Journal of King Saud University – Computer and Information Sciences 35 (2023) 101750

HOSTED BY

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

A performance analysis of transformer-based deep learning models for Arabic image captioning

Ashwaq Alsayed, Thamir M. Qadah *, Muhammad Arif

Computer Science Department, College of Computer and Information Systems, Umm Al-Qura University, Makkah, Saudi Arabia

ARTICLE INFO

Article history:
Received 16 May 2023
Revised 14 August 2023
Accepted 7 September 2023
Available online 14 September 2023

Keywords:
Image captioning
Arabic image captioning
Transformer model
Performance analysis and evaluation
Deep learning
Machine learning
Arabic technologies

ABSTRACT

Image captioning has become a fundamental operation that allows the automatic generation of text descriptions of images. However, most existing work focused on performing the image captioning task in English, and only a few proposals exist that address the image captioning task in Arabic. This paper focuses on understanding the factors that affect the performance of machine learning models performing Arabic image captioning (AIC). In particular, we focus on transformer-based models for AIC and study the impact of various text-preprocessing methods: CAMeL Tools, ArabertPreprocessor, and Stanza. Our study shows that using CAMeL Tools to preprocess text labels improves the AIC performance by up to 34–92% in the BLEU-4 score. In addition, we study the impact of image recognition models. Our results show that ResNet152 is better than EfficientNet-B0 and can improve BLEU scores performance by 9–11%. Furthermore, we investigate the impact of different datasets on the overall AIC performance and build an extended version of the Arabic Flickr8k dataset. Using the extended version improves the BLEU-4 score of the AIC model by up to 148%. Finally, utilizing our results, we build a model that significantly outperforms the state-of-the-art proposals in AIC by up to 196–379% in the BLUE-4 score.
© 2023 The Author(s). Published by Elsevier B.V. on behalf of King Saud University. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

1. Introduction

Image captioning is becoming increasingly important as visual media is created at an unprecedented rate on the web and social media platforms. Research on image captioning targeting English captions has received significant attention in the past decade (Kiros et al., 2014; Xu et al., 2015; Sharma et al., 2018; Cornia et al., 2020; Dubey et al., 2023). Unfortunately, image captioning for the Arabic language did not receive similar attention, and few existing works focused on Arabic image captioning (AIC). To solve the AIC problem, existing models developed for English can be used to generate captions in English, then use a text-to-text language translation model to get the captions in Arabic. However, there are many drawbacks to this approach. First, captioning inaccuracies may result from different sources, namely the captioning and translation models. In addition, using the model can be slower because it performs two steps to get the Arabic text from the image. Furthermore, generating captions in Arabic faces many challenges. The Arabic language has a vast morphology (22,400 tags compared to 48 tags in English) (Habash, 2010). Moreover, Arabic has a fairly complex inflectional system. For example, Arabic verbs have over 5,400 different inflected forms. Thus, generating meaningful Arabic captions that are grammatically correct is challenging. In addition to the challenges that stem from the inherent properties of the Arabic language, we identify three factors that impact machine learning models used for AIC. These factors are (1) the labels' textual preprocessing techniques, (2) the datasets used for building AIC models, and (3) the techniques used to extract the features from the input images to generate the output captions. In this paper, we give an in-depth performance analysis of these factors. We focus our investigation on transformer-based image captioning models. Inspired by recent advances in machine translation (e.g., (Bahdanau et al., 2014; Sutskever et al., 2014; Cho et al., 2014)), deep learning models that "translates" images to text descriptions (captions) of the given images are proposed (Xu et al., 2015; Vinyals et al., 2015; Chu et al., 2020; Sharma et al., 2018; Li et al., 2019; Cornia et al., 2020; Luo et al., 2021; Zhou et al., 2021). Image captioning involves translating an image into a human-understandable caption consisting of one or more sentences. The following equation can describe this process.

$$s = translate(m, i) \quad (1)$$

* Corresponding author.
E-mail addresses: s44280168@st.uqu.edu.sa (A. Alsayed), tmqadah@uqu.edu.sa (T.M. Qadah), mahamid@uqu.edu.sa (M. Arif).
URL: https://thamir.qadah.com (T.M. Qadah).

https://doi.org/10.1016/j.jksuci.2023.101750
1319-1578/© 2023 The Author(s). Published by Elsevier B.V. on behalf of King Saud University.
This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

In the above equation, m represents the model that performs the translation process, $i = (p_1, \ldots, p_n)$ is the input image represented as a sequence of pixels, and $s = (w_1, \ldots, w_m)$ is the output sequence of words. The main contributions of this paper are summarized as follows:

1. We propose a transformer-based encoder-decoder model on AIC that uses pluggable image feature-extraction models. We focus on studying two models, namely, ResNet152 (He et al., 2016) and EfficientNet-B0 (Tan and Le, 2019), and use these models as feature extractors for the images. Our results show that the feature extractors have a significant impact on the performance results of AIC models.
2. We build an extended dataset based on the Arabic version of the Flickr8k dataset (Hodosh et al., 2013). The main feature of our extended dataset (Flickr8k-Rev) is that it has *five captions for each image* created and validated by native Arabic speakers. The total number of labels in the extended training dataset is 32465. This dataset will be made available to the research community in AIC.
3. We study the impact of three preprocessing tools: the CAMeL Tools (Obeid et al., 2020), the ArabertPreprocessor (Antoun et al., 2020), and Stanza (Qi et al., 2020). Our study shows that preprocessing of textual labels in different ways can have a significant impact on the performance of the models.
4. We performed extensive experiments. Our results show that using a carefully handwritten dataset and applying certain preprocessing steps with CAMeL Tools improves the AIC performance by up to 85% and 379% in the BLEU-1 and BLEU-4 scores, respectively, over existing proposals in the literature.

The remainder of this paper is organized as follows. Section 2 highlights the related work. Section 3 gives an overview of the transformer-based model used for AIC in this paper. Section 4 describes the general preprocessing pipeline for AIC. Section 5 gives an overview of Arabic preprocessing tools. Section 6 presents the details of the datasets, including our newly created extended dataset. Section 7 provides details of our prototype implementation. Section 8 presents our performance analysis that impacts different configurations of AIC models. Section 9 concludes the paper and gives future directions.

2. Related work

This section with give a brief overview of recent work in image captioning (IC). It uses techniques from Natural Language Processing and Computer Vision to understand the semantics of the image and create precise and accurate descriptions of that image. We focus our discussion on related works that use deep-learning techniques for IC. Recurrent neural networks (RNNs) and long short-term memory (LSTMs) models are used for image captioning in English (e.g., Vinyals et al., 2015 and Xu et al., 2015). Vinyals et al. (2015) present Neural Image Caption (NIC), based on a convolution neural network (CNN) that encodes an image into a compact representation; it is fed to the LSTM to generate a corresponding sentence by maximizing the likelihood of the sentence. Xu et al. (2015) uses a similar approach but incorporates an attention mechanism over the image. Unlike Xu et al. (2015) and Vinyals et al. (2015), we use a transformer-based model for image captioning in Arabic.

CNN-based feature extraction. In addition, some existing work studied various versions of residual networks as feature extractors for English image captioning, such as ResNet50, ResNet101, and Resnet152. Chu et al. (2020) propose an encoder-decoder model based on ResNet50 and LSTM with soft attention. ResNet50, a CNN-based encoder, uses a fixed-length vector to build a thorough representation of a given image. The decoder uses LSTM, and a soft attention mechanism to selectively focus attention over key sections of an image to predict the following sentence. Atliha and Šešok (2020) implement Show, Attend, and Tell architecture to compare two different image encoders such as VGG19 and ResNet101. As a result, the model with a ResNet101 encoder outperformed the model with a VGG19 encoder by 0.008 in the BLEU-4 score. Cankocagil, 2023 investigate an encoder-decoder framework. The encoder employs CNN to encode images into latent space representations. They examine ResNet152, AlexNet, VGG19-Net, DenseNet, and SqueezeNet as the encoder. The decoder utilizes RNN models to decode feature representations and build language models. Specifically, LSTM and Gated Recurrent Unit (GRU) are used as RNN models with an attention mechanism and teacher forcer algorithm. After such experiments, ResNet152 for the encoder and GRU for the decoder performed better. In this paper, we use two different models (ResNet152 and EfficientNet-B0) for feature extraction and study their impact.

Transformer Models. Proposed works utilizing the transformer model Vaswani et al. (2017) are also proposed. Sharma et al. (2018) suggest using it for English image captioning. They used Inception-ResNet0-v2 for image feature extraction and a transformer for the sequence modeling. The multi-modal Transformer architectures have been studied extensively more recently Li et al. (2019); Cornia et al. (2020); Luo et al. (2021) to implicitly capture visual correlations in image captioning tasks. Cornia et al. (2020) propose Meshed-Memory Transformer; unlike the original transformer, a memory vector is included in the encoder, and a mesh-like structure connects the encoder and the decoder. In order to achieve a better balance between speed and quality, Zhou et al. (2021) develop a semi-autoregressive model for image captioning (SATIC) on the MSCOCO dataset. This model maintains the autoregressive behavior globally but creates words parallelly locally. Luo et al. (2021) developed a hybrid technique that exploits the complementing advantages of region and grid features by combining them. Two self-attention modules are applied to each feature type separately, and a cross-attention module merges their interactions locally. In contrast to the proposals above, our study focuses on Arabic image captioning.

Arabic Image Captioning. Only a few studies tackled the challenge of image captioning in Arabic. In Jindal (2017), Jindal proposes a three steps root word based for automatic AIC generation. First, the proposal fragments the image and maps image fragments onto root words in Arabic deep belief networks pre-trained by Restricted Boltzmann Machines. Then, find the most acceptable words for an image by selecting a set of vowels that must be added to the root words. Finally, Arabic captions are built using the dependency tree relationships between these words. In Jindal (2018), Jindal proposes the idea of replacing the deep belief network with LSTM. BLEU-n is used for evaluation, and the experimental results show promising performance. However, it differs from ours in architecture as it is compositional, which consists of several independent functional building blocks.

