

Deep learning based Arabic short answer grading in serious games

Younes Alaoui Soulimani¹, Lotfi El Achaak², Mohammed Bouhorma¹

¹Smart Systems and Emergent Technologies Team, Faculty of Sciences and Techniques of Tangier, Abdelmalek Essaâdi University, Tangier, Morocco

²Data and Intelligent Systems Team, Faculty of Sciences and Techniques of Tangier, Abdelmalek Essaâdi University, Tangier, Morocco

Article Info

Article history:

Received Dec 26, 2022

Revised May 3, 2023

Accepted Jun 4, 2023

Keywords:

Automated short answer grading
Bidirectional encoder representations from transformers
Long-short-term-memory
Machine learning
Natural language processing
Serious games
Transformer

ABSTRACT

Automatic short answer grading (ASAG) has become part of natural language processing problems. Modern ASAG systems start with natural language preprocessing and end with grading. Researchers started experimenting with machine learning in the preprocessing stage and deep learning techniques in automatic grading for English. However, little research is available on automatic grading for Arabic. Datasets are important to ASAG, and limited datasets are available in Arabic. In this research, we have collected a set of questions, answers, and associated grades in Arabic. We have made this dataset publicly available. We have extended to Arabic the solutions used for English ASAG. We have tested how automatic grading works on answers in Arabic provided by schoolchildren in 6th grade in the context of serious games. We found out those schoolchildren providing answers that are 5.6 words long on average. On such answers, deep learning-based grading has achieved high accuracy even with limited training data. We have tested three different recurrent neural networks for grading. With a transformer, we have achieved an accuracy of 95.67%. ASAG for school children will help detect children with learning problems early. When detected early, teachers can solve learning problems easily. This is the main purpose of this research.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Younes Alaoui Soulimani

Smart Systems and Emergent Technologies Team, Faculty of Sciences and Techniques of Tangier,

Abdelmalek Essaâdi University

P.O. Box 416, Tangier, Morocco

Email: younes.alaoui@amana.ac.ma

1. INTRODUCTION

Assessment and evaluation of learning are important steps in learning and knowledge transmission processes. Instructional design processes that focus on designing and developing learning systems [1], include always a phase called “develop assessment instruments” [2]. Teachers and instruction designers will create assessment tools like exams, assignments, or quizzes. They will usually create different types of questions, which answers are true/false, multiple choice, matching, numerical values, essay, or short answers. Short answer questions require students to answer in free text composing some sentences, typically one or two. This type of question has the advantage of requiring students to construct an answer by themselves, rather than selecting answers from predetermined lists.

Serious games refer currently to video games designed to train people or to transmit learning. Serious games can complement classroom transmission or help with distance learning. The development of

serious games involves pedagogy, didactic, learning design, and game design [3]. Assessment of learning within games is an important feature that helps make learning effective in serious games [4].

Answers written in free text such as short answers and essays have traditionally been absent from computerized tests and serious games because they were considered difficult to evaluate and grade automatically [5], [6]. Because of this challenge, automatic short answer grading (ASAG) has become a research problem. Burrows *et al.* [5] classified the different approaches tried on ASAG problems in five eras, the fourth one being machine learning. Recent advances in natural language processing (NLP) as well as in machine learning applied to NLP are providing promising results on ASAG problems [7]–[9]. Different ASAG applications started emerging and active research in this field has developed [5].

The objective of this research is to extend automated grading based on machine learning to questions and answers written in Arabic. Recent research has tested deep learning approaches on ASAG of answers written in English mainly. The approaches of these researchers seemed generic enough to adapt to Arabic. We wanted also to extend and test the same approaches on answers collected initially in Arabic and not originating from datasets translated from English.

This research targeted questions and answers aimed at schoolchildren from fifth and sixth grade and aged 11 and 12 years old. We have collected a dataset for this research. We have used a standard NLP pipeline. We have leveraged an existing machine-learning algorithm to project Arabic words on numerical vectors that deep-learning algorithms can work with. To grade our short answers initially written in Arabic, we have tested three deep learning approaches namely long-short-term-memory (LSTM), transformers, and bidirectional encoder representations from transformers (BERT). We have deployed our automatic grading in an operational environment and tested this grading in the context of continuous learning evaluation and serious games. We have also made the dataset available to other research projects.

The organization of the paper is as follows. We review the state of the art in section 2. We present our research method in section 3. We discuss the results of this research in section 4. We summarize this work and describe possible enhancements and future work in the last section.

2. STATE OF THE ART

ASAG grades answers written in free text and leverages the approach from NLP. The NLP approaches currently used for ASAG are exploring deep learning models with recurrent neural networks. Kumar *et al.* [10], Prabhudesai and Duong [11], and Xia *et al.* [12] have explored LSTM-based models for ASAG. Alikaniotis *et al.* [13] have introduced a model based on LSTM for text scoring and are able to discover which specific words impact the score. Roy *et al.* [14] have proposed a technique to overcome the need to have labeled training data and graded student answers for every assessment. In the first stage, they used a classifier of student answers coupled with a classifier of similarity with respect to model answers. In the second stage, they used a canonical correlation analysis based on transfer learning to build the classifier ensemble for questions having no labeled data.

Riordan *et al.* [7] have carried out a series of experiments across several short answer scoring datasets. They took as a reference the architecture of the neural network used by Taghipour and Ng [15]. This neural network provided good performances on automated essay scoring. The network leveraged a convolutional neural networks (CNN) architecture with regression and a simple LSTM. Zhang *et al.* [16] addressed the grading of open-ended questions. These questions do not usually have a limited number of reference answers. Students can express opinions or personal thoughts on these questions. They have used a deep learning model that integrates both domain-general and domain-specific information. The proposed model used an LSTM to classify word sequence information. The dataset had about 16,000 sample answers related to seven reading comprehension questions.

