Journal of King Saud University – Computer and Information Sciences 34 (2022) 2651–2665

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

TAAM: Topic-aware abstractive arabic text summarisation using deep recurrent neural networks

Dimah Alahmadi, Arwa Wali, Sarah Alzahrani

Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia

ARTICLE INFO

Article history:
Received 18 January 2022
Revised 28 February 2022
Accepted 27 March 2022
Available online 18 April 2022

Keywords:
Natural language processing (NLP)
Automatic summarisation
Arabic Abstractive summarisation
Topic aware
Recurrent Neural Networks
Deep learning

ABSTRACT

Abstractive text summarisation is essential to producing natural language summaries with main ideas from large text documents. Despite the success of English language-based abstractive text summarisation models in the literature, they are limitedly supporting the Arabic language. Current abstractive Arabic summarisation models have several unresolved issues, a critical one of which is syntax inconsistency, which leads to low-accuracy summaries. A new approach that has shown promising results involves adding topic awareness to a summariser to guide the model by mimicking human awareness. Therefore, this paper aims to enhance the accuracy of abstractive Arabic summarisation by introducing a novel topic-aware abstractive Arabic summarisation model (TAAM) that employs a recurrent neural network. Two experiments were conducted on TAAM: quantitative and qualitative. Based on a quantitative approach using ROUGE matrices, the TAAM model achieves 10.8% higher accuracy than other existing baseline models. Additionally, based on a qualitative approach that captures users' perspectives, the TAAM model is capable of producing a coherent Arabic summary that is easy to read and captures the main idea of the input text.

1. Introduction

The world is facing an information overload crisis because so many channels are generating data, including news platforms, Google searches, social media, and messaging. Every second, these channels generate large amounts of text data, most of which is unstructured (Chowdhary, 2020). Therefore, today's machines must handle large volumes of text data using Natural Language Processing (NLP). NLP is used to program computers to analyse and process natural language data. Despite the complexity and diversity of human languages, NLP can analyse large amounts of text data consistently (Chowdhary, 2020; Mohammad, 2020). There are many NLP applications, including machine translation, information extraction, sentiment analysis, chatbots, and automatic text summarisation (Lee, 2020). Specifically, NLP for automatic text summarisation has attracted considerable attention because retrieving useful information from the huge amounts of text data generated daily is extremely challenging (Guo et al., 2019). Generally, for automatic text summarisation, two techniques are adopted: extractive and abstractive (Khan et al., 2018). Extractive text summarisation (ETS) captures the primary summarised sentences from a given text-based document (Yang et al., 2020), but a disadvantage is its inability to retain general ideas from the whole of a given text. Moreover, the sentences in summaries produced using ETS models may not be semantically coherent (Song et al., 2019). Abstractive text summarisation (ATS), by contrast, aims to condense salient information from a given text and produce a summary containing new vocabulary and sentences that differ from the input text (Yang et al., 2020). In recent years, ATS has garnered more interest than ETS because it overcomes ETS's inability to capture the main ideas in a text (Guo et al., 2019).

Deep learning has been applied effectively in many applications. Deep learning is essentially a neural network with three or more layers, which attempt to simulate the behaviour of the human brain by learning from large amounts of data (Goodfellow et al., 2016). Deep learning has been effectively used in various fields. For example, incorporating deep learning into customer service, using virtual assistants like Apple's Siri, Amazon Alexa, or Google Assistant. Also, financial institutions use deep learning to

E-mail addresses: Dalahmadi@kau.edu.sa (D. Alahmadi), amwali@kau.edu.sa (A. Wali), salzahrani1331@stu.kau.edu.sa (S. Alzahrani)
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.jksuci.2022.03.026
1319-1578/

predict analytics to drive algorithmic trading of stocks, assess business risks for loan approvals, or detect fraud (Goodfellow et al., 2016). One of the effective deep learning models is sequence-to-sequence (seq2seq) encoder-decoder models. Recently, researchers have adopted seq2seq for ATS, which has been a successful approach in many studies (Yao et al., 2018). ATS based on seq2seq models can learn the context of a given text and be trained to understand the text sufficiently enough to generate a novel summary (Zaki et al., 2019). In practice, the encoder takes the whole input sentence and calculates a fixed dimensional feature vector; then the decoder uses the feature vector to generate the output sequence (Yao et al., 2018). Research to explore and achieve better performance with seq2seq models for English ATS has led the way, but ATS for other languages, including Arabic, is lagging (Ibrahim et al., 2017). However, seq2seq models have been applied to ATS for multiple languages, including in a few studies considering the Arabic language (Ouyang et al., 2019; Elmadani et al., 2020). Despite some success in generating abstractive Arabic summaries with seq2seq models, the summaries are relatively inaccurate according to the most frequently used measure for abstractive text summarisation, the Recall-Oriented Understudy for Gisting Evaluation (ROUGE) scoring algorithm (Suleiman and Awajan, 2020; Keneshloo et al., 2019). ROUGE works by comparing the produced summary with the ground truth summary as a reference (Shi et al., 2021). To the best of our knowledge, the highest accuracy obtained for abstractive Arabic summarisation is 60%, according to the ROUGE measure (Zaki et al., 2019), indicating a need for further investigation.

Arabic was the fourth most frequently spoken language worldwide, according to one study (Top Ten Languages Used in the Web. Available, n.d.) for the years 2000–2015 and the Arabic language is the fastest growing language on the web (Al-Saleh and Menai, 2016; Allahyari et al., 2017). Therefore, our study focused on using ATS for Arabic text documents to enrich the research on NLP for Arabic text summarisation. The significant lack of Arabic ATS has several causes, such as a limited standard Arabic corpus and a lack of research on Arabic text summarisation (Belkebir and Guessoum, 2018). Many researchers claim that the main reason for deficiencies in Arabic abstractive summarisation models is the rich morphology and complex word-formation structures of Arabic (Azmi and Altmami, 2018; Belkebir and Guessoum, 2018). Accordingly, producing cohesive abstractive Arabic summaries is highly challenging due to the following:

- Arabic has no capital letters, which makes it difficult to extract nouns for annotation in the summarisation process (Azmi and Altmami, 2018).
- Arabic uses diacritical marks to clarify the meaning of a word in a sentence because the same word can have several meanings; for example, the word (عقد) can mean complicated, contract, or necklace. Unfortunately, most Arabic texts do not use diacritical marks, leaving the reader to deduce the meaning of a word through its context. This can cause words to be annotated incorrectly during the summarisation process (Belkebir and Guessoum, 2018).
- Words in Arabic have roots and affixes, and although the latter can help to clarify meaning, they can also change words' meanings entirely. For example, the root (درس) means study; with the addition of the letter (م), it becomes (مدرس), which means teacher. This limits the overall number of root vocabularies that summarisation models can use to generate summaries, as a word can be repeated many times with different affixes (Zaki et al., 2019).
- Arabic's complex morphology makes it possible to convey a whole sentence with only one word; for example, the word (أنلزمكموها) means 'should we compel you to accept it.' Such words make ATS models extremely challenging to develop (Azmi and Altmami, 2018).

Due to these problems, abstractive Arabic summarisation models continue to generate low-accuracy summaries with syntactic errors, such as repeated words, incomplete sentences, and unintelligible summaries (Elmadani et al., 2020; Suleiman and Awajan, 2020; Zaki et al., 2019). To clarify more, Table 1 represents two examples of Arabic text with the ground truth summary and the generated summary from two recent abstractive Arabic summarization models (Suleiman and Awajan, 2020; Zaki et al., 2019). The text and the summaries in the two examples are translated from the Arabic language to the English language using Google translator. As shown in Table 1, the first example of outcome suffers from an incomplete summary, and the second example suffers from a repeated word in the generated summary. Hence, the previous examples demonstrate the need to enhance the accuracy of generated summaries and tackle the problem of inherent syntactic inconsistency in abstractive Arabic summarisation models. To address the previously explained challenges, this paper introduces a novel abstractive seq2seq summariser for Arabic texts.

Deep recurrent neural networks (RNNs) based on seq2seq have been utilised successfully in many fields, such as for machine translation and abstractive text summarisation (Guo et al., 2019). RNNs use an internal state (memory) to process variable-length sequences of inputs, which in our case means that the RNN model processes a text to produce a summary (Paulus et al., 2017). However, abstractive text summarization models have been mostly applied to a general domain dataset, such as news articles, which makes it challenging to apply these models to a specific domain because applying these models in a specific domain requires handling the domain-specific vocabulary and phrasing, while also requiring domain expertise to evaluate the generated summary (Gigioli et al., 2018). Most of the previous studies that generated an English summary based on various categories led to producing summaries missing a salient entity in the given input (Yang et al., 2020). Consequently, a new point view of an abstractive English summarization model appears in topic-aware models (Gigioli et al., 2018). Recent abstractive English summarisation research has applied a topic aware on RNN summariser model, such as adding a text categorization vector in the encoder as (Yang et al., 2020) or uses a multi-step attention mechanism and biased probability generation process to generate the model's topic-aware information as (Wang et al., 2018). Adding topic aware, prove that an abstractive English summariser with awareness of the generated summary topic leads to a better and more accurate summary, and outperform general summarisers (Gigioli et al., 2018; Wang et al., 2018; Yang et al., 2020). To the best of our knowledge, the topic aware model has never been applied to an abstractive Arabic summariser. Thus, by combining the strength of a deep RNN with a topic-ware module, this study developed a novel architecture for a topic-aware seq2seq summariser to enhance abstractive Arabic text summarisation. In this study, the TAAM model addresses the following research questions:

RQ1: can a topic-aware summarisation model that employs a deep RNN enhance the accuracy of overall abstractive Arabic text summarisation?

RQ2: can a topic-aware summarisation model improve the quality of generated summaries in terms of readability and relevance?

The existence of a powerful and effective ATS model for Arabic text documents is a remarkable step forward in the automatic text summarisation field. In this research, the main aims is to expand the research of abstractive Arabic text summarization in the automatic text summarization field and improve abstractive text

Table 1
Examples of recent outcomes of abstractive Arabic summarization.

Example 1 (Suleiman and Awajan, 2020)

Source text in Arabic:
أشارت توصيات أصدرتها لجنة مستقلة من الخبراء في مجال الطب إلى أن الأشخاص بين عاما 50 و59 من المعرضين بصورة أكبر للإصابة بأمراض القلب والسكتة الدماغية عليهم تناول جرعة يومية من الأسبرين المنخفض الجرعة وقالت قوة عمل الخدمات الوقائية الأميركية التي تحظى بمساندة الحكومة إنه علاوة على الحيلولة دون الإصابة بالأزمات القلبية والسكتات الدماغية فإن من يتبعون هذا النظام تقل لديهم فرص الإصابة بسرطان القولون إذا داوموا على جرعات الأسبرين هذه لمدة عشر سنوات على الأقل تأتي هذه التوصيات على نحو أضيق نطاقا ية مختلفةً من توصيات سابقة أصدرتها قوة العمل والتي ربطت مقترحاتها بالنوع وبشريحة عمر وتستند تعديلات التوصيات إلى إضافة مخاطر الإصابة بسرطان القولون وإضافة نتائج أربع تجارب إكلينيكية بشأن الأسبرين منذ 2009 وقال دوج اوينز عضو اللجنة من نوصيهم بأن يتناولوا الأسبرين هم الذين يتعرضون لمخاطر متزايدة للإصابة بأمراض القلب والأوعية الدموية ومن هم ليسوا عرضة لمضاعفات النزيف ورفضت الإدارة الأميركية للغذاء والدواء العام الماضي وصف الأسبرين لمنع الإصابة بالأزمات القلبية والسكتة الدماغية من جهة أخرى أوضحت نتائج دراسات أجريت على حيوانات إلى أن تناول مرضى السرطان للأسبرين العادي الزهيد الثمن يمكن أن يقوي فاعلية العقاقير الحديثة الباهظة التكلفة التي تساعد جهاز المناعة لديهم على مكافحة الأورام.