The encoder-decoder architecture is used by many existing proposals (e.g., Al-Muzaini et al., 2018; Mualla and Alkheir, 2018; ElJundi et al., 2020; and Hejazi and Shaalan, 2021). Typically, pre-trained image models are used to extract image features. Then, these image features are fed to an LSTM network along with image descriptions. Unfortunately, the datasets used by Al-Muzaini et al. (2018) and Mualla and Alkheir (2018) are not publicly available. In contrast, ElJundi et al. (2020) provide the first significant Arabic captioning dataset to the public. However, unlike us, their proposal involved using an LSTM network. Hejazi and Shaalan (2021) evaluated LSTM and GRU for sequence modeling, Inception V3 and VGG16 for image feature extraction, with and without dropout,

and four methods for Arabic text preprocessing and noted that the use of text preprocessing and VGG16 improved the quality of descriptions. Unlike Hejazi and Shaalan (2021), this paper uses state-of-the-art text-processing frameworks and tools which involve additional preprocessing functions such as tokenization. Moreover, we use state-of-the-art image feature extraction models such as ResNet152 and EfficientNet-B0. We also use two different datasets for studying and evaluating the AIC models.

Emami et al. (2022) focus on Arabic image captioning, utilizing pre-trained bidirectional transformers. They use the X152-C4 for feature extraction and demonstrate that it is possible to obtain state-of-the-art outcomes with a minimal preprocessing strategy and by converting English captioning models to other languages using publicly available dataset benchmarks. Similar to them, we also use transformer-based models for AIC. However, in contrast, we study different preprocessing mechanisms for Arabic text labels. The work by Emami et al. (2022) is limited to task-specific preprocessing, which adds start and end tokens to the sentence and performs simple tokenization of sentences. We explain the details of the task-specific preprocessing in Section 4.

The generated captions are typically evaluated using the BLEU-n score. The state-of-the-art score for the root word-based is Jindal (2018) BLEU-1 score of 65.8, and for the encoder-decoder approach is Al-Muzaini et al. (2018) BLEU-1 score of 46. However, neither the code nor the datasets are available to the public, making it difficult to reproduce and validate their results. The dataset and source code of ElJundi et al. (2020) are both open-sourced, indicating that their work is repeatable and may be used as a baseline for future study.

AIC is still considered an unexplored research territory with few proposals addressing its challenges. One of the main issues is the lack of decent datasets, which we address in this paper by contributing a new curated dataset. Furthermore, unlike previous work, we systematically study AIC performance.

3. A Transformer-based Architecture for AIC

Proposed by Vaswani et al. (2017), the transformer model is one of the deep-learning models that can be used for machine translation. It has been used in a wide range of NLP applications Wang et al. (2019); Raganato and Tiedemann (2018). In addition to NLP applications, the transformer model has also been used for image captioning Sharma et al. (2018). We propose a variant based on Sharma et al. (2018).

Our proposed transformer-based architecture is shown in Fig. 1. The input image is translated into a vector with feature embedding using a CNN-based model. Then, the transformer encoder takes in the feature embedding of the image extracted by CNN, $x = (x_1, \ldots, x_n)$. Thus, the image's feature vector x is computed based on Eq. 2.

$$x = CNN(i) \tag{2}$$

$CNN(i)$ in Eq. 2 represents an abstraction over any CNN-based model that is used for image feature extraction.

After computing x, x is used as input to the encoder block, which computes z according to Eq. 3.

$$z = Encoder(x) \tag{3}$$

$Encoder(x)$ in Eq. 3 represents an abstraction of the set of encoder blocks. The encoder block generally creates an attention-based representation that can find a particular piece of data within a context that could be "infinitely" huge. The encoder is composed of one or more identical layers in a stack. Each layer has a simple position-wise, completely connected feed-forward network, and a multi-head self-attention layer. Each sub-layer adopts layer normalization and a residual connection. The same dimension of data is output by each sublayer. In our architecture, the encoder block with multi-head attention converts x to the attended visual representation vector followed by feed-forward, which is applied to every attention vector to transform it into a format that the following (encoder or decoder) layer can understand, $z = (z_1, \ldots, z_t)$. z is the output vector of the encoder block and is fed to the decoder block.

$$y = Decoder(z, s) \tag{4}$$

$Decoder(z, s)$ in Eq. 4 represents an abstraction of the set of encoder blocks. In our architecture, the decoder block uses z along with the word embedding vector, s, of the input image's captions to generate the output sequence $y = (y_1, \ldots, y_m)$. Word embedding is improved by positional encodings to take advantage of order information. y is the output sequence of words of the decoder block according to Eq. 4.

In general, the decoder is composed of one or more identical layers in a stack. Each layer has one sub-layer of a fully connected feed-forward network and two sub-layers of multi-head attention mechanisms. Each sub-layer adopts a residual connection and a layer normalization, just like the encoder. The initial multi-head attention sub-layer is adjusted to avoid positions from paying attention to subsequent positions so that the future of the target sequence is hidden when predicting the current position.

The word embedding goes through a masked multi-head attention mechanism which is different from a typical multi-head attention mechanism. The decoder block generates the sequence word by word, so it needs to prevent it from conditioning to future tokens. A second multi-headed attention layer takes the outputs of the encoder and the masked multi-headed attention layer to match the encoder's input to the decoder's input, enabling the decoder to select the encoder input that should receive the most attention. Then the output passes through a point-wise feed-forward layer.

4. Text Preprocessing for AIC

In this section, we shed light on an important task which is preparing and preprocessing the dataset for model training and testing. While being a crucial element of any AIC machine-learning pipeline, details of the preprocessing step are often not discussed in the literature. In this paper, we study various techniques for text prepossessing for AIC.

Most of the existing work in AIC preprocesses the Arabic text using standard preprocessing similar to English text. We refer to this as *general preprocessing*. They segment the labels using a segmentation tool used for the English language. This approach may result in semantic errors in training labels. For example, the definite article "ال" (equivalent to **the** in English) is frequently prefixed to other words. It is not an integral element of that word. For instance, both "حاسوب" and "الحاسوب" (with the "ال" prefix) must be included in the vocabulary, resulting in a substantial quantity of redundancy. To solve this problem, we segment the Arabic labels into their component prefixes, stems, and suffixes, along with other preprocessing steps. In this paper, we develop a unified framework for preprocessing labels in AIC. Then, we use our framework to investigate the effect of using different approaches (i.e., CAMeL Tools Obeid et al., 2020, ArabertPreprocessor Antoun et al., 2020, and Stanza Qi et al., 2020) for preprocessing Arabic text labels on the performance of AIC.

In general, image captioning models take text labels along with images as input to train the model, and this text must be preprocessed and cleaned to reduce the text's noise and vocabulary size. We propose a framework that abstracts the preprocessing steps

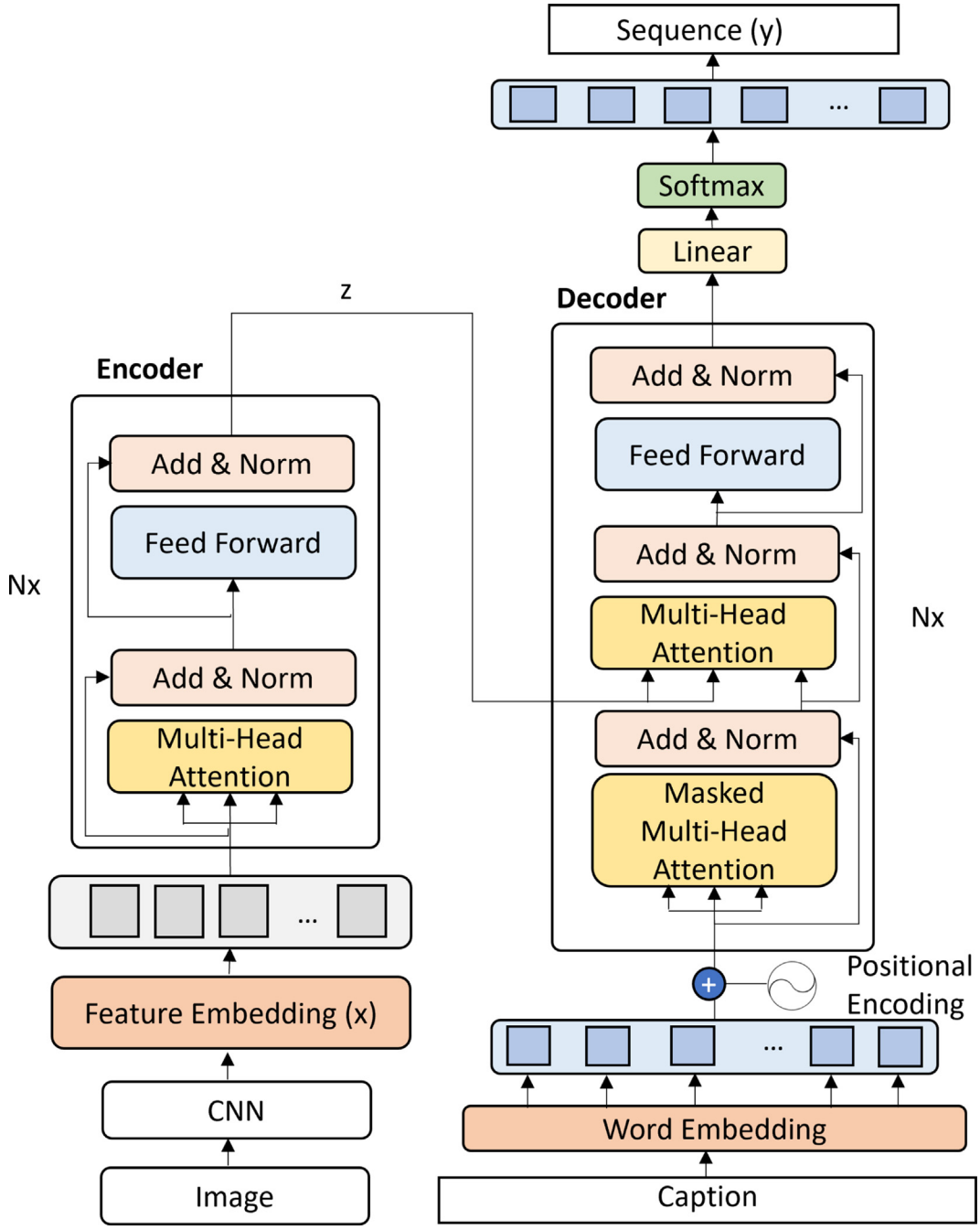

Fig. 1. The encoder and decoder in the transformer model architecture are shown on the left and right. Each encoder contains two sublayers, feed forward neural network and self-attention. Therefore, the feed-forward layer is applied after the self-attention layer. The decoder includes the same layers as the encoder and a third layer that sits between the two that were just mentioned; this layer is known as the encoder–decoder attention to help the decoder to focus on relevant parts of the input sentence.

into three stages. These stages are general preprocessing (GPP), language-specific preprocessing (LSPP), and task-specific preprocessing (TSPP). It is illustrated in Fig. 2 and explained in detail below.

