TCE at Qur'an QA 2023 Shared Task: Low Resource Enhanced Transformer-based Ensemble Approach for Qur'anic QA

Mohammed ElKomy, Amany Sarhan
Department of Computer Engineering, Faculty of Engineering
Tanta University, Egypt
{mohammed.a.elkomy,amany_sarhan}@f-eng.tanta.edu.eg

Abstract

In this paper, we present our approach to tackle Qur'an QA 2023 shared tasks **A** and **B**. To address the challenge of low-resourced training data, we rely on transfer learning together with a voting ensemble to improve prediction stability across multiple runs. Additionally, we employ different architectures and learning mechanisms for a range of Arabic pre-trained transformer-based models for both tasks. To identify unanswerable questions, we propose using a thresholding mechanism. Our top-performing systems greatly surpass the baseline performance on the hidden split, achieving a MAP score of 25.05% for task **A** and a partial Average Precision (pAP) of 57.11% for task **B**.

1 Introduction

Ad hoc search is a fundamental task in Information Retrieval (IR) and serves as the foundation for numerous Question Answering (QA) systems and search engines. Machine Reading Comprehension (MRC) is a long-standing endeavor in Natural language processing (NLP) and plays a significant role in the framework of text-based QA systems. The emergence of **B**idirectional **E**ncoder **R**epresentations from **T**ransformers (BERT) and its family of transformer-based pre-trained language models (LM) have revolutionized the landscape of transfer learning systems for NLP and IR as a whole (Yates et al., 2021; Bashir et al., 2021).

Arabic is widely spoken in the Middle East and North Africa, and among Muslims worldwide. Arabic is known for its extensive inflectional and derivational features. It has three main variants: Classical Arabic (CA), Modern standard Arabic (MSA), and Dialectal Arabic (DA).

Qur'an QA 2023 shared task **A** is a passage retrieval task organized to engage the community in conducting ad hoc search over the Holy Qur'an (MALHAS, 2023; Malhas and Elsayed, 2020). While Qur'an QA 2023 shared task **B** is a ranking-based MRC over the Holy Qur'an, which is the second version of Qur'an QA 2022 shared task (Malhas et al., 2022; MALHAS, 2023).

This paper presents our approaches to solve the two tasks **A** and **B**. For task **A**, we explore both dual-encoders and cross-encoders for ad hoc search (Yates et al., 2021). For task **B**, we investigate LMs for extractive QA using two learning methods (Devlin et al., 2019). For both tasks, we utilize various pre-trained Arabic LM variants. Moreover, we adopt external Arabic resources in our fine-tuning setups (MALHAS, 2023). Finally, we employ an ensemble-based approach to account for inconsistencies among multiple runs. We contribute to the NLP community by releasing our experiment codes and trained LMs to GitHub [1].

In this work, we address the following research questions [2]:

RQ1: What is the impact of using external resources to perform pipelined fine-tuning?
RQ2: How does ensemble learning improve the performance obtained?
RQ3: What is the effect of thresholding on zero-answer questions?
RQ4^{A}: What is the impact of hard negatives on the dual-encoders approach?
RQ5^{B}: What is the impact of multi answer loss method on multi-answer cases?
RQ6^{B}: How is post-processing essential for ranking-based extractive question answering?

The structure of our paper is as follows: Sections 2 and 3 provide an overview of the datasets used in our study. In Section 4, we present the system design and implementation details for both tasks. The main results for both tasks are presented in Section 5. Section 6 focuses on the analysis and discussion of our research questions **RQ**s. Finally, Section 7 concludes our work.

[1] https://github.com/mohammed-elkomy/quran-qa

[2] A superscript at the end of a RQ refers to one of the tasks. No superscript means the RQ applies for both tasks.

Proceedings of the The First Arabic Natural Language Processing Conference (ArabicNLP 2023), pages 728–742

Split		Training	Development
# Question-passage relevance pairs		972	160
# Questions	Multi-answer	105 (60%)	15 (60%)
	Single-answer	43 (25%)	6 (24%)
	Zero-answer	26 (15%)	4 (16%)
	Total	174	25

Table 1: Task **A** dataset relevance pairs distribution across training and development splits. We also include the distribution of answer types per split.

Split		Training	Development
# Question-passage-answer Triplets		1179	220
# Question-passage Pairs	Multi-answer	134 (14%)	29 (18%)
	Single-answer	806 (81%)	124 (76%)
	Zero-answer	52 (5%)	10 (6%)
	Total	992	163

Table 2: Task **B** dataset pairs and triplets distribution across training and development splits. For questions-passage pairs, we show the distribution of answer types.

2 Task A Dataset Details

Qur'an QA 2023 shared task **A** serves as a test collection for the ad hoc retrieval task. The divine text is divided into segments known as the Thematic Qur'an Passage Collection (QPC), where logical segments are formed based on common themes found among consecutive Qur'anic verses (Malhas et al., 2023; Swar, 2007). In this task, systems are required to provide responses to user questions in MSA by retrieving relevant passages from the QPC when possible. This suggests there is a language gap between the questions and the passages, as the passages are in CA. Table 1 presents the distribution of the dataset across the training and development splits. The majority of questions in the dataset are multi-answer questions, meaning that systems can only receive full credit if they are able to identify all relevant passages for these queries. Additionally, Table 1 provides information on zero-answer questions, which are unanswerable questions from the entire Qur'an. (More information about the dataset distribution of topics in Appendix A.1)

Task **A** is evaluated as a ranking task using the standard mean Average Precision (MAP) metric. (Additional information about the evaluation process including zero-answers cases can be found in Appendix A.2)

3 Task B Dataset Details

Qur'an QA 2023 shared task **B** is a ranking-based SQuADv2.0-like MRC over the Holy Qur'an, which extends to the Qur'an QA 2022 (Malhas et al., 2022; Rajpurkar et al., 2016). The dataset is also referred to as Qur'an reading comprehension dataset v1.2 (QRCDv1.2). The same questions from task **A** are organized as answer span extraction task from relevant passages (Malhas and Elsayed, 2020; Malhas et al., 2022). (See the dataset distribution of topics in Appendix A.1)

Table 2 depicts the distribution of dataset pairs and triplets across the training and development splits. In addition, the table presents the distribution of answer types for the dataset pairs.

Although zero-answer questions account for 15% of the questions in task **A** test collection, they only contribute to 5% of the question-passage pairs in task **B**. Furthermore, task **B** has a limited number of unique questions in comparison to their corresponding question-passage pairs as seen from Tables 1 and 2, respectively. As a consequence, task **B** can have repeated questions and passages among different samples and can be even *leaked* among training and development splits (Keleg and Magdy, 2022). Keleg and Magdy (2022) analyzed this phenomenon and identified sources of *leakage* in Qur'an reading comprehension dataset v1.1 (QRCDv1.1). In QRCDv1.1, leakage is defined as the presence of passages, questions, or answers that are shared among multiple samples (Keleg and Magdy, 2022). This can lead to LMs memorizing or overfitting leaked samples (Keleg and Magdy, 2022). Keleg and Magdy (2022) categorized QRCDv1.1 into four distinct and mutually exclusive categories based on the type of leakage: pairs of passage-question, passage-answer, or just questions. (For more information about leakage in task **B**, please refer to Appendix A.4)

We extend the analysis made by Keleg and Magdy (2022) for QRCDv1.2. Our main observation is that 90% of the samples with no answer belong to the trivial leakage group called $D_{(1)}$. This group refers to samples with duplicate passage-answer or question-answer pairs. This indicates that zero-answer questions are not just less prevalent in task **B** but also present a greater challenge in terms of generalization. Given the four groups defined by Keleg and Magdy (2022), they proposed a data re-splitting mechanism for QRCDv1.1 called *faithful* splits. In this work, we extend their re-splitting approach and create faithful splits for QRCDv1.2. (Please refer to Appendix A.4 for more details about faithful splitting)

Task **B** is evaluated as a ranking task as well, using a recently proposed measure called pAP (Malhas and Elsayed, 2020; MALHAS, 2023). (More details about this measure and zero-answer sample evaluation can be found in Appendix A.3)

4 System Design

In this work, we fine-tune a variety of pre-trained Arabic LMs, namely AraBERTv0.2-base (Antoun et al., 2020), CAMeLBERT-CA (Inoue et al., 2021), and AraELECTRA (Antoun et al., 2021). We utilize transfer learning and ensemble learning for both tasks. To determine zero-answer cases, we apply a thresholding mechanism. (Additional information on transfer learning and ensemble learning can be found in Appendices B and C, respectively)

4.1 Task A Architecture

We examine two distinct approaches for neural ranking in ad-hoc search: dual-encoders and cross-encoders approaches (Yates et al., 2021).

