What does BERT learn from Arabic machine reading comprehension datasets?

Eman Albilali, Nora Al-Twairesh and Manar Hosny
College of Computer and Information Sciences
King Saud University, KSA
{ealbilali, twairesh, mifawzi}@ksu.edu.sa

Abstract

In machine reading comprehension tasks, a model must extract an answer from the available context given a question and a passage. Recently, transformer-based pre-trained language models have achieved state-of-the-art performance in several natural language processing tasks. However, it is unclear whether such performance reflects true language understanding. In this paper, we propose adversarial examples to probe an Arabic pre-trained language model (AraBERT), leading to a significant performance drop over four Arabic machine reading comprehension datasets. We present a layer-wise analysis for the transformer's hidden states to offer insights into how AraBERT reasons to derive an answer. The experiments indicate that AraBERT relies on superficial cues and keyword matching rather than text understanding. Furthermore, hidden state visualization demonstrates that prediction errors can be recognized from vector representations in earlier layers.

1 Introduction

The rise of pre-trained Language Models (LM), including ELMo (Peters et al., 2018), BERT (Devlin et al., 2019), GPT (Radford et al., 2019), and RoBERTa (Liu et al., 2019), has shifted the field of Natural Language Processing (NLP) from training complex neural models from scratch to fine-tuning on downstream tasks. These contextual embeddings are generated by training on a general language model task, such that these vectors encode the structure and meaning of each word in the context. This approach has replaced the effort needed to tailor complex neural models with a simple fine-tuning using a relatively small amount of data for a specific task. As the use of LMs is growing in various NLP tasks, it is important to understand how they achieve high performance. This understanding can be used to leverage their strengths, identify their vulnerabilities, and rectify their weaknesses.

Morphologically Rich Languages (MRL) are languages in which significant information is expressed morphologically via word variation, such as in Arabic, Hebrew, and Turkish, rather than syntactically, as in English (Habash, 2010). The diversity of word forms requires machine learning models to cope with the extreme data sparseness that follows from the complex word structure, which further complicates automatic semantic identification. Furthermore, as each word can appear in multiple forms, some are unseen in the annotated data, which amplifies the Out-of-Vocabulary (OOV) problem.

Recently, a considerable number of research studies have sought to analyze different LMs in the context of the English language, while other languages have received limited attention. Most studies, including (Hewitt and Manning, 2019), (Tenney et al., 2019a), and (Tenney et al., 2019b), use black-box external probes to assess model robustness and to identify its weaknesses. The present study follows this direction and proposes adversarial examples as external probes to investigate what underpins the reported performance of Arabic pre-trained LMs.

This paper focuses on the task of Arabic Machine Reading Comprehension (MRC), which requires the model to extract a span from the text as the answer, given a question and a passage. Our approach involves probing the fine-tuned AraBERT (version 0.1) (Antoun et al., 2020a) model by appending distractors containing keywords to examples in the test set and, in turn, evaluating model robustness. The goal is to mislead the model into making incorrect predictions by adding uninterpretable information to the passage, noting that humans can still easily find the correct answer. To the best of our knowledge, no previous research study

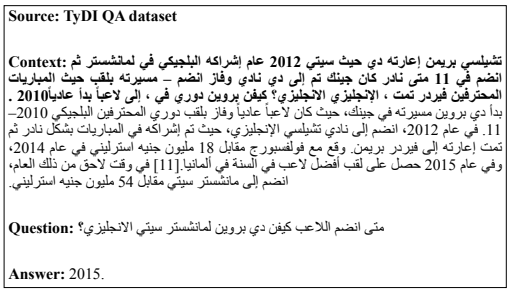


Figure 1: An instance from the TyDI QA MRC dataset with a distractor sentence (presented in bold) appended to the beginning of the passage.

analyzes the Arabic pre-trained LM fine-tuned on MRC task.

Figure 1 shows an example of a distractor sentence appended to the passage. We construct an unreadable distractor by concatenating a variation of keywords from the question, answer, and a random sentence from the passage, and shuffling the word order in the input. Our experimental results show a significant performance drop, implying that AraBERT (version 0.1) relies on statistical cues and keyword matching and, moreover, can be distracted easily when irrelevant information is added to the passage.

