagging, NER, and diacritization.
- There should be a focus on enhancing one QA component, and the three components should be evaluated separately and together.
- The answer extraction component should be enhanced by using different techniques that actually extract or generate answers and not only display top-ranked sentences and paragraphs that contain the answers.
- Different NLP techniques, such as the preprocessing technique mentioned previously, parsing, NER, inflectional QE, and resource-based QE, should be used in question analysis.
- ML-based techniques or even a combination of ML- and rule-based techniques should be used in question classification, domain classification, and answer extraction.
- Systems that have the option not to answer a question if none of the retrieved documents contain the right answer should be created.

In other languages, the concentration is mainly on the answer extraction component by creating reading comprehension models that are trained using a dataset that consists of not only QA pairs but also the paragraphs wherein the questions are asked about. Accordingly, one can create his/her own model that is trained on extracting answers from paragraphs and then test it within a full QA system to determine how the model can perform with articles retrieved using the retrieval component [77].

This does not neglect the importance of having question analysis and document retrieval components. Improving the performance of question analysis and creating queries that can retrieve related documents can improve the quality of answers extracted by the answer extraction component. Generally, the results given by closed-domain QA systems are more accurate than those given by open-domain QA systems since the text becomes less ambiguous. Creating closed-domain QA systems involves using ontologies and domain-specific datasets. However, creating open-domain QA systems may involve exploiting linked open data [96] and using mainly Web-search or open-domain datasets.

IX. CONCLUSION

This paper reviewed the work in Arabic QA. This paper provided a brief introduction on the Arabic language, NLP, and the challenges associated with the Arabic language. It then presented the paper selection criteria and some statistics about the selected papers. Subsequently, this paper presented

the related work conducted and addressed the gaps to advise those who would like to work in the same area. The total number of Arabic QA datasets proposed to be used as a benchmark was six. The total number of QA systems proposed for Arabic text was 25. The authors concluded that the available resources for QA were limited and had small sizes. Some of the resources were developed but were not used in experimentation. Most researchers worked on Arabic QA systems and created their own resources that were unavailable online. ML algorithms were insufficiently used in Arabic QA systems. Experiments to evaluate QA systems can be focused on three main components: question analysis, information retrieval, and answer extraction.

REFERENCES

[1] I. Srba and M. Bielikova, "A comprehensive survey and classification of approaches for community question answering," *ACM Trans. Web*, vol. 10, no. 3, pp. 1–63, Aug. 2016.
[2] I. Vlad Serban, A. García-Durán, C. Gulcehre, S. Ahn, S. Chandar, A. Courville, and Y. Bengio, "Generating factoid questions with recurrent neural networks: The 30M factoid question-answer corpus," 2016, *arXiv:1603.06807*. [Online]. Available: http://arxiv.org/abs/1603.06807
[3] A. Moubaiddin, O. Shalbak, B. Hammo, and N. Obeid, "Arabic dialogue system for hotel reservation based on natural language processing techniques," *Computación y Sistemas*, vol. 19, no. 1, pp. 119–134, Mar. 2015.
[4] B. A. Shawar, "A Chatbot as a natural Web Interface to Arabic Web QA," *Int. J. Emerg. Technol. Learn. (iJET)*, vol. 6, no. 1, pp. 37–43, 2011.
[5] M. F. Al-Jouie and A. M. Azmi, "Automated evaluation of school children essays in Arabic," *Procedia Comput. Sci.*, vol. 117, pp. 19–22, 2017.
[6] H. Rababah and A. T. Al-Taani, "An automated scoring approach for Arabic short answers essay questions," in *Proc. 8th Int. Conf. Inf. Technol. (ICIT)*, May 2017, pp. 697–702.
[7] W. H. Gomaa and A. A. Fahmy, "Automatic scoring for answers to Arabic test questions," *Comput. Speech Lang.*, vol. 28, no. 4, pp. 833–857, Jul. 2014.