4.1. General preprocessing

General text preprocessing is a standard step used for text preprocessing in NLP and AI applications. It is suitable for most languages, like lower-casing all the words, removing punctuation, removing URLs, removing stop words, and removing numeric. Some general preprocessing steps are applied, such as eliminating punctuation and numeric, as shown in the first part (General Preprocessing) of Fig. 2.

4.2. Language-specific preprocessing

Language-specific preprocessing comprises steps that are specific to a language. In our case, the language is Arabic, as shown in the second part of Fig. 2. However, our framework can be used for other languages as well. The morphology of Arabic is especially significant because the language is highly ordered and derivational. Arabic is written from right to left. The Arabic alphabet has 28 letters in total, including Hamza (ء), and Arabic letters have various

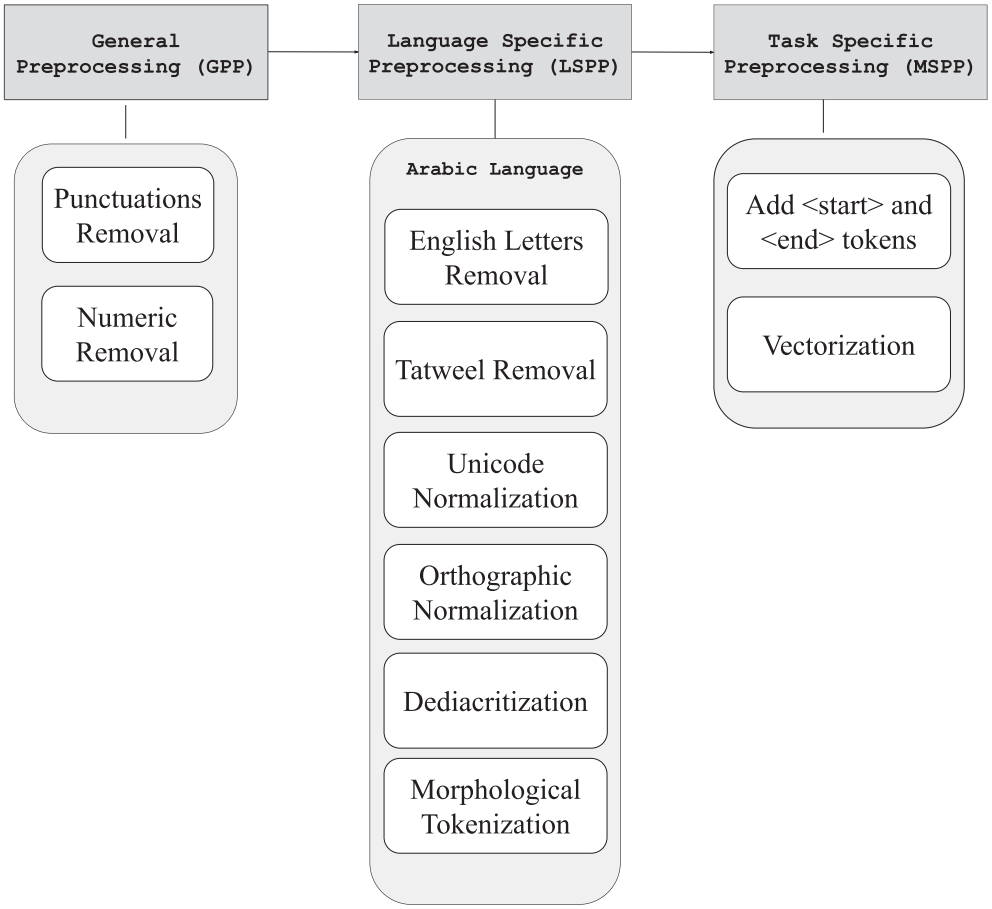

Fig. 2. Preprocessing pipeline describes the steps involved in preprocessing where GPP is applied, which is the critical step for any language. After that, the processing steps for the Arabic language, which is LSPP, have been applied, and finally, TSPP makes the text fit to the image captioning model.

shapes depending on where they are in a word (beginning, middle, or end). For instance, the letters "ع" and "ك", as shown in Table 1, have different shapes depending on their position in the word. The complicated concatenative structure of Arabic is responsible for the Arabic language's lexical sparsity Al-Sallab et al. (2017). Words can take on various shapes while yet having the same meaning. For instance, although the definite article "ال", which corresponds to "the" in English, is usually prefixed to other words, it is not an essential component of those words. So Arabic NLP applications must handle the Arabic language's complexity. The following are some of the steps that have been taken to preprocess the Arabic text and reduce its complexity.

Non-Arabic Letters Removal. In this step, the English letters are deleted. Since this study focuses on Arabic, it is essential to delete non-Arabic letters to reduce noise Shoukry and Rafea (2012).

Tatweel Removal. Some Arabic characters are stretched using the symbol known as the tatweel. In informal Arabic, the tatweel symbol is frequently used to express a sentiment or meaning. The tatweel must be eliminated because it generates redundant spellings of the same word AlOtaibi and Khan (2017).

Unicode Normalization. Unicode normalization transforms Unicode strings into their conventional form by normalizing them (characters that can be written as a combination of Unicode characters are converted into their decomposed form) Obeid et al. (2020).

Orthographic Normalization. Orthographic normalization is the process of unifying different letter forms or visually comparable letters. For example, all "ء" characters are removed, the letter "ة" is replaced by "ه" and the letter "ى" is replaced by "ي". Table 2 summarizes the letter normalization scheme.

Dediacritization. The elimination of Arabic diacritical marks is known as dediacritization. Most Arabic NLP algorithms eliminate them because they make the data sparser Obeid et al. (2020). For instance, the word مَكْتَبَةٌ is converted to مكتبة.

Morphological Tokenization. Standard word tokenization is the process of splitting sentences by whitespace. Morphological tokenization is a different kind of tokenization that separates Arabic

Table 1
Examples of letter shapes according to their position in the word. Where the Arabic letters differ in shape according to their position at the beginning, middle, or end, it also varies according to their connection or separation from the letter before or after.

Letter	Beginning	Middle	End
ع	عـ	ـعـ	ـع or ع
ك	كـ	ـكـ	ـك

Table 2
Letters that are normalized in the Arabic language after unifying different letter forms or visually comparable letters.

Letter	Normalized to
ا or أ or إ or آ	ا
ه or ة (ha or ta'a marbouta)	ه
ي or ى (ya or alef maqsoura)	ي

words into their components, such as prefixes, stems, and suffixes. For example, the word "المكتبة" is rewritten as three tokens "ال+", "ة", and "ة+". The added "+" symbol denotes where the segment was cut off, allowing us to de-segment the text afterward.

4.3. Task-specific preprocessing

In the task-specific preprocessing stage, the steps that make the text suitable for the image captioning model are applied, as shown in the third part of Fig. 2.

Markup Tokens. This step adds < start > and < end > tokens to each caption so that the model knows where the caption begins and ends.

Vectorization. The TextVectorization step associates a unique integer value with each token. It also unifies the caption vectors' length to a specific length, cutting off more extended captions, padding the shorter ones with zeros, and then transforming them into a vector of integers.

5. Preprocessing tools for AIC model training

In this paper, we aim to study the impact of three preprocessing tools, namely, the CAMeL Tools Obeid et al. (2020), ArabertPreprocessor Antoun et al. (2020), and Stanza Qi et al. (2020).

5.1. Preprocessing tools

This subsection gives a brief overview of each preprocessing tool we used in our study.

5.1.1. CAMeL Tools

CAMeL Tools Obeid et al. (2020) is an open-source Python-based toolkit that is used for preprocessing Arabic text and dialects, morphological modeling, dialect identification, named entity recognition, and sentiment analysis. These utilities are covered by the command-line interfaces (CLIs) and application programming interfaces (APIs) provided by CAMeL Tools. The CAMeL Tools toolkit has several features that make it flexible in use and reuse, modular, high-performance, and easy to use for beginners.

5.1.2. ArabertPreprocessor

ArabertPreprocessor is the text preprocessing component of AraBERT which is a transformer-based model designed for Arabic language understanding Antoun et al. (2020). It provides multiple functions for Arabic text-processing and uses FARASA Abdelali et al. (2016) for segmentation. The Farasa segmenter Abdelali et al. (2016) is an Arabic word segmenter used to segment text into stems, prefixes, and suffixes. In addition, apply a specific pre/post-fixed indicator "+" to the segmented articles so that users can reattach the segments to the original term.

5.1.3. Stanza

Stanza Qi et al. (2020) is created by the Stanford NLP Group Manning et al. (2014). It is a set of specific and efficient tools for analyzing the linguistics of various human languages. Tokenization, multi-word token (MWT) expansion, lemmatization, part-of-speech (POS) and morphological features tagging, dependency parsing, and named entity recognition are some techniques it contains that can be utilized in a pipeline.

5.2. Comparison of preprocessing tools

Several differences exist between the aforementioned tools. The CAMeL Tools toolkit provides utilities for preprocessing, morphological modeling, dialect identification, Diacritization, named entity recognition, tokenization, and sentiment analysis. ArabertPreprocessor provides features like diacritization, tatweel removal, and tokenization. Stanza provides multiple features, including MWT expansion, tokenization, lemmatization, morphological features tagging, and named entity recognition. Table 3 compares the availability of features among the three tools. While these tools may contain functions that can be used for various text-processing tasks, we focus our comparison on the features used for AIC.

Table 3
This table compares the preprocessing functions applied by each of the Arabic language preprocessing tools used in our study.

Text Preprocessing Functions	CAMeL	ArabertPreprocessor	Stanza
Remove Tatweel		✓	
Unicode Normalization	✓		
Orthographic Normalization	✓		
Dediacritization	✓	✓	
Morphological Tokenization	✓	✓	✓

5.3. An example of tokenization

Word and sentence composition in Arabic is more complex than in other languages, such as English, and it can lead to a very large lexicon size. For example, a complete sentence in other languages (e.g., English) can be replaced by a single word in Arabic. Thus, there is a need to tokenize Arabic sentences to allow better semantic representation.

Moreover, a token is described as a sequence of one or more letters (characters) separated by spaces. For non-agglutinative languages like English, this definition works. With the Arabic language, tokenization is challenging due to the rich and complex morphology of Arabic Farghaly and Shaalan (2009).

The prefix "ال", which leads to redundancy in the vocabulary, also a more complicated example of one Arabic word being translated into a complete sentence in English, such as ""وسيكتبونها which is translated into an English sentence (and they will write it). Tokenization breaks down sentences into their morphemes:

"و+" "سـ+" "يـ+" "كتب" "+ون" "+ها".