Other researchers have used transformer and transfer learning in their systems to train models. Camus and Filighera [17] have fine-tuned existing and already trained transformer based architectures. They have explored the transfer learning from one dataset to another one and its impact on generalization and performance. Condor [18] has used BERT as a tool to assist instructors with ASAG. Condor targeted situations where final human judgment is considered necessary.

2.1. ASAG datasets

There is a variety of datasets already listed in the literature to train and test ASAG models. As we know, well-structured datasets lead to good results. The Hewlett Foundation [19] has released a dataset called ASAP to train and benchmark ASAG systems. ASAP is currently available on Kaggle. This dataset contains about 10,686 samples belonging to 8 different sets of essays. Each essay has an average of 150 to 550 words response. Each essay is followed by one or more scores given by human graders. The objective is to match the “expert human graders for each essay.”

Some researchers have used datasets collected during university courses. The dataset used by Mohler and Mihalcea [20] at the University of Texas consists of 80 questions collected from a computer science course named Data Structures. The questions were used in multiple assignments and two examinations. They have collected answers through an online learning system. The size of the dataset is 80 questions, 2,273 responses provided by 31 students and 2 human expert graders. Menini *et al.* [21] released a dataset named Statistics to test short answer grading. They have built the dataset from statistics exams.

2.2. ASAG in Arabic

Gomaa and Fahmy [22] have pioneered ASAG for Arabic. They have collected a dataset in Arabic that has 61 questions. Each question had about ten student answers, and all answers were labeled with grades. They have built the dataset from the Environmental Science course of the Egyptian curriculum. For grading, they have used a text similarity-based grading that measures the similarity between student answers and reference answers. They have not used any automatic grading via a machine learning approach.

Nael *et al.* [23] researched a deep learning-based system to score short answers in Arabic and achieved good performances. However, they have not used a dataset built out of questions and answers written initially in Arabic. They have used a translated version to Arabic of an English test dataset called ASAP short answer scoring.

As we wanted to explore automated grading for schoolchildren in Arabic, we wanted to know how such data would look alike in reality. Our objective motivated us to collect and build our own dataset following the best practices already mentioned in the literature. Because data is key to machine learning algorithms, we targeted collecting real and genuine data from schoolchildren in Arabic to create good models that can handle our problems well.

3. OUR RESEARCH METHOD

All the datasets cited in the literature were collected either manually through forms or dedicated web applications [24], [25] or using an automated mechanism like web scraping. We have built our dataset manually from answers provided by schoolchildren aged between 11 to 12 years old and graded by a teacher. Figure 1 illustrates an excerpt of the dataset. We have used Google Forms to collect the answers. All participants were studying in the sixth grade of primary education in Morocco. The schoolchildren answered 18 questions related to the Islamic education course. We have collected 1,276 answers. A teacher has evaluated and graded all answers. The grades were between 0 and 2: 0 for completely incorrect, 1 for partially correct, and 2 for correct.

	id_question	stu_answer	grade
0	18	ز	0
1	18	جبرائيل	1
2	18	جبريل	2
3	18	جبريل عليه السلام	2
4	18	سيدنا جبريل عليه السلام	2
...
1255	1	الجنة	2
1256	1	جزاء الجنة و رضى الله	2
1257	1	..كما قال الرسول صلى الله عليه وسلم : " صبرا آل	2
1258	1	جزاء الجنة	2
1259	1	الجنة	2

Figure 1. Islamic Education short answer dataset (answers 1 to 4 read Gabriel, Gabriel, Gabriel peace be upon him, our master Gabriel peace be upon him, respectively)

Schoolchildren have answered these questions at home on a computer or on mobile. We have noticed that 75% of the answers have 8 words or less. Figure 2 shows the number of answers for a given number of words. Table 1 statistical indicators on the length of answers provide some statistical indicators

related to the number of characters and the number of words in the dataset. Compared to other datasets collected in university or from adult responses, the answers that we have collected have fewer words. We also looked at the data from the score perspective to ensure that all scores were present. We found out that it is important to ensure that all scores are present in the dataset to help machine learning algorithms work correctly. Table 2 provides the number of answers per score.

Table 1. Statistical indicators on the length of answers

	Number of characters	Number of words
Mean value	29.0	5.6
Standard deviations	32.4	6.2
Quartile 1	6	1
Quartile 2	18	3
Quartile 3	40	8
Minimal value	0	0
Maximal value	311	55