[8] W. Bakari, P. Bellot, and M. Neji, "Literature review of Arabic question-answering: Modeling, generation, experimentation and performance analysis," in *Flexible Query Answering Systems*. Cham, Switzerland: Springer, 2015, pp. 321–334.
[9] D. Jurafsky and J. H. Martin, *Speech & Language Processing*. London, U.K.: Pearson, 2017.
[10] S. K. Ray and K. Shaalan, "A review and future perspectives of Arabic question answering systems," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 12, pp. 3169–3190, Dec. 2016.
[11] A. Mishra and S. K. Jain, "A survey on question answering systems with classification," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 28, no. 3, pp. 345–361, Jul. 2016.
[12] M. Biltawi, A. Awajan, and S. Tedmori, "Evaluation of question classification," in *Proc. 2nd Int. Conf. New Trends Comput. Sci. (ICTCS)*, Oct. 2019, pp. 1–7.
[13] Y. H. Phuong and L. G. T. Nguyen, "English teachers'questions in a vietnamese high school reading classroom," *JEELS (J. English Educ. Linguistics Stud.)*, vol. 4, no. 2, pp. 129–154, 2018.
[14] K. C. Ryding, *A Reference Grammar of Modern Standard Arabic*. Cambridge, U.K.: Cambridge Univ. Press, 2005.
[15] F. Aouladomar, "Towards answering procedural questions," in *Proc. IJCAI Workshop Knowl. Reasoning Answering Questions*, 2005, pp. 1–11.
[16] H.-J. Oh, C.-H. Lee, H.-J. Kim, and M.-G. Jang, "Descriptive question answering in encyclopedia," in *Proc. ACL Interact. Poster Demonstration Sessions (ACL)*, 2005, pp. 1–4.
[17] A. Farghaly and K. Shaalan, "Arabic natural language processing: Challenges and solutions," *ACM Trans. Asian Lang. Inf. Process.*, vol. 8, no. 4, pp. 1–22, 2009.
[18] E. Saiegh-Haddad and R. Henkin-Roitfarb, "The structure of Arabic language and orthography," in *Handbook Arabic Literacy*. Dordrecht, The Netherlands: Springer, 2014, pp. 3–28.
[19] M. Biltawi, A. Awajan, and S. Tedmori, "Towards building a frame-based ontology for the Arabic language," in *Proc. ACIT Int. Arab Conf. Inf. Technol.*, 2017, pp. 1–7.
[20] M. A. Yaghan, "'Arabizi': A contemporary style of Arabic slang," *Des. Issues*, vol. 24, no. 2, 2008, pp. 39–52.
[21] M. Alian and A. Awajan, "Arabic tag sets: Review," in *Proc. SAI Intell. Syst. Conf.*, 2018, pp. 592–606.
[22] M. Biltawi, A. Awajan, S. Tedmori, and A. Al-Kouz, "Exploiting multilingual wikipedia to improve Arabic named entity resources," *Int. Arab J. Inf. Technol.* , vol. 14, no. 4A, pp. 598–607, 2017.
[23] H. K. Azad and A. Deepak, "Query expansion techniques for information retrieval: A survey," *Inf. Process. Manage.*, vol. 56, no. 5, pp. 1698–1735, Sep. 2019.
[24] A. M. Ezzeldin and M. Shaheen, "A survey of Arabic question answering: Challenges, tasks, approaches, tools, and future trends," in *Proc. 13th Int. Arab Conf. Inf. Technol. (ACIT)*, 2012, pp. 1–8.
[25] M. Shaheen and A. M. Ezzeldin, "Arabic question answering: Systems, resources, tools, and future trends," *Arabian J. Sci. Eng.*, vol. 39, no. 6, pp. 4541–4564, 2014.
[26] W. Bakari, P. Bellot, and M. Neji, "Researches and reviews in Arabic question answering: Principal approaches and systems with classification," in *Proc. Int. Arab Conf. Inf. Technol. (ACIT)*, 2016, pp. 1–9.