Source text in English:
Recommendations by an independent panel of medical experts indicated that people between 50 and 59 years of age who are at higher risk of heart disease and stroke should take a daily dose of low-dose aspirin and the US-backed Preventive Services Task Force said that in addition to preventing infection with heart attacks and strokes, those who follow this system have a lower chance of developing colon cancer if they keep these aspirin doses for at least ten years. These recommendations come in a narrower range than previous recommendations issued by the labour force that linked a proposal. The type and chip different age and adjustments are based on recommendations in addition to the risk of colon cancer and adding the results of four clinical trials on aspirin since 2009, committee member Doug Owens said, "We recommend that they take aspirin, those who are at increased risk of cardiovascular disease, and who are not exposed to complications of bleeding." The US Food and Drug Administration refused last year to describe aspirin to prevent heart attacks and stroke. On the other hand, the results of studies conducted on animals showed that consuming cancer patients with low-cost regular aspirin can strengthen the effectiveness of expensive modern drugs that help their immune system to fight tumors.

Ground truth summary in Arabic:
الأسبرين يحمي من الجلطات والسرطان

Ground truth summary in English:
Aspirin protects against clots and cancer

Generated summary in Arabic:
الاسبرين يقلل من السكته الدماغيه

Generated summary in English:
Aspirin reduces stroke

Example 2 (Zaki et al., 2019)

Source text in Arabic:
يقدم موقع الدستور بثاً مباشراً لمباراة مانشستر يونايتد ضد ستوك سيتي ، والتي من المقرر أن تقام في أولد ترافورد في الجولة السابعة من الدوري الإنجليزي.

Source text in English:
Al Dostour offers a live broadcast of Manchester United's match against Stoke City, which is due to take place at Old Trafford in the seventh round of the Premiership.

Ground truth summary in Arabic:
بث مباشر "بدون تقطيع" لمانشستر يونايتد وستوك سيتي في الدوري الممتاز

Ground truth summary in English:
Live broadcast" without chopping" for Manchester United and Stoke City in the Premiership

Generated summary in Arabic:
ستوك سيتي الدوري الإنجليزي vs مانشستر يونايتد ضد ستوك سيتي

Generated summary in English:
Manchester United vs Stoke City vs Stoke English Premier League

summarization for an Arabic text by introducing a novel topic-aware abstractive Arabic summarization model (TAAM). To the best of our knowledge, this paper is the first to build and explore the benefits of combining a topic-aware module and deep RNN to produce abstractive Arabic summaries. The main contributions of this work are as follows:

1. Development of a topic-aware abstractive summarisation model employing a deep RNN to generate high-quality summaries of Arabic texts based on quantitative and qualitative techniques.
2. Proposal of a novel framework for constructing and evaluating the proposed topic-aware abstractive Arabic summarisation model (TAAM).
3. Evaluation of different classification algorithms to select the best classifier for an Arabic topic-aware abstractive summariser.
4. Comparative analysis of the TAAM model and other abstractive summariser models based on quantitative results.

This paper consists of seven sections, including the introduction, and is organised as follows: Section 2 provides an overview of the background to the study, Section 3 presents the study's related work, Section 4 introduces the architecture of the TAAM, Section 5 explains the phases this study followed to build the TAAM model and the utilised dataset, Section 6 presents the TAAM evaluation and discusses the results, and Section 7 provides a summary of the study and makes recommendations for future research.

2. Background

The key utility of an abstractive text summarizer is the generation of summaries using novel words that do not exist in the original text (Khan et al., 2018). This gives abstractive models the strength to generate high-quality summaries closer to a human-produced product that is verbally and contextually innovative (Hou et al., 2017). Abstractive text summarisation does not merely copy sentences from a given text; instead, it produces new words from a vocabulary to construct an innovative sentence (Bhat et al., 2018). This capability requires a deep analysis of the original text, which requires a good understanding of word meanings using NLP methods (Abdolahi and Zahedh, 2017). To achieve this task, the abstractive text summarizer performs two steps. First, features are extracted from words to construct a semantic representation of the text via word embedding. This step is performed during the pre-processing phase. Second, natural language generation techniques are used to produce a novel summary during the post-processing phase (Chitrakala et al., 2016). The abstractive text summarization framework is implemented using a variety of methods divided into three categories: structure-, semantic-, and deep-learning-based (Gupta and Gupta, 2019). Structure-based methods apply predefined structures, such as trees, ontologies, and templates. Semantic-based methods use natural language generation capabilities of semantic representation. Examples of semantic-based methods include predicate arguments and semantic graphs (Ermakova et al., 2019). Deep-learning methods are based on neural or classical deep-learning techniques. RNN-based seq2seq models have recently been used to accomplish promising

results in text summarization studies (Hou et al., 2017; Shi et al., 2021). This section provides a background of deep RNN components, evaluation metrics, and challenges.

2.1. Deep RNN components

To understand the deep RNN-based seq2seq model, the seq2seq model must be discussed. A seq2seq model is a model that processes items in the form of a sequence and produces output in a sequence of items. The model works by combining two elements: an encoder and a decoder. The encoder takes tokens of the input sequence in the form of a hidden-state vector and transmits them to the decoder, which then generates an output sequence (Shi et al., 2021). The hidden-state vector considers the hidden memory that retains information about previous data the model has handled (Guo et al., 2019). A basic neural network accomplishes NLP tasks by learning and understanding the meaning of each word in a text. It processes text sequentially based on the seq2seq model; hence, the order determines the output, which depends on the previous output from the hidden state (Giles et al., 1994; Robinson, 1994), and the meaning of a word in a text depends only on earlier words. For example, if an input text is the two sentences sequentially; "Riyadh is the capital", and "the capital is Riyadh". A basic neural network considers the word 'Riyadh' as a named entity just in the first sentence depending on its location, and in the second sentence, the word 'Riyadh' is considered as a not named entity. An RNN handles this situation by using multiple time slots to perform a step. It considers two objects: the output from the previous step and the time. The RNN goes through many iterations to consider the two objects, the temporality of which gives RNN models the ability to recognise words' meanings regardless of the words' locations (Li et al., 2020). For example, if two previous sentences about Riyadh enter an RNN, the RNN considers the word 'Riyadh' as a named entity in both sentences (see Fig. 1). Moreover, an RNN can contain many layers of hidden states, resulting in a *deep RNN*. The hidden states in a deep RNN learn different features, with each layer taking the last hidden state of the layer as the whole input for its layer. In a deep RNN, the addition of layers can improve model results, but it is computationally expensive (Min-Yuh et al., 2018).

The basic unidirectional RNN produces a hidden state by reading the text in one direction. Consequently, any error in a prior hidden state will propagate to the forward hidden states, negatively affecting the final output (Al-Sabahi et al., 2018). To overcome this issue, the RNN must be able to read input text both ways. Hence, a bidirectional RNN works by employing forward and backward RNNs. The bidirectional RNN overcomes the issue of accumulated errors from the unidirectional RNN by reading input text from both sides (Bahdanau et al., 2014). Its reads input from both sides, where forward RNN creates hidden states that depend on text from reading input text from left to right. Otherwise, backward RNN produces hidden states from reading input text from right to left.

After items go through the process of encoding the task of the decoder part in the RNN model, the next step is to calculate the probability of each word for output in a certain sentence. Normally, a technique called Greedy Search selects the most likely words to generate the output sentence. The greedy search works by considering all possibilities for each word to form one sentence (Min-Yuh et al., 2018). In this way, the greedy search considers all the most likely possibilities for each word. So, if the vocabulary is 10 k, then computationally each step costs 10 k to the power of 10 k. Theoretically, this technique is perfect but in practice does not perform well (Min-Yuh et al., 2018). To produce a summary practically, we must consider only the most likely words, for instance, three words. Then, for the next step, we consider only the three most likely words for the previous three words and follow the same process for the next steps. This approach is called a Beam Search and the number of words is called Beam Width (Min-Yuh et al., 2018). Computationally, if the vocabulary is 10 k, the cost is 30 K.

2.2. Recurrent neural network gates

As the main task of an RNN seq2seq model is sequence-based, both encoder and decoder tend to use some form of gate, such as a Long Short-Term Memory (LSTM) or a Gated Recurrent Unit (GRU), achieving state-of-the-art performance (Keneshloo et al., 2019). GRU and LSTM gates are applied to an RNN model as memory cells, allowing the model to save predicted information from early words from the input text. The memory cell in a gate allows the RNN to control the number of data that transfers between hidden states (Rush et al., 2015). This helps RNNs overcome the issue of the vanishing gradient, which is the inability of an RNN normal gate to remember entity information predicted earlier (Hochreiter and Schmidhuber, 1997). This is related to the many RNN layers

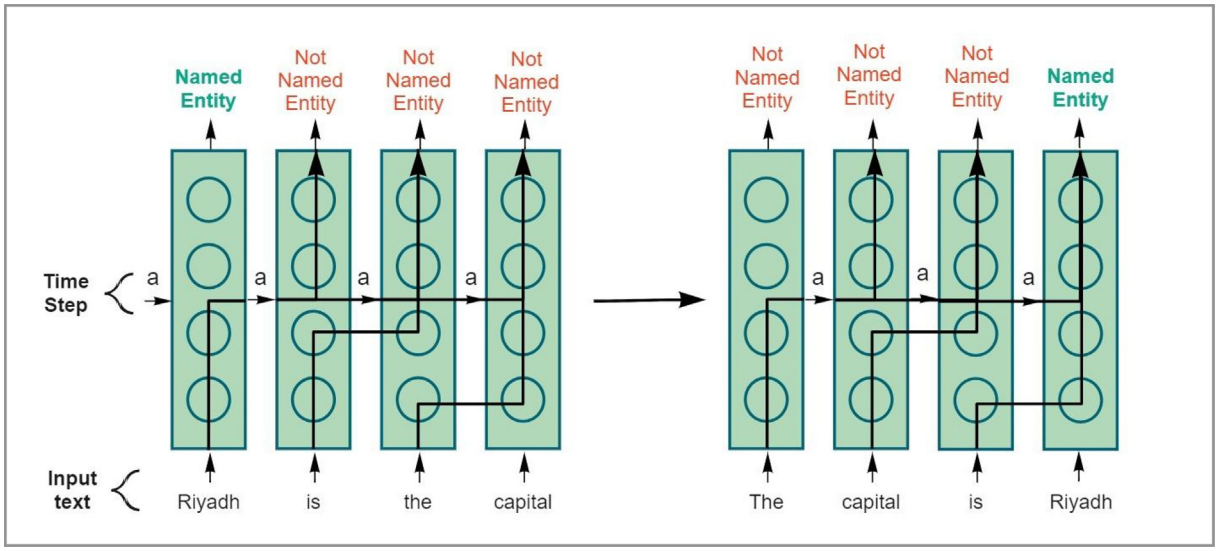

Fig. 1. Example of recurrent neural network.

and affects the prediction of words that depend on hidden states placed earlier in the input text. It is important to solve this problem, especially if the input text is long. Hence, both GRU and LSTM are frequently used in abstractive summarizers. The LSTM offers more memory for data control, making it easier to tune parameters and gain better accuracy (Min-Yuh et al., 2018).