Table 4 shows examples of words before and after going through CAMeL Tools' tokenizer, ArabertPreprocessor segmenter, and Stanza tokenizer. As we can see on the first word in the table, the CAMeL Tools' tokenizer and ArabertPreprocessor divid the word in the same way, where the word is divided into three parts, the first is "ال+" (equivalent to the definition "the" in English) as prefixes, then the stem "فتي", then "ات+" (of the feminine plural) at the end of the word as suffixes.

The second word is divided into three parts, the prefixes "ي+" indicates the masculine present tense, then the stem quotes "لتقط", then the suffixes "ون+" indicates the masculine plural. The fourth word is divided into four parts, "ف+" denotes the order, "ي+" indicates the masculine present tense, the stem "قرؤ", and the suffix "ون+" which indicates the masculine plural. Note that the second and fourth words are divided by CAMeL Tools only.

Table 4
Examples that show some words segmented by CAMeL Tools, ArabertPreprocessor, and Stanza.

	Word	CAMeL	ArabertPreprocessor	Stanza
1.	الفتيات	ات+ فتي +ال	ات+ فتي +ال	الفتيات
2.	يلتقطون	ون+ لتقط +ي	يلتقطون	يلتقطون
3.	ويلعب	لعب +ي +و	يلعب +و	و يلعب
4.	فيقرؤون	ون+ قرؤ +ي +ف	فيقرؤون	فيقرؤون

As for the third word, the letter "+و" (equivalent to "and" in English), it is separated in all three tools (but in Stanza, the + sign was not added), and in CAMeL Tools the masculine present "+ي" is also separated as in the previous word.

6. Datasets for AIC research

In this section, we go over the datasets used in our study. In our study, we use two datasets. The first one is provided by ElJundi et al. (2020) and is denoted as Flickr8k. The second dataset is denoted Flickr8k-Rev, which is a complete revision of Flickr8k. Flickr8k is the first publicly accessible dataset for AIC. It is created by translating the Flickr8k dataset Hodosh et al. (2013). The English captions are translated using the Google Translate API, then corrected and validated by a professional Arabic translator to avoid translation errors. Fig. 3 depicts a few examples from the dataset and the original English captions.

Flickr8k is one of two available Arabic datasets for AIC. The other one is based on the MS COCO dataset Lin et al. (2014), which is Arabic–COCO, Canesee-project, 2023. The Flickr8k dataset has 8,000 images with 24,276 captions. Thus, there are three captions per image. In our revised dataset, Flickr8k-Rev, we extended the number of captions to 40,000 (i.e., five captions per image) and validated them using native Arabic speakers. The Arabic–COCO dataset has 82,783 images and 413,915 captions (5 captions per image). However, the captions in the Arabic–COCO dataset are machine-translated from its original dataset using Google's Machine Translation API, and human validators do not validate it.

Table 5
Some examples of grammatical errors in the Arabic Flicker8k dataset. The blue text indicates grammatical errors, and the red text indicates spelling errors.

Grammatical Errors
صبيان يرتديان الزي الأخضر والأبيض كرة السلة يلعبان مع صبيان يرتديان الزي الأبيض والأزرق
لاعبي كرة قدم يعالج مشكلة مع لاعب كرة قدم آخر
كلبينيتسابقان علع مدمار سباق

The captions in Flickr8k are further validated by human validators after being machine-translated. However, unfortunately, after reviewing the captions in the Flickr8k dataset, we discovered many spelling and grammatical errors. Examples of these errors are shown in Table 5. For example, in the first sentence, we notice a grammatical error in terms of sentence order (subject + verb + ob ject) in the sentence: the object "كرة السلة" precedes the verb "يلعبان". In this case, the correction is to rearrange the sentence:

صبيان يرتديان الزي الأخضر والأبيض يلعبان كرة السلة مع صبيان يرتديان الزي الأبيض والأزرق

- فتاة صغيرة مغطاة بالطلاء تجلس أمام قوس قزح
- فتاة صغيرة تجلس أمام قوس قزح ملون كبير
- فتاة أمام لوحة قوس قزح

- A little girl covered in paint sits in front of a painted rainbow with her hands in a bowl
- A little girl is sitting in front of a large painted rainbow
- A small girl in the grass plays with fingerpaints in front of a white canvas with a rainbow on it
- There is a girl with pigtails sitting in front of a rainbow painting
- Young girl with pigtails painting outside in the grass

- رجل يرتدي سترة حمراء يجلس على مقعد بينما يطبخ وجبة
- رجل يجلس على مقعد ، يطبخ بعض الطعام
- رجل يجلس على مقعد

- A man in a red jacket is sitting on a bench whilst cooking a meal
- A man is sitting on a bench , cooking some food
- A man sits on a bench
- A man wearing a red jacket is sitting on a wooden bench and is cooking something in a small pot
- a man wearing a red jacket sitting on a bench next to various camping items

Fig. 3. A sample of the translated and original dataset ElJundi et al. (2020)Hodosh et al. (2013).

In the second sentence, we show another example of a grammatical error regarding a mismatch of the singular and plural forms. The first word "لاعبي", which means a group of players, is plural, while the verb in the middle of the sentence "يعالج" is singular. In this case, the correction is to modify the singular or plural based on the image described in the database. Also, the word "لاعبي" in the nominative case is written "لاعبو". Thus, the correct caption is as follows:

لاعبو كرة قدم يعالجون مشكلة مع لاعب كرة قدم آخر

In the third sentence, we show an example of an erroneous caption with two different types of errors, i.e., grammatical and spelling errors. The first is the grammatical error in the word's nominative, accusative, and prepositional cases. In this case, the word "كلبين" is in the nominative case, and the correct way to write it is "كلبان". As for the spelling error, it is shown in "علع مدمار". The correction is "على مضمار". Table 6 displays some spelling errors, its correction, and its translation in English. These spelling errors increased the number of unique words in the dataset and impacted the trained model negatively.

Building the Flickr8k-Rev dataset. When building the Flickr8k-Rev dataset, we developed a GUI-based application using GUIzero, 2023 to facilitate writing captions for each image. The user interface displays the image and provides text boxes to add the five captions to the displayed image, save it with the click of a button, and then move to the next image with the "Next" button. Fig. 4 shows a screenshot of the GUI. A user interface was created that enables us to see the current image and add five captions to it, and then press the "Save" button to save each of the five captions along with the path of the image in a separate line in a file and then users can press the "Next" button to move to the following image.

Table 6
Some examples of the spelling errors in the Arabic Flicker8k Dataset.

Spelling Error	Correction (English translation)
تتسابق	تتسابق (she is racing)
لعبة	لعبة (game)
يجدان	يجدفان (they row)
كصارعة	مصارعة (wrestling)
ترشاهدها	تشاهدها (she is watching it)
كلا	كلاب (dogs)

Table 7
Flickr8k-Rev Dataset specifications summary.

Number of images	8091
Number of captions/image	5
Total number of captions	40455
Number of unique words	7684
Number of words in the longest sentence	19
Number of words in the shortest sentence	2

Table 7 presents the dataset specifications. Our Flickr8k-Rev dataset contains 7,684 unique vocabularies, and the length of each caption ranges from 2 to 19. The mean, variance, and standard deviation values of the length of captions are 6.1152, 3.7905, and 1.9469, respectively. In Fig. 5, we show the distribution of the caption lengths in Flickr8k-Rev. Captions in our dataset are verified by printing all unique vocabulary and ensuring they are free from spelling errors, wrongly repeated letters, single letters, extra spaces, and missing captions.

7. Prototype implementation

The implementation architecture is composed of a two-step pipeline, as shown in Fig. 6:

1. After extracting the image features from the CNN pre-trained model, the encoder receives an image's feature vector.
2. The decoder takes the sequence vector after it is preprocessed along with the image representation (the output from the encoder) and then extracts the sequence output.

As shown in Fig. 6, we must prepare the dataset containing images and text before training the model. In our experiment, we use Arabic Flickr8k-Rev and the Arabic Flickr8k datasets, as explained in the previous section. Moreover, we preprocess the images and text, as described below.

Fig. 4. Our GUI-based application to facilitate writing captions for images in Flickr8k-Rev. It contains five text-boxes that allow adding five image captions and two buttons, one for saving the added captions "Save" and the other for moving to the next image "Next", in addition to a counter showing progress according to the number of images.

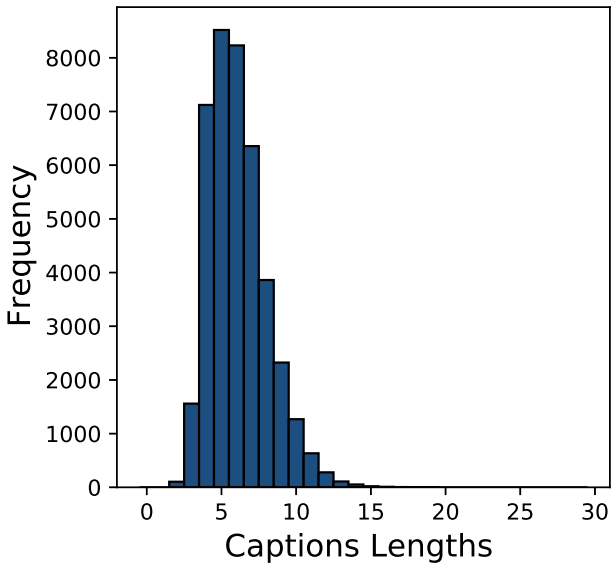

Fig. 5. A histogram of captions lengths in our Flickr8k-Rev dataset. The length of the sentences ranges from 2 to 19 words in the sentence. The most frequent length is five words in the sentence.

7.1. Image preprocessing

The dataset contains different sizes of RGB images. The following image preprocessing steps are used to preprocess the images.

- Decode a JPEG-encoded image to a uint8 tensor and three color channels.
- Resize all images to the same size, 299x299 pixels.
- Convert the image to dtype, normalized to floating point values in the range [0, 1).
- Pass the resulting tensors through the CNN model, where the output layer is the last convolutional layer to extract the feature vector.

We use pre-trained CNN models without the fully connected layers for feature extraction. In our prototype, we use the ResNet-152 He et al. (2016) and the EfficientNet model Tan and Le (2019). Next, give a brief description of these two models.

ResNet-152 He et al. (2016)**.** By learning the residual representation functions rather than the signal representation directly, ResNet creates an intense network with up to 152 layers. To fit the input from the previous layer to the next layer without modifying the input, ResNet introduces the idea of skipping connections which leads to deeper networks.