[27] B. A. B. Sati, M. A. S. Ali, and S. M. Abdou, "Arabic text question answering from an answer retrieval point of view: A survey," *Int. J. Adv. Comput. Sci. Appl.*, vol. 7, no. 7, pp. 478–484, 2016.
[28] A. Bouziane, D. Bouchiha, N. Doumi, and M. Malki, "Question answering systems: The story till the Arabic linked data," *Int. J. Artif. Intell. Soft Comput.*, vol. 6, no. 1, pp. 24–42, 2017.
[29] B. F. Green, Jr., A. K. Wolf, C. Chomsky, and K. Laughery, "Baseball: An automatic question-answerer," presented at the Western Joint IRE-AIEE-ACM Comput. Conf., 1961.
[30] B. Hammo, H. Abu-Salem, and S. Lytinen, "QARAB: A: Question answering system to support the Arabic language," in *Proc. ACL Workshop Comput. Approaches Semitic Lang.*, 2002, pp. 1–11.
[31] Y. Benajiba, P. Rosso, and A. Lyhyaoui, "Implementation of the ArabiQA question answering system's components," in *Proc. Workshop Arabic Natural Lang. Process., 2nd Inf. Commun. Technol. Int. Symp. (ICTIS)*, 2007, pp. 1–5.
[32] W. Brini, M. Ellouze, S. Mesfar, and L. H. Belguith, "An Arabic question-answering system for factoid questions," in *Proc. Int. Conf. Natural Lang. Process. Knowl. Eng.*, Sep. 2009, pp. 1–7.
[33] W. Brini, M. Ellouze, O. Trigui, S. Mesfar, L. H. Belguith, and P. Rosso, "Factoid and definitional Arabic question answering system," in *Proc. Post NOOJ*, 2009, pp. 8–10.
[34] O. Trigui, L. H. Belguith, and P. Rosso, "DefArabicQA: Arabic definition question answering system," in *Proc. 7th Workshop Lang. Resour. Hum. Lang. Technol. Semitic Lang. (LREC)*, 2010, pp. 40–45.
[35] M. Akour, S. O. Abufardeh, K. Magel, and Q. Al-Radaideh, "QArabPro: A rule based question answering system for reading comprehension tests in Arabic," *Amer. J. Appl. Sci.*, vol. 8, no. 6, pp. 652–661, 2011.
[36] O. Trigui, L. H. Belguith, P. Rosso, H. B. Amor, and B. Gafsaoui, "Arabic QA4MRE at CLEF 2012: Arabic question answering for machine reading evaluation," in *Proc. CLEF*, 2012.
[37] L. Abouenour, K. Bouzoubaa, and P. Rosso, "IDRAAQ: New Arabic question answering system based on query expansion and passage retrieval," in *Proc. CLEF*, 2012.
[38] A. M. Ezzeldin, M. H. Kholief, and Y. El-Sonbaty, "ALQASIM: Arabic language question answer selection in machines," in *Proc. Int. Conf. Cross-Lang. Eval. Forum Eur. Lang.*, 2013, pp. 100–103.
[39] W. N. Bdour and N. K. Gharaibeh, "Development of yes/no Arabic question answering system," 2013, *arXiv:1302.5675*. [Online]. Available: http://arxiv.org/abs/1302.5675
[40] S. Bekhti and M. Al-Harbi, "AQuASys: A question-answering system for Arabic," in *Proc. WSEAS Int. Conf. Recent Adv. Comput. Eng.*, 2013, pp. 1–10.
[41] A. I. Kamal, M. A. Azim, and M. Mahmoud, "Enhanced Arabic question answering system," in *Proc. Int. Conf. Comput. Intell.Commun. Netw.*, 2014, pp. 641–645.
[42] A. Azmi and N. AlShenaifi, "Handling, 'why' questions in Arabic," in *Proc. 5th Int. Conf. Arabic Lang. Process. (CITALA)*, 2014.
[43] H. Abdelnasser, M. Ragab, R. Mohamed, A. Mohamed, B. Farouk, N. M. El-Makky, and M. Torki, "Al-Bayan: An Arabic question answering system for the Holy Quran," in *Proc. EMNLP Workshop Arabic Natural Lang. Process. (ANLP)*, 2014, pp. 1–8.