2.3. Evaluation metrics

To evaluate new abstractive text summarization models, qualitative and quantitative approaches are usually used (Siyao Li et al., 2019). For the qualitative approach, humans read and evaluate the generated summaries (Shi et al., 2021). The humans are language experts or others whose dataset language is in their native language. For the quantitative approach, there are many evaluation metrics (Shi et al., 2021). The most popular and widely used are Recall-Oriented Understudy for Gisting Evaluation (ROUGE), bilingual evaluation understudy (BLUE), and Metric for Evaluation of Translation with Explicit ORdering (METEOR) (Keneshloo et al., 2019). ROUGE is a set of metrics measures that calculate the respective precision between the ground-truth text and the output produced by the summarization model (Lin, 2004). ROUGE's most-used metrics are ROUGE-N and ROUGE-L. ROUGE-N is the overlap of N-gram between the generated summary and reference. ROUGE-N is divided into ROUGE-1 reflect of unigram, and ROUGE-2 reflect of bigrams (Lin and Och, 2004). ROUGE-L refers to the longest common subsequence (Lin and Och, 2004). ROUGE and BLUE measures are similar; the only difference is that BLUE generates higher accuracy output, closer to a human's judgment (Keneshloo et al., 2019). METEOR measures precision and recall; however, it assigns a higher significance to recall over precision (Shi et al., 2021).

2.4. Challenges

Despite the success of RNNs in abstractive text summarization, they still suffer from the following problems:

- Syntactic issues: Syntactic information is used improperly in abstractive text summarization, which leads to incomplete sentences, repetitive words, and incoherent phrases (Yao et al., 2018; Yang et al., 2020). These issues are especially prominent in long summaries, which lead to an insufficient reflection of the main idea of the given text (Yang et al., 2020; Ibrahim et al., 2017).
- Vocabulary issues: Abstractive text summarization models face difficulties handling rare words because a word's importance is determined by its number of appearances, which is not the case from a human perspective. Additionally, the inability to cope with out-of-vocabulary (OOV) words prevents models from learning representations for novel vocabulary through training (Song et al., 2019).
- Exposure bias issue: At each time step during training, the decoder takes the word vector of the previous ground-truth word. However, in testing, the decoder receives the preceding word released by the model as input, leading to an exposure-bias problem (Yang et al., 2020).
- Differences between training and testing measurements: The models trained by the cross-entropy approach, which maximises the likelihood of the next token given the previous one, are evaluated by non-differentiable and discrete evaluation metrics (e.g., ROUGE) (Yang et al., 2020).

Therefore, recent studies of abstractive text summarization have focused primarily on tackling either one or multiple issues to enhance the accuracy of abstractive text summarization (Keneshloo et al., 2019). In this study, we focus on solving syntactic issues to enhance the accuracy of abstractive Arabic text summarization.

3. Related work

This section compares previous abstractive Arabic text summarisation studies. Recent studies show that the BERT (Bidirectional Encoder Representations from Transformers) model is a state-of-the-art method for abstractive text summarization, but the Arabic language has never been tested. Therefore, Elmadani et al. (2020) introduced the first abstractive Arabic summarization model that works with multilingual BERT. The authors want to prove how multilingual BERT can be effective in the case of a low-resource language, such as Arabic. The normal BERT is implemented using an encoder and a decoder. The encoder works by adding several symbols to learn sentence representations, while also using interval segmentation embeddings to distinguish between multiple sentences. The decoder works in six-layered transformers, which are initialized randomly. However, normal BERT can be applied only to the English language. Thus, the authors use multilingual BERT, which is similar to the normal one but has been trained in multiple languages. The model was trained and tested using the KALIMAT dataset, a Multipurpose Arabic Corpus containing 20,291 articles. However, the test results on the KALIMAT dataset generated accuracy, around 12.21%.

A few abstractive Arabic summarization models are introduced using attention-based features. Attention-based for Abstractive Arabic text summarization model was introduced for the first time by Yehia Khoja et al. (2017), The researchers introduced two abstractive summarizers to reflect the benefit of Attention-based for Abstractive Arabic text summarization. The first model built using a basic seq2seq summarizer model. The second model, build using a seq2seq summarizer with Attention-based. To evaluate the two models, a dataset of 30 K was utilized. The dataset is pairs of articles and headlines extracted from Saudi Newspapers Arabic Corpus. After evaluating the two models using BLUE, the researchers found that both can generate relevant headlines, but the summarizer with Attention-based can generate higher accuracy.

One of the models using attention-based is presented by Suleiman and Awajan (2020). The researchers proposed an architecture for abstractive Arabic text summarization consisting of two layers for the encoder and one layer for the decoder. Both layers for the encoder use bidirectional LSTM. The evaluation was conducted on a dataset deemed proper for abstractive Arabic summarization; it comprised various resources, such as Reuters, Aljazeera, and others. Each text in the dataset was paired with one summary, and the number of texts was 79,965. However, this evaluation, using the ROUGE measurement, generated an accuracy of 46.4%. Another is presented by Ouyang et al. (2019), who produce a summarization system for low-resource languages. They constructed a block abstractive cross-lingual summarization system that can summarise a document in a language available only in another language. The system works by deleting phrases that are difficult to translate, while generating new phrases in their place. The authors implement the system by translating and then summarising; the summarization part is a standard copy-attention summarizer, applied using a pointer-generator network. The system was trained and evaluated on four low-resource languages: Arabic, Somali, Swahili, and Tagalog. The experiments demonstrated that the model generates fluent summaries from noisy text. To evaluate the system in the Arabic language, they used the DUC 2004 Task 3 dataset, which consists of real-world Arabic news articles translated into English. The Arabic experiment generated an accuracy of 29.43% based on ROUGE measurements. The authors

claim that the results in Arabic ranked as the first in summarising machine-translated documents.

Moreover, there are another two studies presenting abstractive Arabic summarization models with Attention-based. The first study by Abdullah Alharbi et al. (Wazery et al., 2022), The researcher presents an Arabic summarization model with Attention-based. However, the aim of presenting this model is to investigate which one of the deep artificial neural networks performs better. Where they apply different layers of recurrent neural network gates to develop the encoder and decoder: GRU, LSTM, and Bidirectional Long Short-Term Memory (BiLSTM). Two datasets were utilised to conduct the experiment. First, is the Arabic Headline Summary (AHS) dataset, which contains 300 k Arabic articles and their titles. Second, is the Arabic Mogalad_Ndeef (AMN) dataset, which contains 265 k Arabic articles and their summaries. The AMN dataset was used in the study (Zaki et al., 2019). The experiment reveals that the summarizer applied BiLSTM gate performers better than the other two gates, where the highest result records 54.95% on the AHS dataset. The second study by Al-Maleh and Desouki (2020), the researchers aim to introduce a new Arabic dataset for Abstractive Arabic summarizers. The dataset is an Arabic dataset of summarised article headlines that consists of approximately 300 thousand articles. Then they apply two summarizers. The first, is an abstractive seq2seq model with the attention mechanism and coverage mechanism. The second, is an abstractive seq2seq model with the attention mechanism and copy mechanism. After applying the two models on the dataset, the second model records the highest accuracy, which is around 62%.

Another abstractive Arabic summarizer using an attention-based model is presented by Zaki et al. (2019), who focus on solving three abstractive summarization issues: OOV, exposure bias, and inconsistency between train/test measurements. The authors built three models for abstractive text summarization for the English and Arabic languages. Each is built to address one of the previously mentioned problems. To build the three models, the researchers first applied a multi-layer bidirectional seq2seq model with attention as a base for all three models. The three models are a pointer generator model, curriculum learning scheduled-sampling model, and policy-gradient model. To evaluate the three models, the researchers conducted two experiments—one for each language. In both languages, ROUGE was adopted to evaluate the three models. In the Arabic experiment, the researchers applied the models on the Arabic dataset that they collected, which consisted of 267 K Arabic news articles. Unexpectedly, the best accuracy was 60.70% from the basic model applied as a stone for the other three models, which is a multi-layer bidirectional seq2seq model with attention. Table 2 presents an overview of all the developed Arabic abstractive summarization mentioned in this subsection, alongside the ROUGE measures. The comparison between studies was based specifically on the ROUGE-1 metric because it was used in most of the studies.

According to Table 2, the multilingual BERT model had the lowest accuracy compared to attention-based models. Nevertheless, strangely, the attention-based models had significantly different levels of accuracy, ranging from around 30% to 60%. According to the ROUGE scores, the highest accuracy (60.7%) was achieved by Zaki et al. (2019), who developed three models based on a basic seq2seq model. Once again, the basic seq2seq model generated the most accurate summaries, but unusually, the authors claimed that the other two models—the scheduled-sampling and policy gradient models—generated better sentence quality. This indicates a need for further investigation into abstractive Arabic summarisation models and a new abstractive Arabic summariser that generates syntactically consistent sentences and high-quality summaries. This paper proposes a topic-aware abstractive Arabic summariser based on a seq2seq deep RNN and contributes to Arabic research in the NLP field.

4. TAAM architecture

The main task of TAAM is to generate a shorter Arabic text containing the main idea of the original long text. Basically, TAAM is a topic aware model that uses a many-to-many seq2seq architecture, which allows the model to process inputs of different lengths (Al-Saleh and Menai, 2016). The TAAM processes input sentences through four modules to generate output sentences (i.e., summaries) with lengths of 20–35 words. Each module consists

Table 2
Overview of abstractive Arabic summarisers.

Approach	Reference	Year	Framework	Dataset	ROUGE		
					R1	R2	RL
BERT	Elmadani et al. (2020)	2020	The encoder works by adding several symbols for learning sentence representations, while the decoder works with six-layer transformers that are initialised randomly.	KALIMAT dataset	12.21	4.36	12.19
Attention-based	Suleiman and Awajan (2020)	2020	The architecture consists of two layers for the encoder and one layer for the decoder. Both layers of the encoder use bidirectional LSTM: unidirectional LSTM and an attention mechanism.	79,965 Arabic documents	46.4	–	–
	Ouyang et al. (2019)	2019	The model uses a standard copy-attention summariser that applies a pointer generator network.	DUC 2004	29.43	7.02	19.89
	Wazery et al. (2022)	2022	an Arabic summarization model with Attention-based, applied with different recurrent neural network gates: GRU, LSTM, and BiLSTM.	265 k Arabic articles, and 300 k Arabic articles	54.95	13.1	37.84
	Al-Maleh and Desouki (2020)	2020	Two models; The first, is an abstractive seq2seq model with the attention mechanism and coverage mechanism. The second, is an abstractive seq2seq model with the attention mechanism and copy mechanism.	300 k Arabic articles	62.13	34.46	44.23
	Zaki et al. (2019)	2019	Three models based on multilayer bidirectional seq2seq with attention. The three models are a pointer generator model, a curriculum learning scheduled-sampling model, and a policy gradient model.	267 K Arabic news articles	60.79	41.28	50.08
Topic-awareness based	This paper	2022	A topic-aware abstractive Arabic summariser based on a deep RNN.	267 K Arabic news articles	**71.6**	58.6	70.1

of multiple layers to complete a certain task. Fig. 2 illustrates the architecture of the TAAM summariser model. First a word embedding module that converts the input text into many dimension vectors to enable the model to understand the meaning of words. Then simultaneously the dimension vectors of the input text enter the encoder module and topic aware module. The encoder module calculates the hidden state and the context vector of the input text. The encoder module consists of multiple RNN layers with LSTM gates that calculates the hidden state of the input text, where the context vector is created using an attention layer to pay attention to important words in the input text. On the other hand, the topic aware module uses an RNN topic classifier to establish more informative features of data called a guide vector. The guide vector helps to guide the model in a direction that reflects human expertise. There are three outputs of the encoder module and the topic aware module: the hidden state, the context vector, and the guide vector. Next all these three outputs transfer to the decoder module. The decoder uses these outputs to calculate the probability that each word will be in the produced summary. The decoder module consists of multiple RNN layers with LSTM gates. Eventually, the decode module utilises a beam search technique to choose the most appropriate output summary sentence. In this section, we explain and discuss each module in detail.