Although the task of Arabic MRC has gained popularity in the research community, the question of what information must be captured by a model to achieve high performance remains unclear. We extend our analysis on adversarial examples by examining the hidden state vectors between encoder layers. At this end, we visualize vector transformations in each layer of the transformer for correctly-predicted and falsely-predicted answers. This visualization exposes incorrect predictions in the earlier layers and reveals the part of the context that the model regards as supporting facts.

We summarize the main contributions of this research as follows:

- We conduct a deep analysis of the state of the art Arabic LM (namely, AraBERT (Antoun et al., 2020a)) on Arabic MRC datasets.
- We construct adversarial examples for Arabic MRC that fool the model and undermine its performance substantially.
- We visualize and analyze the hidden state vectors in each layer of the transformer model for Arabic MRC.

The rest of the paper is organized as follows: Section 2 reviews the related work. Section 3 discusses the methodology to construct adversarial examples and visualizes the transformer's hidden state vectors. We conduct several experiments and discuss our findings in section 4. Finally, we conclude the paper in section 5.

2 Related Work

The interpretability and probing of models for NLP tasks fall into three areas: probing tasks to understand linguistic properties captured by the model; adversarial examples to understand the flaws and weaknesses of the model; and assessing dataset quality by conducting partial training and constructing challenging sets (Belinkov and Glass, 2019). We discuss the first two directions as they are the focus of our work.

Probing tasks The most common approach used to understand the linguistic properties captured by a model is to investigate its hidden states by designing probing tasks. The authors in (Tenney et al., 2019b) designed a novel edge probing task to measure the effectiveness of a pre-trained model in capturing linguistic information, which involved probing core NLP tasks encompassing diverse syntactic and semantic phenomena. In (Tenney et al., 2019a), the authors discovered that BERT follows a classical NLP pipeline in an interpretable and localizable way. The authors in (van Aken et al., 2019) applied a set of general and QA-specific probing tasks to discover information in each representation layer. An attention-based probing classifier was proposed by (Clark et al., 2019) to demonstrate syntactic information and coreference captured by BERT's attention. Through structural probes, (Hewitt and Manning, 2019) suggested that syntactic information can be recovered from BERT token representations.

As multiple probing studies report that BERT possesses syntactic, semantic, and world knowledge information, (Tenney et al., 2019a) emphasized that the absence of linguistic patterns in a probing classifier cannot lead one to conclude that the information is not there, nor can the presence of such information reveal how it is used. Furthermore, it is worth questioning the extent to which a probing classifier can be complex such that the recovered information is extracted from the original model rather than the probing classifier (Rogers et al., 2020). To rectify these issues,

(Elazar et al., 2020) proposed amnesic probing, which involves removing certain information from the model (e.g., part-of-speech tags) and evaluating the performance change.

Adversarial examples Adversarial examples have gained immense popularity in the NLP field, particularly as a way to understand model failures. Most research studies have focused on constructing black-box attacks due to the nature of the text (Belinkov and Glass, 2019). First, the discrete nature of the input complicates the task of measuring the distance between the original and the adversarial example. Second, it is difficult to formulate minimizing the distance as an optimization problem for a discrete input. In (Jia and Liang, 2017), it was shown that appending a distractor sentence to the passage in an MRC task lowers the performance for state-of-the-art models. The authors in (Alzantot et al., 2018) proposed a black-box genetic algorithm to construct semantically and syntactically similar examples, the aim being to fool sentiment analysis and textual entailment models. The performance of BERT dropped when evaluated on unreadable adversarial examples, which were generated by appending keywords to candidate answers in multiple-choice reading comprehension datasets (Si et al., 2019). Recently, model-in-the-loop adversarial annotation, which has been investigated by (Wallace et al., 2019), (Nie et al., 2020), and (Kaushik et al., 2020), is a new direction that has been applied to annotate challenging datasets using an adversarial model to retrieve hard examples. The annotators are allowed to interact with the adversary during the annotation process, and they can use model feedback to inform the generation process.

Only one study, that of (Alshemali and Kalita, 2019), has investigated adversarial examples in the Arabic language. Specifically, noun-adjective agreement in Arabic was violated in order to construct a perturbed input text. This attack fooled a word-level Bidirectional Long Short-term Memory (BiLSTM) model and a word-level Convolutional Neural Network (CNN) model, and it successfully reduced model performance when evaluated on a sentiment analysis task.