In dual-encoders, documents and queries are encoded separately into dense vectors, which are then compared using a metric learning function, such as cosine distance. We utilize **S**table **T**raining **A**lgorithm for dense **R**etrieval (STAR) with a batch size of 16 queries to train our dense retrievers (Zhan et al., 2021; Yates et al., 2021).

In contrast cross-encoders involve encoding positive and negative pairs of documents and questions, assigning a relevance score. This method packs a document and a question into a single input for a sentence similarity LM (Yates et al., 2021). Both methods require negative relevance signals during training. (Please refer to Figures 4a and 4b in Appendix for both approaches. Additionally, see Appendix D for more details about negative selection criteria and zero-answer prediction)

Although cross-encoders have a higher computational overhead compared to dual-encoders when used for ranking, the former has a quadratic complexity while the latter has a linear complexity. However, both methods are still feasible for low-resource datasets (Yates et al., 2021). In both approaches, we utilize the cumulative predicted scores of the top K documents to calculate the likelihood of each question having an answer. We then apply a threshold ζ to identify zero-answer questions.

4.2 Task B Architecture

We fine-tune pre-trained LMs for span prediction as in SQuADv2.0 (Rajpurkar et al., 2018; Devlin et al., 2019). We use two different fine-tuning methods: First answer loss (FAL) and Multi answer loss (MAL). The FAL method focuses on optimizing for the first answer in the ground truth answers, which is the default approach in standard span prediction implementations for SQuAD (Devlin et al., 2019; Wolf et al., 2019). In contrast, MAL optimizes for multiple answers simultaneously for the multi-answer samples in QRCDv1.2. This helps prevent the trained systems from being overly confident in a single span and distributes the predicted probability among different spans. (Refer to Appendix E for more information about these learning methods)

It is worth noting that raw predictions from span prediction LMs are suboptimal for ranking MRC, as many of them have overlapping content. To address this, we follow a post-processing mechanism proposed by Elkomy and Sarhan (2022). (See Appendix E.1 for implementation details)

Similar to task **A**, we perform thresholding by a hyperparameter ζ to determine zero-answer samples using LM null answer **[CLS]** token probability (Rajpurkar et al., 2018; Devlin et al., 2019). (See Appendix E.2 for more details on zero-answer cases)

5 Results

The results tables for both tasks use the following notational format: We use short forms to refer to combinations of LMs and their fine-tuning approaches using superscripts and subscripts.

The subscripts $\sim$ and $\approx$ denote direct fine-tuning and pipelined fine-tuning, respectively. Additionally, the arrows in model names subscripts indicate the stages of pipelined fine-tuning, with the learning resources names listed. Superscripts are used to denote the architectures employed for task **A** and the learning methods for task **B**.

Tables 3 and 4 present our detailed results on the development split for both tasks for single and self-ensemble models. Table 3 shows the results for cross encoder and dual-encoders for task **A**. Our best single model, (ARB$^{\otimes}_{\approx}$), achieved a MAP of 34.83% and an MRR of 47.09%. (ARB$^{\otimes}_{\approx}$) self-ensemble achieved the best MAP of 36.70%. Table 3 also presents the R@10 and R@100 metrics. This represents the upper bound on the reranking

Short Form	Systems	Single Model								Self Ensemble	
		MAP	MRR	R@10	R@100	MAP$^{\star}_{\zeta}$	MAP (Question Type)			MAP	MAP$^{\star}_{\zeta}$
							Zero	Single	Multi		
Lexical Baseline											
BM$_{\sim}$	BM25	18.43	26.40	19.98	19.98	26.40	25.00	16.67	17.39	N/A	N/A
Dual-encoder											
ARB$^{\circledcirc}_{\sim}$	AraBERTv0.2-base *TASK A+ Random Neg*	20.02	42.87	29.72	48.23	20.02	0.00	35.42	19.20	N/A	N/A
ARB$^{\circledcirc}_{\approx}$	AraBERTv0.2-base *TASK A+ Hard Neg*	**24.44**	35.17	36.09	43.96	24.44	0.00	45.00	22.73	N/A	N/A
Cross Encoder											
ELC$^{\otimes}_{\sim}$	AraELECTRA *TASK A*	8.96	16.51	19.13	42.49	16.48	3.00	10.32	10.01	12.18	16.18
ELC$^{\otimes}_{\approx}$	AraELECTRA *TyDi QA$_{\text{AR}}$→Tafseer→TASK A*	26.60	41.61	38.52	59.19	31.91	19.00	38.31	23.94	29.13	36.56
CAM$^{\otimes}_{\sim}$	CAMeLBERT-CA *TASK A*	23.16	33.52	37.06	55.12	27.45	13.00	36.92	20.36	27.57	32.02
CAM$^{\otimes}_{\approx}$	CAMeLBERT-CA *TyDi QA$_{\text{AR}}$→Tafseer→TASK A*	29.34	42.17	39.93	57.23	33.81	18.00	**51.40**	23.54	32.77	36.77
ARB$^{\otimes}_{\sim}$	AraBERTv0.2-base *TASK A*	31.76	41.93	**46.55**	**62.71**	34.27	**46.00**	28.16	**29.41**	36.09	36.87
ARB$^{\otimes}_{\approx}$	AraBERTv0.2-base *TyDi QA$_{\text{AR}}$→Tafseer→TASK A*	**34.83**	**47.09**	39.99	60.82	**37.55**	43.00	46.22	28.10	**36.70**	**40.70**

Table 3: Dev split evaluation results for task **A**. **MAP** means ζ is set to mark 15% of questions as unanswerable. $\bigstar$ accompanied by ζ refers to applying the best ζ (see Appendix F). Average performance is reported for multiple runs of single models. Superscripts $\circledcirc$ and $\otimes$ in short form refer to dual-encoder and cross encoder, respectively. Subscripts $\sim$ and $\approx$ denote direct fine-tuning and pipelined fine-tuning, respectively.

stage performance that we can obtain (Yates et al., 2021).

Table 4 summarizes the results for task **B**. Our best performing model over the standard split, (ELC$^{M}_{\approx}$), attained a pAP of 53.36% and 55.21% for single model and self-ensemble models, respectively. Table 4 also presents results for the faithful validation split we defined previously. (ARB$^{M}_{\approx}$) is our best performing single model for the faithful split, achieving a pAP score of 54.19%.

Both tables present comprehensive results for different question types, as well as the outcomes for a manually set threshold ζ and $\zeta^{\star}$, i.e., the threshold that yields the best performance.
(See Appendix F for more details about ζ and optimal ζ selection)

Considering the question types , experiments of (ARB$^{\otimes}_{\sim}$) and (ARB$^{\otimes}_{\approx}$) obtains the best MAP performance for zero-answer and multi-answer questions for task **A**.

With regard to the hidden split, Tables 5 and 6 provide a summary of our official submissions.

In task **A**, as shown in Table 5, we made 3 cross-encoder submissions: MIX$^{\otimes}_{\approx}$, which is an ensemble combining runs from CAM$^{\otimes}_{\approx}$ and ARB$^{\otimes}_{\approx}$ cross encoders. MIX$^{\otimes}_{\approx}$ achieved a MAP of 25.05%. In comparison, the TF-IDF baseline only achieved a MAP of 9.03%.

On the other hand, in task **B**, we experimented with our two best performing models in Table 4. As shown in Table 6, (ARB$^{M}_{\approx}$) outperformed (ELC$^{M}_{\approx}$) with a pAP of 57.11%. This result is consistent with the findings from the faithful validation split (Keleg and Magdy, 2022) in Table 4 for (ARB$^{M}_{\approx}$) and (ELC$^{M}_{\approx}$). Specifically, the MAL method outperformed FAL for all of our models in the faithful validation split (underlined in Table 4).

6 Analysis and Discussion

Regarding **RQ1**, external resources always bring significant improvements to the same LM for both tasks. For task **A**, we have three stages of fine-tuning as indicated by arrows in Table 3. For example, when (ELC$^{\otimes}_{\sim}$) is fine-tuned with external resources into (ELC$^{\otimes}_{\approx}$) the MAP performance improves from 8.96% to 26.60% for single models as in Table 3. In similar fashion for task **B**, (ELC$^{M}_{\approx}$) outperforms (ELC$^{M}_{\sim}$) by almost 13% for the standard split in Table 4.