[44] N. S. Fareed, H. M. Mousa, and A. B. Elsisi, "Syntactic open domain Arabic question/answering system for factoid questions," in *Proc. 9th Int. Conf. Inform. Syst.*, Dec. 2014, p. 1.

[45] H. Kurdi, S. Alkhaider, and N. Alfaifi, "Developemnt and evaluation of a Web based question answering system for Arabic language," *Comput. Sci. Inf. Technol. (CS & IT)*, vol. 4, no. 2, pp. 187–202, 2014.
[46] W. Bakari, O. Trigui, and M. Neji, "Logic-based approach for improving Arabic question answering," in *Proc. IEEE Int. Conf. Comput. Intell. Comput. Res.*, Dec. 2014, pp. 1–6.
[47] F. T. AL-Khawaldeh, "Answer extraction for why Arabic questions answering systems: EWAQ," 2019, *arXiv:1907.04149*. [Online]. Available: http://arxiv.org/abs/1907.04149
[48] W. Ahmed and P. BabuAnto, "Answer extraction for how and why questions in question answering systems," *Int. J. Comput. Eng. Res.*, vol. 12, no. 6, pp. 18–22, 2016.
[49] Z. Neji, M. Ellouze, and L. H. Belguith, "IQAS: Inference question answering system for handling temporal inference," in *Proc. Int. Symp. Innov. Intell. Syst. Appl. (INISTA)*, Aug. 2016, pp. 1–5.
[50] A. Albarghothi, F. Khater, and K. Shaalan, "Arabic question answering using ontology," *Procedia Comput. Sci.*, vol. 117, pp. 183–191, 2017.
[51] W. Ahmed, A. Ahmed, and A. P. Babu, "Web-based Arabic question answering system using machine learning approach," *Int. J. Adv. Res. Comput. Sci.*, vol. 8, no. 1, pp. 1–6, 2017.
[52] M. Nabil, A. Abdelmegied, Y. Ayman, A. Fathy, G. Khairy, M. Yousri, N. M. El-Makky, and K. Nagi, "AlQuAnS-an Arabic language question answering system," in *Proc. KDIR*, 2017, pp. 1–11.
[53] W. Ahmed, P. A. Bibin, and B. P. Anto, "Question answering system based on neural networks," *Int. J. Eng. Res.*, vol. 6, no. 3, pp. 142–144, 2017.
[54] H. Mozannar, K. El Hajal, E. Maamary, and H. Hajj, "Neural Arabic question answering," 2019, *arXiv:1906.05394*. [Online]. Available: http://arxiv.org/abs/1906.05394
[55] A. Abdi, S. Hasan, M. Arshi, S. M. Shamsuddin, and N. Idris, "A question answering system in hadith using linguistic knowledge," *Comput. Speech Lang.*, vol. 60, Mar. 2020, Art. no. 101023.
[56] F. Gey and D. Oard, "The TREC-2001 cross-language information retrieval track: Searching Arabic using English, French or Arabic queries," in *Proc. Text REtrieval Conf. (TREC)*, 2001, p. 78.
[57] D. W. Oard and F. C. Gey, "The TREC 2002 Arabic/English CLIR track," in *Proc. TREC*, 2002, pp. 1–10.
[58] A. Peñas, E. H. Hovy, P. Forner, Á. Rodrigo, R. F. Sutcliffe, C. Forascu, and C. Sporleder, "Overview of QA4MRE at CLEF 2011: Question answering for machine reading evaluation," in *Proc. CLEF*, 2011, pp. 1–10.
[59] R. F. Sutcliffe, A. Peñas, E. H. Hovy, P. Forner, Á. Rodrigo, C. Forascu, Y. Benajiba, and P. Osenova, "Overview of QA4MRE main task at CLEF 2013," in *Proc. CLEF*, 2013, pp. 1–30.
