

Article

A Metamorphic Testing Approach for Assessing Question Answering Systems

Kaiyi Tu, Mingyue Jiang * and Zuohua Ding

School of Information Science and Technology, Zhejiang Sci-Tech University, Hangzhou 310018, China; 201930605023@mails.zstu.edu.cn (K.T.); zuohuading@zstu.edu.cn (Z.D.)

* Correspondence: mjiang@zstu.edu.cn

Abstract: Question Answering (QA) enables the machine to understand and answer questions posed in natural language, which has emerged as a powerful tool in various domains. However, QA is a challenging task and there is an increasing concern about its quality. In this paper, we propose to apply the technique of metamorphic testing (MT) to evaluate QA systems from the users' perspectives, in order to help the users to better understand the capabilities of these systems and then to select appropriate QA systems for their specific needs. Two typical categories of QA systems, namely, the textual QA (TQA) and visual QA (VQA), are studied, and a total number of 17 metamorphic relations (MRs) are identified for them. These MRs respectively focus on some characteristics of different aspects of QA. We further apply MT to four QA systems (including two APIs from the AllenNLP platform, one API from the Transformers platform, and one API from CloudCV) by using all of the MRs. Our experimental results demonstrate the capabilities of the four subject QA systems from various aspects, revealing their strengths and weaknesses. These results further suggest that MT can be an effective method for assessing QA systems.



Citation: Tu, K.; Jiang, M.; Ding Z. A Metamorphic Testing Approach for Assessing Question Answering Systems. *Mathematics* **2021**, *9*, 726. <https://doi.org/10.3390/math9070726>

Academic Editors: Vassilis C. Gerogiannis and Tadashi Dohi

Received: 8 February 2021
Accepted: 25 March 2021
Published: 28 March 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: textual question answering; visual question answering; metamorphic testing; metamorphic relations; quality assessment

1. Introduction

Question answering (QA) [1,2] focuses on returning right answers to given questions. Among various QA systems, the textual question answering (TQA) and visual question answering (VQA) represent a typical paradigm that enables the machine to answer a question in natural language by referring to the given contents (i.e., text or image). As shown in Figure 1, TQA [3] focuses on answering a question about a passage of text, which is also known as an NLP task of machine reading comprehension; while VQA [4] focuses on answering a question based on an image, which leverages techniques from the domains of NLP and computer vision. Both TQA and VQA have various potential applications. For example, TQA has been widely adopted by conversational agents [5] and customer service support [6]; VQA has a broad range of applications in the autonomous agents and virtual assistants [7]. On the other hand, a large number of neural network models have been created for implementing both TQA and VQA. For instances, BiDAF [8], BERT [9], RoBERTa [10] for TQA, and ViLBERT [11] for VQA.

Due to the importance and popularity of QA, it is critical to properly assess QA systems in order to demonstrate their capabilities and limitations. QA systems are commonly evaluated by a test dataset. However, the dataset may not necessarily be representative of the real world. Due to this, various different approaches have been proposed and applied to evaluate QA systems, revealing a series of problems concerning different aspects. Jia et al. [12] proposed an adversarial evaluation scheme to investigate whether QA can answer questions about passages containing adversarially inserted sentences, and their experimental results revealed that the QA models under investigation had poor performance. Divyansh et al. [13] investigated popular QA benchmarks and then revealed that

TQA might ignore the passage of text when answering questions. Mudrakarta et al. [14] proposed to apply the notion of attribution to generate adversarial questions, based on which it was observed that QA systems often ignored important terms in questions. On the other hand, recent studies investigated the robustness of QA systems [15,16] and further proposed strategies for improving their robustness [17].

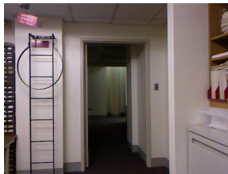
Textual Question Answering (TQA)	Visual Question Answering (VQA)
Passage: New York City has over 28,000 acres (110 km ²) of municipal parkland and 14 miles (23 km) of public beaches. Parks in New York City include Central Park, Prospect Park, Flushing Meadows Corona Park, Forest Park, and Washington Square Park. The largest municipal park in the city is Pelham Bay Park with 2700 acres (1093 ha).	Image: 
Question: How long are all the public beaches together in miles?	Question: how many ladders are in the picture?
Answer: 14.	Answer: 1.

Figure 1. Textual question answering (TQA) and visual question answering (VQA): TQA answers a question with reference to a passage, while VQA answers a question with respect to an image.

This study focuses on assessing TQA and VQA systems from the users' perspective in order to reveal to which degree QA systems satisfy the users' expectations. This kind of assessment is helpful for the users to better understand QA systems such that they are able to select appropriate QA systems for their specific needs. To this end, we propose to adopt the technique of metamorphic testing (MT). MT is a property based testing technique, which has shown promising effectiveness in various software engineering activities, such as testing [18], fault localization [19], and program repair [20,21]. The key component of MT is metamorphic relations (MRs), which encode system properties via the relationship among multiple related inputs and outputs. MT is originally applied for software verification. In recent year, it has been successfully extended to software validation and system comprehension [22,23].

In this study, we identify a total number of 17 MRs for QA systems. These MRs respectively focus on different aspects of TQA and VQA, which can help the users to understand the capability of TQA and VQA systems from different perspectives, and can also provide guidances for the users to select appropriate systems to satisfy their specific needs. We conduct experiments by employing four QA systems (two TQA APIs provided by AllenNLP [24] and Transformers [25], and two VQA APIs provided by AllenNLP and CouldCV) using all of the MRs, demonstrating the capabilities and limitations of the QA systems under investigation. To summarize, the paper makes three major contributions.

- We proposed to apply the technique of metamorphic testing to assess QA systems from the users' perspectives, and presented 17 MRs by considering different aspects of QA systems.
- We conducted experiments on four common QA systems (two TQA systems and two VQA systems), demonstrating the feasibility and effectiveness of MT in assessing QA systems.
- We conducted comparison analysis among subject QA systems to reveal their capabilities of understanding and processing the input data, and also demonstrated how the analysis results can help the user to select appropriate QA system for their specific needs.

The remainder of the paper is organized as follows. Section 2 introduces the technique of metamorphic testing. Section 3 clarifies the overall approach, and Section 4 presents a list of MRs identified for QA systems. Our experimental setup is introduced in Section 5, and the experimental results are presented and analyzed in Section 6. Section 7 discusses related work, and Section 8 concludes the present study.