EfficientNet model Tan and Le (2019)**.** The EfficientNet model is a neural search that uses a compound scaling method with a compound coefficient to optimize accuracy and efficiency. The mobile-size-baseline EfficientNet helps optimize the accuracy and efficiency of classification, model scaling, and identifying balancing network depth, width, and resolution. Data augmentation will also be applied to increase the number of data and rotation in different angles to avoid the overfitting problem and give a better generalization. The authors created a family of EfficientNets from B0 to B7, which achieved state-of-the-art accuracy on ImageNet while being very efficient compared to its competitors. B0 is used as a baseline in their work.

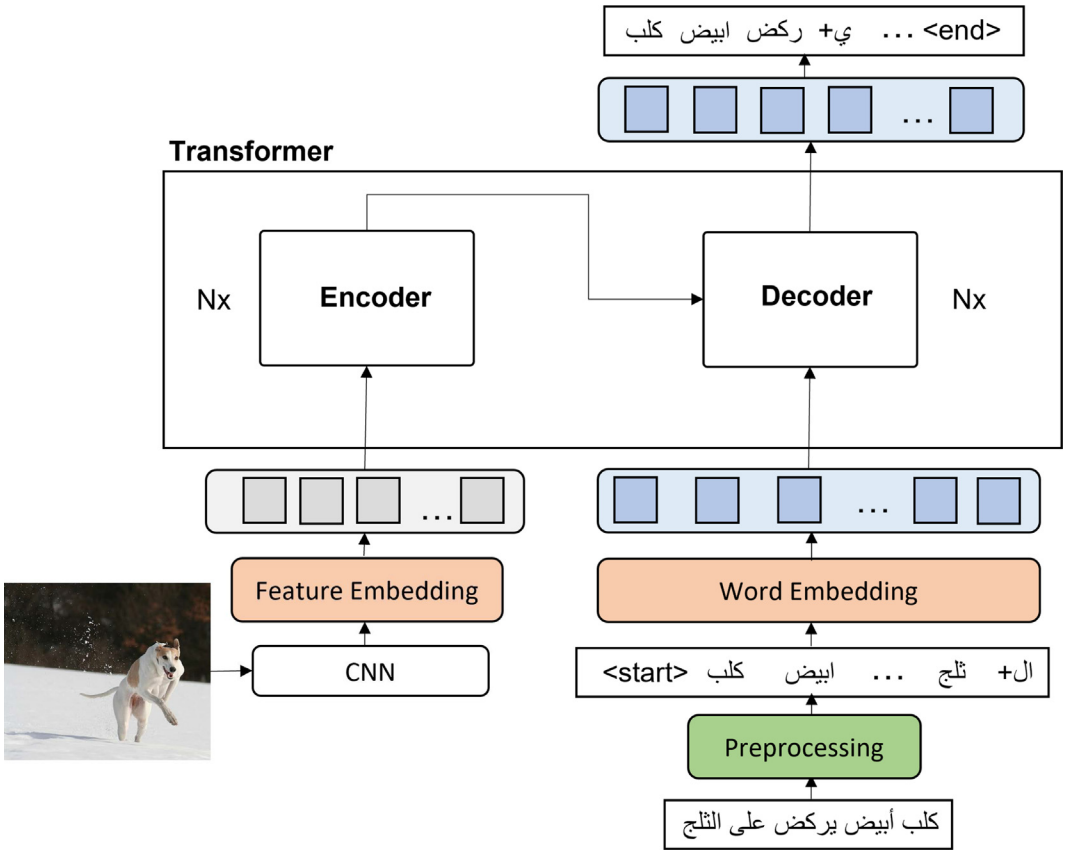

Fig. 6. Prototype implementation architecture. It is made up of a multi-layer encoder and decoder. Self-attended visual features are encoded using the encoder. The decoder then performs cross-attention using the textual and visual features from the encoder. Sentence creation would then be done using the output of the decoder's last layer.

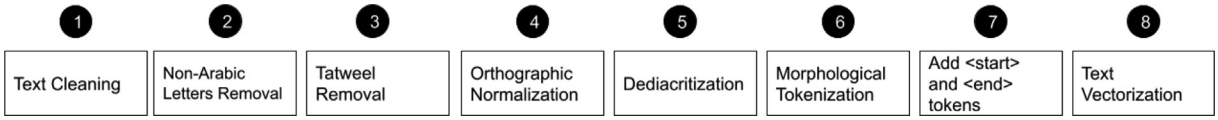

Fig. 7. The above are the set of preprocessing steps that can be applied in general for preprocessing text labels for AIC.

7.2. Text preprocessing

The Arabic text goes through several preprocessing steps, as shown in Fig. 7. We defined the following three mechanisms for preprocessing captions.

Mechanism 1: (CAMeL) The first mechanism uses only CAMeL Tools functions. Some of its functions are used in the following preprocessing steps: orthographic normalization, dediacritization, and morphological tokenization.
Mechanism 2: (ARABERT) The second mechanism is based on ArabertPreprocessor, and its functions used for the following preprocessing steps: tatweel removal, dediacritization, and morphological tokenization.
Mechanism 3: (STANZA) The last mechanism is based on Stanza in the morphological tokenization step. Stanza does not have functions that can be used in other steps.

Some steps are common to all mechanisms, such as text cleaning and non-Arabic letter removal. Text cleaning is the step mentioned in GPP and is applied similarly in the three mechanisms. The method by Shoukry and Rafea (2012) is followed in non-Arabic letters removal for all three mechanisms, orthographic normalization for the second and third mechanisms, and de-diacritization for the third. Tatweel removal has been implemented in a second mechanism only. The TSPP steps are applied similarly for all three mechanisms: the addition of <start> and <end> tokens and text vectorization.

7.3. Datasets and training methodology

In our study, we use a splitting ratio of 80%,10%, and 10% for training, validation, and testing, respectively. First, we divide the two datasets into two partitions with a ratio of 90% and 10%. We divide the partition containing the 90% further into two partitions such that the partition used for training is 80% of the original dataset size, and the remaining is used for validation. The partition used for testing corresponds to 10% of the original dataset size. We followed this approach to remain comparable to ElJundi et al. (2020) where the dataset is split with a ratio of 90% for training and 10% for testing, but they don't use a validation set. Moreover, this splitting approach allows us to apply the k-fold cross-validation technique, which takes a portion from the training set for validation, as shown in Table 8.

8. Experimental evaluation

In this section, we present our empirical study. We start by describing the evaluation metrics used in our experimental evaluation. Table 9 summarizes the notations used throughout this subsection.

Table 9
Summary of notations used in this paper.

Text Preprocessing Mechanism 1 using CAMeL Tools	CAMEL
Text Preprocessing Mechanism 2 using ARABERT Preprocessor	ARABERT
Text Preprocessing Mechanism 3 using STANZA	STANZA
GPP and TSPP	GTSPP
BiLingual Evaluation Understudy	B1-B4
Metric for Evaluation of Translation with Explicit Ordering	M
Recall-Oriented Understudy for Gisting Evaluation	R
Consensus-based Image Description Evaluation	C

8.1. Evaluation metrics

We focus on four metrics commonly used in image captioning research evaluation.

BiLingual Evaluation Understudy (BLEU). BLEU Papineni et al. (2002), which we denote as B, is a metric for evaluating machine-translated text automatically. The BLEU score is between 0 and 1, indicating how closely the machine-translated text resembles a collection of high-quality reference translations. A number of 0 indicates that the machine-translated output has no overlap with the reference translation (poor quality), while a value of 1 indicates that the overlap is excellent (high quality). It is the most frequently used metric for IC evaluation and is used to examine the n-gram correlation between the assessed translation statement and the reference translation statement.

Metric for Evaluation of Translation with Explicit Ordering (METEOR). METEOR Banerjee and Lavie (2005), which we denote as M, is created to address various flaws with the BLEU metric. The creators of the METEOR metric claim that METEOR can capture the correlation with human judgments better than the BLEU metric. The metric also gives more focus on recall over precision. METEOR is based on unigram-precision and unigram-recall.

Recall-Oriented Understudy for Gisting Evaluation (ROUGE). ROUGE Lin (2004), which we denote as R, consists of a set of metrics for evaluating automatic text summarization and is also applied for image captioning. These metrics compare reference (human-produced) summaries or captions with summaries or captions generated automatically. It calculates the longest matching sequence of words between the generated caption and the reference caption to determine their similarity.

Consensus-based Image Description Evaluation (CIDEr). CIDEr Vedantam et al. (2015), which we denote as C. CIDEr is based on computing the cosine similarity between the candidate caption and the set of reference captions associated with the image's Term Frequency-Inverse Document Frequency weighted n-grams, thus accounting for both precision and recall.

8.2. Performance analysis experiments

We designed several experiments to study the factors that impact AIC performance. Our experimental study concerns several

Table 8
Data split configuration.

Dataset	Training Images	Training Captions	Validation Images	Validation Captions	Testing Images	Testing Captions
Flickr8k	6493(80%)	19479	800(10%)	2400	800(10%)	2400
Flickr8k-Rev	6493(80%)	32465	800(10%)	4000	800(10%)	4000

Table 10
This table shows the mean and standard deviation of B1, B2, B3, B4, M, R, and C scores of the ten-fold cross-validation as a summary of the experiments. The first group represents the impact of the training dataset. The second group shows the impact of image feature extraction models. The last two groups for the impact of text preprocessing. The (∗) sign indicates using Flickr8k-Rev dataset.