[60] W. S. Ismail and M. N. Homsi, "DAWQAS: A dataset for Arabic why question answering system," *Procedia Comput. Sci.*, vol. 142, pp. 123–131, 2018.
[61] W. Bakari, P. Bellot, and M. Neji, "AQA-WebCorp: Web-based factual questions for Arabic," in *Proc. KES*, 2016, pp. 275–284.
[62] A. Aouichat and A. Guessoum, "Building TALAA-AFAQ, a corpus of Arabic FActoid question-answers for a question answering system," in *Proc. Int. Conf. Appl. Natural Lang. Inf. Syst.*, 2017, pp. 380–386.
[63] A. Miller, A. Fisch, J. Dodge, A.-H. Karimi, A. Bordes, and J. Weston, "Key-value memory networks for directly reading documents," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016.
[64] A. Bordes, N. Usunier, S. Chopra, and J. Weston, "Large-scale simple question answering with memory networks," 2015, *arXiv:1506.02075*. [Online]. Available: http://arxiv.org/abs/1506.02075
[65] P. Li, W. Li, Z. He, X. Wang, Y. Cao, J. Zhou, and W. Xu, "Dataset and neural recurrent sequence labeling model for open-domain factoid question answering," 2016, *arXiv:1607.06275*. [Online]. Available: http://arxiv.org/abs/1607.06275
[66] C. Tan, F. Wei, N. Yang, B. Du, W. Lv, and M. Zhou, "S-net: From answer extraction to answer generation for machine reading comprehension," 2017, *arXiv:1706.04815*. [Online]. Available: http://arxiv.org/abs/1706.04815
[67] S. Teufel, "An overview of evaluation methods in TREC ad hoc information retrieval and TREC question answering," in *Evaluation of Text and Speech Systems*. Dordrecht, The Netherlands: Springer, 2007, pp. 163–186.
[68] L. Gillard, P. Bellot, and M. El-Bèze, "Question answering evaluation survey," in *Proc. LREC*, 2006, pp. 1133–1138.
[69] P. Forner, D. Giampiccolo, B. Magnini, A. Peñas, Á. Rodrigo, and R. Sutcliffe, "Evaluating multilingual question answering systems at CLEF," in *Proc. CLEF*, 2010.
[70] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQUAD: 100,000+ questions for machine comprehension of text," 2016, *arXiv:1606.05250*. [Online]. Available: https://arxiv.org/abs/1606.05250
[71] F. A. Mohammed, K. Nasser, and H. M. Harb, "A knowledge based Arabic question answering system (AQAS)," *ACM SIGART Bull.*, vol. 4, no. 4, pp. 21–30, Oct. 1993.
[72] B. Hammo, S. Abuleil, S. Lytinen, and M. Evens, "Experimenting with a question answering system for the Arabic language," *Comput. Humanities*, vol. 38, no. 4, pp. 397–415, Nov. 2004.
[73] S. Abuleil and M. Evens, "Discovering lexical information by tagging Arabic newspaper text," in *Proc. Workshop Comput. Approaches to Semitic Lang. (Semitic)*, 1998, pp. 1–7.
[74] G. Kanaan, A. Hammouri, R. Al-Shalabi, and M. Swalha, "A new question answering system for the Arabic language," *Amer. J. Appl. Sci.*, vol. 6, no. 4, pp. 797–805, Apr. 2009.
[75] N. S. Fareed, H. M. Mousa, and A. B. Elsisi, "Enhanced semantic Arabic question answering system based on Khoja stemmer and AWN," in *Proc. 9th Int. Comput. Eng. Conf. (ICENCO)*, Dec. 2013, pp. 85–91.
[76] W. Bakari, P. Bellot, and M. Neji, "A logical representation of Arabic questions toward automatic passage extraction from the Web," *Int. J. Speech Technol.*, vol. 20, no. 2, pp. 339–353, Jun. 2017.