Transformer models Our analysis focuses on AraBERT (version 0.1) (Antoun et al., 2020a), an Arabic pre-trained language model based on the BERT-BASE architecture (Devlin et al., 2019). BERT is a stack of transformer encoder layers with multiple self-attention heads (Vaswani et al., 2017). In (Antoun et al., 2020a), the authors trained the BERT-BASE architecture on manually-scraped Arabic news websites, the public-words Arabic corpus, and the Open Source International Arabic News Corpus (OSIAN). Since then, several Arabic transformer models have been proposed. Safaya et al. (2020) introduced an Arabic BERT, a pre-trained BERT-BASE on Arabic Wikipedia dump and the Arabic version of OSCAR (Ortiz Suárez et al., 2020). In ARBERT and MARBERT, (Abdul-Mageed et al., 2020) pre-trained BERT model on Modern Standard Arabic (MSA) and dialects. Antoun et al. (2020b) generated AraGPT2 trained from scratch on large Arabic corpora following the architecture and training procedure of GPT2 (Radford et al., 2019).

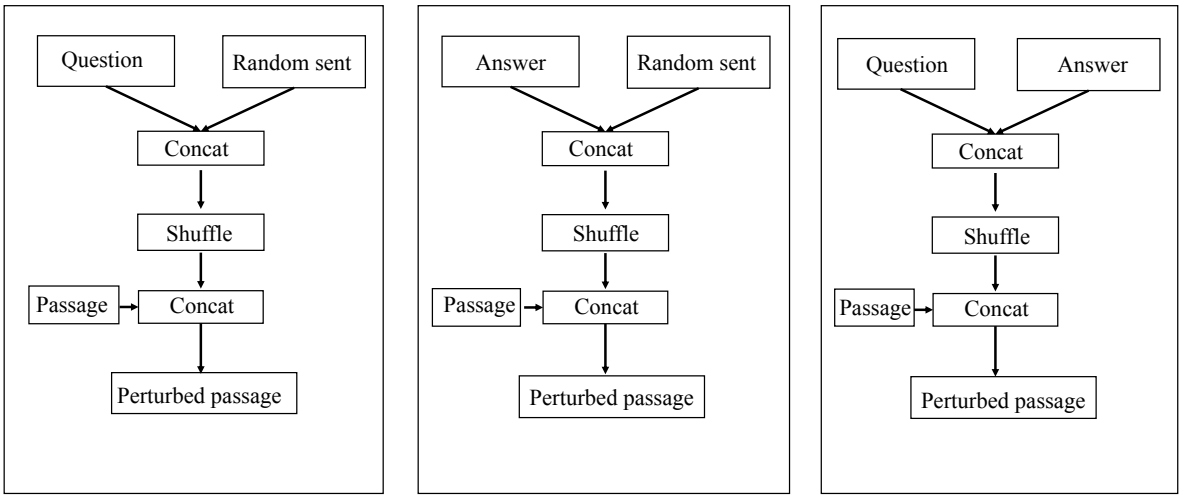
To the best of our knowledge, no research study has been undertaken to analyze recent Arabic pre-trained language models. Arabic adversarial examples were constructed for sentiment analysis task (Alshemali and Kalita, 2019), but no previous study has examined adversarial examples for Arabic MRC.

3 Methodology

We focus our analysis on AraBERT (version 0.1) fine-tuned on the task of Arabic MRC using four different Arabic MRC datasets. We propose two approaches to investigate the model's failures and to study the transformation of hidden vectors in each layer. First, we construct an unreadable perturbed sentence and append it to the context to generate adversarial examples. Following this, we qualitatively analyze the vector transformations by examining their position in vector space.

3.1 Adversarial Examples

This subsection introduces the method for constructing adversarial examples. We first fine-tune AraBERT (version 0.1) on the original training set and, in turn, evaluate it on the perturbed set. To generate an adversarial example, we append an unreadable perturbed sentence to the passage. To obtain the distractor sentence, we concatenate different parts from the example, as shown in Figure 2. We generate an ungrammatical sentence by randomly shuffling the sentence and changing the word order in the input. Note that we only shuffle the distractor sentence and leave the answer unchanged, as



(a) Question-Sent-Shuffled (b) Answer-Sent-Shuffled (c) Question-Answer-Shuffled

Figure 2: Procedure for constructing adversarial examples for MRC.

well as the other sentences. For this reason, the example's answer remains the same.