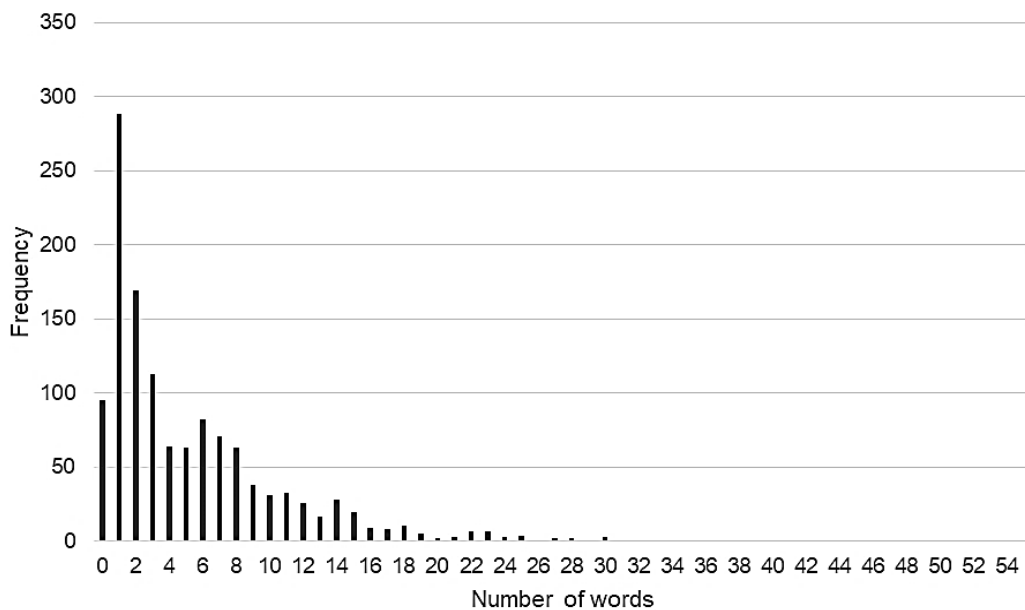


Figure 2. Number of answers per number of words

Table 2. Number of answers per score

Scores	Number of answers
0	27%
1	20%
2	52%

3.1. NLP pipeline for ASAG

For Arabic NLP, researchers are currently using pipelines and architectures similar to what is being used for English NLP [26]. On the other hand, the NLP pipelines used for ASAG are similar to the generic pipelines used for other NLP applications. For this research, we have used a pipeline similar to the ones used for English ASAG when leveraging NLP and machine learning. The adopted NLP pipeline was composed of three main stages as illustrated in Figure 3. The classification or grading happens in the third stage. We have used two stages upfront to preprocess the text and transform words into numerical vectors before classification can be applied.

The first stage is text processing and includes tasks like segmentation, tokenization, stop word removal, and stemming or lemmatization. The output of this first stage is the list of important words composing the initial answer but reduced to their root words. Stemming and lemmatization are both used in NLP to normalize words by reducing each word to its root or dictionary form. Stemming algorithms chop

off suffixes and are fast, but they may reduce words to wrong roots or non-existing words. Lemmatization algorithms apply a contextual analysis to words and link them on average to more appropriate root words. However, if the text is long, then lemmatization takes considerably more time.

The second stage called feature extraction or embedding is where we map each word with a numerical vector belonging to a relatively low-dimensional continuous space, called embedding space. An important requirement for this mapping is that words sharing similar meanings or semantics should translate in the embedding space to numerical vectors that are close to each other [26]. Each dimension of the embedding space is usually linked to some semantic features of our vocabulary. Table 3 shows the example of embedding vectors associated with six different words and provides an example of six words projected on an embedding space of three dimensions where each dimension is associated with a pure semantic feature, namely {Person; Location; Duration}.

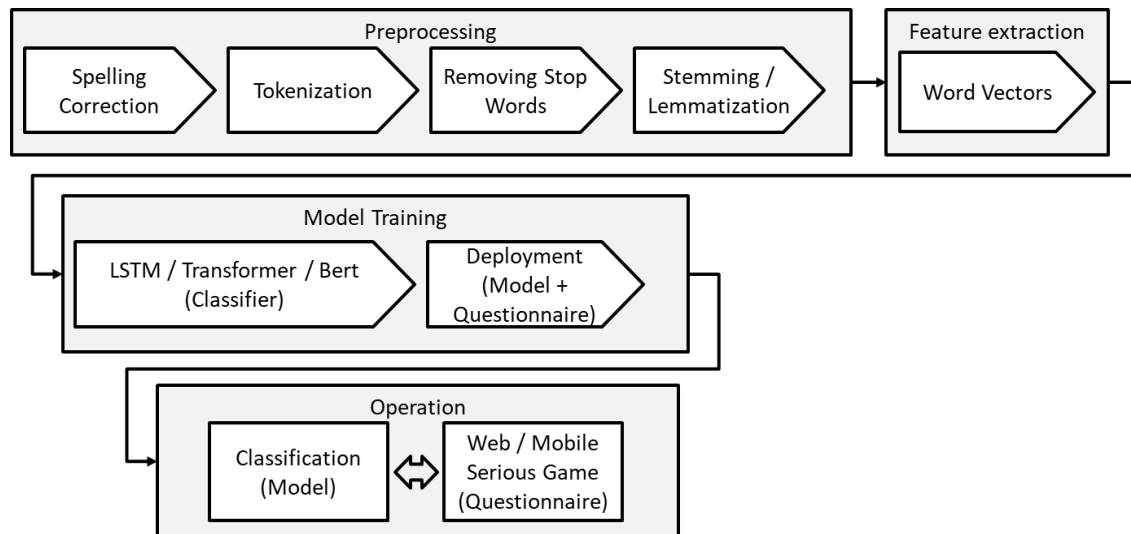


Figure 3. NLP pipeline architecture for ASAG

Table 3. Example of embedding vectors associated with six different words

		Vocabulary					
		Prophet	Messenger	Mecca	Year	Medina	Preach
Embedding dimensions	Person	0.97	0.95	0.01	0.06	0.07	0.26
	Location	0.12	0.19	0.76	0.23	0.72	0.11
	Duration	0.36	0.43	0.23	0.98	0.12	0.24

Words used in similar contexts usually have similar meanings or semantics, thus these words must be close to each other along some dimensions of the embedding space. Different techniques are used for feature extraction. In this research, we have used word2vec [27] to generate the embedding vectors. The word2vec algorithm leverages machine-learning techniques and is key to NLP. The dimension of the embedding space was 300. This means that all the words of the Arabic corpus that we have used were projected on vectors of dimension 300.

The third stage performs the classification task. This stage leverages deep learning algorithms and implements our machine learning models. We have first trained these models on our data. We then tested them on unseen answers. For both training and testing, we have fed these models with data that went through the two first stages of our pipeline.