To answer our **RQ2**, ensemble learning consistently outperforms single models for both tasks. For instance, (CAM$^{\otimes}_{\approx}$) ensemble surpasses its single model by 3.5% for the MAP metric for task **A**. Similarly, (ELC$^{M}_{\approx}$) ensemble outperforms its corresponding single model by almost a pAP of 2% for task **B**.

With regard to **RQ3**, the hyperparameter ζ affects the zero answer type evaluation scores for both tasks. We make best use of the available data by employing a quantile method to determine the threshold ζ for both tasks. However, (ARB$^{\otimes}_{\approx}$) model MAP performance improves by 3% when the optimal $\zeta^{\star}$ is employed for task **A**. This suggests that there is a room for improvement for the

Short Form	Systems		Single Model								Self Ensemble Model		
	Model	Method	Faithful		Standard Development Split								
			pAP	pAP_{Post}	pAP	pAP_{Post}	$pAP^{\star}_{\zeta}$	pAP (Sample Type) Zero	Single	Multi	pAP	pAP_{Post}	$pAP^{\star}_{\zeta}$
$ELC^{F}_{\sim}$	AraELECTRA *TASK B*	FAL	34.97	41.23	38.27	44.40	39.26	18.67	41.51	31.18	41.16	46.50	41.72
$ELC^{M}_{\sim}$		MAL	37.44	42.63	40.55	45.56	41.48	14.67	43.69	36.04	42.01	47.21	43.90
$ELC^{F}_{\approx}$	AraELECTRA *TyDi QA_{AR}→TASK B*	FAL	52.76	55.45	49.76	53.70	51.99	10.33	54.36	43.69	50.66	55.35	52.75
$ELC^{M}_{\approx}$		MAL	53.15	55.43	**53.36**	**56.42**	**55.10**	18.33	**56.61**	**51.55**	**55.21**	**58.38**	**57.05**
$CAM^{F}_{\sim}$	CAMeLBERT-CA *TASK B*	FAL	41.45	45.76	37.63	42.04	38.36	11.00	40.83	33.13	42.51	45.50	43.18
$CAM^{M}_{\sim}$		MAL	43.54	47.36	38.57	43.38	39.38	12.67	40.52	39.20	41.66	45.39	43.80
$CAM^{F}_{\approx}$	CAMeLBERT-CA *TyDi QA_{AR}→TASK B*	FAL	50.64	53.12	41.59	46.50	42.39	13.67	44.36	39.39	47.03	49.37	47.12
$CAM^{M}_{\approx}$		MAL	52.14	54.01	40.08	44.80	41.30	15.00	41.61	42.18	42.75	46.87	44.23
$ARB^{F}_{\sim}$	AraBERTv0.2-base *TASK B*	FAL	44.81	48.93	45.66	49.34	46.60	23.67	49.29	37.74	49.38	53.05	50.01
$ARB^{M}_{\sim}$		MAL	47.41	50.62	45.71	47.69	46.85	25.67	48.43	41.03	49.69	52.03	51.28
$ARB^{F}_{\approx}$	AraBERTv0.2-base *TyDi QA_{AR}→TASK B*	FAL	52.97	55.86	50.62	54.43	51.28	**35.33**	53.78	42.39	52.20	55.77	53.45
$ARB^{M}_{\approx}$		MAL	**54.19**	**56.55**	50.51	53.32	51.35	31.33	53.22	45.54	52.13	54.94	52.94

Table 4: Dev split evaluation results for task **B**. **pAP** means fixing ζ to 0.8. ***Post*** subscript identifies post-processing. ★ accompanied by ζ refers to applying the best ζ (see Appendix F). Average performance is reported for multiple runs of single models. Superscripts F and M in short form indicate FAL and MAL methods, respectively. Subscripts $\sim$ and $\approx$ denote direct fine-tuning and pipelined fine-tuning, respectively. Underlined values refer to the higher performance when comparing the two learning methods.

Short Form	Self Ensemble Model	MAP	MRR
TF-IDF Baseline		9.03	22.60
$CAM^{\otimes}_{\approx}$	CAMeLBERT-CA *TyDi QA_{AR}→Tafseer→TASK A*	23.02	47.06
$ARB^{\otimes}_{\approx}$	AraBERTv0.2-base *TyDi QA_{AR}→Tafseer→TASK A*	24.64	**49.39**
$MIX^{\otimes}_{\approx}$	$CAM^{\otimes}_{\approx}$ + $ARB^{\otimes}_{\approx}$	**25.05**	46.10

Table 5: Results on the hidden split for task **A**. ζ is set to mark 15% of questions as unanswerable.

Short Form	Method	Self Ensemble Model	pAP@10
Full-passage Baseline			32.68
$ELC^{M}_{\approx}$	MAL	AraELECTRA *TyDi QA_{AR}→TASK B*	53.10
$ARB^{M}_{\approx}$		AraBERTv0.2-base *TyDi QA_{AR}→TASK B*	**57.11**
$MIX^{M}_{\approx}$		$ELC^{M}_{\approx}$ + $ARB^{M}_{\approx}$	56.43

Table 6: Results on the hidden split for task **B**. ζ is set to mark 5% of pairs as unanswerable.

ζ parameter. (Please refer to Appendix F for more details about ζ selection and **RQ3**).

In Table 3, we experimented with dual-encoders using both random and hard negatives (Zhan et al., 2021) to address **RQ4**. ($ARB^{\odot}_{\approx}$) outperforms ($ARB^{\odot}_{\sim}$) by almost 4.5% when we perform hard negatives mining using a fine-tuned checkpoint ($ARB^{\odot}_{\sim}$).

In Table 4, MAL learning method consistently brings significant improvements to the final performance for all models over the faithful split. Moreover, it consistently outperforms FAL learning method for the multi-answer type of samples. For instance, ($ELC^{M}_{\approx}$) performs better than ($ELC^{F}_{\approx}$), achieving a pAP score of 51.55% compared to 43.69% achieved by ($ELC^{F}_{\approx}$) for the subset of multi-answer samples. However, due to the fact that multi-answer samples make up only 18% of the development samples in the standard split (Table 2), MAL does not always outperform FAL for the standard split overall performance. This finding addresses **RQ5**.

With regard to **RQ6**, the post-processing approach proposed by Elkomy and Sarhan (2022) always surpasses the raw prediction score for both single and ensemble models. This is represented by ***Post*** subscript in Table 4. For example, post-processing improves ($ARB^{M}_{\approx}$) both single model and self-ensemble pAP performance by almost 3%.

7 Conclusion

In this paper, we have presented our solution for both task **A** and task **B** of Qur'an QA 2023 shared tasks. We explored various Arabic LMs using different training approaches and architectures. Our best performing systems are ensemble-based, enhanced with transfer learning using external learning resources. Lastly, we addressed a set of **RQ**s that highlight the main strengths of our work.

Limitations

In this paper, we have adapted conventional learning-based architectures for Arabic QA tasks, specifically for MRC and ad hoc search. However, we faced several challenges throughout our study. One significant challenge was the scarcity of training resources, along with the imbalanced

distribution of topics and question types. This was particularly evident in the zero-answer cases. As a consequence, our zero-answer thresholding mechanism demonstrated high sensitivity to each individual model.

Additionally, we noticed significant performance variations due to the small size of the datasets. In order to tackle the problem of variations and noisy predictions, we investigated an ensemble approach. However, we still suggest that the results we obtained during the development phase may not accurately reflect the actual performance of learning systems. Despite the effectiveness of faithful splits for task **B**, we still suggest exploring n-fold cross-validation for both tasks. However, our computation resources were significantly limited during the competition phase.

For task **B**, our models trained for MRC were found to be suboptimal for ranking tasks. Although our post-processing technique improved the raw predictions, this indicates the necessity for other ranking-based MRC approaches. Furthermore, we would like to explore the performance of large LMs on this particular task.

Ethics Statement

The paper contains facts and beliefs that do not necessarily reflect the views or opinions of the authors. The information presented is based on objective analysis and does not aim to promote or endorse any particular religious interpretation.