4.1. Word-embedding module

The first module consists of a word-embedding layer, the green box in Fig. 2, whose main task is to convert each word in the input sentences $X = [X_1, X_2, \cdots X_n]$, where n is the length of the input, (see Fig. 2) into an embedding vector that enables an abstractive summarizer to understand the meaning of words to generate a novel summary (Sun et al., 2020). Word embedding works by substituting each word in a word dictionary with a list of numbers; the generated list represents the syntax and semantics of words (Mikolov et al., 2013). In the RNN model, word embedding occurs prior to the text entering the encoder (Pennington et al., 2014). Following the study (Zaki et al., 2019), the Word Embedding module is built using a Word2Vec embedding algorithm that converts each word into a float numbered list depending on the relationships between words. All the lists are collected into one dictionary. Then the embedding words are inserted into the encoder layer.

4.2. Encoder module

The encoder module consists of two layers (the red box in Fig. 2). The first layer contains RNN layers that receive the embedding sentence from the word-embedding module as an input and then calculate the hidden state, $h = [h_1, h_2, \cdots h_n]$, for each word in the input, and send it to the decoder. Based on a study (Yang et al., 2020), the RNN layer uses bidirectional LSTM gates, which are beneficial for reading inputs from both sides of the input text. When processing an input, the RNN considers two objects: the time step and the output from the previous step (Fig. 3). The RNN layer of the TAAM depends on Equations (1) and (2).

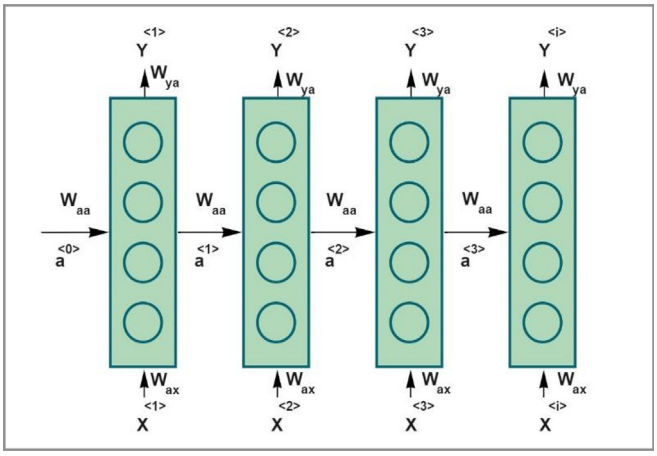

Fig. 3. Objects in the RNN layers.

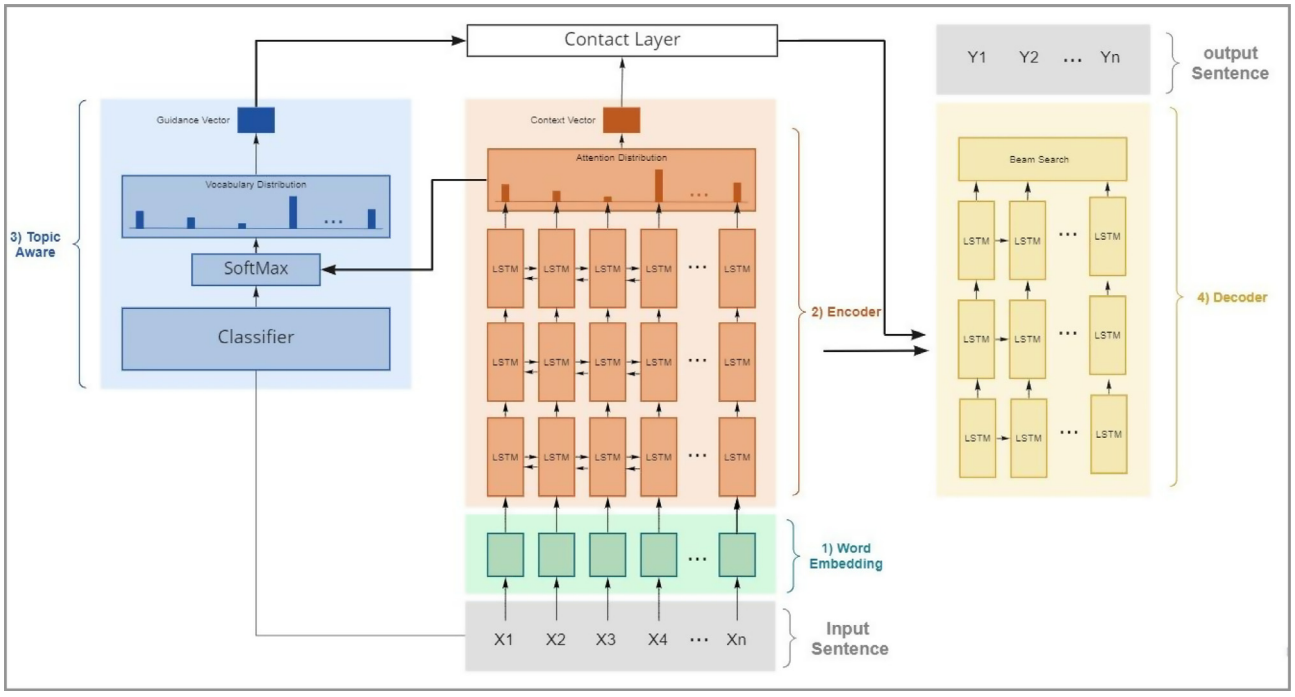

Fig. 2. The architecture of the topic-aware Arabic summariser.

$$a^{<i>} = g(W_{aa}a^{<i-1>} + W_{ax}X^{<i>} + b_a) \quad (1)$$

$$y^{<i>} = g(W_{ya}a^{<i-1>} + b_y) \quad (2)$$

Equation (1) calculates the next activation $a^{<i>}$ from the previous activation and input with bias b_a by utilising the following:

- W_{aa} is the weight of activation multiplied by the activations for $a^{<i-1>}$.
- $a^{<i-1>}$ is activation from the previous step.
- W_{ax} is the weight of activation multiplied by input $X^{<i>}$.
- $X^{<i>}$ is the word from the input sentence.
- g is the tanh activation function.

Equation (2) calculates the next output of time step $y^{<i>}$ from the previous activation with b_y bias by utilising the following:

- W_{ay} is the weight of activation multiplied by the output $y^{<i>}$.
- $a^{<i-1>}$ is activation from the previous step.
- g is the tanh activation function.

Following a study (Zaki et al., 2019) to build a deep RNN, we used a stack of RNN layers to obtain higher accuracy. Fig. 4 and Equation (3) show how the deep RNN works; for example, to calculate activation $a^{(2)<2>}$ at layer two, it uses the g tanh activation function, W_a weight of activation, activation $a^{(2)<1>}$ from the previous step at layer two, activation $a^{(1)<2>}$ at layer one, and b_a bias.

$$a^{(2)<2>} = g(W_a\left[a^{(2)<1>} + a^{[1]<2>}\right] + b_a) \quad (3)$$

Based on studies (Suleiman and Awajan, 2020; Yang et al., 2020; Zaki et al., 2019), the second layer is the attention distribution layer. The attention layer teaches the model to pay attention to important words in the input text rather than the whole text (i.e., important words and neighbouring words). The output of this layer is the context vector representing the amount of attention given to each word, context vector C^t is a weighted sum of the hidden states of the RNN-encoded input. Consequently, the encoder

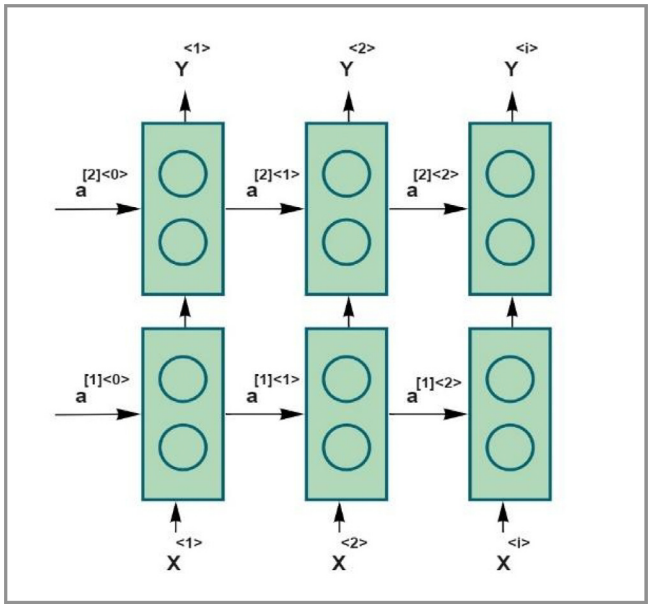

Fig. 4. Basic architecture of the deep RNN.

module sends two outputs to the decoder module: a hidden state h from the RNN layers, and a context vector from the attention layer C^t.

4.3. Topic-aware module

A topic-aware summarizer uses a classification model (classifier) to feed and guide the summarization model with topic information that allows the summarizer to pay more attention to the topic when generating a summary, which reflects human knowledge (Lee et al., 2011). To add a topic-aware layer to the proposed Arabic summarizer, we follow Yang et al. (2020) by adding a new guidance vector that creates new vocabulary distribution using topic information features. The topic information features will guide the decoder to place more attention on words belonging to the same topic, which results in a better summary. Following two previous studies (Kim et al., 2020; Yang et al., 2020), this module consists of three layers, which are in the blue box in Fig. 2: topic classifier, Softmax, and new vocabulary distribution. The module output is a guidance vector G as the representation of topics that provides attention to the words belonging to the same topic when generating a summary. This module works by first inputting a sentence into the topic classifier to generate topic information features. Then, the produced topic information inserts into a Softmax function to predict the category probability distribution. Softmax is often used as the last activation function of a neural network to normalise the network output to a probability distribution over predicted output classes (Yang et al., 2020). Finally, by the predicted output classes, a new vocabulary distribution is created, which constructs the guidance vector G. Fig. 5 shows the final contexts sent to the decoder, where C^t is the context vector from the encoder module and G is the guidance vector from this module.

4.4. Decoder module

The final module is the decoder shown in the yellow box in Fig. 2 that receives the three objects. The context vector and hidden states from the encoder module also use a guidance vector from the topic-aware module to finally calculate the probability of each word to generate a one-sentence summary. Based on the study (Yang et al., 2020), the decoder consists of RNN layers and a beam search. The RNN layers are constructed of stacked RNNs with LSTM gates that decode all three received objects to calculate the probability of each word. Then, to construct the summary output $Y = [Y_1, Y_2, \cdots . Y_n]$ (see Fig. 5), the beam search considers the most likely words according to the beam width.

To summarise the flow of the TAAM, first, the input articles are fed into the word-embedding module, which converts them to many dimensions embedding vectors. Then the encoder receives the embedding input to calculate the hidden states and the context vector following the same input articles fed into the topic-aware module that calculates the guidance vector. Finally, the three objects produced from the encoder and the topic-aware module guide the decoder to generate the final summaries.