For both LSTM and transformer models, we have used the Gensim toolkit during lemmatization, tokenization, and word embedding [28]. Gensim addresses many common NLP tasks and provides an implementation of the word2vec algorithm. We usually train the word2vec algorithm on a corpus and associated texts to generate a word vector encoding how to map each word from the corpus on a numerical low-dimension vector. In our research, we have used word2vec with “Wiki.ar.vec” as the pre-trained word vector [26]. “Wiki.ar.vec” was trained on ar.wikipedia. For the BERT model, we have performed the tokenization task using a pre-trained model called “Bashar-talafha/multi-dialect-bert-base-arabic” [29].

3.2. Deep learning architectures used for ASAG

Our approach to Arabic ASAG was to test and adapt the models used for English ASAG. The first unknown was the quality of the word vectors. A second one was how the models would behave on answers written in simple words by schoolchildren. We have tested an LSTM architecture [30], a transformers-based architecture [31], and a transfer learning by fine-tuning a BERT pre-trained model [32]. This section presents the results.

3.3. LSTM model

The architecture based on the LSTM model is composed of 7 layers and is described in Figure 4. The input layer has 54 nodes because the longest response in our system can have 54 words. LSTM is a recurrent network and will iterate on 54 words. The embedding layer has 300 nodes, 300 being the length of the vector after word2vec encoding. The LSTM layer has 64 units, followed by two dropout layers with 64 and 32 nodes, followed by one flattened layer with 3456 nodes. As we have three possible final grades {0, 1, 2}, the output layer has 3 nodes to provide the result of our classification task. In total, the trainable parameters of the LSTM model are around 204,163 parameters.

Layer (type)	Output Shape	Param #
input_5 (InputLayer)	[None, 54]	0
embedding_4 (Embedding)	(None, 54, 300)	383700
lstm_4 (LSTM)	(None, 54, 64)	93440
dropout_8 (Dropout)	(None, 54, 64)	0
flatten_4 (Flatten)	(None, 3456)	0
dense_8 (Dense)	(None, 32)	110624
dropout_9 (Dropout)	(None, 32)	0
dense_9 (Dense)	(None, 3)	99
Total params: 587,863		
Trainable params: 204,163		
Non-trainable params: 383,700		

Figure 4. LSTM layer design for ASAG

For the ASAG problem, we found that the hyper-parameters used with LSTM have an impact on learning and test results. We have tested different hyper-parameters in Table 4. The Hyper-parameter of LSTM Architecture for ASAG. 1 is the best hyper-parameter found to train the LSTM model for Arabic ASAG. We present the performances achieved with these parameters in section 6.

Table 4. Hyper-parameter of LSTM architecture for ASAG

Batch size	Learning Rate	Beta 1/2	Epochs	Optimizer	Regularization
256	0.001	0.9	150	Adam	Dropout early stopping

3.4. Transformer model

The architecture based on transformer model is composed of 6 layers as shown in Figure 5. The input layer has 54 nodes, 54 being the length of the longest response of our dataset. Follows a token position embedding layer with 300 nodes where 300 is the size of the computed embedding vectors. The transformer layer also has 300 nodes, followed by a max-pooling layer for dimensionality reduction, then one dropout layer with 300 nodes and finally a 3 nodes output layer to perform the classification task. The trainable parameters of the transformer model are about 768,311 parameters.

For transformer, we have also tested several hyper-parameters and compared results. Table 5 hyper-parameter of transformer architecture for ASAG presents the hyper-parameters that provided the best performances. We discuss the performances achieved in section 6.

```

Model: "model_Transformer"
-----
Layer (type)                Output Shape                Param #
-----
input_14 (InputLayer)       [(None, 54)]                0
token_and_position_embedding (None, 54, 300)      399900
transformer_block_13 (Transf (None, 54, 300)    367508
global_max_pooling1d (Global (None, 300)        0
dropout_53 (Dropout)        (None, 300)                0
dense_53 (Dense)            (None, 3)                   903
-----
Total params: 768,311
Trainable params: 768,311
Non-trainable params: 0

```

Figure 5. Transformer layer design for ASAG

Table 5. Hyper-parameter of transformer architecture for ASAG

Batch size	Learning Rate	Beta 1/2	Epochs	Optimizer	Regularization
256	0.001	0.9	25	Adam	Dropout early stopping

3.5. BERT model

The last tested architecture is based on BERT Model. We have used an architecture composed of 6 layers, see Figure 6. The input layer has 309 nodes. The BERT layer has 110,617,344 parameters, followed by two dense layers with 64 and 32 nodes and two dropout layers with 64 and 32 nodes. The output layer has 3 nodes to perform the classification task. Overall, the trainable parameters of the BERT model are about 51,395 parameters.

```

Model: "model_BERT"
-----
Layer (type)                Output Shape                Param #    Connected to
-----
input_9 (InputLayer)       [(None, 309)]              0
input_10 (InputLayer)      [(None, 309)]              0
tf_bert_model_2 (TFBertModel) TFBertModelOutputWit 110617344  input_9[0][0]
                                                                    input_10[0][0]
dense_8 (Dense)            (None, 64)                  49216
tf_bert_model_2[0][1]
dropout_115 (Dropout)      (None, 64)                  0          dense_8[0][0]
dense_9 (Dense)            (None, 32)                  2080
dropout_115[0][0]
dropout_116 (Dropout)      (None, 32)                  0          dense_9[0][0]
dense_10 (Dense)           (None, 3)                   99         dropout_116[0][0]
-----
Total params: 110,668,739
Trainable params: 51,395
Non-trainable params: 110,617,344

```

Figure 6. BERT layer design for ASAG

As for LSTM and transformer, the performance of a BERT model on ASAG varies with the parameters used. We have tested and experimented with different parameters before finding a set of parameters that provided good performances on our ASAG problem. We have listed these parameters in Table 6. We present and discuss the performances achieved with these parameters in section 4.