We construct three variants for the adversarial attack based on the information appended to the distractor. *Q-Sent-Shuffled* attack is constructed by concatenating the question and a random sentence from the passage that does not contain the answer. In *Answer-Sent-Shuffled* attack, the distractor is formed by mixing the answer and random sentence words. Even if the answer appears twice in the context, the model should extract the answer from the relevant context rather than from the unreadable text. The shuffled question and answer are appended to the text in *Q-Answer-Shuffled* attack.

To ensure that the distractor is ungrammatical and difficult to interpret, we reshuffle the sequence based on the shuffle degree as follows:

$$ShuffleDegree = \frac{MinimumEditDistance}{SequenceLength}$$

The Minimum Edit Distance (MED) is calculated between the original sequence and the shuffled sequence. The original sequence is re-shuffled until the shuffle degree exceeds a threshold value (in this case, 0.65). The main trigger, here, is that certain words from the question appear in the passage with other random sentences. Ideally, if the pre-trained LM understands the text and does not rely on keyword matching, it will not be fooled by the distractor sentence.

3.2 Visualization of Transformed Tokens

BERT's architecture is based on a stacked transformer that allows the vector transformation to be traced for each token traversing from the bottom to the top layers. Following (van Aken et al., 2019), we use this characteristic and analyze the vector transformation for correctly-predicted and falsely-predicted answers.

To analyze vector transformation for the Arabic pre-trained LM fine-tuned on an MRC task, we randomly select correct and incorrect answer samples from the respective test sets. We extract the vector representation for each token in the sequence from each layer and remove padding. We consider the distance between vectors in the vector space as an indication of the semantic relation. The BERT-BASE model uses hidden vectors with a dimension of 768, and we need to visualize the relation between vectors within a 2-dimensional (2D) space. Accordingly, we apply Principal Component Analysis (PCA) to reduce the vector dimensions into 2D in each layer.

4 Experiments and Analysis

4.1 Datasets

In this paper, we analyze four Arabic MRC datasets. We briefly introduce each one in the following subsections.

	AR TyDI QA dev	AR XQuAD	AR MLQA	Arabic WikiReading	Average
AraBERTv0.1	68.94	25.88	23.73	63.01	-
Q-Sent-Shuffled	48.47 -29.7%	**13.89** **-46.33%**	**12.66** **-46.65%**	**53.07** **-15.1%**	-34.45%
Answer-Sent-Shuffled	50.97 -26.07%	22.98 -11.21%	20.00 -15.72%	58.37 -6.62%	-14.9%
Q-Answer-Shuffled	**44.08** **-36.06%**	18.74 -27.59%	19.48 -17.91	-	-27.19%
Average drop	-30.61%	-28.38%	-26.76%	-10.86%	-25.51%

Table 1: EM scores for fine-tuned AraBERT (version 0.1) on the original and perturbed test sets with adversarial examples. Italicized numbers represent the percentage of performance drop relative to the original performance. Bold performance values indicate the most effective method to distract the model for each dataset.

TyDI QA is a multi-lingual dataset covering 11 topological diverse languages (Clark et al., 2020). The dataset was collected by annotators who were presented with a short prompt from Wikipedia and asked to write a question inspired by the prompt. The Wikipedia article that best matched the question using Google search was returned as the context. After that, the annotator selected the passage containing the answer, if any, along with the minimal span containing the answer. This dataset evaluates three tasks: passage selection, minimal answer selection, and gold passage. We are interested in the gold passage task; to extract the answer from a passage rather than from a long text to facilitate the comparison with other datasets.

XQuAD is a cross-lingual MRC dataset composed of 240 paragraphs and 1,190 question-answer pairs translated from SQuADv1.1 by professional translators into 10 languages, one of which is Arabic (Artetxe et al., 2020).

MLQA is a multilingual QA dataset comprising 7 languages (Lewis et al., 2020). Question-answer pairs were collected by crowd-workers in the English language, after which they were translated into the target languages by professional translators.