5. Experiment setup

To build the proposed TAAM, this study followed three phases, as illustrated in Fig. 6. First is the data pre-processing phase, which involves two steps to prepare the dataset for the TAAM. The steps are regular text cleaning and Arabic text cleaning. Second is the data classification phase, which involves three steps to create the classifier that will be used to create topic information in the topic aware module of the TAAM (see Fig. 2). The steps are extracting topics, annotation, and classification. Third is the implementation

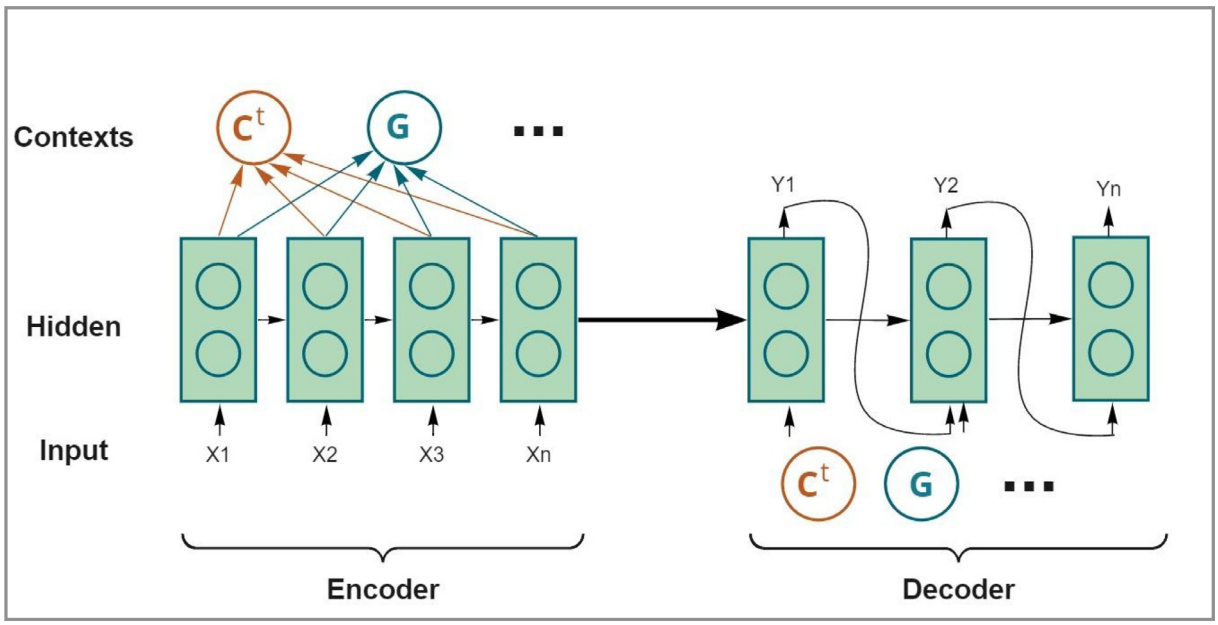

Fig. 5. The contexts sent to the decoder.

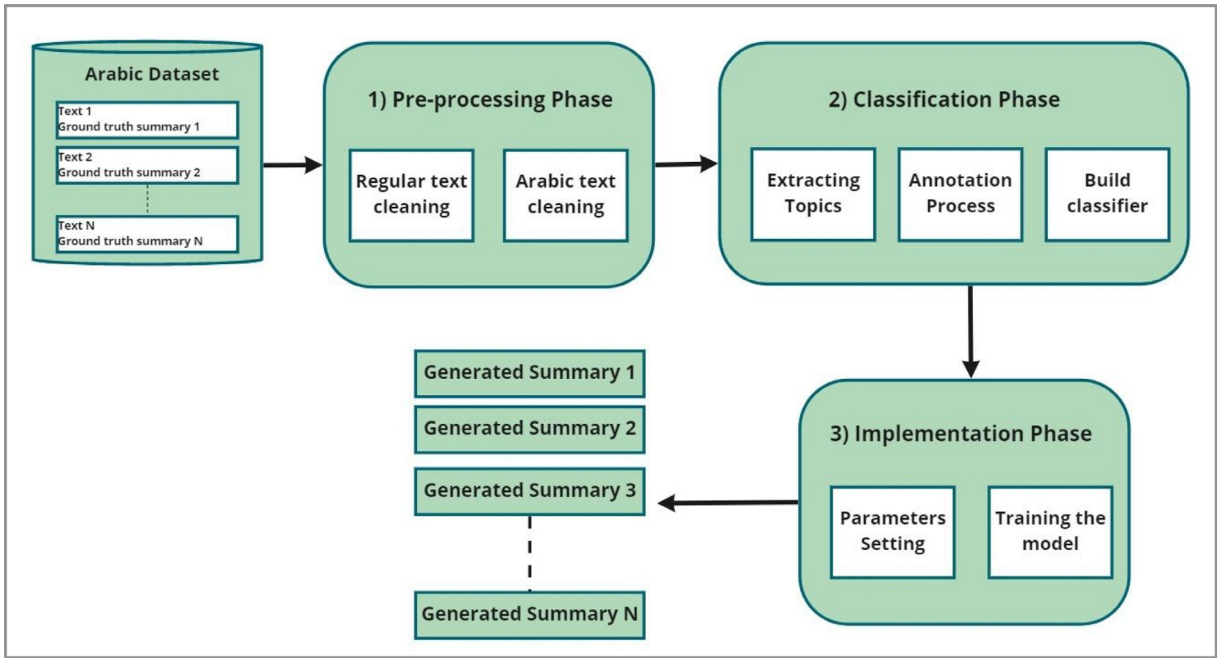

Fig. 6. Phases to build the proposed TAAM summarizer.

phase, i.e., the implementation of the four TAAM modules (see Fig. 2), which provides the parameter setting and training TAAM. This section starts by explaining the Arabic dataset utilised to train and test the TAAM. Then it discusses and explains, in detail, the three phases followed to build the TAAM.

5.1. Dataset

The dataset for this study is taken from a previous study conducted by Amr Zaki et al. (Zaki et al., 2019), who published an Arabic dataset for training and testing abstractive Arabic summarisation. The dataset is a large collection of 267,000 Arabic news articles, each associated with a sentence representing a ground truth summary. The articles and summaries are written in Modern Standard Arabic (MSA). Of the 267,000 news articles, comprising a total of 65,856,051 words, 236,000 relate to Arabic news and 31,000 specifically to Saudi news. The length of articles and summaries can affect the accuracy of generated summaries (Paulus et al., 2017), and Figs. 7 and 8 present the word distributions of the ground truth summaries and the articles across the dataset. As shown in Fig. 7, the number of words in each summary varied, with the average distribution ranging from 5 to 35 words per summary, and the longest summary being around 38 words. Fig. 8 shows that the number of words in each article also varied,

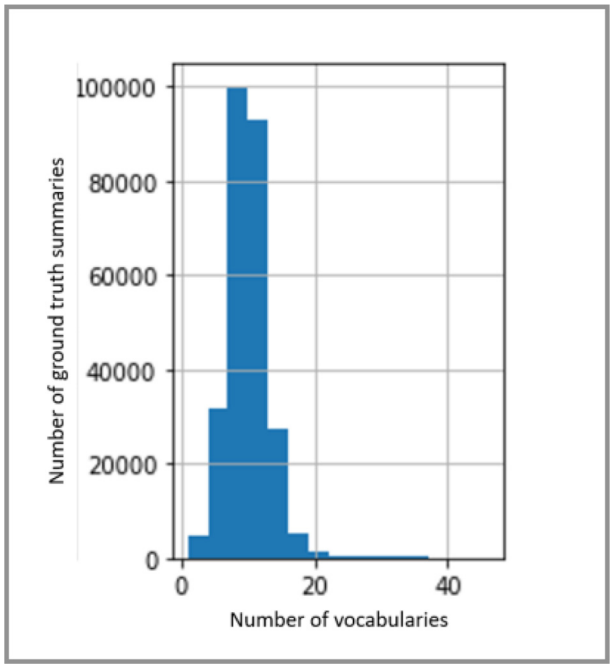

Fig. 7. Word distribution of the ground truth summaries.

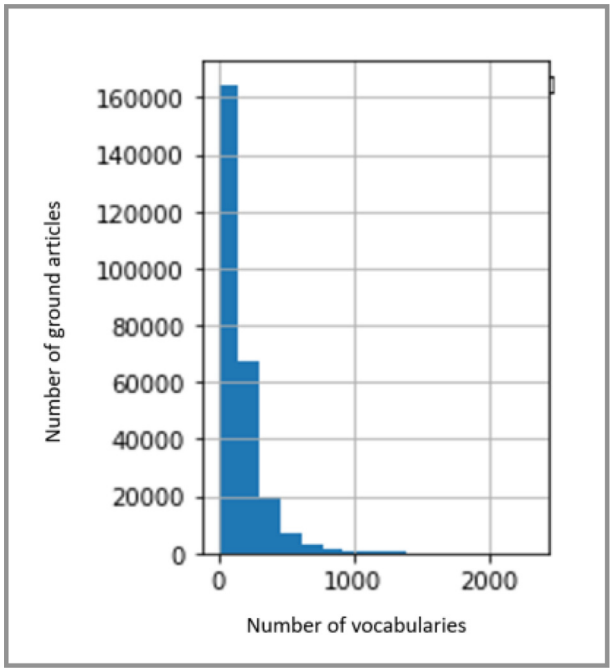

Fig. 8. Word distribution of the articles.

with an average distribution of 200–1,000 words per article, and the longest article being around 1,500 words.

5.2. Pre-processing phase

To pre-process the dataset, we use different machine learning algorithms. The pre-processing is completed in the following two steps:

5.2.1. Regular text cleaning

The data cleaning step aims to remove from the dataset all unwanted and noisy data that makes it ambiguous. In this step, the dataset is cleaned not only for the proposed model but also to make the texts clear for annotators in the data classification phase. In this step, the following are removed:

- Null values
- White space (extra spaces and new lines)
- Punctuation marks (! @#$\%^\&*()_+<>?:,; -}{؛ ، ؟)
- URLs
- Capital and small English letters

5.2.2. Arabic text cleaning

Advanced cleaning for the Arabic language is considered an essential step because the Arabic language is an agglutinative language, which means that words are composed of pieces that add to the meaning of the word (Zaki et al., 2019). Thus, this step aims to standardise all letters and words in the dataset through lemmatization and normalisation. The lemmatization approach is used to shrink words to a proper abstract form that is appropriate for the machine learning model (Paulus et al., 2017). For the lemmatization, the Farasa Library (Arabic segmentation) is utilised. The Farasa tool is a fast and accurate text-processing toolkit for Arabic text. The Farasa Library is chosen for lemmatization of the dataset because it has been proven to be significantly better than state-of-the-art Arabic segmentation tools such as Stanford and MADAMIRA at machine translation and information retrieval tasks (Paulus et al., 2017). On the other hand, normalisation is used to standardise the form of Arabic letters and words to be represented in one form without changing the meaning of the word (Lulu and Elnagar, 2018). Tashaphyne is an Arabic light stemmer and is segmented. It supports mainly light stemming (removing prefixes and suffixes) and provides all possible segmentations. Tashaphyne comes with default prefixes and suffixes and accepts the use of a customised prefixes and suffixes list, which allows it to handle more aspects and make customised stemmers without changing code. The dataset is normalised by adopting the Tashaphyne library in Python (Lulu and Elnagar, 2018). The normalisation process used in the study was introduced by Lulu and Elnagar (2018) and applied to normalise the following:

- Arabic diacritic format
- Letter elongation
- Repeated letters
- Three Arabic letters: Ha'a (ه), ya'a (ي), and Alef (ا)

Table 3 presents an example of article text of the dataset before and after each step in the pre-processing phase.