Table 6. Hyper-parameter of BERT architecture for ASAG

Batch size	Learning rate	Beta 1/2	Epochs	Optimizer	Regularization
256	0.001	0.9	100	Adam	Dropout early stopping

3.6. Deployment to an operating environment

We wanted to test the deployment and the behavior of the grading service in an operating environment. In operations, short answers need to go through text processing and feature extraction before classification. We have embedded the 3 stages of our NLP pipeline in our deployment server, as shown in Figure 7.

To execute the machine learning models in our operating environment, we have installed TensorFlow [33] on our server and used it as an inference engine to execute our classification models. We have deployed our models from Colab and Kaggle after training and tuning to TensorFlow in a ‘H5’ container [34]. We wanted to make the automatic grading service available to multiple front ends. We made this service available through a service-oriented architecture (REST API). We consumed this service through a web and a mobile application. The service-oriented architecture has proved flexible to deploy and operate the trained models.

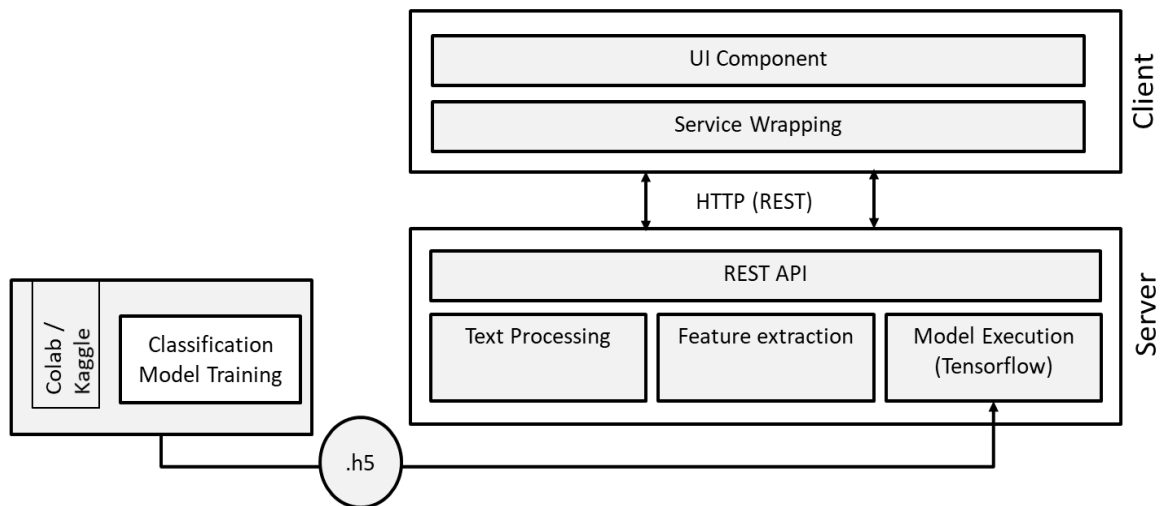


Figure 7. Web application architecture

4. RESULTS AND DISCUSSION

To evaluate the performance of each model, we have used four evaluation metrics: accuracy, precision, recall, and Cohen kappa. The values of these metrics for the LSTM model are listed in Table 7, while Figure 8 plots the metrics against the number of epochs. For the transformer model, Table 8 provides the metrics, and Figure 9 plots the metrics against epochs. For the BERT model, Table 9 and Figure 10 provide the values of the metrics and the graph plotting the metrics against epochs.

Table 7. LSTM model for ASAG metrics results

	Accuracy	Precision	Recall	Cohen Kappa	Loss
Training	83.95%	86.34%	78.78%	71.11%	0.4305
Test	69.62%	73.50%	62.03%	45.39%	0.727