Acknowledgements

We would like to extend our heartfelt appreciation to Dr. Moustafa El Zantout for his invaluable support and insights during the course of this work. We would also like to express our deep gratitude to the organizers of Qur'an QA for their efforts in promoting research in Arabic in general, and the most significant Arabic text, the Holy Qur'an.

References

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. Arabert: Transformer-based model for arabic language understanding. pages 9–15. European Language Resource Association.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2021. Araelectra: Pre-training text discriminators for arabic language understanding. pages 191–195. Association for Computational Linguistics.

Muhammad Huzaifa Bashir, Aqil M Azmi, Haq Nawaz, Wajdi Zaghouani, Mona Diab, Ala Al-Fuqaha, and Junaid Qadir. 2021. Arabic natural language processing for qur'anic research: A systematic review.

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*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. pages 4171–4186. Association for Computational Linguistics.

Mohamemd Elkomy and Amany M Sarhan. 2022. Tce at qur'an qa 2022: Arabic language question answering over holy qur'an using a post-processed ensemble of bert-based models. pages 154–161. European Language Resources Association.

Siddhant Garg, Thuy Vu, and Alessandro Moschitti. 2020. Tanda: Transfer and adapt pre-trained transformer models for answer sentence selection. volume 34, pages 7780–7788.

Go Inoue, Bashar Alhafni, Nurpeiis Baimukan, Houda Bouamor, and Nizar Habash. 2021. The interplay of variant, size, and task type in arabic pre-trained language models. Association for Computational Linguistics.

Amr Keleg and Walid Magdy. 2022. Smash at qur'an qa 2022: Creating better faithful data splits for low-resourced question answering scenarios. pages 136–145. European Language Resources Association.

Rana Malhas and Tamer Elsayed. 2020. AyaTEC: Building a Reusable Verse-based Test Collection for Arabic Question Answering on the Holy Qur'an. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 19(6):1–21.

Rana Malhas and Tamer Elsayed. 2022. Arabic Machine Reading Comprehension on the Holy Qur'an using CL-AraBERT. *Information Processing & Management*, 59(6):103068.

Rana Malhas, Watheq Mansour, and Tamer Elsayed. 2022. Qur'an QA 2022: Overview of the First Shared Task on Question Answering over the Holy Qur'an. In *Proceedings of the 5th Workshop on Open-Source Arabic Corpora and Processing Tools (OSACT5) at the 13th Language Resources and Evaluation Conference (LREC 2022)*, pages 79–87.

Rana Malhas, Watheq Mansour, and Tamer Elsayed. 2023. Qur'an QA 2023 Shared Task: Overview of Passage Retrieval and Reading Comprehension Tasks over the Holy Qur'an. In *Proceedings of the First Arabic Natural Language Processing Conference (ArabicNLP 2023)*, Singapore.

RANA R MALHAS. 2023. *ARABIC QUESTION ANSWERING ON THE HOLY QUR'AN*. Ph.D. thesis.

Ali Mostafa and Omar Mohamed. 2022. Gof at qur'an qa 2022: Towards an efficient question answering for the holy qu'ran in the arabic language using deep learning-based approach. pages 104–111. European Language Resources Association.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. 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)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. pages 2383–2392. Association for Computational Linguistics.

Omer Sagi and Lior Rokach. 2018. Ensemble learning: A survey. *WIREs Data Mining and Knowledge Discovery*, 8(4):e1249.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*.

Ahmed Sleem, Eman Mohammed lotfy Elrefai, Marwa Mohammed Matar, and Haq Nawaz. 2022. Stars at qur'an qa 2022: Building automatic extractive question answering systems for the holy qur'an with transformer models and releasing a new dataset. pages 146–153. European Language Resources Association.

Marwan N. Swar. 2007. *Mushaf Al-Tafseel Al-Mawdoo'ee*. Dar Al-Fajr Al-Islami, Damascus.

Tanzil. 2007-2023. Tanzil - quran translations. `https://tanzil.net/trans/`. Electronic Quranic Resources.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Andrew Yates, Rodrigo Nogueira, and Jimmy Lin. 2021. Pretrained transformers for text ranking: BERT and beyond. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Tutorials*, pages 1–4, Online. Association for Computational Linguistics.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2021. Optimizing dense retrieval model training with hard negatives. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 1503–1512, New York, NY, USA. Association for Computing Machinery.

Appendix

A Dataset Additional Details

AyaTEC is a dataset designed to evaluate the performance of retrieval-based Arabic QA systems over the Holy Qur'an. It contains 207 questions and 1,762 corresponding answers, which are categorized into 11 topics covering different aspects of the Qur'an. The dataset caters to the information needs of two types of users: skeptical and curious (Malhas and Elsayed, 2020). The dataset includes single-answer and multi-answer questions, as well as questions that have no answer. Both Qur'an QA 2023 shared tasks are primarily based on an adapted version of AyaTEC (MALHAS, 2023; Malhas et al., 2022). Figure 1 illustrates an example from task **A**. The question asks whether there is a reference in the Qur'an to the body part used for reasoning. Four relevant Qur'anic segments are annotated to have an answer for this question. Figure 2 depicts a question-passage-answer triplet from task **B**. The question in this case is about creatures capable of praising God, within the context of the given passage.

A.1 Topic Distribution for tasks

AyaTEC covers 11 diverse topics referenced in the Holy Qur'an. Figure 3 illustrates the imbalanced nature of those different topics. Furthermore, the representation of unique questions is significantly limited in comparison to question-passage-answer triplets. Additionally, it is evident that the ratio of triplets to unique questions varies for each respective topic. In task **B**, these factors give rise to common questions across various passages. Consequently, they result in data leakage between the training and development splits (Keleg and Magdy, 2022). (Further information regarding this can be found in Appendix A.4)

A.2 Task A Evaluation Measures

For this ranking task, systems are expected to return up to 10 Qur'anic passages for each question when possible. If the system determines that the question is unanswerable from the entire Qur'an, a null document is only returned, indicated by -1. The primary measure for the task is MAP, which gives full credit only if all relevant documents are retrieved at the top of the ranked answer list. For