[77] D. Chen, A. Fisch, J. Weston, and A. Bordes, "Reading wikipedia to answer open-domain questions," 2017, *arXiv:1704.00051*. [Online]. Available: http://arxiv.org/abs/1704.00051
[78] Z. Salem, J. Sadek, F. Chakkour, and N. Haskkour, "Automatically finding answers to 'Why' and 'how to' questions for Arabic language," in *Proc. Int. Conf. Knowl.-Based Intell. Inf. Eng. Syst.*, 2010, pp. 586–593.
[79] J. Sadek, F. Chakkour, and F. Meziane, "Arabic rhetorical relations extraction for answering 'why' and 'how to' questions," in *Proc. Int. Conf. Appl. Natural Lang. Inf. Syst.*, 2012, pp. 385–390.
[80] J. Sadek and F. Meziane, "A discourse-based approach for Arabic question answering," *ACM Trans. Asian Low-Resource Lang. Inf. Process.*, vol. 16, no. 2, pp. 1–18, Dec. 2016.
[81] S. Khoja and R. Garside, "Stemming Arabic text," Comput. Dept., Lancaster Univ., Lancashire, U.K., Tech. Rep., 1999.
[82] A. M. Azmi and N. A. Alshenaifi, "Answering Arabic why-questions: Baseline vs. RST-based approach," *ACM Trans. Inf. Syst.*, vol. 35, no. 1, pp. 1–19, 2016.
[83] A. M. Azmi and N. A. Alshenaifi, "*Lemaza*: An Arabic why-question answering system," *Natural Lang. Eng.*, vol. 23, no. 6, pp. 877–903, 2017.
[84] M. Biltawi, A. Awajan, and S. Tedmori, "Towards building an open-domain corpus for Arabic reading comprehension," in *Proc. 35th Int. Bus. Inf. Manage. Assoc. (IBIMA)*, 2020, pp. 1–27.
[85] D. A. Ferrucci, "Introduction to 'this is Watson,'" *J. Res. Develop.* , vol. 56, nos. 3–4, p. 1, 2012.
[86] A. Lally, J. M. Prager, M. C. McCord, B. K. Boguraev, S. Patwardhan, J. Fan, P. Fodor, and J. Chu-Carroll, "Question analysis: How Watson reads a clue," *IBM J. Res. Develop.*, vol. 56, nos. 3–4, pp. 1–2, 2012.
[87] M. C. McCord, J. W. Murdock, and B. K. Boguraev, "Deep parsing in Watson," *IBM J. Res. Develop.*, vol. 56, nos. 3–4, pp. 1–3, 2012.
[88] S. K. Bhaskaran, C. Sreejith, and P. C. Rafeeque, "Neural networks and conditional random fields based approach for effective question processing," *Procedia Comput. Sci.*, vol. 143, pp. 211–218, 2018.
[89] M. W. Bilotti, B. Katz, and J. Lin, "What works better for question answering: Stemming or morphological query expansion," in *Proc. Inf. Retr. Question Answering (IR4QA) Workshop SIGIR*, 2004.
[90] D. Lukovnikov, A. Fischer, J. Lehmann, and S. Auer, "Neural network-based question answering over knowledge graphs on word and character level," in *Proc. 26th Int. Conf. World Wide Web*, Apr. 2017, pp. 1211–1220.
[91] G. Zhang, X. Fan, C. Jin, and M. Wu, "Open-domain document-based automatic QA models based on CNN and attention mechanism," in *Proc. IEEE Int. Conf. Big Knowl. (ICBK)*, Nov. 2019, pp. 326–332.
[92] A. B. Soliman, K. Eissa, and S. R. El-Beltagy, "AraVec: A set of Arabic word embedding models for use in Arabic NLP," *Procedia Comput. Sci.*, vol. 117, pp. 256–265, 2017.
[93] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Trans. Assoc. Comput. Linguistics*, vol. 5, pp. 135–146, Dec. 2017.
[94] F. Baly and H. Hajj, "raBERT: Transformer-based model for Arabic language understanding," in *Proc. 4th Workshop Open-Source Arabic Corpora Process. Tools, With Shared Task Offensive Lang. Detection*, 2020, pp. 9–15.