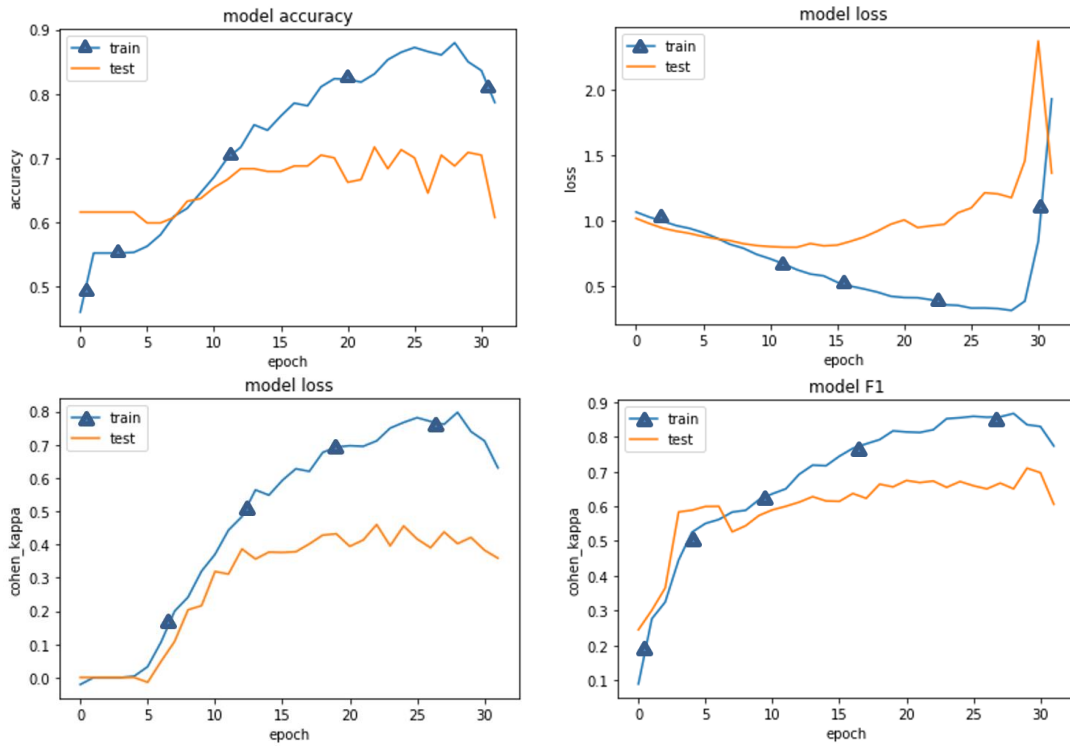


Figure 8. LSTM model for ASAG metrics' graphs

Table 8. Transformer model for ASAG metrics results

	Accuracy	Precision	Recall	Cohen Kappa	Loss
Training	95.67%	95.67%	95.67%	92.49%	0.090
Test	77.22%	77.35%	76.37%	59.96%	0.9340

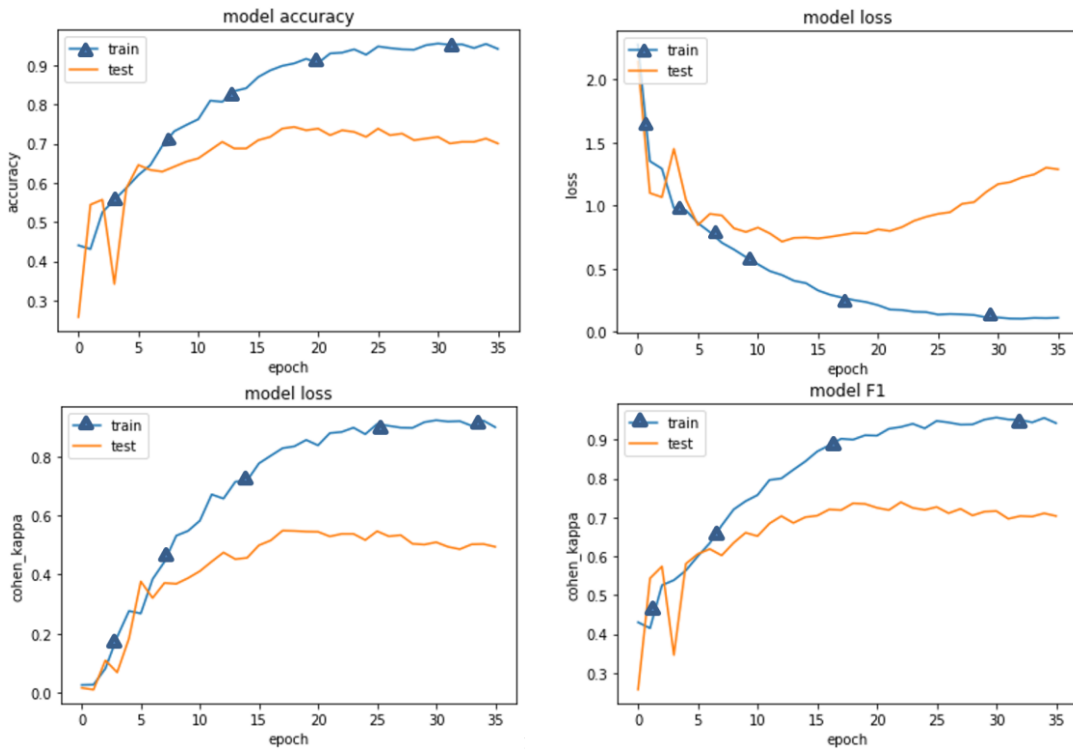


Figure 9. Transformer model for ASAG metrics' graphs

Table 1. BERT model for ASAG metrics results

	Accuracy	Precision	Recall	Cohen Kappa	Loss
Training	85%	88.68%	78.56%	73.84%	0.4157
Test	71.31%	77.08%	66.67%	47.61%	0.4761