2. Metamorphic Testing

Metamorphic testing (MT) [26,27] is a property based testing technique. MT proposes to describe the necessary properties of the target system through the relationships among inputs and outputs of multiple executions. Such properties are expressed by metamorphic relation (MRs). Specifically, an MR describes how to construct the follow-up input from the given input (which is known as the source input), and also encodes the relationship among the source and follow-up outputs (namely, the outputs for the source and follow-up inputs respectively). As an example for illustration, consider the program *Max* that implements the algorithm of finding the maximum value among two input values. An MR for *Max* can be “Suppose that the source input is $t_s = (x, y)$, where x and y can be arbitrary numeric values, and the follow-up input t_f is constructed by swapping the two input values of t_s (that is, $t_f = (y, x)$). As a result, the source and follow-up outputs are expected to be identical”.

Generally, MRs can be identified by referring to the system’s requirements or based on the users’ expectations on the system. Given an MR and a set of its source inputs (which can be generated by arbitrary strategies), MT can be conducted as below. At first, the corresponding follow-up inputs are constructed based on the source inputs according to the MR. After that, for every group of source and follow-up inputs, MT respectively runs the target program on both source and follow-up inputs, yielding the source and follow-up outputs. MT finally checks each group of source and follow-up inputs and outputs against the relevant MR to see whether or not the MR is violated. Any group of source and follow-up inputs with which the program violates the MR is regarded to incur an MR violation. Specifically, an MR violation is an indicator of the existence of defects in the target system if the relevant MR is identified with reference to the system’s requirements. Nevertheless, an MR violation reveals either the existence of defects or the discrepancies between the system behavior and the users expectations if the MR is identified with respect to the users’ expected characteristics of the system.

Different from traditional testing techniques that check the correctness of the output of individual inputs, MT checks the satisfaction of MRs on individual groups of source and follow-up executions. Because of this, MT can be conducted without using oracles, and has been applied for software verification and validation [18,22] as well as for helping users to understand the system behaviors [23]. It is also noted that after MRs are identified, the whole procedure of MT can be easily automated.

3. Methodology

This study proposes to apply MT to evaluate QA systems by considering different users’ requirements. An overview of the approach is presented in Figure 2. Given a set of source inputs (namely, passage-question pairs for TQA and image-question pairs for VQA) and a list of MRs, a corresponding set of perturbed passage-question pairs and image-question pairs are generated, which are respectively the follow-up inputs for TQA and VQA. By executing the TQA and VQA systems with source and follow-up inputs that are relevant to individual MRs, their source and follow-up answers are collected. Since both TQA and VQA provide a phrase or a sentence as an output answer, we conduct semantic similarity analysis on groups of original and follow-up answers with respect to the relevant MR to determine the testing result. At last, for each MR and every TQA and VQA system under investigation, we calculate the violation rate, which denotes the rate of occurrence of MR violations. A higher violation rate indicates a higher degree to which the system’s behaviors deviate from the users’ expectations. Based on the evaluation data, we further conduct comparison analysis to reveal the capabilities of QA systems under investigation. Our analysis mainly focuses on three aspects: both TQA and VQA’s capabilities of understanding and answering questions, TQA’s capabilities of understanding and processing passages, and VQA’s capabilities of understanding and processing images. We also demonstrate how our analysis results can help the users to select appropriate QA systems according to their specific needs.

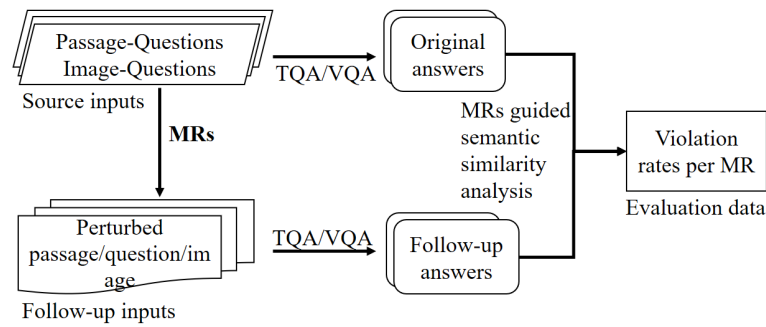


Figure 2. Overview of how metamorphic testing (MT) is applied to evaluate QA systems.

The key task of applying MT to QA systems lies in the identification of MRs by considering the characteristics of QA systems as well as the users' expectations on these systems. Moreover, upon the identification of MRs, the whole evaluation procedure can be automated.

4. Metamorphic Relations of Question Answering Systems

In order to evaluate QA systems by MT, we defined a series of MRs. These MRs consider the users' expected characteristics of QA systems, and thus the satisfaction and violation of these MRs can help users to better understand the capability and limitations of QA systems. In total, 17 MRs are identified, each of which focuses on some aspects of QA. This section presents the details of these MRs, and also gives illustrative examples for some MRs.

4.1. Output Relationships

Let t_s and t_f be a group of source and follow-up inputs of a QA system with respect to an MR, and let A_s and A_f be the corresponding source and follow-up outputs. In this study, we consider the following relationships between A_s and A_f .

- *Equivalent*: A_s and A_f are regarded to be equivalent if they have similar semantics.
- *Different*: A_s and A_f are regarded to be different if they have distinct semantics.

In order to determine whether two answers A_s and A_f have similar semantics, we first transform them into vector representations. This is done by employing the bert-as-service API [28], which encodes a sentence with a fixed length vector by using the BERT model [9]. BERT is a pre-trained transformer network built upon the attention mechanism [29]. The model has multiple layers, each of which consists of an attention sub-layer and a feed-forward network sub-layer. The former helps the model to gain a broad range of information from the input. For an input, the attention sub-layer extracts three vectors, namely, the query vector, key vector and value vector, and packs them together into matrices Q , K , and V , respectively. Based on this, it conducts the self-attention calculation as below [29].

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \quad (1)$$

where d_k represents the dimension of keys of the input, and *softmax* is a learned normalized exponential function. Specifically, BERT adopts a multi-head attention mechanism, which concatenates multiple attention calculations of linearly transformed queries, keys and values. The output of the attention sub-layer is provided for another sub-layer that contains a feed-forward network, which is responsible for conducting linear transformations as below [29].

$$FFN(x) = max(0, xW_1 + b_1)W_2 + b_2. \quad (2)$$

Based on the digital vectors yielded by BERT for A_s and A_f , we further apply the cosine similarity analysis [30] to decide whether or not they are semantically equivalent. Suppose that the size of the resulting vectors is n , let $vs = [vs_1, \dots, vs_n]$ and $vf = [vf_1, \dots, vf_n]$ be the vectors representing A_s and A_f , respectively. The semantic similarity of A_s and A_f is measured by

$$\text{sim}(A_s, A_f) = \frac{\sum_{i=1}^n vs_i \times vf_i}{\sqrt{\sum_{i=1}^n vs_i^2} \sqrt{\sum_{i=1}^n vf_i^2}}. \quad (3)$$

As a result, a similarity score that is higher than a threshold value indicates the equivalence of A_s and A_f in terms of their semantics.