5.3. Classification phase

A topic-aware summarizer uses a classification model (classifier) to feed and guide the summarization model with topic information reflecting human knowledge to generate a summary (Lee et al., 2011). To create topic information, we need a classification model (classifier) to generate accurate topics for the articles. The topic classifier used in this study is a type of supervised model, which means the training data must be associated with predefined label "topics" to train the topic classifier (Oladipupo, 2010). Thus, this phase involves three steps to create the classifier that will be used later to construct the TAAM. The steps are extracting topics, annotation, and classification.

5.3.1. Extracting topics

The first step is extracting topics, which means defining the number and types of topics in the dataset. The latent Dirichlet allocation (LDA) algorithm was used to extract latent topics from the dataset. LDA automatically analyses text in a dataset to create

Table 3
Example Before and After Advanced Cleaning step.

Before the pre-processing phase
أعلن مركز توثيق تابع لمنظمة التحرير الفلسطينية اليوم الأحد في تقريره الشهري حول الأنتهاكات الإسرائيليه بحق الفلسطينين, أن شهداء أرتقوا علي أيدي قوات الاحتلال خلال شهر سبتمبر الماضي. و أوضح المركز حسبما ذكرت وكاله الأنباء الفلسطينية وفا, ان الشهداء من بينهم اطفال ارتقوا علي أيدي قو ات الأحتلال في الضفة الغربية و القدس وقطاع غزه خلال الشهر الماضي, حيث اعدم منهم بدم بارد علي الحواجز الاسرائيلية المنتشره في الضفه الغربيه و القدس, مشيرا الي أن عدد الشهداء ارتفع بعد مرور عام علي اندلاع الهبه الشعبيه في مطلع اكتوبر بينهم اطفال و لا زالت سلطات الاحتلال تحتجز جثامين الشهيدا بعد أن سلمت جثامين شهداء القدس القدامي.
After Regular Text Cleaning
أعلن مركز توثيق تابع لمنظمة التحرير الفلسطينية اليوم الأحد في تقريره الشهري حول الأنتهاكات الإسرائيليه بحق الفلسطينين, أن شهداء أرتقوا علي أيدي قوات الاحتلال خلال شهر سبتمبر الماضي. و أوضح المركز حسبما ذكرت وكاله الأنباء الفلسطينية وفا, ان الشهداء من بينهم اطفال ارتقوا علي أيدي قو ات الأحتلال في الضفة الغربية و القدس وقطاع غزه خلال الشهر الماضي, حيث اعدم منهم بدم بارد علي الحواجز الاسرائيلية المنتشره في الضفه الغربيه و القدس, مشيرا الي أن عدد الشهداء ارتفع بعد مرور عام علي اندلاع الهبه الشعبيه في مطلع اكتوبر بينهم اطفال و لا زالت سلطات الاحتلال تحتجز جثامين الشهيدا بعد أن سلمت جثامين شهداء القدس القدامي.
After Lemmatization
اعلن مركز توثيق تابع منظم تحرير فلسطينيه يوم احد في تقرير شهري حول انتهاك اسرائيليه حق الفلسطينين ان شهيد ارتقوا علي ايدي قوات احتلال خلال شهر سبتمبر ماضي و أوضح مركز حسبما ذكر ال الانباء فلسطينيه وفا ان شهيد من بين أطفال ارتقو ا علي ايدي قو ات احتلال في ضفه غربي و قدس قطاع غز خلال شهر ماضي حيث أعدم من دم بارد علي حواجز اسرائيليه منتشره في ضفه غربي و قدس مشير الي ان عدد شهيد ارتفع بعد مرور عام علي اندلاع الهب شعبيه في مطلع أكتوبر الي بين طفل و لا زال سلطة احتلال احتجز جثامين شهيد بعد ان سلم جثامين شهيد قدس قدام
After Normalization
اعلن مركز توثيق تابع منظم تحرير فلسطينيه يوم احد فى تقرير شهرى حو ل انتهاك اسرائيليه حق الفلسطينين ان شهيد ارتقو ا على ايدى قو ات احتلال خلال شهر سبتمبر ماضى و اوضح مركز حسبما ذكر ال الانباء فلسطينيه وفا ان شهيد من بين اطفال ارتقوا على ايدى قو ات احتلال فى ضفه غربى و قدس قطاع غز خلال شهر ماضى حيث اعدم من دم بارد على حو اجز اسرائيليه منتشره فى ضفه غربى و قدس مشير الى ان عد شهيد ارتفع بعد مرور عام على اندلاع الهب شعبيه فى مطلع اكتوبر الى شهيد بين طفل و لا زال سلطه احتلال احتجز جثامين شهيد بعد ان سلم جثامين شهيد قدس قدام

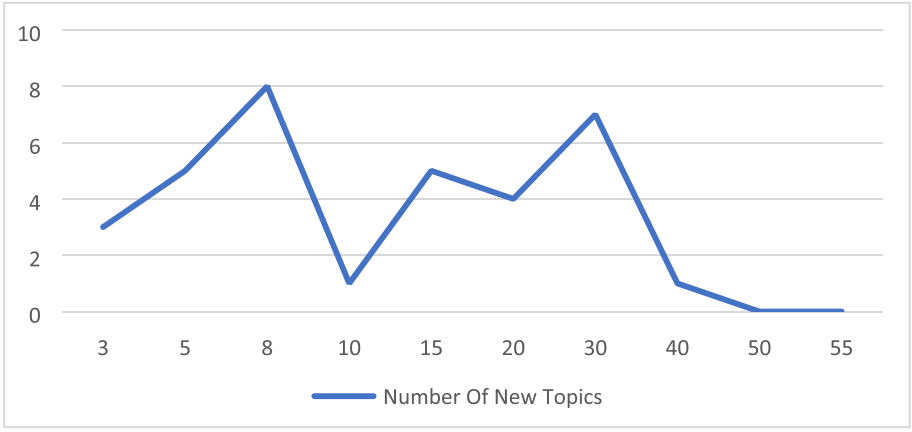

Fig. 9. Number of new topics exceeding the K number.

clusters of words, with each cluster representing a topic (Bastani et al., 2019). The LDA is considered a type of unsupervised machine learning, which does not require a predefined dataset to train a model (Bastani et al., 2019). To build an LDA model, one parameter must be defined as 'K,' which is the number of clusters the model should generate. Each cluster is then defined manually according to a certain topic (Bastani et al., 2019). Based on Kaveh Bastani et al.'s study (Bastani et al., 2019), we used trial and error procedures to find the K number. We observed that each time K increased, new topics appeared, so we kept increasing the K number until the words in the generated clusters were ambiguous and could no longer define new topics. Notably, the number of topics did not match the K number because some clusters did not represent a certain topic; we called them non-topic clusters. Fig. 9 shows the number of topics that appeared each time K increased. After K = 50 was reached, no new topics appeared; therefore, we stopped at K = 50, where the number of extracted topics reached 27, as follows: (1. Crimes, 2. Economy, 3. Accidents, 4. Conferences, 5. Art, 6. Court Cases, 7. Politics, 8. Celebrations, 9. Education, 10. AL sham Wars, 11. Health, 12. Egypt News, 13. Sport (Africa), 14. Technology, 15. Saudi Arabia News, 16. Yemen War, 17. Sport (Europe), 18. Stock Market, 19. General Information, 20. America News, 21. News for Women, 22. Sports (Saudi Arabia), 23. Weather, 24. Fashion, 25. Terrorism, 26. Farming, and 27. Tourism).

5.3.2. Annotation process

To train the topic classifier, an annotation process was applied to the dataset, which involved manually labelling the dataset according to the previously extracted 27 topics. The annotation process was carried out by volunteer annotators using an in-house labelling method. To ensure that the results of the annotation process were valid, the process was carried out twice to confirm the reliability of the selected annotators, each time with a different group of annotators. To determine the degree of agreement between the two groups of annotators, the Kappa coefficient was calculated; the value was 0.632 and the percentage of agreement was 74%, indicating moderate agreement according to (Landis and Koch, 1977). After the dataset was fully labelled, the data was grouped according to topic, with one topic for each group. Fig. 10 shows the distribution of the 27 topics across the dataset.

5.3.3. Build classifier

The final step involved applying multiple classification algorithms to the labelled dataset to find the best classification algorithm for building the classifier. Table 4 presents the results of the classification algorithms applied to the labelled 27-topic dataset according to the F1 score, recall, and precision. As shown, the classification algorithms were categorised into two types: traditional and deep learning. For each type, we applied the most frequently used algorithms according to the study (Raj and Rajesh, 2012). For the traditional type, we applied naive bayes, support vector machines, logistic regression, and stochastic gradient descent algorithms. For deep learning, we applied an RNN with LSTM gates. The dataset was split into 80% training data and 20% testing data. The highest accuracy for the 27-topic labelled dataset was generated by the RNN model with LSTM gates, which achieved 84.17% accuracy.

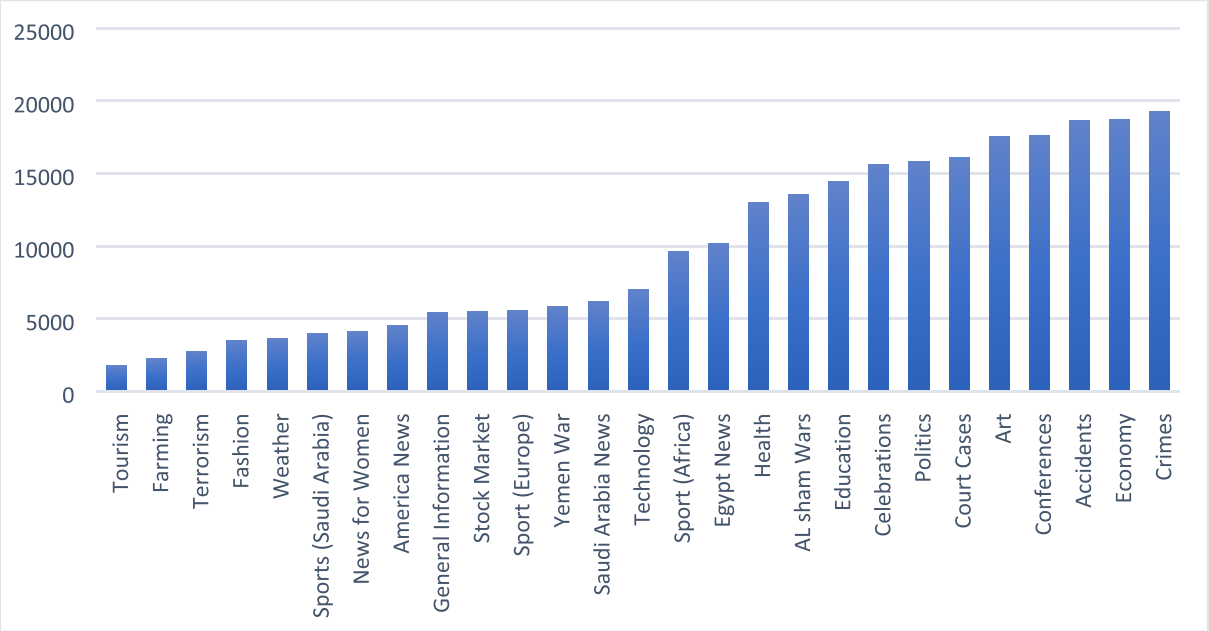

Fig. 10. Distribution of the 27 topics across the dataset.

Table 4
Results for the classification algorithms applied to the 27-topic labelled dataset.