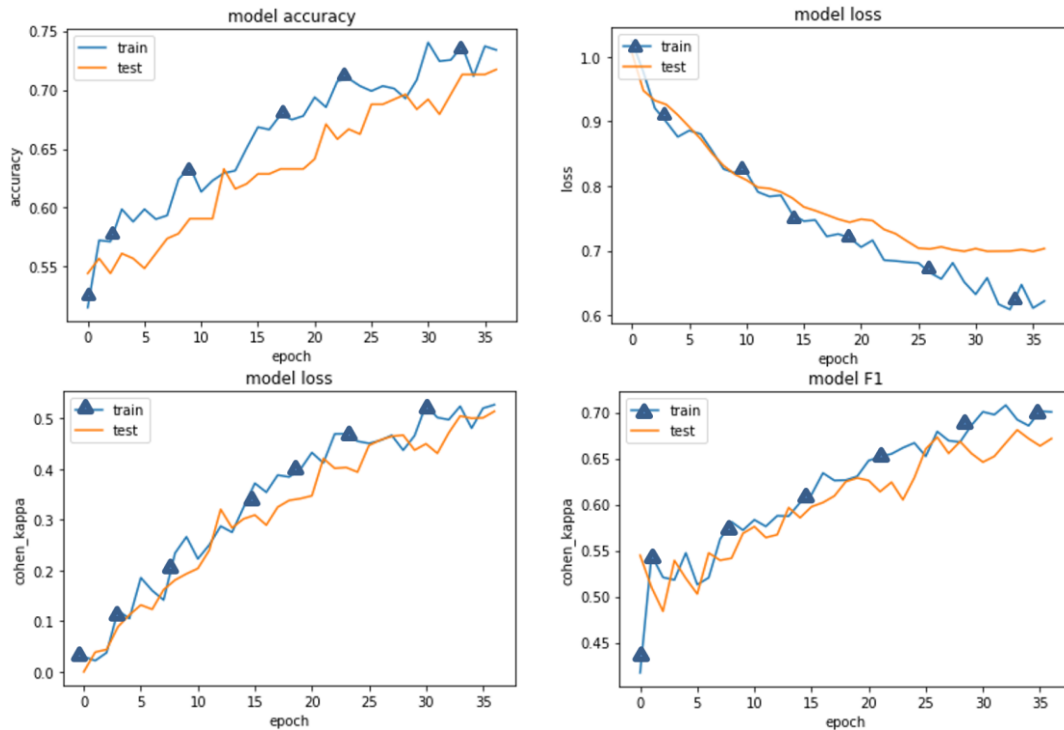


Figure 10. BERT model for ASAG: metrics' graphs

4.1. Discussion

From the results of the model evaluation section, we notice that the transformers model has the best accuracy with 95.67%, followed by the BERT model with an accuracy of 88.68%, and then LSTM with an accuracy of 83.95%. The same remark applies to the other metrics like precision, recall, and Cohen kappa. When we see the metrics graphs of each proposed model, we notice that the transformer model overfits faster compared to both LSTM and BERT. The difference between the training curve and the test curve becomes larger as the epochs increase. This is due to the complexity of the model architecture and the size of the used dataset. In order to deal with this problem, we have used two technics usually used for this kind of problem. The first one consisted of using the dropout layers to reduce the complexity of the model. The second one consisted of using an early stopping technique.

We have compared our models with other models from the literature as illustrated in Table 10. As for accurate results on ASAG, we can conclude that we have followed a good approach and achieved results comparable to the other research. We can improve our results by varying some hyper-parameters like architecture or regularization techniques, by adding more data to the dataset, or by using dedicated lemmatization and word embedding.

Table 10. Accuracy results on ASAG

Model	Accuracy
ASAG based Bi-LSTM, Conneau <i>et al.</i> [35]	76 %
ASAG based LSTM, Saha <i>et al.</i> [6]	79.26 %
ASAG based LSTM, Liu <i>et al.</i> [36]	88.9 %
Proposed ASAG based LSTM	83.95%
ASAG based transformers, Wang <i>et al.</i> [37]	80.17 %
ASAG based transformers, Camus and Filighera [17]	79.7 %
Proposed ASAG based transformers	95.67%
Proposed ASAG based BERT	85%

5. CONCLUSION

We have shown in this research that we can set up an ASAG system for schoolchildren and for Arabic. ASAG systems in Arabic can leverage and adapt natural language processing pipelines and deep learning architectures used for English ASAG. We have used a lemmatization adapted to Arabic to transform words into their dictionary forms. As deep learning algorithms require us to map or embed Arabic words into low-dimension numerical vectors, we have used the open-source word2vec algorithm trained on Arabic Wikipedia to compute these numerical embedding vectors. Our ASAG system targeted schoolchildren from fifth and sixth grades aged 11 and 12 years old. The answers given by these schoolchildren turned out to be short and composed of 5.6 words on average. The collected dataset proved large enough to train our model and to provide good results. We have also made this dataset public and available for future research projects. Moreover, service-oriented architecture proved beneficial to deploying our models to production in an environment providing ASAG services. We were able to consume the ASAG service via different front ends.

We have used a word-embedding algorithm trained on Arabic Wikipedia. As the style and expressions used by schoolchildren are different from what we can find in Wikipedia, one can explore training a word-embedding algorithm on a corpus made out of school textbooks in Arabic. We can also improve our ASAG system by adding correction capabilities. This means that the system will correct wrong or ambiguous answers and propose to schoolchildren how to improve their responses. Finally, Our ASAG system was trained on one chapter of the curriculum of the fifth grade. One can extend this to cover all chapters of the fifth grade or of primary education. Such extension will make a continuous evaluation of learning in primary schools easy and will help teachers detect early schoolchildren with learning problems. Once detected, teachers can help these schoolchildren overcome their difficulties. With the recent emergence of large language models, developing ASAG systems to cover full primary curricula and for continuous evaluation of learning seems a promising direction.