4.2. MRs for QA Systems

The input of TQA consists of a passage and a question, and the input of VQA contains an image and a question. As such, we use P_s (or I_s) and Q_s to denote the passage (or image) and question in t_s , and use P_f (I_f) and Q_f to denote the corresponding information in t_f . That is, $t_s = (P_s, Q_s)$ and $t_f = (P_f, Q_f)$ for TQA, while $t_s = (I_s, Q_s)$ and $t_f = (I_f, Q_f)$ for VQA. Different MRs may operate on different input parameters of t_s to construct t_f , leading to discrepancies between t_s and t_f . According to this, we classify all MRs into three categories, which are summarized in Table 1 and are explained as below.

- MR1.x has $t_s = (P, Q_s)$ and $t_f = (P, Q_f)$ or $t_s = (I, Q_s)$ and $t_f = (I, Q_f)$. That is, t_s and t_f of MR1.x have the same P (or I), but different questions Q_s and Q_f . This category of MRs operates on Q_s to construct Q_f . Hence, they focus on QA's capability of understanding and answering questions
- MR2.x has $t_s = (P_s, Q)$ and $t_f = (P_f, Q)$. That is, t_s and t_f of MR2.x have the same Q, but different passages P_s and P_f . This category of MRs operate on P_s to construct P_f , and they focus on the TQA's capability of processing and understanding the input passage.
- MR3.x has $t_s = (I_s, Q)$ and $t_f = (I_f, Q)$. That is, t_s and t_f of MR3.x have the same Q, but different images I_s and I_f . This category of MRs operate on I_s to construct I_f , and they concentrate on the VQA's capability of processing and understanding the input image.

Table 1. Summary of metamorphic relations (MRs).

	Source and Follow-Up Inputs	Number of MRs
MR1.x	$t_s = (P, Q_s), t_f = (P, Q_f)$ $t_s = (I, Q_s), t_f = (I, Q_f)$	4 (MR1.1–MR1.4)
MR2.x	$t_s = (P_s, Q), t_f = (P_f, Q)$	5 (MR2.1–MR2.5)
MR3.x	$t_s = (I_s, Q), t_f = (I_f, Q)$	8 (MR3.1–MR3.8)

4.2.1. MR1.x

This category of MRs are designed to investigate the QA's capability of understanding and answering questions. For each MR, t_s and t_f use the same input passage or image but different questions, that is, (P, Q_s) and (P, Q_f) for TQA, while (I, Q_s) and (I, Q_f) for VQA. Different MRs alter Q_s in different ways to construct Q_f and also encode the relationship that is expected to be satisfied by A_s and A_f . We identify four MRs, which are described as follows.

MR1.1 (Capitalization): Q_f is constructed by replacing lowercase letters of Q_s with the corresponding uppercase letters. As a result, A_f is expected to be equivalent to A_s .

MR1.2 (Rephrasing comparative question): Suppose that Q_s contains comparative phrases. Q_f is constructed by rephrasing Q_s without changing the meaning of Q_s . As a result, A_f is expected to be equivalent to A_s .

MR1.3 (Replacing the comparative word with its antonym): Suppose that Q_s contains comparative words. Q_f is constructed by replacing a comparative word in Q_s with its

antonym such that Q_f expresses a different meaning from Q_s . As a result, A_f is expected to be different from A_s .

MR1.4 (Changing the subject of a question): Q_f is constructed by changing the subject of Q_s with another noun. This change leads to different meanings of these two questions. As a result, A_f is expected to be different from A_s .

Table 2 shows some illustrative examples of Q_s and Q_f of MR1.1–MR1.4, where Q_f is highlighted with underlines. For each MR, the interpretation of MR violations is also presented.

Table 2. Interpretations and illustrations of MR1.x.

MRs	Interpretation of MR Violation	Examples of Pairs of (Q_s, Q_f)
MR1.1	QA is sensitive to the letter case of a question.	What song won Best R&B Performance? <u>WHAT SONG WON BEST R&B PERFORMANCE?</u>
MR1.2	QA is sensitive to questions using different comparative descriptions.	In how many years will A remain higher than B in population? <u>In how many years will B remain lower than A in population?</u>
MR1.3	QA cannot properly understand the questions expressed via different comparative words.	What type of residents tend to be more fluent than rural ones? <u>What type of residents tend to be less fluent than rural ones?</u>
MR1.4	QA cannot properly understand the questions involving different subjects.	What is the name of the final studio album from Destiny's Child? <u>What is the name of the final studio album from Bob's Child?</u>

4.2.2. MR2.x

This category of MRs are identified to study the TQA's capability of processing and understanding the input passage. For each MR, the source input is $t_s = (P_s, Q)$, and the corresponding follow-up input is $t_f = (P_f, Q)$. Every MR proposes a way of altering P_s to construct P_f and also predicts the relationships between the corresponding A_s and A_f . Table 3 summarizes this category of MRs, the details of which are presented as below.

Table 3. Summary of MR2.x.

MRs	Interpretation of MR Violations	Operation Used for Constructing P_f
MR2.1	TQA is sensitive to the letter case of a passage.	Capitalization
MR2.2	TQA is sensitive to the order of sentences in a passage.	Order reversing
MR2.3	TQA is sensitive to the added sentences that are irrelevant to the question.	Addition
MR2.4	TQA is sensitive to the deleted sentences that are irrelevant to the question.	Removal
MR2.5	TQA is incapable of properly understanding and processing the question related texts.	Replacement

MR2.1 (Capitalization): P_f is constructed by replacing lowercase letters of P_s with the corresponding uppercase letters. As a result, A_f is expected to be equivalent to A_s .

MR2.2 (Reversing the order of sentences): P_f is constructed by reversing the order of sentences of P_s . As a result, A_f is expected to be equivalent to A_s .

MR2.3 (Addition of irrelevant sentences): P_f is constructed by adding some sentences that are irrelevant to the question into P_s . As a result, A_f is expected to be equivalent to A_s .

MR2.4 (Removal of irrelevant sentences): P_f is constructed by removing sentences that are irrelevant to the question from P_s . As a result, A_f is expected to be equivalent to A_s .

MR2.5 (Replacing the answer-related words): Suppose that A_s is a numeric value, which is an answer to questions of types of how many, how old, how long, or when. P_f is constructed by replacing A_s in P_s with $A_s + n$, where n is a randomly selected numeric

constant, which makes $A_s + n$ a numeric value that is different from A_s and is also unique in P_s . As a result, A_f is expected to be different from A_s but is equal to $A_s + n$.