[95] A. Abdelali, K. Darwish, N. Durrani, and H. Mubarak, "Farasa: A fast and furious segmenter for Arabic," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Demonstrations*, 2016, pp. 1–7.
[96] M. Latifi, H. R. Hontoria, and M. Sànchez-Marrè, "ScoQAS: A semantic-based closed and open domain question answering system," Procesamiento de Lenguaje Natural, Tech. Rep. 59, 2017, pp. 73–80.
[97] D. L. Rubin, N. F. Noy, and M. A. Musen, "Protégé: A tool for managing and using terminology in radiology applications," *J. Digit. Imag.*, vol. 20, no. S1, pp. 34–46, Nov. 2007.
[98] E. Alatrish, D. Tosic, and N. Milenkovic, "Building ontologies for different natural languages," *Comput. Sci. Inf. Syst.*, vol. 11, no. 2, pp. 623–644, 2014.
[99] M. Silberztein, "NooJ: A linguistic annotation system for corpus processing," in *Proc. HLT/EMNLP Interact. Demonstrations*, 2005, pp. 1–2.
[100] A. Boudlal, A. Lakhouaja, A. Mazroui, A. Meziane, M. O. A. O. Bebah, and M. Shoul, "Alkhalil morpho sys1: A morphosyntactic analysis system for Arabic texts," in *Proc. Int. Arab Conf. Inf. Technol.*, 2010, pp. 1–6.
[101] M. Boudchiche, A. Mazroui, M. O. A. O. Bebah, A. Lakhouaja, and A. Boudlal, "AlKhalil morpho sys 2: A robust Arabic morpho-syntactic analyzer," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 29, no. 2, pp. 141–146, Apr. 2017.
[102] M. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi, "Bidirectional attention flow for machine comprehension," 2016, *arXiv:1611.01603*. [Online]. Available: http://arxiv.org/abs/1611.01603
[103] D. Weissenborn, G. Wiese, and L. Seiffe, "Making neural QA as simple as possible but not simpler," 2017, *arXiv:1703.04816*. [Online]. Available: http://arxiv.org/abs/1703.04816

MARIAM M. BILTAWI received the B.Sc. degree in computer science from Princess Sumaya University for Technology (PSUT), Jordan, in 2005, and the M.Sc. degree from Al-Balqa Applied University (BAU), Jordan, in 2011. She is currently pursuing the Ph.D. degree in computer science with PSUT. Since November 2006, she has been working as a Computer Laboratory Supervisor with the Computer Science Department, PSUT. During her position as a Laboratory Supervisor, she works as a part-time Lecturer with PSUT. Her research interests include natural language processing, data mining, machine learning, image processing, and operating systems.

SARA TEDMORI received the B.Sc. degree in computer science from the American University of Beirut, Lebanon, in 2001, and the M.Sc. degree in multimedia and Internet computing and the Ph.D. degree in computer science from Loughborough University, U.K., in 2003 and 2008, respectively. She is currently an Associate Professor with the Computer Science Department, Princess Sumaya University of Technology, Jordan. Her research interests include natural language processing, sentiment analysis, data mining, image processing, social networks, web technologies, and classification.

ARAFAT AWAJAN received the Ph.D. degree in computer science from the University of Franche-Comte, France, in 1987. He has held various administrative and academic positions at the Royal Scientific Society and Princess Sumaya University for Technology (PSUT). From 2000 to 2003, he was the Head of the Department of Computer Science. From 2005 to 2006, he was the Head of the Department of Computer Graphics and Animation. From 2004 to 2007, he was the Dean of the King Hussein School for Information Technology. From 2008 to 2010, he was the Director of the Information Technology Center, RSS. From 2011 to 2014, he was the Dean of Student Affairs. From 2014 to 2017, he was the Dean of the King Hussein School for Computing Sciences. From 2017 to 2020, he was the Vice President of PSUT. He is currently the President of Mutah University. He is also a Full Professor with PSUT. His research interests include natural language processing, Arabic text mining, and digital image processing.