Classification Algorithms	F1 Score	Recall	Precision
Traditional Algorithms			
Naive bayes	63.46	65.79	64.28
Support vector machine	69.60	70.17	71.48
Logistic regression	68.48	69.44	69.39
Stochastic gradient descent	69.79	70.59	70.37
Deep Learning Algorithm			
RNN	83.69	84.13	**84.17**

However, we noticed that several of the 27 topics in the dataset could be merged to obtain greater classification accuracy (e.g., the topics of terrorism and the Yemen and Sham Wars had similar vocabularies relating to war). Therefore, we merged the 27 topics into 14 new topics, as follows: (1. Economy, 2. Arab World News, 3. Wars, 4. Politician, 5. General Information, 6. Crimes, 7. Sports, 8. Accidents, 9. Conferences, 10. Art, 11. Court Cases, 12. Celebrations, 13. Education, and 14. Health). After the topics had been merged, we applied the same classification algorithms used for the 27-topic dataset to the new 14-topic labelled dataset. Table 5 presents the results of the classification algorithms applied to the 14-topic labelled dataset according to the F1 score, recall, and precision. The highest accuracy for the 14-topic labelled dataset was generated by the RNN model with LSTM gates, which achieved 87.72% accuracy. Finally, the classifier was built using the RNN model with LSTM gates and the full 14-topic labelled dataset because they generated the highest accuracy. The developed classifier was used as one layer for constructing the proposed TAAM summariser.

5.4. Implementation phase

We implemented the TAAM on the Google Collab server. The hardware was an NVIDIA-SMI GPU server, and 27.3 gigabytes of runtime were available on the RAM. We ran the code on Google Colab Jupyter notebook and used the Python 3.1 programming language. For the library we utilised the TensorFlow 1.2.0 deep learning framework to install the following packages: Keras, Layers, Models, and Callbacks. To construct the TAAM, we followed the settings given in (Zaki et al., 2019): a 300-dimensional embedding layer in Word2Vec, three stacked RNN layers in the encoder, a decoder with a hidden-state size of 256, and a beam search with a beamwidth of 3 during decoding.

In the training phase, based on two studies (Yang et al., 2020) (Zaki et al., 2019), the batch size was 64, the learning rate was 0.005, and we used 40 epochs. As per Section 5.1, the longest article in the dataset was around 1,500 words; therefore, we set the maximum word length for articles at 1,500 words to cover most articles in the dataset. Because the length of articles and summaries can affect the summaries' accuracy (Paulus et al., 2017), we trained the model multiple times, each time changing the maximum length for generated summaries to find the most suitable number of words. Based on the utilised dataset, the TAAM can generate a summary with lengths of 20–35 words; therefore, we trained the model four times for 20, 25, 30, and 35 words, respectively. We set the range at 20–35 because it was the range for the ground truth summaries in the dataset, as explained in Section 5.1. The TAAM trained on 236,020 samples (80% of the dataset), and the remaining 26,215 samples (10% of the dataset) were left for validation. In detail, the model trained on 42,661,006 trainable parameters, which took about 5.40 h. Regarding the model's performance, Fig. 11 shows the loss values for the training and testing. There were small differences between the training and testing values, and the loss value almost reached zero for the final training.

Table 5
Results for the classification algorithms applied to the 14-topic labelled dataset.

Classification Algorithms	F1 Score	Recall	Precision
Traditional Algorithms			
Naive bayes	76.81	77.05	77.41
Support vector machine	77.73	77.89	78.60
Logistic regression	76.81	77.05	77.41
Stochastic gradient descent	76.62	76.81	77.50
Deep Learning Algorithm			
RNN	85.02	87.13	**87.72**

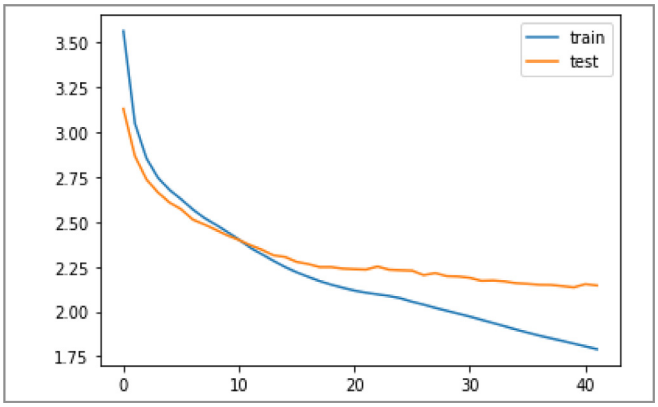

Fig. 11. Loss values for the training and testing.

6. Results and discussion

To evaluate the TAAM, we conducted two experiments: (1) a quantitative experiment using the most common automatic measure for abstractive text summarisation (ROUGE) (Al-Saleh and Menai, 2016) and (2) a qualitative experiment whereby volunteers manually evaluated a sample-generated summary. The following subsections discuss the two conducted experiments.

6.1. Quantitative results

We used ROUGE to evaluate the model automatically because it is widely employed in the abstractive text summarisation field (Al-Saleh and Menai, 2016). As explained previously, the TAAM can generate summaries with lengths ranging from 20 to 35 words. We trained the model four times on 20, 25, 30, and 35 words, respectively. Table 6 presents the results for the generated summaries according to the ROUGE-1, ROUGE-2, and ROUGE-L matrices. As shown, the most suitable number of words for the generated summary was 30, which achieved 71% accuracy according to ROUGE-1.

To compare the TAAM with other models, we chose the four models represented in the study (Zaki et al., 2019) as the baseline models for this study. We chose these models because they achieved the highest accuracy among the abstractive Arabic summarisers presented in Section 2. We also applied these four models to the same dataset that was used for the rest of this study. The four baseline models were as follows:

- **An attention-based seq2seq model**—a multilayer seq2seq bidirectional encoder decoder with attention functionality to pay attention to specific words.
- **A pointer generator model**, using extractive and abstractive approaches, which allows the model to copy words from input text through pointing while generating words from a fixed vocabulary.
- **A scheduled-sampling curriculum learning model**, trained by depending entirely on the ground truth, with the model trained by gradually introducing the error of the model to itself.
- **A policy-gradient** model that works by trying to minimise the loss function to optimise the randomly sampled output.

Table 6
ROUGE results for the generated summaries.

ROUGE	20 words	25 words	30 words	35 words
ROUGE-1	62.5	61.9	**71.6**	70.6
ROUGE-2	59.6	58.9	58.6	58.6
ROUGE-L	62.1	61.4	71.1	70.1

Table 7 presents the comparison between the four baseline models' and the TAAM's output accuracies according to the ROUGE-1, ROUGE-2, and ROUGE-L matrices. The results answered the first research question: 'Can a topic-aware summarisation model that employs a deep RNN enhance the accuracy of overall abstractive Arabic text summarisation?' As shown, the TAAM achieved the highest accuracy of the evaluated models according to the results.

6.2. Qualitative results

To bring a human perspective to the performance of the TAAM, we asked two Arabic-national volunteers to manually evaluate the generated summaries. The sample consisted of 5000 articles and their associated generated summaries. Similar to (Yang et al., 2020), where we randomly selected 358 summaries from each topic. Based on (Yang et al., 2020), the volunteers then evaluated each summary according to two criteria: 1) the relevance—how relevant the generated summary was to the article and 2) the readability—how easy it was to read the generated summary. They evaluated the relevance and readability of each generated summary by assigning a number from one to five, as follows: one (very bad), two (poor), three (fair), four (good), and five (excellent). We sent instruction guidelines with examples to the volunteers before the evaluation to ensure that they understood the evaluation process. The total average scores assigned by the two volunteers according to the two criteria are shown in Table 8. These results answered the second research question: 'Can a topic-aware summarisation model improve the quality of generated summaries in terms of readability and relevance?' The results revealed that the TAAM could generate summaries consisting of easy-to-read sentences that were relevant to the article texts.

6.3. Discussion

After observing the results from both quantitative and qualitative perspectives, we applied the ROUGE metrics to the same samples evaluated by the volunteers to examine the difference between the two methods. The highest accuracy, according to ROUGE, was 70.0, revealing a difference between the two methods. We believe that the main reason for this difference was that the ROUGE metrics were not flexible enough when comparing the output and the reference. ROUGE counts the overlaps of words between generated summaries and ground truth summaries without considering the importance of words or that there may

Table 7
Results for baseline models and the TAAM.

ROUGE	ROUGE-1	ROUGE-2	ROUGE-L
Attention-based seq2seq model	60.79	41.28	50.08
Pointer generator model	48.78	32.13	41.63
Scheduled-sampling curriculum learning model	52.97	36.68	45.82
Policy-gradient model	43.08	27.92	37.34
Topic-aware Arabic summariser model (TAAM)	**71.6**	58.6	70.1

Table 8
Total averages for the two volunteers.

Volunteers	Readability	Relevance
Volunteer 1	4.26	4.61
Volunteer 2	4.34	4.42
Total	**4.3 (86%)**	**4.5 (90%)**

Table 9
An example of a generated summary.

Example
Source text in Arabic:
اسفرت حمله مباحث التموين بالغربيه علي مدن ومراكز المحافظه امس السبت لضبط السلع المغشوشه و الفاسده بالاسو اق و المتلاعبين بالاسعار عن تحرير محضرا بمخالفات تموينيه متنوعه كان العميد احمد الخو اجه مدير مباحث التموين بالغربيه قد تلقي اخطارا من ضباط الاداره بنتائج الحمله و التي اسفرت عن ضبط مخالفه متنوعه شملت مخالفات البيع بازيد من التسعيره وعدم الاعلان عن الاسعار وتجميع خبز مدعم وتجميع خبز بلدي مجفف الي جانب ضبط سلع مجهو له المصدر ومغشوشه ومنتهيه الصلاحيه وتلاعب في الاوزان وتم تحرير المحاضر اللازمه لجميع المخالفين وجار اخطار النيابات المختصه للمراكز و الاقسام
Source text in English:
The campaign of Supply Investigations in Gharbia against the cities and centres of the governorate yesterday, Saturday, to seize fraudulent and corrupt goods in the market and price manipulators, resulted in the issuance of a report of various catering violations. Miscellaneous violations included selling more than the price, not declaring prices, assembling subsidized bread, assembling dried local bread, in addition to seizing goods of unknown origin, fraudulent and expired, and manipulation of weights. The necessary records have been edited for all violators, and the competent prosecution offices of the centres and departments are being notified.
Generated summary in Arabic:
التحقيق في التوريد بالغربية يضبط بضائع مزورة وفاسدة في السوق
Generated summary in English:
An investigation into the supply in Gharbia seizes counterfeit and corrupt goods in the market
Ground truth summary in Arabic:
ضبط مخالفه تسعيره وسلع مغشوشه وفاسده في حمله تموينيه بالغربيه
Ground truth summary in English:
A violation of its pricing and fraudulent and corrupt goods was caught in a catering campaign in Gharbia

be different words with the same meaning. Table 9 shows an example of an article paired with the generated summary and the ground truth summary, the text and the summaries in the example are translated from the Arabic language to the English language using Google translator. Humans can capture the same idea of an article from two sentences, but when ROUGE analyses the same two sentences, its accuracy is very low (around 12%) because the words in the two sentences are not the same; hence, from the perspective of ROUGE, the two sentences are different, which severely affects the accuracy of the abstractive text summarisation model. However, the results of TAAM can be improved by conducting more experiments that are dedicated to the effectiveness of changing parameters such as the batch size, the learning rate, the beam width, and the number of epochs.