MR2.5 is designed by considering a special case where TQA returns a numeric value as an answer to a given question. In this study, we consider four types of questions, namely, how many, how old, how long, and when. An illustrative example of MR2.5 is presented in Table 4, which demonstrates the way of constructing P_f based on both P_s and A_s . Obviously, MR2.5 can only be applied to source inputs that contain the aforementioned four types of questions.

Table 4. Example P_s and P_f of MR2.5 (n is set to be 3).

<p>After graduating from high school, West received a scholarship to attend Chicago's American Academy of Art in 1997 and began taking painting classes, but shortly after transferred to Chicago State University P_s: to study English. He soon realized that his busy class schedule was detrimental to his musical work, and at 20 he dropped out of college to pursue his musical dreams. This action greatly displeased his mother, who was also a professor at the university.</p> <p>Q_s: How old was Kanye when he dropped out of college?</p> <p>A_s: 20</p>	<p>After graduating from high school, West received a scholarship to attend Chicago's American Academy of Art in 1997 and began taking painting classes, but shortly after transferred to Chicago State University P_f: to study English. He soon realized that his busy class schedule was detrimental to his musical work, and at 23 he dropped out of college to pursue his musical dreams. This action greatly displeased his mother, who was also a professor at the university.</p>
---	--

4.2.3. MR3.x

This category of MRs are identified for evaluating the VQA's capability of processing and understanding the input image. For each MR, the source input is $t_s = (I_s, Q)$, and the corresponding follow-up input is $t_f = (I_f, Q)$. Accordingly, each MR designs a way of altering I_s to construct I_f and also predicts the relationships between source and follow-up outputs. Researchers have proposed a series of operations, such as image scaling and image rotation, to perturb images for evaluating deep neural network based models [31]. In this study, we consider 2D input images, and identify MRs by adopting some of the operations.

We first consider the rotation operation. To rotate an image with a given angle, a rotation matrix is constructed and applied on the image (<https://github.com/jrosebr1/imutils>, accessed on 8 October 2020). Suppose that c is the center of the rotation, θ is the rotation angle, and x denotes the scale factor. The rotation matrix is as follows:

$$\begin{bmatrix} \alpha & \beta & (1 - \alpha) * c.x - \beta * c.y \\ -\beta & \alpha & \beta * c.x + (1 - \alpha) * c.y \end{bmatrix}, \quad (4)$$

where $\alpha = x * \cos\theta$ and $\beta = x * \sin\theta$. Three MRs, namely, MR3.1–MR3.3, are identified by adopting varying rotation angles.

MR3.1: I_f is constructed by rotating I_s by 90 degrees. As a result, A_f is expected to be equivalent to A_s .

MR3.2: I_f is constructed by rotating I_s by 180 degrees. As a result, A_f is expected to be equivalent to A_s .

MR3.3: I_f is constructed by rotating I_s by 270 degrees. As a result, A_f is expected to be equivalent to A_s .

We next consider the changing of RGB images into grayscale images. This can be implemented by using the ITU-R 601-2 (Luma transform <https://github.com/python-pillow/Pillow>, accessed on 8 October 2020), where each pixel of an image is expressed as 8-bits, and is transformed as below.

$$L = R * 299/1000 + G * 587/1000 + B * 114/1000, \quad (5)$$

where R , G , and B are the RGB values in range of 0–255, and L is the resulting single channel output. Based on this, MR3.4 is identified.

MR3.4: Suppose that I_s is a RGB image. I_f is constructed by converting I_s to its corresponding grayscale image. As a result, A_f is expected to be equivalent to A_s .

We further consider another two types of images operations, image flipping and resizing. Flipping an image utilizes a similar method as for rotating images but with different parameter configurations, while resizing an image can be implemented by adopting scale factors along the horizontal and vertical axes. Based on these two types of operations, the following four MRs are identified.

MR3.5: I_f is constructed by flipping I_s horizontally. As a result, A_f is expected to be equivalent to A_s .

MR3.6: I_f is constructed by flipping I_s vertically. As a result, A_f is expected to be equivalent to A_s .

MR3.7: I_f is constructed by magnifying the size of I_s by 1.5 times. As a result, A_f is expected to be equivalent to A_s .

MR3.8: I_f is constructed by reducing the size of I_s by 1.5 times. As a result, A_f is expected to be equivalent to A_s .

5. Experimental Setup

A series of experiments were conducted to evaluate four QA systems by using all of the 17 MRs. This section presents our experimental setup, including the implementation of MRs, our subject QA systems, the datasets used in the experiments, and the source inputs of MRs.

5.1. MRs Implementation

All of the identified MRs were implemented in order to automatically evaluate QA systems by MT. Some specific MR implementations are presented as below.

MR1.3: MR1.3 replaces the comparative word in Q_s with its antonym for constructing Q_f . To this end, we applied nltk (<http://www.nltk.org/>, accessed on 23 October 2020) for part-of-speech tagging, which can identify comparative form of an adjective or adverb in Q_s . We further searched the antonym of the given word by using PyDictionary (<https://github.com/geekpradd/PyDictionary>, accessed on 23 October 2020).

MR1.4: MR1.4 changes the subject of Q_s to construct Q_f . In this study, we treated a word of Q_s representing the entity of PERSON as the subject of Q_s . To identify and change the subject of Q_s , we applied the Named Entity Recognizer StanfordNERTagger (<https://nlp.stanford.edu/software/CRF-NER.html>, accessed on 2 November 2020). Given a Q_s , StanfordNERTagger was first applied to extract the PERSON entity from Q_s . If the PERSON entity was successfully identified, we further replaced it with another PERSON entity that was not included in the passage.

MR3.1–MR3.3: These MRs rotate I_s to construct I_f . To automate this procedure, we utilized a package called *imutils* (<https://github.com/jrosebr1/imutils>, accessed on 2 November 2020), which provides a function *rotate_bound* for rotating images by given degrees.

MR3.4–MR3.8: MR3.4 changes a RGB image to a grayscale image, MR3.5 and MR3.6 flip I_s to construct I_f , while MR3.7 and MR3.8 enlarge (shrink) I_s to construct I_f . To implement these MRs, we used two libraries PIL (<https://github.com/python-pillow/Pillow>, accessed on 8 October 2020) and OpenCV (<https://opencv.org/>, accessed on 8 October 2020).