Question ID: 428	
Question	
هل أشار القرآن إلى العضو الذي يعقل به الإنسان؟ **Did the Holy Qur'an refer to the body part humans use for reasoning?**	
Answer ID	**Answer Text**
6:22-26	وَيَوْمَ نَحْشُرُهُمْ جَمِيعًا ثُمَّ نَقُولُ لِلَّذِينَ أَشْرَكُوٓا۟ أَيْنَ شُرَكَآؤُكُمُ ٱلَّذِينَ كُنتُمْ تَزْعُمُونَ. ثُمَّ لَمْ تَكُن فِتْنَتُهُمْ إِلَّآ أَن قَالُوا۟ وَٱللَّهِ رَبِّنَا مَا كُنَّا مُشْرِكِينَ. ٱنظُرْ كَيْفَ كَذَبُوا۟ عَلَىٰٓ أَنفُسِهِمْ ۖ وَضَلَّ عَنْهُم مَّا كَانُوا۟ يَفْتَرُونَ. وَمِنْهُم مَّن يَسْتَمِعُ إِلَيْكَ ۖ وَجَعَلْنَا عَلَىٰ قُلُوبِهِمْ أَكِنَّةً أَن يَفْقَهُوهُ وَفِىٓ ءَاذَانِهِمْ وَقْرًا ۖ وَإِن يَرَوْا۟ كُلَّ ءَايَةٍ لَّا يُؤْمِنُوا۟ بِهَا ۚ حَتَّىٰٓ إِذَا جَآءُوكَ يُجَٰدِلُونَكَ يَقُولُ ٱلَّذِينَ كَفَرُوٓا۟ إِنْ هَٰذَآ إِلَّآ أَسَٰطِيرُ ٱلْأَوَّلِينَ. وَهُمْ يَنْهَوْنَ عَنْهُ وَيَنْـَٔوْنَ عَنْهُ ۖ وَإِن يُهْلِكُونَ إِلَّآ أَنفُسَهُمْ وَمَا يَشْعُرُونَ.
7:179-179	وَلَقَدْ ذَرَأْنَا لِجَهَنَّمَ كَثِيرًا مِّنَ ٱلْجِنِّ وَٱلْإِنسِ ۖ لَهُمْ قُلُوبٌ لَّا يَفْقَهُونَ بِهَا وَلَهُمْ أَعْيُنٌ لَّا يُبْصِرُونَ بِهَا وَلَهُمْ ءَاذَانٌ لَّا يَسْمَعُونَ بِهَآ ۚ أُو۟لَٰٓئِكَ كَٱلْأَنْعَٰمِ بَلْ هُمْ أَضَلُّ ۚ أُو۟لَٰٓئِكَ هُمُ ٱلْغَٰفِلُونَ
22:42-46	وَإِن يُكَذِّبُوكَ فَقَدْ كَذَّبَتْ قَبْلَهُمْ قَوْمُ نُوحٍ وَعَادٌ وَثَمُودُ. وَقَوْمُ إِبْرَٰهِيمَ وَقَوْمُ لُوطٍ. وَأَصْحَٰبُ مَدْيَنَ ۖ وَكُذِّبَ مُوسَىٰ فَأَمْلَيْتُ لِلْكَٰفِرِينَ ثُمَّ أَخَذْتُهُمْ ۖ فَكَيْفَ كَانَ نَكِيرِ. فَكَأَيِّن مِّن قَرْيَةٍ أَهْلَكْنَٰهَا وَهِىَ ظَالِمَةٌ فَهِىَ خَاوِيَةٌ عَلَىٰ عُرُوشِهَا وَبِئْرٍ مُّعَطَّلَةٍ وَقَصْرٍ مَّشِيدٍ. أَفَلَمْ يَسِيرُوا۟ فِى ٱلْأَرْضِ فَتَكُونَ لَهُمْ قُلُوبٌ يَعْقِلُونَ بِهَآ أَوْ ءَاذَانٌ يَسْمَعُونَ بِهَا ۖ فَإِنَّهَا لَا تَعْمَى ٱلْأَبْصَٰرُ وَلَٰكِن تَعْمَى ٱلْقُلُوبُ ٱلَّتِى فِى ٱلصُّدُورِ.
47:20-24	وَيَقُولُ ٱلَّذِينَ ءَامَنُوا۟ لَوْلَا نُزِّلَتْ سُورَةٌ ۖ فَإِذَآ أُنزِلَتْ سُورَةٌ مُّحْكَمَةٌ وَذُكِرَ فِيهَا ٱلْقِتَالُ ۙ رَأَيْتَ ٱلَّذِينَ فِى قُلُوبِهِم مَّرَضٌ يَنظُرُونَ إِلَيْكَ نَظَرَ ٱلْمَغْشِىِّ عَلَيْهِ مِنَ ٱلْمَوْتِ ۖ فَأَوْلَىٰ لَهُمْ. طَاعَةٌ وَقَوْلٌ مَّعْرُوفٌ ۚ فَإِذَا عَزَمَ ٱلْأَمْرُ فَلَوْ صَدَقُوا۟ ٱللَّهَ لَكَانَ خَيْرًا لَّهُمْ. فَهَلْ عَسَيْتُمْ إِن تَوَلَّيْتُمْ أَن تُفْسِدُوا۟ فِى ٱلْأَرْضِ وَتُقَطِّعُوٓا۟ أَرْحَامَكُمْ. أُو۟لَٰٓئِكَ ٱلَّذِينَ لَعَنَهُمُ ٱللَّهُ فَأَصَمَّهُمْ وَأَعْمَىٰٓ أَبْصَٰرَهُمْ. أَفَلَا يَتَدَبَّرُونَ ٱلْقُرْءَانَ أَمْ عَلَىٰ قُلُوبٍ أَقْفَالُهَا.

Figure 1: A sample from shared task **A**. We highlight the most relevant part in each Qur'anic segment.

Sample ID: 17:40-44_164	
Passage	
أَفَأَصْفَىٰكُمْ رَبُّكُم بِٱلْبَنِينَ وَٱتَّخَذَ مِنَ ٱلْمَلَٰٓئِكَةِ إِنَٰثًا ۚ إِنَّكُمْ لَتَقُولُونَ قَوْلًا عَظِيمًا. وَلَقَدْ صَرَّفْنَا فِى هَٰذَا ٱلْقُرْءَانِ لِيَذَّكَّرُوا۟ وَمَا يَزِيدُهُمْ إِلَّا نُفُورًا. قُل لَّوْ كَانَ مَعَهُۥٓ ءَالِهَةٌ كَمَا يَقُولُونَ إِذًا لَّٱبْتَغَوْا۟ إِلَىٰ ذِى ٱلْعَرْشِ سَبِيلًا. سُبْحَٰنَهُۥ وَتَعَٰلَىٰ عَمَّا يَقُولُونَ عُلُوًّا كَبِيرًا. تُسَبِّحُ لَهُ ٱلسَّمَٰوَٰتُ ٱلسَّبْعُ وَٱلْأَرْضُ وَمَن فِيهِنَّ ۚ وَإِن مِّن شَىْءٍ إِلَّا يُسَبِّحُ بِحَمْدِهِۦ وَلَٰكِن لَّا تَفْقَهُونَ تَسْبِيحَهُمْ ۚ إِنَّهُۥ كَانَ حَلِيمًا غَفُورًا.	
Question	
ما المخلوقات التي تسبح الله؟ **What creatures are capable of praising God?**	
Answer	
١ - ٱلسَّمَٰوَٰتُ ٱلسَّبْعُ وَٱلْأَرْضُ ٢ - إِن مِّن شَىْءٍ إِلَّا يُسَبِّحُ بِحَمْدِهِۦ	

Figure 2: A sample from shared task **B**. We highlight the ground truth answers in the Qur'anic passage.

the zero-answer questions, full credit is given to successful systems only when they are unable to find any relevant Qur'anic passage to answer the question, and return the null document. In addition to MAP, mean Reciprocal Rank (MRR) is also reported, which gives credit just for the first relevant document from the ranked list (Yates et al., 2021).

In formal notation, we begin by defining the function $\alpha(q, p)$, which is a binary relevance function that indicates whether a passage p is annotated as relevant to a question q in the test collection. Equ.(1) represents the function that calculates the total number of relevant Qur'anic passages from the QPC to q.

$$\psi(q) = \sum_{p \in \mathcal{QPC}} \alpha(q, p) \tag{1}$$

Zero-answer questions have a zero value for the function ψ, and their MAP score is calculated in a different way. Equ.(2) shows the evaluation measure for MAP for answerable questions. For a ranked list R, we calculate the precision at each possible cutoff $@i$ at which a relevant document is present (Yates et al., 2021).

$$\text{MAP}(R, q) = \frac{\sum_{(i,p) \in R} \text{Prec}\,@i(R, q) \cdot \alpha(q, p)}{\psi(q)}, \tag{2}$$

Equ.(3) illustrates the combined MAP evaluation measure for task **A**. In this measure, zero-answer questions are given full credit only when R is the null document, represented by -1 in the official evaluation script [3] (MALHAS, 2023).

$$\text{MAP}_{\text{A}}(R, q) = \begin{cases} \mathbb{1}_{R \equiv [-1]} & \text{if } \psi(q) = 0 \\ \text{MAP}(R, q) & \text{Otherwise} \end{cases} \tag{3}$$

$\mathbb{1}_C$ is an *indicator* function, which returns 1 if the binary condition C holds and 0 otherwise.

A.3 Task B Evaluation Measures

Standard MRC tasks, like SQuADv2.0, are evaluated based only on the first prediction. In contrast, task **B** is evaluated as a ranking task against a ranked list, rather than relying solely on the top prediction. As in task **A**, systems are expected to return up to 10 answer spans from a given Qur'anic passage to answer a question when possible. The primary evaluation metric for this task is pAP (Malhas and Elsayed, 2020; MALHAS, 2023). This metric incorporates partial matching with the traditional rank-based Average Precision measure, i.e., MAP. In the case of unanswerable samples, the system receives a full score if it only returns and empty ranked list.