7. Conclusion and future work

In this study, we propose TAAM; a new topic-aware abstractive summarization model for the Arabic language based on a deep recurrent neural network (RNN). The proposed model generates a summary containing words that do not exist in the input text by mimicking human knowledge and identifying topic information features, which generate better summaries.

The TAAM consists of four modules: 1) a word-embedding module that converts the input text into many dimension vectors to enable the model to understand the meaning of words; 2) an encoded module consisting of multiple RNN layers with LSTM gates to calculate a hidden state of the input, with an attention layer paying attention to important words in the input text; 3) a topic-aware module consisting of an RNN topic classifier that establishes more informative features of data, which help to guide the model in a direction that reflects human expertise and where the module creates a new guide vector for the decoder; and 4) a decoder module consisting of multiple RNN layers with LSTM gates to calculate the probability of each word, with a beam search to choose the most appropriate output sentence.

To construct the proposed TAAM, we followed three phases. First was pre-processing the utilized dataset. Second was constructing a classifier to generate topic information by applying multiple classification algorithms to a labelled dataset to select the most suitable classifier for generating topic information in the TAAM model. Third was the implementation of the proposed TAAM.

To evaluate the TAAM, we conducted two experiments: quantitative and qualitative. Based on ROUGE matrices, the quantitative approach revealed that the TAAM achieved 10.8% higher accuracy than the other baseline models. The qualitative approach recorded 86% for readability (How easy the summary can be read?) and 90% for relevance (How relevance the summary to the input text?), which reflected a human perspective. The qualitative approach showed that the model generated a coherent Arabic summary that is easy to read and captures the main idea of the input text. But there is a noticeable difference between the two evaluations. We believe that the main reason for this difference was that the ROUGE metrics were not flexible enough when comparing the output and the reference, because the ROUGE counts the overlaps of words between generated summaries and ground truth summaries without considering the importance of words or that there may be different words with the same meaning.

However, there are limitations to the TAAM model. First, the model was developed entirely on one dataset that contains 27 topics and uses a specific framework for the dataset. This limits the model capability because it can be generalised by applying the model on a various dataset that contains more topics. Second, as the TAAM architecture uses the seq2seq model, it takes a long period of time to train the model. Also, as the TAAM architecture uses Word2vec for word embedding, it represents words with multiple meanings that are conflated into a single representation, which affects the output summary. The TAAM architecture can be improved by using and testing other RNN models and word embedding techniques.

In our future work, we intend to apply additional abstractive text summarisation techniques to the TAAM to add more layers to the modules, including adding a pointer generator layer to the encoder module and BERT to the word-embedding module. In general, we aim to expand Arabic research in the automatic text summarisation field.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Abdolahi, M., Zahedh, M., 2017. Sentence matrix normalization using most likely n-grams vector. In: 2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation (KBEI), pp. 40–45.

Allahyari, M., Pouriyeh, S., Assefi, M., Safaei, S., Trippe, E. D., Gutierrez, J. B., & Kochut, K. (2017). Text summarization techniques: a brief survey. *ArXiv Preprint ArXiv:1707.02268*.

Al-Sabahi, K., Zuping, Z., & Kang, Y. (2018). Bidirectional attentional encoder-decoder model and bidirectional beam search for abstractive summarization. *ArXiv Preprint ArXiv:1809.06662*.

Al-Saleh, A.B., Menai, M.E.B., 2016. Automatic Arabic text summarization: a survey. Artif. Intell. Rev. 45 (2), 203–234.

Azmi, A.M., Altmami, N.I., 2018. An abstractive Arabic text summarizer with user controlled granularity. Inf. Process. Manage. 54 (6), 903–921.

D. Bahdanau K. Cho Y. Bengio Neural machine translation by jointly learning to align and translate ArXiv Preprint 2014 ArXiv:1409.0473.

Belkebir, R., Guessoum, A., 2018. TALAA-ATSF: a global operation-based arabic text summarization framework. In: Intelligent Natural Language Processing: Trends and Applications. Springer, pp. 435–459.

Bhat, I.K., Mohd, M., Hashmy, R., 2018. Sumitup: A hybrid single-document text summarizer. In: Soft computing: Theories and applications. Springer, pp. 619–634.

Chitrakala, S., Moratanch, N., Ramya, B., Raaj, C.G.R., Divya, B., 2016. In: Concept-based extractive text summarization using graph modelling and weighted iterative ranking. Communication and Applications, pp. 149–160.

Chowdhary, K.R., 2020. Natural language processing. In *Fundamentals of artificial intelligence*. In: Chowdhary, K.R. (Ed.), Fundamentals of Artificial Intelligence. Springer India, New Delhi, pp. 603–649.

Suleiman, D., Awajan, A., 2020. Deep learning based abstractive arabic text summarization using two layers encoder and one layer decoder. J. Theor. Appl. Inform. Technol. 98 (16).

Ermakova, L., Cossu, J.V., Mothe, J., 2019. A survey on evaluation of summarization methods. Inf. Process. Manage. 56 (5), 1794–1814.

Gigioli, P., Sagar, N., Rao, A., Voyles, J., 2018. Domain-aware abstractive text summarization for medical documents. IEEE International Conference on Bioinformatics and Biomedicine (BIBM) 2018, 2338–2343.

Giles, C.L., Kuhn, G.M., Williams, R.J., 1994. Dynamic recurrent neural networks: Theory and applications. IEEE Trans. Neural Networks 5 (2), 153–156.

Guo, Q., Huang, J., Xiong, N., Wang, P., 2019. MS-pointer network: Abstractive text summary based on multi-head self-attention. IEEE Access 7, 138603–138613.

Gupta, S., Gupta, S.K., 2019. Abstractive summarization: An overview of the state of the art. Expert Syst. Appl. 121, 49–65.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9 (8), 1735–1780.

Hou, L., Hu, P., Bei, C., 2017. Abstractive document summarization via neural model with joint attention. National CCF Conference on Natural Language Processing and Chinese Computing, 329–338.

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press.

Ibrahim, M.N., Maria, K.A., Jaber, K.M., 2017. A Comparative Study for Arabic Multi-Document Summarization Systems (AMD-SS). In: 2017 8th International Conference on Information Technology (ICIT), pp. 1013–1022.

Bastani, K., Namavari, H., Shaffer, J., 2019. Latent Dirichlet allocation (LDA) for topic modeling of the CFPB consumer complaints. Expert Syst. Appl. 127, 256–271.

Keneshloo, Y., Shi, T., Ramakrishnan, N., Reddy, C.K., 2019. Deep reinforcement learning for sequence-to-sequence models. IEEE Trans. Neural Networks Learn. Syst. 31 (7), 2469–2489.

Khalid Elmadani, Mukhtar Elgezouli, & Anas Showk. (2020). BERT Fine-tuning For Arabic Text Summarization. *ArXiv Preprint ArXiv:2004.14135*.

Khan, A., Salim, N., Farman, H., Khan, M., Jan, B., Ahmad, A., Ahmed, I., Paul, A., 2018. Abstractive text summarization based on improved semantic graph approach. Int. J. Parallel Prog. 46 (5), 992–1016.

S.-E. Kim N. Kaibalina S.-B. Park A Topical Category-Aware Neural Text Summarizer Applied Sciences 10 16 (20202). 5422.

Raj, K., Rajesh, V., 2012. Classification algorithms for data mining: A survey. International Journal of Innovations in Engineering and Technology (IJIET) 1 (2), 7–14.

Lee, K., Palsetia, D., Narayanan, R., Patwary, M.M.A., Agrawal, A., Choudhary, A., 2011. Twitter trending topic classification. In: 2011 IEEE 11th International Conference on Data Mining Workshops, pp. 251–258.

Lee, R.S.T. (Ed.), 2020. Artificial Intelligence in Daily Life. Springer Singapore, Singapore.

Lulu, L., Elnagar, A., 2018. Automatic Arabic Dialect Classification Using Deep Learning Models. Procedia Comput. Sci. 142, 262–269.

Li, Z., Peng, Z., Tang, S., Zhang, C., Ma, H., 2020. Text summarization method based on double attention pointer network. IEEE Access 8, 11279–11288.

Lin, C.-Y., & Och, F. J. (2004). Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, 605–612.

T. Mikolov K. Chen G. Corrado J. Dean Efficient estimation of word representations in vector space ArXiv Preprint 2013 ArXiv:1301.3781.

Min-Yuh, Chen, & Chao-Yu. (2018). *Artificial intelligence for automatic text summarization*. IEEE.

Mohammad, S. M. (2020). Examining citations of natural language processing literature. *ArXiv Preprint ArXiv:2005.00912*.

Al-Maleh, M., Desouki, S., 2020. Arabic text summarization using deep learning approach. J. Big Data, 7–109.

Ouyang, J., Song, B., & McKeown, K. (2019). A robust abstractive system for cross-lingual summarization. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2025–2031.

Paulus, R., Xiong, C., & Socher, R. (2017). A deep reinforced model for abstractive summarization. *ArXiv Preprint ArXiv:1705.04304*.

Pennington, J., Socher, R., Manning, C.D., 2014. Glove: Global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543.

Landis, J.R., Koch, G.G., 1977. The Measurement of Observer Agreement for Categorical Data. *Biometrics*. Biometrics 33 (1), 159.

Robinson, A.J., 1994. An application of recurrent nets to phone probability estimation. IEEE Trans. Neural Networks 5 (2), 298–305.

Rush, A. M., Chopra, S., & Weston, J. (2015). A neural attention model for abstractive sentence summarization. *ArXiv Preprint ArXiv:1509.00685*.

Shi, T., Keneshloo, Y., Ramakrishnan, N., Reddy, C.K., 2021. Neural abstractive text summarization with sequence-to-sequence models. ACM Trans. Data Sci. 2 (1), 1–37.

S. Li D. Lei P. Qin W. Yang . Wang1. Deep reinforcement learning with distributional semantic rewards for abstractive summarization. ArXiv Preprint ArXiv:1909.00141 2019.

Song, S., Huang, H., Ruan, T., 2019. Abstractive text summarization using LSTM-CNN based deep learning. Multimedia Tools and Applications 78 (1), 857–875.

Sun, C., Lv, L., Tian, G., Wang, Q., Zhang, X., Guo, L., 2020. Leverage Label and Word Embedding for Semantic Sparse Web Service Discovery. Math. Problems Eng. 2020, 1–8.

Oladipupo, T., 2010. Types of machine learning algorithms Vol. 3, 19–48.

Top Ten Languages Used in the Web. Available. (n.d.).

Wang, L., Yao, J., Tao, Y., Zhong, L., Liu, W., Du, Q., 2018. A reinforced topic-aware convolutional sequence-to-sequence model for abstractive text summarization. ArXiv Preprint. ArXiv:1805.03616.

Yang, M., Wang, X., Lu, Y., Lv, J., Shen, Y., Li, C., 2020. Plausibility-promoting generative adversarial network for abstractive text summarization with multi-task constraint. Inf. Sci. 521, 46–61.

Yao, K., Zhang, L., Du, D., Luo, T., Tao, L., Wu, Y., 2018. Dual encoding for abstractive text summarization. IEEE Trans. Cybern. 50 (3), 985–996.

Wazery, Y.M., Saleh, M.E., Alharbi, A., Ali, A.A., Khalil, A.M., 2022. Abstractive Arabic Text Summarization Based on Deep Learning. Comput. Intell. Neurosci. 2022, 1–14.

Zaki, A.M., Khalil, M.I., Abbas, H.M., 2019. Deep architectures for abstractive text summarization in multiple languages. In: 2019 14th International Conference on Computer Engineering and Systems (ICCES), pp. 22–27.