To automatically check the relationship of A_s and A_f , we employed the bert-as-service API [28], which determines the degree to which the given two sentences have similar semantics. This API represented a sentence as a fixed length vector according to BERT [9], based on which we calculated the cosine similarity of vectors of A_s and A_f to determined whether they are equivalent or different.

5.2. Subject QA Systems

In the experiments, two TQA APIs and two VQA APIs were employed as our subject systems, which are listed as below:

- AllenNLP-TQA (<https://demo.allennlp.org/reading-comprehension>, accessed on 10 November 2020), which is a TQA API at the AllenNLP platform [24]. AllenNLP-TQA is an implementation of the BiDAF model [8] with ELMo embeddings.
- Transformers-TQA (<https://github.com/huggingface/transformers>, accessed on 10 November 2020), which is a TQA API at the Transformers platform [25]. This API is built upon the the DistilBERT model [32].
- AllenNLP-VQA (<https://demo.allennlp.org/visual-question-answering>, accessed on 10 November 2020), which is a VQA API at the AllenNLP platform [24]. This API is built upon the ViLBERT model [11].
- CloudCV-VQA (<http://vqa.cloudcv.org/>, accessed on 10 November 2020), which is an API provided by the CloudCV. This API utilizes the Pythia model [33].

5.3. Datasets and Source Inputs of MRs

The SQuAD 2.0 dataset [34] was used for preparing source inputs of TQA. SQuAD2.0 contains over 150,000 questions. For VQA, we utilized the DAQUAR dataset [35], which contains 1449 images and 12,468 questions. A source input obtained from the SQuAD 2.0 dataset was a passage-question pair, while a source input extracted from the DAQUAR dataset was an image-question pair.

Nine MRs, namely, MR1.1–MR1.4 and MR2.1–MR2.5, were used to evaluate TQA systems, while 12 MRs, namely, MR1.1–MR1.4 and MR3.1–MR3.8, were used to evaluated TQA systems. Each MR was applied to individual source inputs in order to generate the relevant follow-up inputs. Note that MRs may not be applicable to some source inputs due to its preconditions and the operations used for constructing follow-up inputs. For example, MR1.3 operates on comparative words, and thus it cannot be applied to source inputs whose questions contain no comparative word. As a result, different MRs may have varying numbers of groups of source and follow-up inputs. In total, over 50,000 groups of source and follow-up inputs are used for evaluating TQA systems, and over 80,000 groups of source and follow-up inputs are used for evaluating VQA systems.

6. Results and Analysis

In this section, the MT results of evaluating the four subject QA systems are presented. Then, the capabilities of our subject QA systems are analyzed and discussed with respect to relevant MRs.

6.1. MT Results for QA Systems

To evaluate QA systems, the violation rate (VR) was used as the evaluation metric. Given an MR and a QA system, let y be the total number of groups of source and follow-up inputs of the MR that were applied to test the QA system, and x be the number of groups of source and follow-up inputs with which the system violated the MR. The VR of this QA system with respect to this MR is $\frac{y}{x}$. Obviously, a lower VR value indicated a higher degree to which the QA system conformed to the relevant MR, revealing a higher satisfaction of the users' needs. Oppositely, a higher VR value denoted that the QA system was more sensitive to the MR operations, and thus was more likely to produce unexpected answers for the given question. Particularly, a violation rate of 0 means that no violation of the relevant MR was revealed in our experiments, suggesting that the system was likely to be robust with respect to the MR and all of its source and follow-up inputs.

Table 5 summarizes the VR values of four QA systems with respect to all identified MRs. It is observed that all of the QA systems violated some MRs with different degrees, showing VR values ranging from 0.61% to 92.98%. Consider, for example, the VR value (65.10%) of AllenNLP-TQA with respect to MR1.1. This VR value indicated that among all groups of source and follow-up inputs of MR1.1 that were applied to test AllenNLP-TQA, 65.10% revealed MR violations. It can also be found from Table 5 that every QA system violated different MRs with varying VR values and that different QA systems also violated

the same MR with varying VR values. This results further suggest that the proposed MRs were capable of reflecting the QA systems' capability from different aspects.

Table 5. Violation rates of question answering (QA) systems (“*” denotes that the number of source input is 0, while “-” means that the relevant MR is not applicable to the system).

	AllenNLP-TQA	Transformers-TQA	AllenNLP-VQA	CloudCV-VQA
MR1.1	65.10%	91.11%	10.34%	20.14%
MR1.2	42.86%	7.14%	*	*
MR1.3	92.98%	3.51%	*	*
MR1.4	86.97%	68.45%	*	*
MR2.1	67.37%	86.86%	-	-
MR2.2	8.12%	6.14%	-	-
MR2.3	2.05%	0.61%	-	-
MR2.4	3.73%	1.18%	-	-
MR2.5	33.18%	23.99%	-	-
MR3.1	-	-	81.10%	66.14%
MR3.2	-	-	80.79 %	62.71%
MR3.3	-	-	33.58%	66.42%
MR3.4	-	-	55.25 %	47.68%
MR3.5	-	-	48.10%	20.86%
MR3.6	-	-	79.54%	62.74%
MR3.7	-	-	32.06%	29.73%
MR3.8	-	-	32.68%	31.51%
Average	44.71%	32.11%	56.68%	48.47 %

6.2. Further Analysis

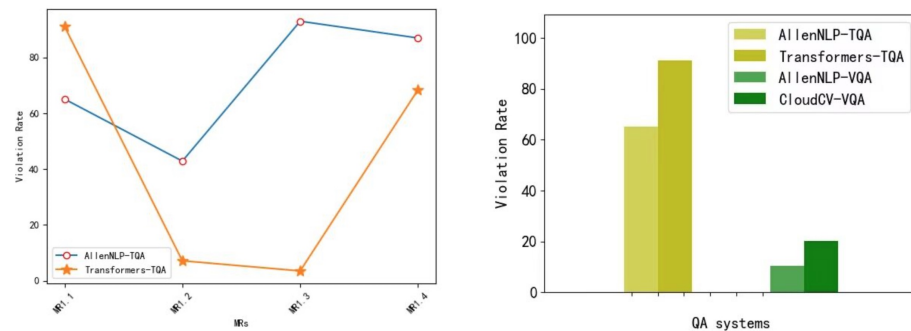
Based on the MT results reported in Table 5, an in-depth analysis was conducted to reveal the capabilities of the four QA systems from different perspectives. Each VR value reported in Table 5 represents the extent to which a system deviated from the properties specified by the relevant MR. Furthermore, as described and explained in Section 4, different MRs handled varying input parameters and also referred to different capabilities of QA. More importantly, a system may have performed well in some aspects but may have had bad performance in some other aspects, while different users may have had concern with varying QA capabilities due to their distinct application scenarios. It was therefore important for the users to know the strength and weakness of different systems such that appropriate systems could be selected to satisfy their needs. Because of this, we compared subject QA systems by inspecting VR values of MRs pertaining to specific QA capabilities in order to reveal the strength and weakness of individual systems from different aspects.