Formally, partial matching is performed over token indexes of two substrings extracted from a given supporting passage. Based on Malhas and Elsayed (2020), F_1 is used to calculate the similarity between the two substrings R_k and g. R_k represents the k^{th} answer from a ranked list R, and g refers to any ground truth answer from the set of ground truth answers G.

$$\mathcal{F}_{1_k}^{R} = \max_{g \in G} \{F_1(R_k, g)\} \tag{4}$$

In terms of Equ.(4), we can define a partial matching version of precision at cutoff K, i.e., pPrec (Malhas and Elsayed, 2020; MALHAS, 2023).

$$\text{pPrec}\,@K(R) = \frac{1}{K} \sum_{i=1}^{K} \mathcal{F}_{1_i}^{R} \tag{5}$$

In their study, MALHAS (2023) introduced a method for handling multi-answer samples. They proposed a string splitting mechanism that ensures only one correct answer is matched in each entry of R. Equ.(6) presents the pAP evaluation metric for multi-answer ranking MRC in terms of pPrec (Malhas and Elsayed, 2022), which stands as a token-level partial matching version of Equ(2).

$$\text{pAP}(R) = \frac{\sum_{i \in R} \text{pPrec}\,@i(R) \cdot \beta(R, i)}{|G|}, \tag{6}$$

$\beta(R, i)$ is a binary function that returns one if R_i is a partially relevant answer. More specifically,

$$\beta(R, k) = \mathbb{1}_{\mathcal{F}_k^R > 0} \tag{7}$$

In similar fashion, Equ.(8) presents the complete pAP evaluation measure for task **B**. In this measure, zero-answer samples are given full credit only when R is an empty list (MALHAS, 2023).

$$\text{pAP}_{\text{B}}(R) = \begin{cases} \mathbb{1}_{R \equiv [\,]} & \text{if } |G| = 0 \\ \text{pAP}(R) & \text{Otherwise} \end{cases} \tag{8}$$

[3] The symbol $\equiv$ signifies the equivalence operator between two lists.

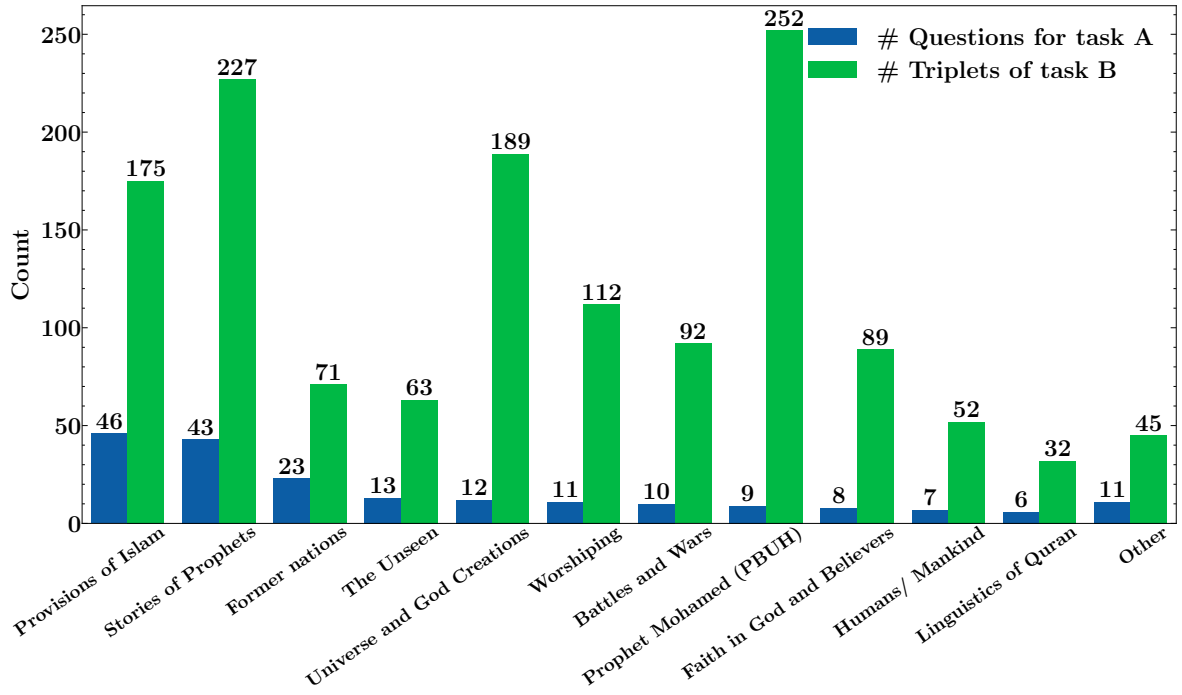


Figure 3: Distribution of QRCDv1.2 over the 11 topics for task **A** questions and task **B** triplets.

A.4 Leakage in QRCDv1.2

Keleg and Magdy (2022) analyzed QRCDv1.1 and identified instances where passages and questions were repeated. They classified QRCDv1.1 into four logical mutually-exclusive categories according to their complexity. Table 7 provides a summary of the criteria used and the expected behavior of trained LMs for each category. Additionally, symbols are employed to indicate the levels of complexity within each category, as determined by performance scores obtained by Keleg and Magdy (2022). Based on their analysis, Keleg and Magdy (2022) solely utilized $D_{(3)\text{ ood + hard}}$ for their final development split for QRCDv1.1.

In this work, we extend their approach for QRCDv1.2. We slightly modify this by considering both $D_{(2)}$ and $D_{(3)}$ for the development split. In addition, we employ disjoint set algorithm to find all leakage groups in $D_{(1)}$. We use those groups to balance the zero-answer questions ratio in the development split. This is because 90% of zero-answer questions belong to the trivial leakage group $D_{(1)}$.

In their work, Keleg and Magdy (2022) also proposed a resplitting approach for QRCDv1.1. They reorganized training and development splits using the four logical groups to create what they called *faithful* splits for QRCDv1.1. Faithful splits aim to create more representative evaluations for QRCDv1.1 dataset. Table 8 summarizes the modifications we made for performing evaluation using faithful splits. Table 9 presents the distribution of our faithful split for QRCDv1.2 based on our modified splitting strategy outlined in Table 8. It also includes the distribution of zero-answer samples within each group. As in Table 9, we preserve the original ratio of training to development splits. Additionally, the percentage of zero-answer samples within each split is preserved compared to the original distribution in Table 2.

A.5 External Learning Resources

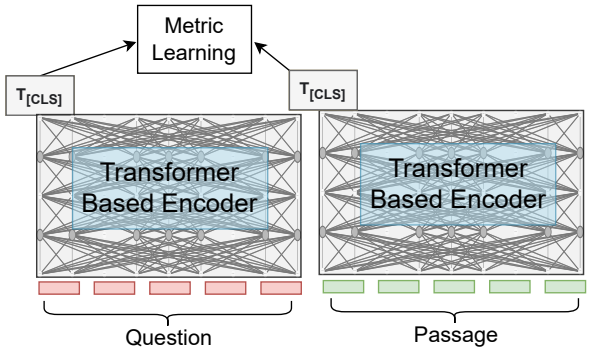
We leverage external resources to perform pipelined fine-tuning for both tasks **A** and **B**. For task **A**, we utilized interpretation resources (tafseer) from both Muyassar and Jalalayn, obtained from Tanzil (2007-2023). We created pairs of QPC Qur'anic passages and their corresponding interpretations, resulting in approximately 2.5K relevant pairs. Additionally, we used the Arabic TyDI-QA Gold**P** dataset (Clark et al., 2020) to generate pairs of relevant questions and their supporting evidence passages, resulting in 15K relevant pairs. For task **B**, we solely relied on the Arabic subset of the TyDI-QA Gold**P** MRC dataset (Clark et al., 2020). This dataset consists of approximately 15K question-passage-answer triplets.

Category	Criteria	Expected LM behavior
$D_{(1)}$ in+leakage	Samples with repeated passage-answer or question-answer pairs	Memorize answers and overfit to training data $\sim$
$D_{(2)}$ in+no leakage	Samples with repeated passages but having unique answers which are different from $D_{(1)}$ answers	Reasoning is required to find the right answer $\approx$
$D_{(3)}$ ood + hard	Samples with unique passages but having rarely repeated questions (appearing 3 times or less)	Some reasoning is required to find the right answer for rare questions $\approx$
$D_{(4)}$ ood + easy	Samples with unique passages but having commonly repeated questions (more than 3 times)	Lexical matching guides trained LMs to find similar answers $\approx$

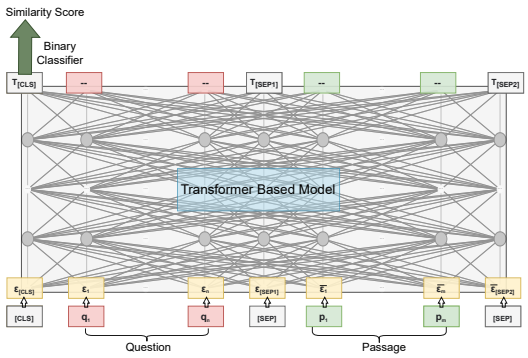
Table 7: Description of the four categories introduced by Keleg and Magdy (2022) over QRCDv1.1 dataset. We show the criteria for identifying each category and the expected behavior for a fine-tuned LM. We denote the complexity of each category using symbols. For instance, $\approx$ represents the most challenging set for learning systems, while $\sim$ refers the least challenging set.