	B1	B2	B3	B4	M	R	C
Flickr8k	0.295± 0.047	0.179± 0.031	0.099± 0.018	0.052± 0.01	0.231± 0.005	0.28± 0.019	0.289± 0.073
Flickr8k-Rev	0.427± 0.014	0.303± 0.012	0.202± 0.01	0.129± 0.008	0.285± 0.003	0.4± 0.008	0.494± 0.027
EfficientNet-B0	0.295± 0.047	0.179± 0.031	0.099± 0.018	0.052± 0.01	0.231± 0.005	0.28± 0.019	0.289± 0.073
ResNet152	0.329± 0.007	0.198± 0.005	0.108± 0.004	0.057± 0.003	0.224± 0.005	0.288± 0.005	0.34± 0.012
GTSPP	0.329± 0.007	0.198± 0.005	0.108± 0.004	0.057± 0.003	0.224± 0.005	0.288± 0.005	0.34± 0.012
CAMEL	0.489± 0.012	0.317± 0.012	0.213± 0.01	0.145± 0.009	0.334± 0.006	0.398± 0.006	0.472± 0.023
ARABERT	0.442± 0.006	0.277± 0.004	0.179± 0.003	0.116± 0.003	0.287± 0.006	0.367± 0.002	0.419± 0.011
STANZA	0.332± 0.010	0.198± 0.008	0.119± 0.007	0.071± 0.005	0.233± 0.005	0.299± 0.005	0.329± 0.015
GTSPP∗	0.444± 0.013	0.318± 0.011	0.212± 0.009	0.136± 0.008	0.283± 0.004	0.403± 0.006	0.547± 0.029
CAMEL∗	0.616± 0.009	0.47± 0.011	0.359± 0.011	0.273± 0.011	0.391± 0.003	0.519± 0.005	0.828± 0.032
ARABERT∗	0.538± 0.016	0.384± 0.013	0.281± 0.012	0.204± 0.011	0.341± 0.04	0.464± 0.038	0.629± 0.036
STANZA∗	0.449± 0.009	0.311± 0.006	0.213± 0.005	0.142± 0.005	0.293± 0.007	0.406± 0.005	0.502± 0.015

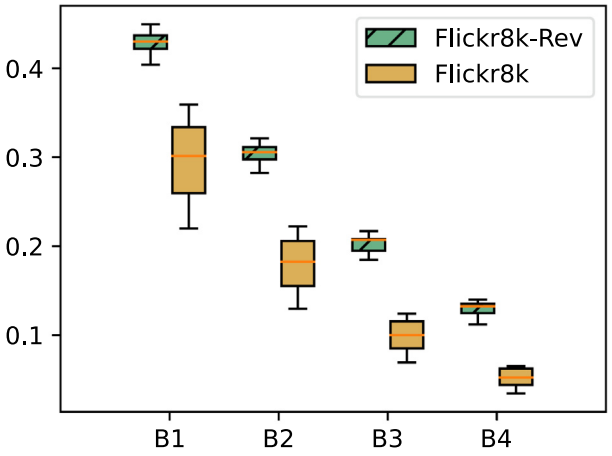

Fig. 8. Comparison of Flickr8k and Flickr8k-Rev with GTSPP for text preprocessing and EfficientNet-B0 for image feature extraction concerning B1, B2, B3, and B4 values for ten-fold cross-validation.

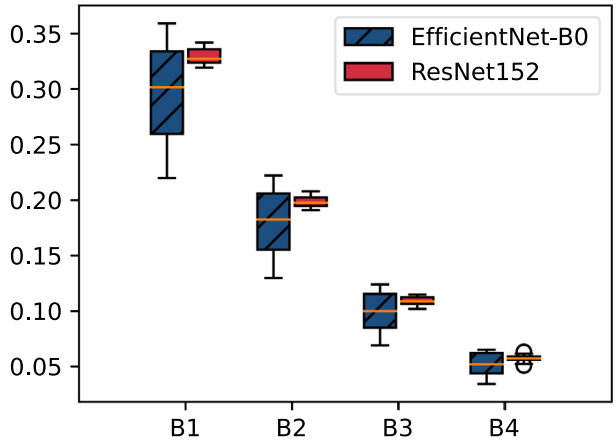

Fig. 10. The comparison of EfficientNet-B0 and ResNet152 for image feature extraction using Flickr8k dataset to demonstrate the impact of image feature extraction models. BLEU scores for ten-fold cross-validation.

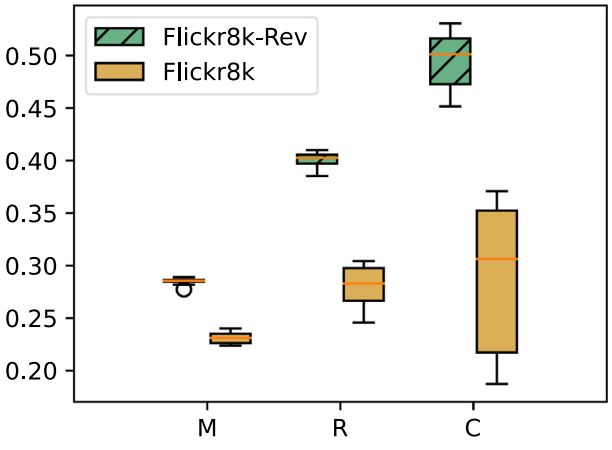

Fig. 9. Comparison of Flickr8k and Flickr8k-Rev with GTSPP for text preprocessing and EfficientNet-B0 for image feature extraction concerning M, R, and C values for ten-fold cross-validation. It showed that using Flickr8k-Rev improves the quality of the captions.

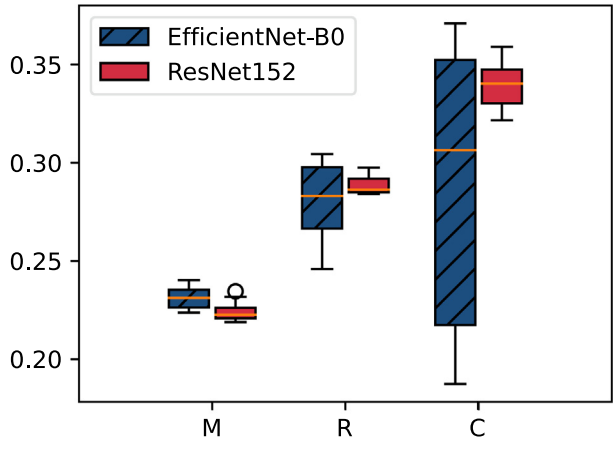

Fig. 11. Comparison of EfficientNet-B0 and ResNet152 for image feature extraction using Flickr8k dataset to demonstrate the impact of image feature extraction models, with respect to M, R, and C values for ten-fold cross-validation.

Table 11
The original number of vocabulary in both datasets versus the number of vocabulary after applying GTSPP, CAMeL, ARABERT, and STANZA

Dataset	Original	GTSPP	STANZA	ARABERT	CAMEL
Flickr8k	11388	10656	9472	5670	4447
Flickr8k-Rev	7684	7176	6501	3690	2779

factors, such as the datasets, the preprocessing mechanism, and the image feature extractors.

We use the Flickr8k and the Flickr8k-Rev datasets to study the impact of the datasets. The preprocessing mechanisms are described in Section 7. In addition, we also include a baseline mechanism (**Mechanism 0**) that does not include LSPP operations and uses only the GPP and TSPP operations. We refer to this approach as GTSPP. EfficientNet-B0 and ResNet152 are used as image feature extractors.

Experiments are carried out in Colab Pro+ with a GPU and a high RAM. The models are trained using ten-fold cross-validation. The hyperparameters set in the model were as follows: the batch size is 64, the number of epochs is 30 with early stopping, the patience parameter is set to 3, and the learning-rate parameter is set to 1e-4. For all experiments, we use the cross-entropy loss function and Adam optimizer. Table 10 presents a numerical summary of our experimental study.

Impact of the training Dataset. We run two experiments with GTSPP using Flickr8k and Flickr8k-Rev to understand the impact of using the different datasets for training the model. After conducting experiments using the two datasets, it turned out that using Flickr8k-Rev improves the quality of the captions, as shown in Fig. 8 and 9. Fig. 8 shows BLEU-n scores for the two datasets for ten-fold cross-validation where the BLEU-1 (B1) values for the dataset Flickr8k dataset range from 0.219–0.359 and BLEU-4 (B4) values range from 0.034–0.065. Flickr8k-Rev dataset BLEU-1 (B1) ranges from 0.404–0.449, and BLEU-4 (B4) ranges from 0.112–0.139 for the testing set. After using the Flickr8k-Rev the results is improved by 18.4% - 9.05% for BLEU-1 and 7.75% - 7.48% for BLEU-4.

Fig. 9 shows the result for M, R, and C scores. The M values for the Flickr8k dataset range from 0.223–0.24, R values range from 0.245–0.304, and C values range from 0.187–0.37. Flickr8k-Rev dataset M values range from 0.277–0.289, R values range from 0.385–0.41, and C values range from 0.451–0.53.

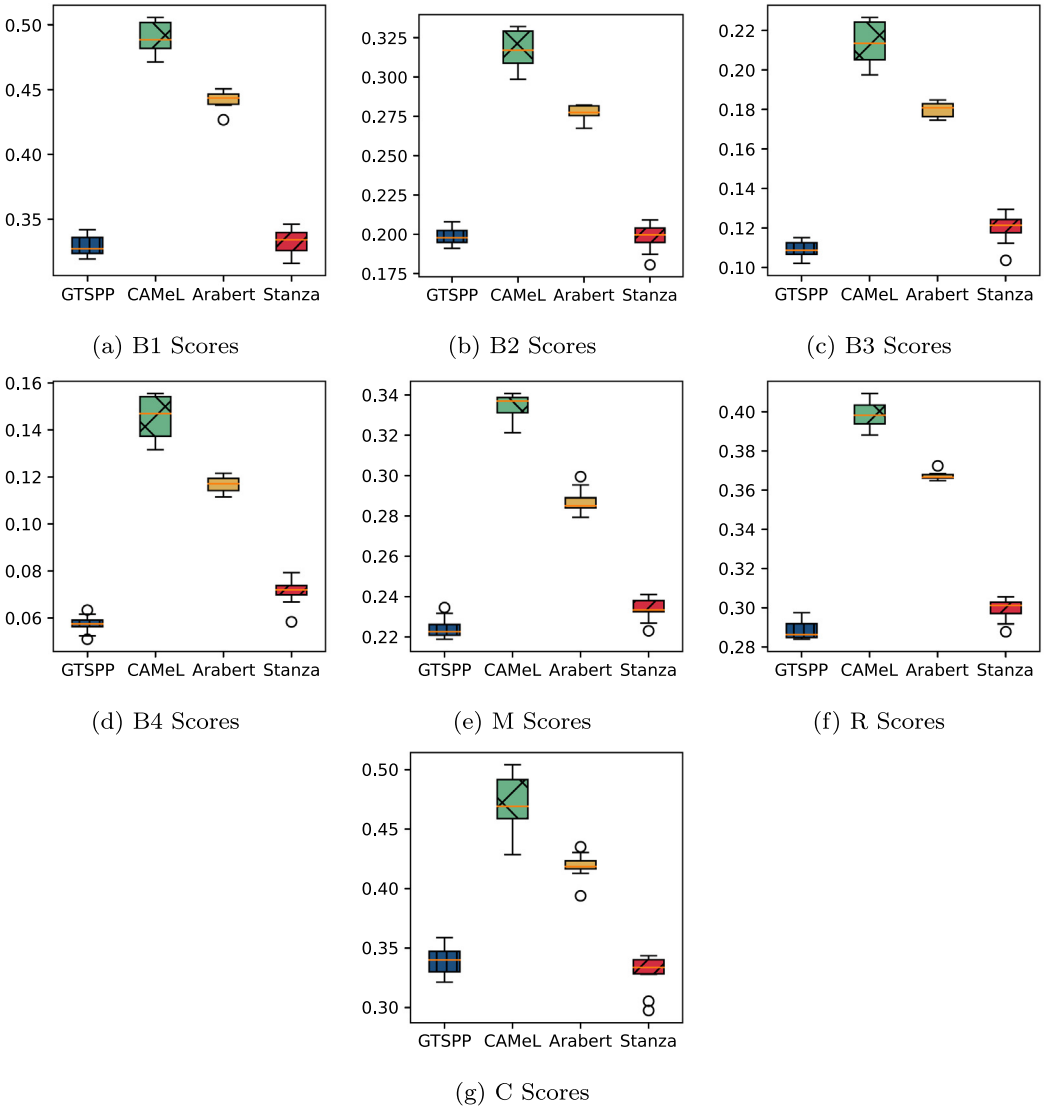

Fig. 12. Comparison of preprocessing mechanisms CAMeL, ARABERT, and STANZA with the baseline GTSPP concerning B1, B2, B3, B4, M, R, C scores for ten-fold cross-validation using the Flickr8k dataset. When applying the preprocessing mechanisms, the results increased, especially with CAMeL of different metrics, except with B2 and C for STANZA.