6.2.1. QA's Capability of Understanding and Answering Questions

Both TQA and VQA have to understand the question and then to give an appropriate answer to the question. When using these systems, the users may want to know which QA system has a better capability of processing questions. Four of the proposed MRs, namely, MR1.1–MR1.4, focus on this aspect by describing the relationships among source and follow-up inputs that differ exactly in the input questions.

Figure 3 compares different TQA systems and VQA systems based on MR1.1–MR1.4. As shown in Figure 3a, Transformers-TQA had lower VR values than AllenNLP-TQA for three out of four MRs. It can be further observed from Table 5 that the average VR value of Transformers-TQA on these four MRs was also much lower than that of AllenNLP-TQA. Therefore, as compared with AllenNLP-TQA, Transformers-TQA exhibited better capabilities of understanding and answering questions. Similarly, as shown in Figure 3b, the two VQA systems also had varying violation rates for MR1.1 (the other three MRs had 0 source input for VQA and thus no data was collected). As compared with CloudCV-VQA,

AllenNLP-TQA had a relatively lower violation rate with respect to MR1.1, suggesting that AllenNLP-VQA was more robust to the letter case of input questions. Moreover, Figure 3b also showed that the two VQA systems under investigation were of better capability of handling questions with lowercase or uppercase letters than the two TQA systems, because the former two had much lower VR values (namely, 10.34% and 20.14%) than the latter (namely, 65.10% and 91.11%) with respect to MR1.1.



(a) Violation rates of TQA with respect to MR1.x. (b) Violation rates of QA with respect to MR1.1.

Figure 3. Violation rates of QA with respect to MR1.1–MR1.4.

6.2.2. TQA's Capability of Understanding and Processing Passages

TQA answers a given question based on a passage, it thus needs to understand and process the passage for exacting information related to the given question. We defined five MRs, MR2.1–MR2.5, for investigating TQA's capability of understanding and processing input passages.

Figure 4 compares the violation rates of the two TQA systems (AllenNLP-TQA and Transformers-TQA) with respect to MR2.1–MR2.5. Firstly, both TQA systems had much lower violation rates for MR2.2–MR2.5 (VR values are lower than 35%) as compared with those for MR2.1 (VR values are higher than 65%). These results reveal that the two TQA systems were much more robust to the adding, removing or replacing some contents of the input passage, but were less robust to the conversion of lowercase letters to uppercase letters of the input passage. Secondly, Transformers-TQA had similar violation rates as AllenNLP-TQA for MR2.2–MR2.4 (the discrepancies between the VR values of the two systems with respect to individual MRs were about 2%), but had a very different violation rates from AllenNLP-TQA for the other two MRs (the VR value of the former was about 20% higher than that of the latter with respect to MR2.1, while the VR value of the former was about 10% lower than that of the latter with respect to MR2.5). In other words, the two TQA systems had equivalent capability of dealing with passages containing sentences of different orders as well as containing more or less irrelevant sentences. Nevertheless, AllenNLP-TQA did better for handling passages containing lowercase or uppercase letters, while Transformers-TQA performed better when dealing with passages containing minor replaced contents.

6.2.3. VQA's Capability of Understanding and Processing Images

While TQA understands and processes the input passage for answering a question, VQA relies on the input image for giving an answer to a question. We identified eight MRs, MR3.1–MR3.8, for investigating the VQA's capability of understanding and processing images.

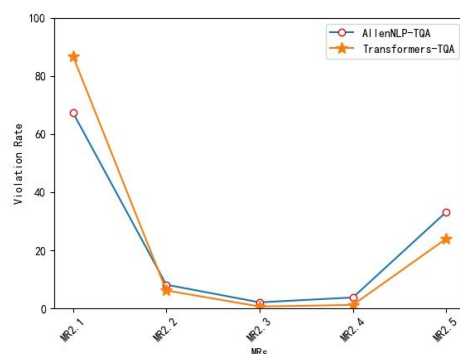


Figure 4. TQA's violation rates with respect to MR2.1–MR2.5.

Figure 5 compares AllenNLP-VQA and CloudCV-VQA with respect to MR3.1–MR3.8. It was observed that except for MR3.3, CloudCV-VQA always had lower violation rates than AllenNLP-VQA, indicating that CloudCV-VQA performed better in terms of MR3.1, MR3.2, MR3.4–MR3.8. On the other hand, both VQA systems had different violation rates for MRs involving the same image perturbation operation, such as rotation and flipping. For example, consider MR3.1–MR3.3, which rotated a source image to construct a follow-up image (but each MR rotated the image by a specific angle, such as 90 degrees, 180 degrees, and 270 degrees). For these three MRs, AllenNLP-VQA had VR values of 81.10%, 80.79% and 33.58%, and CloudCV-VQA had VR values of 66.14%, 62.71% and 66.42%. A similar observation can also be obtained when inspecting these two VQA systems with respect to MR3.4 and MR3.5 that both flipped the source image to construct the follow-up image (but with different flipping directions).

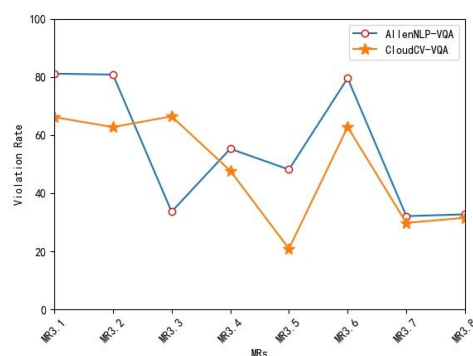


Figure 5. VQA's violation rates with respect to MR3.1–MR3.8.

6.2.4. Further Analysis and Discussion

TQA and VQA had the commonality that they both needed to understand and process the given question. Figure 3b compares our four subject systems with respect to MR1.1, showing that the two VQA systems had relatively better capabilities than the two TQA systems in terms of processing questions containing lowercase or uppercase letters. However, TQA and VQA differed in that the former relied on the passage of text while the latter relied on the image. Concerning these aspects, we respectively used MR2.x and MR3.x for evaluating TQA and VQA. It can still be found from Table 5 that the two TQA systems generally had lower violation rates for MR2.x (which focused on TQA's capability of understanding and processing passages) as compared with the VQA's violation rates for MR3.x (that concentrated on VQA's capability of understanding and processing images). These results indicated that compared with the image processing capability of the two VQA systems, the two TQA systems had better capability of processing passages. Furthermore, Table 5 presents the average violation rates across all applied MRs for individual subject QA systems (as shown in the last row of Table 5). Base on the average VR values, it was

found that the two TQA systems generally performed better than the two VQA system, because the former two had average VR values of 44.71% and 32.11% while the latter two had average VR values of 56.68% and 48.47%.