Category	Splitting Strategy by Keleg and Magdy (2022)	Our Modified Splitting Strategy
$D_{(1)}$ in+leakage	For duplicate question-answer or passage-answer pairs, choose only one sample for training and leave the rest for the development set.	Use it entirely for training, this is due to the fact that $D_{(1)}$ is trivial for development. To balance the zero-answer questions ratio, we take entire zero-answer leakage groups into the development set. We employ disjoint-set algorithm for this purpose.
$D_{(2)}$ in+no leakage	Split randomly with a splitting ratio of 86.7% for training and 13.3% for development, which corresponds to the original ratio of the data.	Split them into two overlapping sets, as such, confusing examples with the same passages are distributed among training and development with different answers.
$D_{(3)}$ ood + hard	Only use it for the development set (removed from training).	Same as Keleg and Magdy (2022)
$D_{(4)}$ ood + easy	Split randomly with a splitting ratio of 86.7% for training and 13.3% for development, which corresponds to the original ratio of the data.	Use it entirely for training, this is due to the fact that $D_{(4)}$ is trivial for development.

Table 8: Description of our modified *faithful* splitting for QRCDv1.2 dataset over the four categories introduced by Keleg and Magdy (2022). We also show their proposed splitting approach (Keleg and Magdy, 2022). Check Table 7 for more details and reasons behind such splitting strategies.



(a) Dual-encoder generic architecture with metric learning for neural ranking.



(b) Cross-encoder generic architecture for an input pair of a question and a passage with a predicted similarity score.

Figure 4: Diagrams for model architectures for task **A**.

Category	Train	Development	Total
$D_{(1)\text{ in+leakage}}$	405 (49)	7 (7)	412 (56)
$D_{(2)\text{ in+no leakage}}$	290 (2)	95 (1)	385 (3)
$D_{(3)\text{ ood + hard}}$	0 (0)	62 (3)	62 (3)
$D_{(4)\text{ ood + easy}}$	296 (0)	0 (0)	296 (0)
Total	991 (51)	164 (11)	1155 (62)
Zero-answer %	5.15 %	6.71 %	5.37 %

Table 9: QRCDv1.2 dataset distribution of pairs for our *faithful* splitting over the four categories introduced by (Keleg and Magdy, 2022). Parenthesized values refer to the number of zero-answer samples within each category for each split.

B Transfer Learning

In order to overcome the limited training resources for both tasks, we incorporate external QA and interpretation resources (tafseer) (Tanzil, 2007-2023). External resources enhance our learning systems in general by leveraging transfer learning across multiple fine-tuning stages (Garg et al., 2020; MALHAS, 2023). We use arrows in subscripts in Tables 3, 4, 5, and 6 to refer to stages of fine-tuning. (More details about external learning resources and their construction in Appendix A.5)

C Ensemble Learning

We utilize a voting self-ensemble technique for a group of fine-tuned models trained with different seeds (Sagi and Rokach, 2018). We use the raw predictions without applying a zero-answer threshold.

In task **A**, for an ensemble $\mathcal{E}$ we aggregate the relevance scores for a Qur'anic passage p and a question q assigned by a model φ. The ensemble relevance score $\mathcal{S}$ between p and q is as follows:

$$\mathcal{S}(q,p) = \sum_{\varphi \in \mathcal{E}} \varphi(q,p) \tag{9}$$

In similar fashion for task **B**, we leverage a span voting ensemble (Elkomy and Sarhan, 2022). For each sample, we aggregate span scores for each span s made by each predictor φ.

$$\mathcal{S}(s) = \sum_{\varphi \in \mathcal{E}} \varphi(s) \tag{10}$$

After that, we apply zero-answer thresholding to the aggregated result.

D Additional System Details for task A

We summarize both architectures for task **A** in Figures 4a and 4b for dual-encoders and cross-encoders, respectively.

D.1 Implementation Details

In our STAR training process, we incorporate both random in-batch negatives and hard negatives. Random negatives involve randomly selecting irrelevant documents for each query, providing positive and negative signals for learning systems (Yates et al., 2021). On the other hand, hard negatives refer to the most offending irrelevant examples predicted by an encoder similarity score (Zhan et al., 2021). In a batch of size 16, we encode 16 different queries with their corresponding positive documents; in addition, in-batch negatives are used for all other queries. These negatives can be chosen randomly or through STAR hard negative mining. We use a learning rate of 5×10^{-5} for all of our dual-encoder experiments. In the case of cross-encoders, we generate question-document pairs. These pairs have a ratio of one positive pair and three randomly selected negative pairs. For all of our cross-encoders, we use a learning rate of 1×10^{-6} with a batch size of 16.

D.2 Zero-answer Prediction

We assign a likelihood for each question q to be answerable using the total relevance scores for the top returned passages R. φ refers to a general relevance predictor between q and a passage p.

$$\gamma(q) = -\sum_{p \in R} \varphi(q,p) \tag{11}$$

The negative sign corresponds to the inverse proportional relationship between high relevance scores and the likelihood of unanswerability. We then normalize those scores for all questions into $\bar{\gamma}(q)$ and apply a no answer threshold ζ. We define a binary threshold function, σ, which applies the threshold to identify unanswerable questions.

$$\sigma(q) = \mathbb{1}_{\bar{\gamma}(q) > \zeta} \tag{12}$$

E Additional System Details for task B

In this work, we fine-tune LMs for extractive MRC as span predictors (Devlin et al., 2019). The fine-tuning process involves packing each question-passage pair x together and feeding it to a LM to predict the start and end token indices from the passage, as shown in Figure 5. To achieve this, a trainable randomly initialized start vector S and end vector E are stacked on top of the LM, having the i^{th} token hidden-representation T_i. The final model with the newly stacked layers has learnable parameters θ.

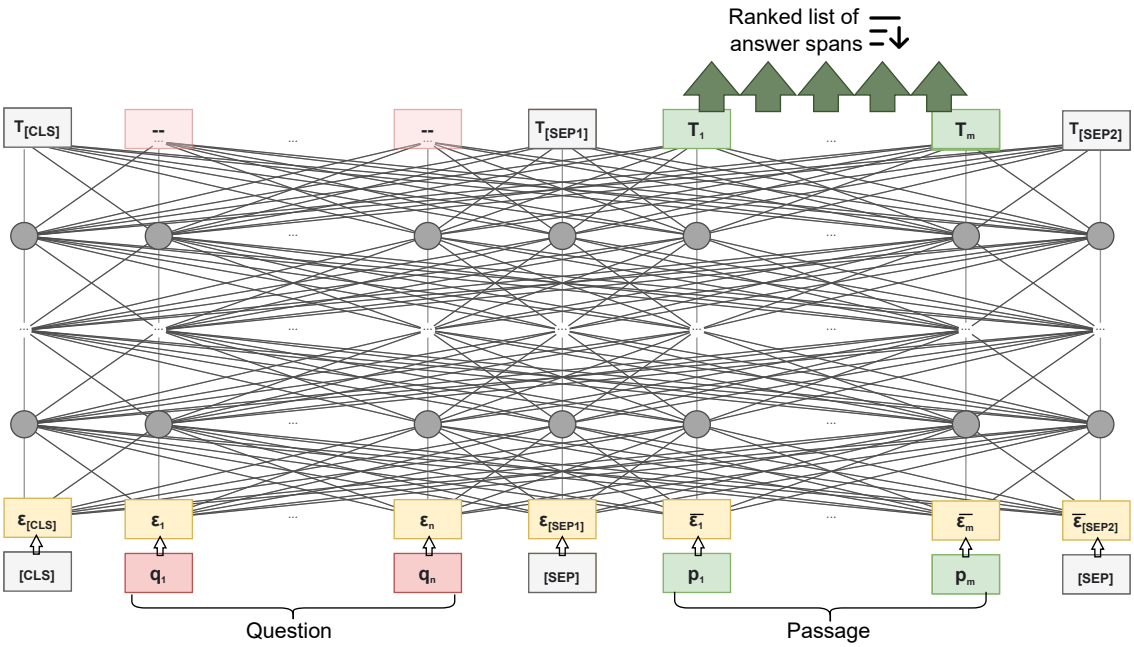
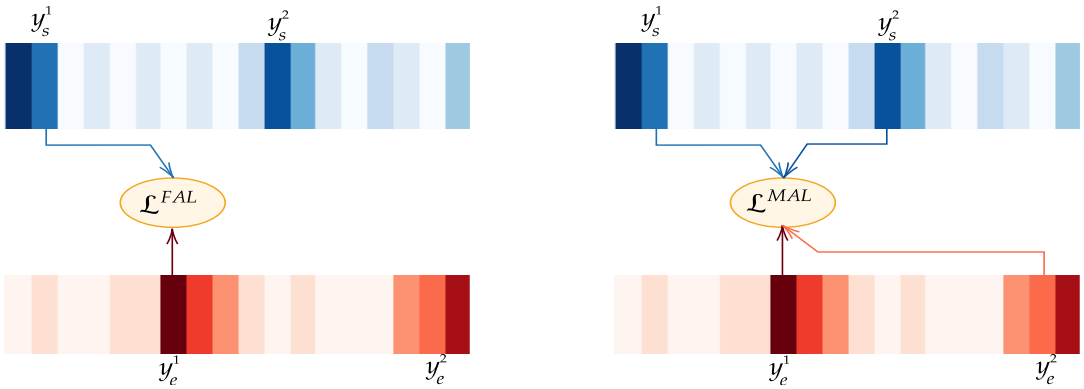