Arabic WikiReading[1] The dataset was automatically constructed under a distant supervision strategy using the Wikidata statement property as a query and the statement value as the ground truth answer. Arabic articles were collected from *arwikiExtracts*[2] by parsing an Arabic *20190920* Wikipedia dump, using the *WikiExtractor*[3] tool to strip away images, tables, info-boxes, and figures. The first paragraph of each article, which captures the essential information of an article, was extracted and paragraphs with fewer than 300 characters were discarded. Arabic Wikidata dump was extracted from the *20190909.JSON* dump using the *Wikidata-filter*[4] tool. All statements with the same item and property were consolidated into a single (item, property, answer) triple. Query-answer pairs were then matched with the relevant paragraph by replacing each Wikidata item in the (item, property, answer) triples with the appropriate Wikipedia curated paragraph, knowing the title of the Wikipedia article that matches the item in the collected triples, and discarding any item not linked to Wikipedia articles to form approximately 98,000 MRC instances.

4.2 Experimental Setup

We base our training code on the PyTorch implementation of Hugging Face transformers[5]. We use AraBERT (version 0.1) (Antoun et al., 2020a), a BERT BASE model (Devlin et al., 2019) pre-trained on an Arabic corpus with 12 layers, 12 self-attention heads, and a total of 110 million parameters. We fine-tune AraBERT (version 0.1) on Arabic MRC datasets. Precisely, we train on the Arabic TyDI QA gold passage (Clark et al., 2020) training set, and we evaluate on the original development set and the perturbed variants, as the test

[1]https://github.com/esulaiman/Arabic-WikiReading-and-KaifLematha-datasets

[2]https://github.com/motazsaad/arwikiExtracts

[3]https://github.com/attardi/wikiextractor

[4]https://github.com/xwhan/wikidata-filter

[5]https://huggingface.co/transformers/

	AR TyDI QA dev	AR XQuAD	AR MLQA	Arabic WikiReading	Average
AraBERTv0.1	81.70	41.04	39.64	68.27	-
Q-Sent-Shuffled	**62.90** **-23.01%**	**22.36** **-45.52%**	**21.61** **-45.48%**	**58.08** **-13.66%**	-31.92%
Answer-Sent-Shuffled	67.18 -17.77%	37.49 -8.66%	35.38 -10.75%	66.81 -0.68%	-9.47%
Q-Answer-Shuffled	68.84 -15.74%	40.12 -2.24	43.66 +10.14	-	-2.6%
Average drop	-18.84%	-18.87%	-15.36%	-7.17%	-14.66%

Table 2: F1 scores for fine-tuned AraBERT (version 0.1) on the original and perturbed test sets with adversarial examples. Italicized numbers represent the percentage of performance drop relative to the original performance. Bold performance values indicate the most effective method to distract the model for each dataset.

set is not publicly available. We further evaluate the model on the original Arabic XQuAD (Artetxe et al., 2020) and Arabic MLQA (Lewis et al., 2020) test sets and their perturbed variants. Additionally, we train on the Arabic WikiReading training set and evaluate using the test set and the constructed adversarial examples. Each training run is performed over 3 epochs with a batch size equals 8, and we follow (Mozannar et al., 2019) by adopting a learning rate of 3e-5. The maximum sequence length chosen is 512. Tokens exceeding this length are truncated, while the input sequence is padded to reach the maximum sequence length.

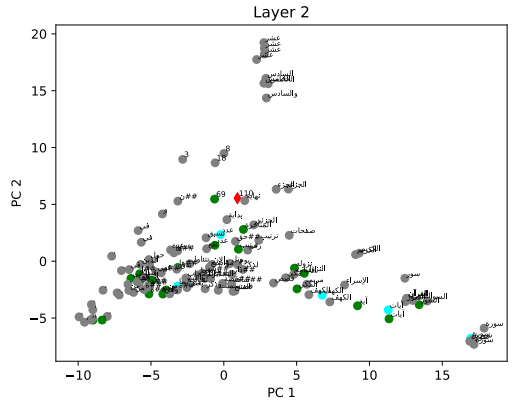
4.3 Results and Discussion

The performance results are reported in Tables 1 and 2. Table 1 shows the EM score for the AraBERT (version 0.1) model tested on the original MRC datasets and the attacked variants with adversarial examples discussed in Section 3. Note that the question and the ground-truth answers are unchanged. AraBERT (version 0.1) achieves F1 scores of 81.70 and 68.27 when tested on Ar-TyDI QA and Arabic WikiReading, respectively, while Ar-XQuAD and Ar-MLQA perform poorly with 41.04 and 39.64 F1 score, respectively. When evaluating using the adversarial examples, the performance drops by around 45% for Ar-XQuAD and Ar-MLQA, and by 23% for TyDI QA for the question-random sentence attack. This indicates that AraBERT (version 0.1) was fooled by the adversarial sentence inserted into the passage, which was unreadable even for humans. This implies that the BERT model relies heavily on statistical cues and keyword matching rather than an understanding of the text. This result is consistent with English LM studies (Si et al., 2019), suggesting that BERT is not robust against adversarial attacks.