In summary, the proposed 17 MRs encoded some characteristics of QA system, based on which MT results revealed the capabilities of our subject TQA and VQA systems from different perspectives. On one hand, the MT results reported the VR values for every subject system with respect to individual MRs, which could help the users to gain a better understanding about the capability and limitations of the relevant systems. For example, by inspecting the VR values of AllenNLP-TQA, the users could find that this system was good at extracting the question-related information from the passage either with or without some irrelevant sentences (as suggested by the VR value of 2.05% of MR2.3), but it was very incapable of properly understanding questions containing comparative words (as indicated by the VR value of 92.98% of MR1.3). On the other hand, the MT results supported the comparison of different QA systems by considering different aspects, which thus provided guidance for the user to select appropriate QA systems for their specific needs. For example, if the users wanted to use VQA systems without concerning the use of lowercase or uppercase letters in the question description, they could check the VQA systems' VR values with respect to MR1.1. The reason for this is that MR1.1 encoded the relationship between source and follow-up inputs to reflect to which degree a QA system was sensitive to the letter case of a question. In our experiments, AllenNLP-VQA had a VR value of 10.34%, while CloudCV-VQA had a VR value of 20.14%, with respect to MR1.1. Based on this result, it was natural that the users would utilize AllenNLP-VQA rather than CloudCV-VQA. Note that different users may have had varying needs and expectations on the QA systems, and thus MT results of different MRs should be referred in different application scenarios.

7. Related Work

A large body of studies focus on assessing the QA systems' robustness. In order to construct input data, various strategies have been proposed, such as adversarially inserting sentences into the input passages of TQA [12], perturbing questions with respect to high attribute terms [14], rephrasing questions by applying linguistic variations [36], introducing noises into questions [15,37], and applying universal adversarial triggers [38]. Another line of work focuses on improving or explaining QA systems' robustness. Chen et al. [17] proposed a model for TQA through sub-part alignment, which was able to filter out bad prediction results and thus was of higher robustness, while Patro et al. [16] proposed a collaborative correlated network for providing visual and textual explanations of the VQA's answers. Although robustness is important for evaluation, these studies are orthogonal to our focus on assessing to what degree QA systems satisfy the users' specific expectations. On the other hand, most of existing studies focused on either of TQA or VQA, and proposed strategies for changing only parts of an input (namely, question or passage). Nevertheless, our study proposed a list of MRs, which involve various operations that can be applied to both the input questions and the input passages (input images) of TQA (VQA).

Apart from focusing on the QA systems' robustness, Ribeiro et al. [39] evaluated the logic consistency of QA systems. They transformed a question and also implied the corresponding answer by considering the positive and negative implications caused by the given question with respect to the context. While useful, this method still did not take the other parts of the input (i.e., passages or images) into account, and thus the evaluation was still restricted to parts of the QA's capabilities.

Ribeiro et al. [15] introduced MT to one of the QA systems, namely, TQA, and proposed to use MT for evaluating the TQA's robustness. However, in their work, only one MR was identified, which introduced a specific type of noises (namely, typos) into the input passage or the input question to generate follow-up inputs. In contrast, our study proposed applying MT as a comprehensive evaluation method for both TQA and VQA in a user-oriented way. We have identified a large number of MRs for QA, including MRs

that reflect systems' robustness (such as the MRs adopting the capitalization operation on the inputs), and also MRs that focus on particular system functionalities (such as the MRs adopting words replacement). Moreover, these MRs are able to construct diverse test data with changes on both the input passages (images) and questions of TQA (VQA).

8. Conclusions

In recent years, question answering (QA) has emerged as a popular and powerful tool in various domains, due to its capability of enabling the machine to understand and answer question posted in natural language. Unfortunately, recent studies have adopted various techniques to evaluate QA systems, revealing a series of problems concerning different aspects. In this paper, we focused on the evaluation of two typical categories of QA systems, namely, the textual QA (TQA) and visual QA (VQA). We applied the technique of metamorphic testing (MT) to QA, and identified 17 metamorphic relations (MRs) by considering the users' varying expectations on QA systems. In the experiments, we evaluated two TQA systems and two VQA systems by using all of the MRs, and our experimental results reveal their capabilities from different perspectives. These results further suggest that the proposed MRs are capable of encoding the expected characteristics of QA and MT can be an effective evaluation method for QA.

Author Contributions: Conceptualization, M.J. and Z.D.; methodology, M.J.; software, K.T.; data curation, K.T.; writing—original draft preparation, M.J.; writing—review and editing, M.J., Z.D. and K.T.; visualization, M.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by National Nature Science Foundation of China(Grant Nos. 61751210 and 61802349), and the Zhejiang Provincial Natural Science Foundation of China(Grant No. LY20F020021).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Bouziane, A.; Bouchiha, D.; Doumi, N.; Malki, M. Question Answering Systems: Survey and Trends. *Procedia Comput. Sci.* **2015**, *73*, 366–375. [\[CrossRef\]](#)
2. Zeng, C.; Li, S.; Li, Q.; Hu, J.; Hu, J. A Survey on Machine Reading Comprehension: Tasks, Evaluation Metrics, and Benchmark Datasets. *Appl. Sci.* **2020**, *10*, 7640. [\[CrossRef\]](#)
3. Liu, S.; Zhang, X.; Zhang, S.; Wang, H.; Zhang, W. Neural Machine Reading Comprehension: Methods and Trends. *Appl. Sci.* **2019**, *9*, 3698. [\[CrossRef\]](#)
4. Antol, S.; Agrawal, A.; Lu, J.; Mitchell, M.; Parikh, D. VQA: Visual Question Answering. *Int. J. Comput. Vis.* **2015**, *123*, 4–31.
5. Reddy, S.; Chen, D.; Manning, C.D. CoQA: A Conversational Question Answering Challenge. *Trans. Assoc. Comput. Linguist.* **2019**, *7*, 249–266. [\[CrossRef\]](#)
6. Cui, L.; Huang, S.; Wei, F.; Tan, C.; Zhou, M. SuperAgent: A Customer Service Chatbot for E-commerce Websites. In Proceedings of the ACL 2017, System Demonstrations, Vancouver, BC, Canada, 30 July–4 August 2017; pp. 97–102.
7. Li, H.; Wang, P.; Shen, C.; Hengel, A.V.D. Visual Question Answering as Reading Comprehension. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 6319–6328.
8. Seo, M.; Kembhavi, A.; Farhadi, A.; Hajishirzi, H. Bidirectional Attention Flow for Machine Comprehension. *arXiv* **2018**, arXiv:1611.01603.
9. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv* **2019**, arXiv:1810.04805.
10. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv* **2019**, arXiv:1907.11692.
11. Lu, J.; Batra, D.; Parikh, D.; Lee, S. ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks. *arXiv* **2019**, arXiv:1908.02265.
12. Jia, R.; Liang, P. Adversarial Examples for Evaluating Reading Comprehension Systems. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 9–11 September 2017; pp. 2021–2031.