Figure 5: Generic architecture illustration of a LM for ranking MRC.



(a) Standard (FAL) (b) MAL

Figure 6: Illustration of Learning Methods.

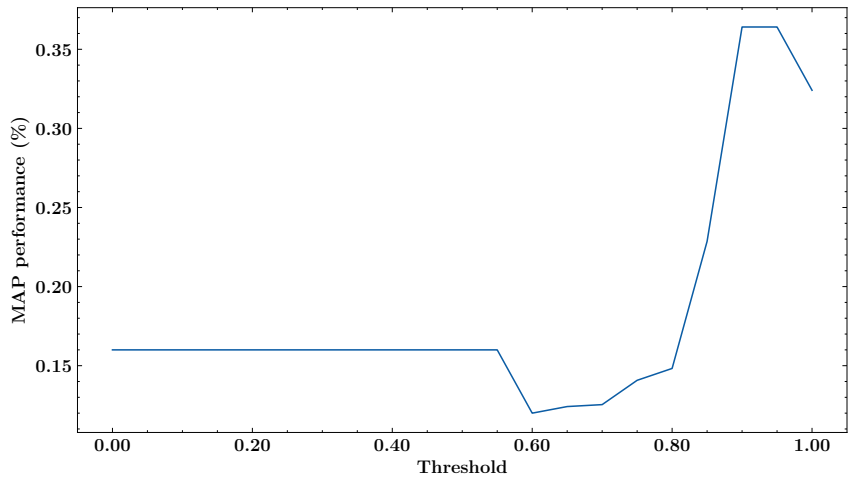


Figure 7: Thresholding effect against MAP performance for one of our fine-tuned models.

The dot product between S and T_i is chosen to determine the score that the i^{th} token is the start of the answer span. These scores for all passage tokens are followed by a softmax layer that produces the probabilities for individual tokens being the start of the answer span (Seo et al., 2016; Devlin et al., 2019). Equ.(13) depicts the probability that the i^{th} token is the start of the answer span.

$$\mathbb{P}\left(i \mid x; \theta\right) = \frac{e^{S \cdot T_i}}{\sum_j^{|T|} e^{S \cdot T_j}} \tag{13}$$

Under full-supervision, the training objective is to optimize the log-likelihoods for both the ground truth start and end positions. For a model with learnable θ, an input x, and a single ground truth answer span y, the log likelihood for the start token position is as follows:

$$\mathcal{L}_{\text{start}}\left(\theta \mid x, y\right) = -\log \mathbb{P}\left(y_s \mid x; \theta\right) \tag{14}$$

where the subscript s in y_s refers to the start position of the answer span y.

If there are multiple answers for a sample x, we rather have a set of plausible answer spans $\mathcal{Y}$. Elkomy and Sarhan (2022); Sleem et al. (2022); Mostafa and Mohamed (2022) in Qur'an QA 2022 tackled this by considering any answer span from $\mathcal{Y}$ by taking one at random or the first answer span, namely, y^1. We denote the i^{th} answer from $\mathcal{Y}$ as y^i. We call this learning method First answer loss (FAL). This can be formulated in terms of $\mathcal{Y}$ as denoted below:

$$\mathcal{L}_{\text{start}}^{\text{FAL}}\left(\theta \mid x, \mathcal{Y}\right) = -\log \mathbb{P}\left(y_s^1 \mid x; \theta\right) \tag{15}$$

Figure 6a illustrates this learning method. However, QRCDv1.2 task **B** considers a multi-answer MRC scenario, this leads to discrepancy between training and testing when FAL learning method is employed for fine-tuning. Towards this end, we define MAL learning method. This learning method takes the multi-answer cases in consideration by optimizing for all answers altogether. Mathematically, this generalizes to any y^i from the set $\mathcal{Y}$ and takes the sum of the log likelihood losses for multiple answers as shown in Equ.(16):

$$\mathcal{L}_{\text{start}}^{\text{MAL}}\left(\theta \mid x, \mathcal{Y}\right) = -\sum_{y^i \in \mathcal{Y}} \log \mathbb{P}\left(y_s^i \mid x; \theta\right) \tag{16}$$

We show the MAL learning method in Figure 6b.

E.1 Implementation Details

To enhance LMs predictions, we employ a post-processing approach. Elkomy and Sarhan (2022) proposed an effective non-maximum suppression post-processing approach at Qur'an QA 2022 (Malhas et al., 2022). They also proposed some operations for rejecting uninformative short answers. For all of our models, we used a learning rate of 2×10^{-5} and a batch size of 16.

E.2 Zero-answer Prediction

MRC for SQuADv2.0-like datasets uses null answer **[CLS]** token probability to give a likelihood for a question to have an answer within the supporting passage (Rajpurkar et al., 2018; Devlin et al., 2019). This works by finding the difference between the null answer score of **[CLS]** token and the non-empty answer span with the highest score. φ is a general span extractor that operates on a question q and a passage p.

$$\gamma(q, p) = \varphi(q, p)_{[CLS]} - \varphi(q, p)_{MAX} \tag{17}$$

Upon calculating scores for all samples, we proceed to normalize them into $\bar{\gamma}(q)$ and then apply a threshold value ζ to determine if there is no answer. To identify unanswerable questions, we use a binary threshold function σ,

$$\sigma(q) = \mathbb{1}_{\bar{\gamma}(q) > \zeta} \tag{18}$$

F ζ Selection and $\zeta^\star$

In this work, we defined ζ hyperparameter for zero-answer thresholding. This hyperparameter controls the proportion of samples that are considered to be zero-answer. Due to the small size of the dataset, we used a quantile method to set ζ. This method marks a proportion of the samples according to the statistics of the dataset. Task **B** is less sensitive to this parameter because almost 5% of the samples are zero-answer. In contrast, task **A** is highly sensitive to this parameter because of the larger proportion of zero-answer cases compared to task **A**. Additionally, We are interested in finding the theoretical upperbound performance for ζ; this is addressed by **RQ3**.

In Tables 3 and 4 we use ★ accompanied by ζ to refer to the optimal performance of the binary classification problem of has-answer vs. has-no-answer, as explained in Appendices D.2 and E.2. Figure 7 illustrates the thresholding effect against

fine-tuned model performance for task **A**; this answers **RQ3**. As we can see, the ζ hyperparameter can not be set arbitrarily. Instead, we can adjust it by considering the outcomes obtained from trained models on the training data. To find the optimal threshold $\zeta^{\bigstar}$ for both tasks, we implemented a greedy optimization algorithm for all possible levels of thresholds made by a given model; check the code for more details [4].

[4] In both code bases, this is performed by function *find_best_thresh*. You may find this function under *metrics* directory in *compute_score_qrcd.py* and *helpers.py* scripts for tasks **A** and **B**, respectively.