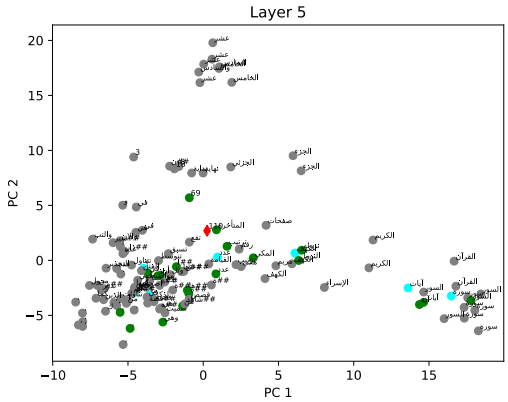
Generally, the inclusion of question words into the perturbed sentence causes the greatest reduction in F1 score over all the datasets. Similarly, when the question words appear in the distractor sentence, the EM score decreases substantially across all the datasets. Adversarial attacks targeted at the Arabic WikiReading dataset yield the lowest performance decrease compared to other MRC datasets. This may be due to the nature of the question, which consists of a small number of tokens compared to other MRC datasets.

Token Transformation Analysis Figure 3 shows the vector representation for the question and the passage tokens reduced into 2D using PCA. Vector transformation from the bottom to the top layers suggests that the model follows several phases to answer the question. Figure 3a indicates that earlier layers in AraBERT (version 0.1) group tokens with similar subjects into clusters, simulating an embedding layer in the neural network architecture.

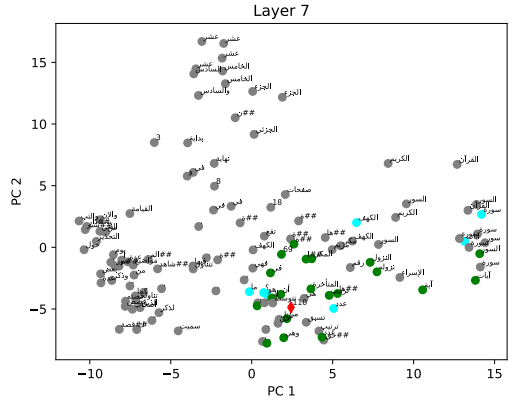
In the middle layers (layer 5 in the figure), we observe that some tokens remain grouped based on topical similarity (sorp, qrAn and Ayp), سورة ، قرآن و آية while other tokens transform to represent the relation between entities given the input context. For instance, the model identifies that the tokens are (mrym and Alkhf) مريم و الكهف related to each other as *sorp* سورة rather than as pronouns and these tokens were observed close to each other in the vector space. The task-specific function of matching the question with the support-



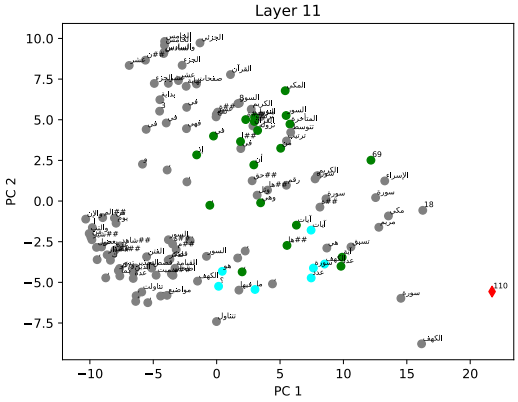
(a) Tokens carrying similar topics are clustered into relevant groups.



(b) Entity matching: tokens related to other entities are located in the same vector space.



(c) Supporting fact matching: question tokens are matched to the relevant sentence from the context.



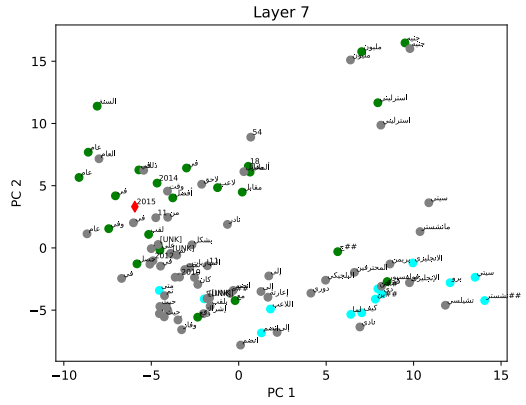
(d) Answer extraction: the answer is separated from other tokens.