13. Kaushik, D.; Lipton, Z.C. How Much Reading Does Reading Comprehension Require? A Critical Investigation of Popular Benchmarks. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 5010–5015.
14. Mudrakarta, P.K.; Taly, A.; Sundararajan, M.; Dhamdhere, K. Did the Model Understand the Question? In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, 15–20 July 2018; pp. 1896–1906.
15. Ribeiro, M.T.; Wu, T.; Guestrin, C.; Singh, S. Beyond Accuracy: Behavioral Testing of NLP Models with CheckList. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020; pp. 4902–4912.
16. Patro, B.N.; Patel, S.; Namboodiri, V.P. Robust Explanations for Visual Question Answering. In Proceedings of the 2020 IEEE Winter Conference on Applications of Computer Vision (WACV), Village, CO, USA, 1–5 March 2020; pp. 1566–1575.
17. Chen, J.; Durrett, G. Robust Question Answering Through Sub-part Alignment. *arXiv* **2020**, arXiv:2004.14648.
18. Zhou, Z.Q.; Sun, L. Metamorphic testing of driverless cars. *Commun. ACM* **2019**, *62*, 61–67. [\[CrossRef\]](#)
19. Xie, X.; Wong, W.E.; Chen, T.Y.; Xu, B.W. Metamorphic slice: An application in spectrum-based fault localization. *Inf. Softw. Technol.* **2013**, *55*, 866–879. [\[CrossRef\]](#)
20. Jiang, M.; Chen, T.Y.; Kuo, F.C.; Towey, D.; Ding, Z. A metamorphic testing approach for supporting program repair without the need for a test oracle. *J. Syst. Softw.* **2017**, *126*, 127–140. [\[CrossRef\]](#)
21. Jiang, M.; Chen, T.Y.; Zhou, Z.Q.; Ding, Z. Input Test Suites for Program Repair: A Novel Construction Method Based on Metamorphic Relations. *IEEE Trans. Reliab.* **2020**. [\[CrossRef\]](#)
22. Zhou, Z.Q.; Xiang, S.; Chen, T.Y. Metamorphic testing for software quality assessment: A study of search engines. *IEEE Trans. Softw. Eng.* **2016**, *42*, 264–284. [\[CrossRef\]](#)
23. Zhou, Z.Q.; Sun, L.; Chen, T.Y.; Towey, D. Metamorphic Relations for Enhancing System Understanding and Use. *IEEE Trans. Softw. Eng.* **2020**, *46*, 1120–1154. [\[CrossRef\]](#)
24. Gardner, M.; Grus, J.; Neumann, M.; Tafford, O.; Dasigi, P.; Liu, N.F.; Peters, M.; Schmitz, M.; Zettlemoyer, L. AllenNLP: A Deep Semantic Natural Language Processing Platform. In Proceedings of the Workshop for NLP Open Source Software (NLP-OSS), Melbourne, Australia, 20 July 2018; pp. 1–6.
25. Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; et al. Transformers: State-of-the-Art Natural Language Processing. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Virtual Conference, 16–20 November 2020; pp. 38–45.
26. Segura, S.; Fraser, G.; Sanchez, A.B.; Ruiz-Cortés, A. A survey on metamorphic testing. *IEEE Trans. Softw. Eng.* **2016**, *42*, 805–824. [\[CrossRef\]](#)
27. Chen, T.Y.; Kuo, F.C.; Liu, H.; Poon, P.L.; Towey, D.; Tse, T.H.; Zhou, Z.Q. Metamorphic Testing: A Review of Challenges and Opportunities. *ACM Comput. Surv.* **2018**, *51*, 4:1–4:27. [\[CrossRef\]](#)
28. Xiao, H. bert-As-Service. 2018. Available online: <https://github.com/hanxiao/bert-as-service> (accessed on 8 November 2020).
29. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. *arXiv* **2017**, arXiv:1706.03762.
30. Reimers, N.; Gurevych, I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, Hong Kong, China, 3–7 November 2019.
31. Tian, Y.; Pei, K.; Jana, S.; Ray, B. DeepTest: Automated Testing of Deep-Neural-Network-Driven Autonomous Cars. In Proceedings of the 40th International Conference on Software Engineering, Gothenburg, Sweden, 27 May–3 June 2018; pp. 303–314.
32. Sanh, V.; Debut, L.; Chaumond, J.; Wolf, T. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. *arXiv* **2020**, arXiv:1910.01108.
33. Singh, A.; Goswami, V.; Natarajan, V.; Jiang, Y.; Chen, X.; Shah, M.; Rohrbach, M.; Batra, D.; Parikh, D. MMF: A Multimodal Framework for Vision and Language Research. 2020. Available online: <https://github.com/facebookresearch/mmf> (accessed on 12 September 2020).
34. Rajpurkar, P.; Jia, R.; Liang, P. Know What You Don’t Know: Unanswerable Questions for SQuAD. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Melbourne, Australia, 15–20 July 2018; pp. 784–789.
35. Malinowski, M.; Fritz, M. A Multi-World Approach to Question Answering about Real-World Scenes based on Uncertain Input. In Proceedings of the 27th International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; pp. 1682–1690.
36. Shah, M.; Chen, X.; Rohrbach, M.; Parikh, D. Cycle-Consistency for Robust Visual Question Answering. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 6642–6651.
37. Huang, J.H.; Dao, C.D.; Alfady, M.; Ghanem, B. A Novel Framework for Robustness Analysis of Visual QA Models. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2019; Volume 33, pp. 8449–8456.
38. Wallace, E.; Feng, S.; Kandpal, N.; Gardner, M.; Singh, S. Universal Adversarial Triggers for Attacking and Analyzing NLP. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; pp. 2153–2162.
39. Ribeiro, M.T.; Guestrin, C.; Singh, S. Are Red Roses Red? Evaluating Consistency of Question-Answering Models. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 6174–6184.