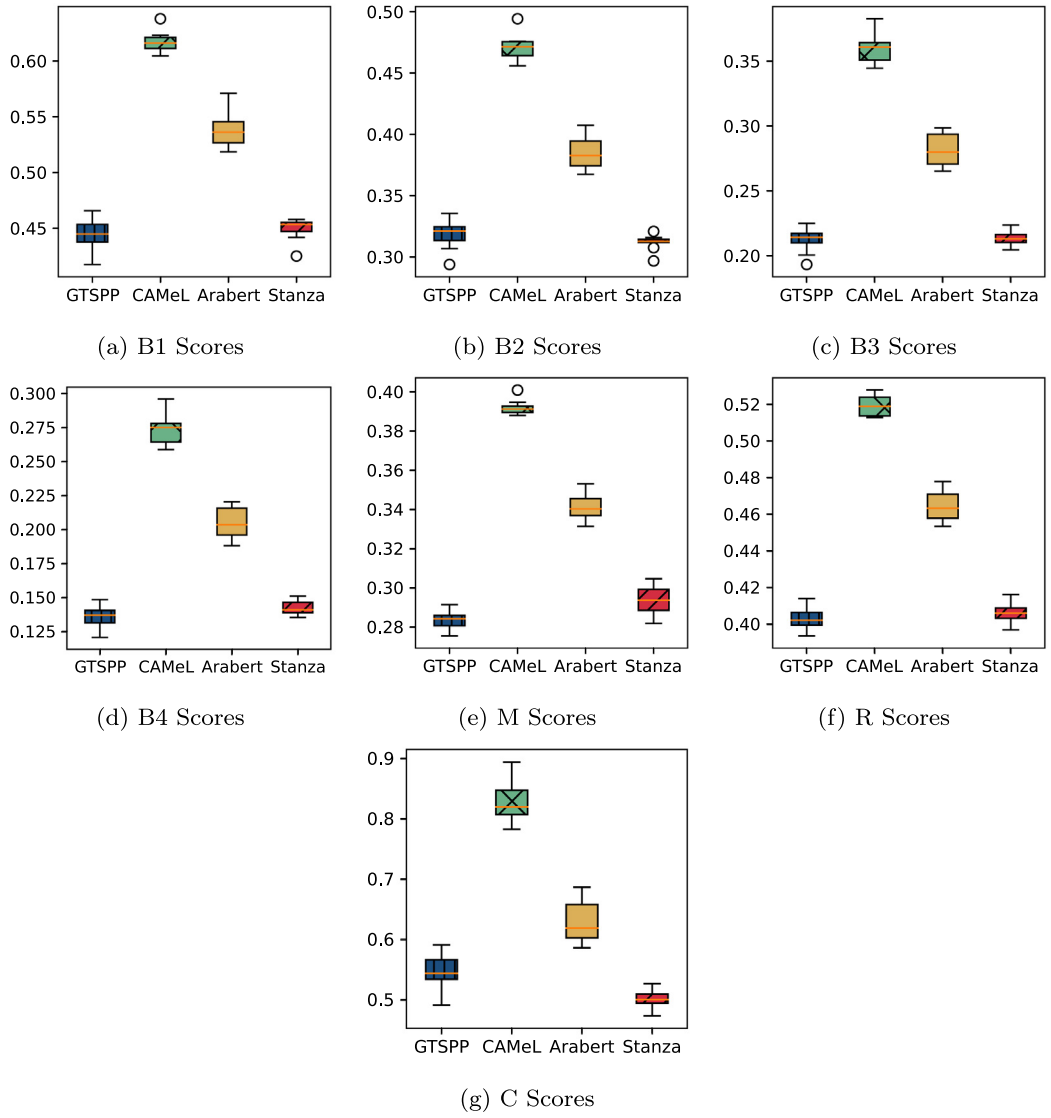

Fig. 13. Comparison of preprocessing mechanisms CAMeL, ARABERT, and STANZA with the baseline GTSPP concerning B1, B2, B3, B4, M, R, C scores for ten-fold cross-validation using the Flickr8k-Rev dataset. When applying the preprocessing mechanisms, the results increased, especially with CAMEL of different metrics, except with B2 and C for STANZA.

Table 12
Performance comparisons of recent studies on the Arabic version of the Flickr8k ElJundi et al. (2020) dataset with our model using CAMeL on Flickr8k ElJundi et al. (2020) and Flickr8k-Rev datasets. It shows Fig. 14 data. Note that, the (∗) sign indicates using Flickr8k-Rev. In all of our configurations, we are using ResNet152 as the feature extractor component.

Model	B1	B2	B3	B4	M	R	C
ElJundi et al. (2020)	0.332	0.193	0.105	0.057	-	-	-
Hejazi and Shaalan (2021)	0.365	0.214	0.12	0.066	-	-	-
Emami et al. (2022)	0.391	0.246	0.150	0.092	0.314	0.331	0.415
CAMEL	0.489	0.317	0.213	0.145	0.334	0.398	0.472
ARABERT	0.442	0.277	0.179	0.116	0.287	0.367	0.419
CAMEL∗	**0.616**	**0.470**	**0.359**	**0.273**	**0.391**	**0.519**	**0.828**
ARABERT∗	0.538	0.384	0.2813	0.204	0.353	0.452	0.629

Impact of Image Feature Extraction Models. In the following experiment, we study the impact of the model used for image feature extraction on the generated captions. We use two image feature extraction models, EfficientNet-B0 and ResNet152. The shape of the feature vector extracted from EfficientNet-B0 and ResNet152 are 81×1280 and 100×2048, respectively. The average performance of the model that uses ResNet152 is better than the model that uses EfficientNet-B0, as shown in Figs. 10 and 11.

Fig. 10, shows the result for B1, B2, B3, and B4 scores. The BLEU average values of the ten-fold cross validation of ResNet152 are

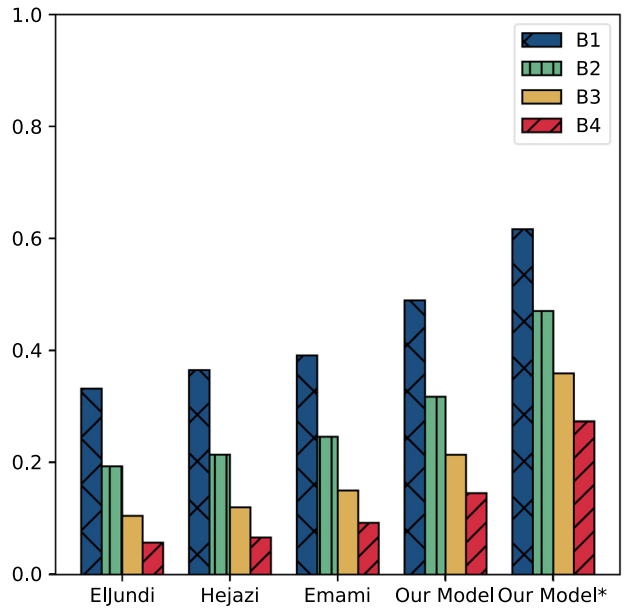

Fig. 14. Performance comparisons of recent studies on the Arabic version of the Flickr8k ElJundi et al. (2020) with our model using CAMeL on Flickr8k ElJundi et al. (2020) and Flickr8k.-Rev datasets.

0.329, 0.198, 0.108, and 0.057 for B1, B2, B3, and B4, respectively. The EfficientNet-B0 scores are 0.295, 0.179, 0.099, and 0.052 for B1, B2, B3, and B4, respectively.

In Fig. 11, the average scores for M, R, and C of the model using ResNet152 are 0.224, 0.288, and 0.34, respectively. While the scores for EfficientNet-B0 are 0.231, 0.28, and 0.289 for M, R, and C, respectively.

We note that all ResNet152 results are higher than EfficientNet-B0 except for R. EfficientNet-B0 is more complex than ResNet152, so it has more variations in data and is more prone to overfitting due to regularization.

Impact of Text Preprocessing. We experimentally study the four aforementioned preprocessing mechanisms. The mechanisms are evaluated to analyze their impact and importance in generating accurate captions.

The different preprocessing mechanisms have an impact on the number of unique vocabulary. In Table 11, we show the number of unique vocabulary after applying each mechanism in Flickr8k and Flickr8k-Rev. The vocabulary size is decreased in the two datasets after using the preprocessing mechanisms on the text. The smallest vocabulary size is with the CAMeL mechanism.

GTSPP is used as a baseline for our quantitative comparison with the rest of the preprocessing mechanisms to study their impact on generating accurate captions. Fig. 12 shows the values of B1, B2, B3, B4, M, R, and C, for the Flickr8k dataset. Generally, we note the better performance results when applying the preprocessing mechanisms CAMeL, ARABERT, and STANZA. Compared to GTSPP, using CAMeL improves the performance by 0.09–0.16. Using ARABERT compared to GTSPP enhances the performance by 0.05–11. For STANZA, the performance improves by 0.003–0.013 except for the B2 and the C scores, which decreases by 0.0004 and 0.011, respectively.

Similar behavior is noted for the Flickr8k-Rev dataset (see Fig. 13). Compared to GTSPP, the improvements are 0.10–0.28, 0.05 to 0.09, and 0.0009 to 0.009 for CAMeL and ARABERT. For STANZA, the improvement is not significant (i.e., 0.0009–0.009).