Figure 3: Vector visualization for the question-context tokens from different AraBERT (version 0.1) layers after applying PCA dimensionality reduction. Red diamond represents the answer token, dark cyan dots indicate supporting fact tokens, turquoise dots are question tokens, and gray dots show other tokens in the sequence.

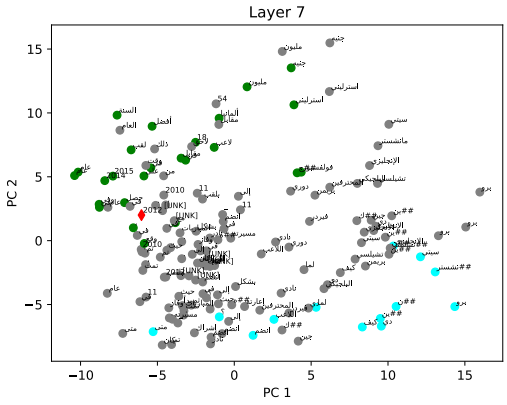
ing facts appears at layer 7. Question tokens are transformed to the same vector space as the context tokens related to the answer. Finally, in the last layer, the model dissolves most of the previous clusters and distinguishes the possible answer from other tokens, while irrelevant tokens form a separate cluster, as depicted in Figure 3d.

Adversarial Examples Analysis We visualize the vector transformation for an incorrectly predicted answer in a perturbed example from the TyDi QA development set depicted in figure 1. The example was generated by concatenating a shuffled question and random sentence to the passage. We further compare it to the original example before adding the perturbed sentence, thereby investigating the effect of the attack on model understanding. Figures 4a and 4b represent the vector space for the question and context tokens in layer 7. As discussed earlier, the question tokens are transformed in the middle layers to group with supporting facts from the context. Figures 4a and 4b show that the question tokens are far from the supporting fact tokens (illustrated with green dots), while tokens from the perturbed sentence in Figure 4b cover the same vector space as the question tokens. This suggests the importance of supporting fact extraction in layer 7. In particular, when the model was unable to extract the supporting fact from the context, it was fooled, and question tokens were matched with the perturbed sentence containing words from the question. As a result, the answer *2012* was selected from the perturbed sentence rather than the correct answer *2015*.

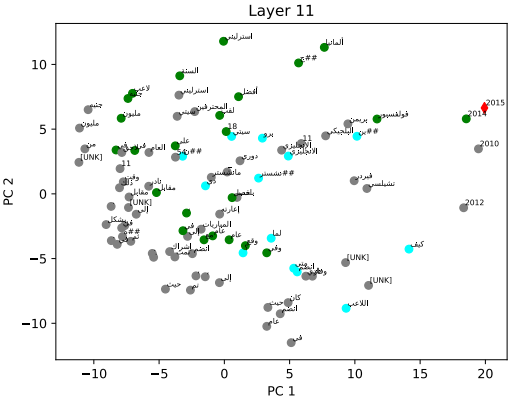
Although the model predicted *2015* as the correct answer in Figure 4c, the vector visualizations indicate that the other candidate answers *2014,*



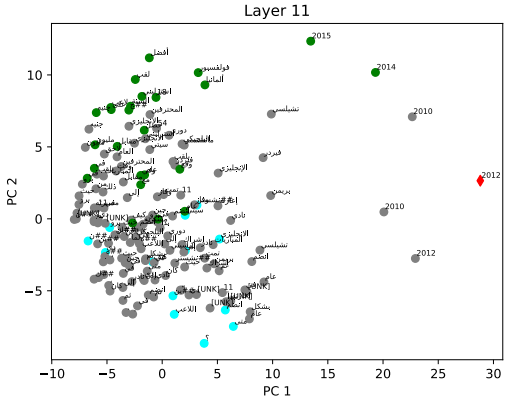
(a) Question tokens are not matched with the supporting fact and located far away in the vector space.



(b) Question tokens are far from supporting facts and matched with perturbed sentence tokens.