The improvement results can be attributed to the tokenization operation in camel because it splits pronouns related to the word while ARABERT and STANZA don't, as we noticed in Table 4. This means

Fig. 15. This figure shows three images with two captions under their respective image. The first caption (labeled with 1) is generated using the model trained with Flickr8k-Rev while the second caption (labeled with 2) is generated using the model trained with Flickr8k.

that CAMeL extracts the root of the word better and thus facilitates the complexity of the Arabic. So, appropriate preprocessing significantly affects the performance and the quality of the resulting captions.

Comparison with the previous studies. Our best-performing model (using ResNet152 and CAMeL preprocessing) is compared with previous studies ElJundi et al. (2020), Hejazi and Shaalan (2021), and Emami et al. (2022).

In all previous works, the Arabic Flickr8k dataset is used for training their models. Table 12 summarizes all the results for all metrics in our study. Fig. 14 shows the comparison results of B1, B2, B3, and B4 scores. Using the Flickr8k dataset, our model scored 0.4894, 0.317, 0.213, and 0.145 for B1, B2, B3, and B4, respectively. We improve the result using the same dataset by 25–47%, 29–64%, 42–103%, and 58–154%, for B1, B2, B3, and B4, respectively. Using the Flickr8k-Rev dataset, our model significantly improved over previous studies and scores 0.616, 0.47, 0.359, and 0.273 for B1, B2, B3, and B4, respectively. This is an performance improvement of 58–86%, 91–144%, 139–242%, and 197–379%, for B1, B2, B3, and B4, respectively. The above results show that our proposed model can achieve better results up to an order of magnitude.

Qualitative Results. Fig. 15 shows three sample images from the test set. The captions labeled with 1 are generated using the Flickr8k-Rev dataset, while the captions labeled with 2 are generated using the Flickr8k dataset. In both cases, we used CAMeL Tools as the preprocessor. In all three examples, the captions with 1 labels are more accurate and have fewer errors. In Fig. 15a, the second sentence suffers from additional "ة" characters in the last word. The second caption in Fig. 15b has a grammatical error. Finally, in Fig. 15c, the first caption also describes the color of the dog as "بني", which means brown. Because Flickr8k has some spelling and grammatical errors, these errors are carried over to the results.

9. Conclusion

This paper systematically analyzes the performance of transformer-based image captioning models for Arabic captioning. We study the impact of text preprocessing, image feature extractors, and datasets on the performance of AIC models. Our study shows that using CAMeL Tools to preprocess text labels improves the AIC performance by up to 34–92% in the BLEU-4 score. In addition, we study the impact of image recognition models. Our results show that ResNet152 is better than EfficientNet-B0 and can improve BLEU scores performance by 9–11%. Furthermore, we investigate the impact of different datasets on the overall AIC performance and build an extended version of the Arabic Flickr8k dataset. Using the extended version improves the BLEU-4 score of the AIC model by up to 148%. Finally, utilizing our results, we build a model that significantly outperforms the state-of-the-art proposals in AIC by up to 196–379% in the BLUE-4 score. Future work includes exploring more language-specific preprocessing techniques to improve the text preprocessing mechanisms and different architectures of transformer-based deep learning models for AIC.

Declaration of Competing Interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: [Thamir M. Qadah reports a relationship with Umm Al-Qura University - College of Computer and Information Systems that includes: employment. Muhammad Arif reports a relationship with Umm Al-Qura University - College of Computer and Information Systems that includes: employment.]

References

Abdelali, A., Darwish, K., Durrani, N., Mubarak, H., 2016. Farasa: A fast and furious segmenter for arabic. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations, pp. 11–16.

Al-Muzaini, H.A., Al-Yahya, T.N., Benhidour, H., 2018. Automatic arabic image captioning using rnn-lst m-based language model and cnn. Int. J. Adv. Comput. Sci. Appl. 9 (6).

AlOtaibi, S., Khan, M.B., 2017. Sentiment analysis challenges of informal arabic language. Int. J. Adv. Comput. Sci. Appl. 8 (2).

Al-Sallab, A., Baly, R., Hajj, H., Shaban, K.B., El-Hajj, W., Badaro, G., 2017. Aroma: A recursive deep learning model for opinion mining in arabic as a low resource language. ACM Trans. Asian Low-Resource Lang. Infor. Process. (TALLIP) 16 (4), 1–20.

Antoun, W., Baly, F., Hajj, H., 2020. Arabert: Transformer-based model for arabic language understanding, arXiv preprint arXiv:2003.00104.

Atliha, V., Šešok, D., 2020. Comparison of vgg and resnet used as encoders for image captioning. 2020 IEEE Open Conference of Electrical, Electronic and Information Sciences (eStream). IEEE, pp. 1–4.

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

Banerjee, S., Lavie, A., 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In: Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization, pp. 65–72.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y., 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation, arXiv preprint arXiv:1406.1078.

Canesee-project, Arabic-COCO. https://github.com/canesee-project/Arabic-COCO

Cankocagil . Convolutional language model for image captioning: Deep learning for vision & language translation models. https://github.com/cankocagil/Image-Captioning/

Chu, Y., Yue, X., Yu, L., Sergei, M., Wang, Z., 2020. Automatic image captioning based on resnet50 and lstm with soft attention. Wireless Commun. Mobile Comput.

Cornia, M., Stefanini, M., Baraldi, L., Cucchiara, R., 2020. Meshed-memory transformer for image captioning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10578–10587.

Dubey, S., Olimov, F., Rafique, M.A., Kim, J., Jeon, M., 2023. Label-attention transformer with geometrically coherent objects for image captioning. Inf. Sci. 623, 812–831.

ElJundi, O., Dhaybi, M., Mokadam, K., Hajj, H.M., Asmar, D.C., 2020. Resources and end-to-end neural network models for arabic image captioning. In: VISIGRAPP (5: VISAPP), pp. 233–241.

Emami, J., Nugues, P., Elnagar, A., Afyouni, I., 2022. Arabic image captioning using pre-training of deep bidirectional transformers. In: Proceedings of the 15th International Conference on Natural Language Generation, pp. 40–51.

Farghaly, A., Shaalan, K., 2009. Arabic natural language processing: Challenges and solutions. ACM Trans. Asian Lang. Infor. Process. (TALIP) 8 (4), 1–22.

GUIzero. https://lawsie.github.io/guizero

Habash, N., 2010. Introduction to Arabic Natural Language Processing. Synthesis Lectures on Human Language Technologies. Springer International Publishing. URL https://books.google.com.sa/books?id=Y9Q5zwEACAAJ.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778.

Hejazi, H., Shaalan, K., 2021. Deep learning for arabic image captioning: A comparative study of main factors and preprocessing recommendations. Int. J. Adv. Comput. Sci. Appl. 12 (11).

Hodosh, M., Young, P., Hockenmaier, J., 2013. Framing image description as a ranking task: Data, models and evaluation metrics. J. Artif. Intell. Res. 47, 853–899.

Jindal, V., 2017. A deep learning approach for arabic caption generation using roots-words. In: Thirty-First AAAI Conference on Artificial Intelligence.

Jindal, V., 2018. Generating image captions in arabic using root-word based recurrent neural networks and deep neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32.

Kiros, R., Salakhutdinov, R., Zemel, R., 2014. Multimodal neural language models. In: International Conference on Machine Learning, PMLR, pp. 595–603.

Li, G., Zhu, L., Liu, P., Yang, Y., 2019. Entangled transformer for image captioning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 8928–8937.

Lin, C.-Y., 2004. Rouge: A package for automatic evaluation of summaries. In: Text Summarization Branches Out, pp. 74–81.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L., 2014. Microsoft coco: Common objects in context. In: European Conference on Computer Vision. Springer, pp. 740–755.

Luo, Y., Ji, J., Sun, X., Cao, L., Wu, Y., Huang, F., Lin, C.-W., Ji, R., 2021. Dual-level collaborative transformer for image captioning. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 2286–2293.

Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J.R., Bethard, S., McClosky, D., 2014. The stanford corenlp natural language processing toolkit. In: Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp. 55–60.

Mualla, R., Alkheir, J., 2018. Development of an arabic image description system. Int. J. Comput. Sci. Trends Technol. (IJCST) 6, 205–213.
Obeid, O., Zalmout, N., Khalifa, S., Taji, D., Oudah, M., Alhafni, B., Inoue, G., Eryani, F., Erdmann, A., Habash, N., 2020. Camel tools: An open source python toolkit for arabic natural language processing. In: Proceedings of the 12th Language Resources and Evaluation Conference, pp. 7022–7032.
Papineni, K., Roukos, S., Ward, T., Zhu, W.-J., 2022. Bleu: a method for automatic evaluation of machine translation. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, pp. 311–318.
Qi, P., Zhang, Y., Zhang, Y., Bolton, J., Manning, C.D., 2020. Stanza: A python natural language processing toolkit for many human languages, arXiv preprint arXiv:2003.07082.
Raganato, A., Tiedemann, J., 2108. An analysis of encoder representations in transformer-based machine translation. In: Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, The Association for Computational Linguistics.
Sharma, P., Ding, N., Goodman, S., Soricut, R., 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 2556–2565.
Shoukry, A., Rafea, A., 2012. Preprocessing egyptian dialect tweets for sentiment mining. In: Fourth Workshop on Computational Approaches to Arabic-Script-based Languages, pp. 47–56.
Sutskever, I., Vinyals, O., Le, Q.V., 2014. Sequence to sequence learning with neural networks. Adv. Neural Infor. Process. Syst. 27.
Tan, M., Le, Q., 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In: International Conference on Machine Learning, PMLR, pp. 6105–6114.
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need. Adv. Neural Infor. Process. Syst. 30.
Vedantam, R., Lawrence Zitnick, C., Parikh, D., 2015. Cider: Consensus-based image description evaluation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4566–4575.
Vinyals, O., Toshev, A., Bengio, S., Erhan, D., 2015. Show and tell: A neural image caption generator. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3156–3164.
Wang, Q., Li, B., Xiao, T., Zhu, J., Li, C., Wong, D.F., Chao, L.S., 2019. Learning deep transformer models for machine translation, arXiv preprint arXiv:1906.01787.
Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., Bengio, Y., 2015. Show, attend and tell: Neural image caption generation with visual attention. In: International Conference on Machine Learning, PMLR, pp. 2048–2057.
Zhou, Y., Zhang, Y., Hu, Z., Wang, M., 2021. Semi-autoregressive transformer for image captioning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 3139–3143.