(c) The answer (2015) is extracted as the answer, while other candidate answers appeared in the same vector space.



(d) A token (2012) from the perturbed sentence is extracted as the answer.

Figure 4: Vector visualization for the question-context tokens from different AraBERT layers after applying PCA dimensionality reduction. Red diamond represents the answer token, dark cyan dots indicate supporting fact tokens, turquoise dots are question tokens, and gray dots show other tokens in the sequence.

2010, and 2012 cover the same vector space as the correct answer, and the other tokens are spread over the vector space rather than forming a homogeneous group. However, the model predicted the answer *110* in Figure 3d with high confidence since that other tokens were separated from the answer and formed groups far away from the answer.

5 Conclusion

This paper analyzed hidden state transformations through layers and adversarial examples to explore what an Arabic pre-trained LM learns from MRC datasets. We proposed a simple yet effective method to construct adversarial examples that fool AraBERT, which resulted in a substantial performance drop. Our analysis suggests that pre-trained LMs achieve competitive performance simply by relying on superficial cues such as lexical overlap or entity type matching.

Qualitative analysis of the hidden states of transformers indicated that uninterpretable information can be used to understand the reasons that underpin model failure and weaknesses. We demonstrated that locating the correct supporting fact for MRC earlier in the middle layers contributes to correct predictions. Our finding suggests that different layers solve different problems. We suggest examining part of the network or connecting non-adjacent layers based on the downstream task at hand.

It would be worthwhile for future work to analyze the degree to which different tokenization methods (input representations) assist in the learning of better morphology and the modeling of infrequent words in Arabic pre-trained LMs. Furthermore, we are interested in extending our adversarial examples to cover other Arabic pre-trained LMs

and constructing unanswerable questions.

References

Muhammad Abdul-Mageed, AbdelRahim Elmadany, and El Moatez Billah Nagoudi. 2020. Arbert & marbert: Deep bidirectional transformers for arabic. *arXiv preprint arXiv:2101.01785*.

Betty van Aken, Benjamin Winter, Alexander Löser, and Felix A. Gers. 2019. How does bert answer questions? a layer-wise analysis of transformer representations. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, page 1823–1832, New York, NY, USA. Association for Computing Machinery.

Basemah Alshemali and Jugal Kalita. 2019. Adversarial examples in arabic. In *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 371–376. IEEE.

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896, Brussels, Belgium. Association for Computational Linguistics.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020a. AraBERT: Transformer-based model for Arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15, Marseille, France. European Language Resource Association.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020b. Aragpt2: Pre-trained transformer for arabic language generation.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637, Online. Association for Computational Linguistics.

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Jonathan H. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. 2020. TyDi QA: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. 2020. When bert forgets how to pos: Amnesic probing of linguistic properties and mlm predictions. *arXiv preprint arXiv:2006.00995*.

Nizar Y Habash. 2010. Introduction to arabic natural language processing. *Synthesis Lectures on Human Language Technologies*, 3(1):1–187.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

Divyansh Kaushik, Eduard Hovy, and Zachary C Lipton. 2020. Learning the difference that makes a difference with counterfactually-augmented data. *International Conference on Learning Representations (ICLR)*.

Patrick Lewis, Barlas Oguz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. 2020. MLQA: Evaluating cross-lingual extractive question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7315–7330, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Hussein Mozannar, Elie Maamary, Karl El Hajal, and Hazem Hajj. 2019. Neural Arabic question answering. In *Proceedings of the Fourth Arabic Natural Language Processing Workshop*, pages 108–118,

Florence, Italy. Association for Computational Linguistics.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

Pedro Javier Ortiz Suárez, Laurent Romary, and Benoît Sagot. 2020. A monolingual approach to contextualized word embeddings for mid-resource languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1703–1714, Online. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Ali Safaya, Moutasem Abdullatif, and Deniz Yuret. 2020. KUISAIL at SemEval-2020 task 12: BERT-CNN for offensive speech identification in social media. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 2054–2059, Barcelona (online). International Committee for Computational Linguistics.

Chenglei Si, Shuohang Wang, Min-Yen Kan, and Jing Jiang. 2019. What does bert learn from multiple-choice reading comprehension datasets? *arXiv preprint arXiv:1910.12391*.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019a. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019b. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008. Curran Associates, Inc.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing NLP. In *Empirical Methods in Natural Language Processing*.