DATA AVAILABILITY

Dataset: <https://github.com/FSTT-LIST/GLUPS-ASAG-Dataset>

Models: <https://www.kaggle.com/code/lotfielaachak/asag-lstm>

<https://www.kaggle.com/code/lotfielaachak/asag-transformer>

<https://www.kaggle.com/code/lotfielaachak/asag-bert>





REFERENCES

- [1] K. Zierer and N. M. Seel, "General didactics and instructional design: eyes like twins a transatlantic dialogue about similarities and differences, about the past and the future of two sciences of learning and teaching," *SpringerPlus*, vol. 1, no. 1, Dec. 2012, doi: 10.1186/2193-1801-1-15.
- [2] W. Dick, L. Carey, and J. O. Carey, "The systematic design of instruction," Sixth Edition, Allyn & Bacon, pp. 1-400, 2005.
- [3] Y. Alaoui, L. El Achaak, A. Belahbib, and M. Bouhorma, "Serious games for sustainable education in emerging countries: An open-source pipeline and methodology," in *Emerging Trends in ICT for Sustainable Development: The Proceedings of NICE2020 International Conference*, Springer International Publishing, 2021, pp. 399–407, doi: 10.1007/978-3-030-53440-0_42.
- [4] F. Bellotti, B. Kapralos, K. Lee, P. Moreno-Ger, and R. Berta, "Assessment in and of serious games: an overview," *Advances in Human-Computer Interaction*, pp. 1–11, 2013, doi: 10.1155/2013/136864.
- [5] S. Burrows, I. Gurevych, and B. Stein, "The eras and trends of automatic short answer grading," *International Journal of Artificial Intelligence in Education*, vol. 25, no. 1, pp. 60–117, Mar. 2015, doi: 10.1007/s40593-014-0026-8.
- [6] S. Saha, T. I. Dhamecha, S. Marvaniya, R. Sindhgatta, and B. Sengupta, "Sentence level or token level features for automatic short answer grading?: Use both," in *Lecture Notes in Computer Science*, Springer International Publishing, 2018, pp. 503–517, doi: 10.1007/978-3-319-93843-1_37.
- [7] B. Riordan, A. Horbach, A. Cahill, T. Zesch, and C. M. Lee, "Investigating neural architectures for short answer scoring," in *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, 2017, pp. 159–168, doi: 10.18653/v1/W17-5017.
- [8] P. Willett, "Recent trends in hierarchic document clustering: A critical review," *Information Processing and Management*, vol. 24, no. 5, pp. 577–597, Jan. 1988, doi: 10.1016/0306-4573(88)90027-1.
- [9] J. P. Callan, "Passage-level evidence in document retrieval," in *SIGIR '94*, London: Springer London, 1994, pp. 302–310, doi: 10.1007/978-1-4471-2099-5_31.
- [10] S. Kumar, S. Chakrabarti, and S. Roy, "Earth mover's distance pooling over Siamese LSTMs for automatic short answer grading," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, Aug. 2017, pp. 2046–2052, doi: 10.24963/ijcai.2017/284.
- [11] A. Prabhudesai and T. N. B. Duong, "Automatic short answer grading using Siamese bidirectional LSTM based regression," in *2019 IEEE International Conference on Engineering, Technology and Education (TALE)*, Dec. 2019, pp. 1–6, doi: 10.1109/TALE48000.2019.9226026.
- [12] L. Xia, M. Guan, J. Liu, X. Cao, and D. Luo, "Attention-based bidirectional long short-term memory neural network for short answer scoring," in *Machine Learning and Intelligent Communications*, Springer International Publishing, 2021, pp. 104–112, doi: 10.1007/978-3-030-66785-6_12.




- [13] D. Alikaniotis, H. Yannakoudakis, and M. Rei, "Automatic text scoring using neural networks," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016, pp. 715–725, doi: 10.18653/v1/P16-1068.
- [14] S. Roy, H. S. Bhatt, and Y. Narahari, "An iterative transfer learning based ensemble technique for automatic short answer grading," *arXiv preprint arXiv:1609.04909*, Sep. 2016.
- [15] K. Taghipour and H. T. Ng, "A neural approach to automated essay scoring," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016, pp. 1882–1891, doi: 10.18653/v1/D16-1193.
- [16] L. Zhang, Y. Huang, X. Yang, S. Yu, and F. Zhuang, "An automatic short-answer grading model for semi-open-ended questions," *Interactive Learning Environments*, vol. 30, no. 1, pp. 177–190, Jan. 2022, doi: 10.1080/10494820.2019.1648300.
- [17] L. Camus and A. Filighera, "Investigating transformers for automatic short answer grading," in *Lecture Notes in Computer Science*, Springer International Publishing, 2020, pp. 43–48, doi: 10.1007/978-3-030-52240-7_8.
- [18] A. Condor, "Exploring automatic short answer grading as a tool to assist in human rating," in *Lecture Notes in Computer Science*, Springer International Publishing, 2020, pp. 74–79, doi: 10.1007/978-3-030-52240-7_14.
- [19] ASAP, "The Hewlett foundation: automated essay scoring," *Kaggle*. <https://www.kaggle.com/competitions/asap-aes> (accessed Dec. 14, 2022).
- [20] M. Mohler and R. Mihalcea, "Text-to-text semantic similarity for automatic short answer grading," in *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics on-EACL '09*, 2009, pp. 567–575, doi: 10.3115/1609067.1609130.
- [21] S. Menini, S. Tonelli, G. De Gasperis, and P. Vittorini, "Automated short answer grading: a simple solution for a difficult task," 2019.
- [22] W. H. Gomaa and A. A. Fahmy, "Automatic scoring for answers to Arabic test questions," *Computer Speech and Language*, vol. 28, no. 4, pp. 833–857, Jul. 2014, doi: 10.1016/j.csl.2013.10.005.
- [23] O. Nael, Y. ELmanyalawy, and N. Sharaf, "AraScore: A deep learning-based system for Arabic short answer scoring," *Array*, vol. 13, Mar. 2022, doi: 10.1016/j.array.2021.100109.
- [24] S. Basu, C. Jacobs, and L. Vanderwende, "Powergrading: A clustering approach to amplify human effort for short answer grading," *Transactions of the Association for Computational Linguistics*, vol. 1, pp. 391–402, Dec. 2013, doi: 10.1162/tacl_a_00236.
- [25] R. D. Nielsen, W. H. Ward, J. H. Martin, and M. Palmer, "Annotating students' understanding of science concepts," in *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, 2008.
- [26] A. B. Soliman, K. Eissa, and S. R. El-Beltagy, "AraVec: A set of Arabic word embedding models for use in Arabic NLP," *Procedia Computer Science*, vol. 117, pp. 256–265, 2017, doi: 10.1016/j.procs.2017.10.117.
- [27] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, Jan. 2013.
- [28] R. Rehurek and P. Sojka, "Gensim-python framework for vector space modelling," *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, vol. 3, no. 2, 2011.
- [29] B. Talafha *et al.*, "Multi-dialect Arabic BERT for country-level dialect identification," *arXiv preprint arXiv:2007.05612*, Jul. 2020.
- [30] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.
- [31] A. Vaswani *et al.*, "Attention is all you need," *Advances in neural information processing systems*, Jun. 2017.
- [32] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *arXiv:1810.04805*, Oct. 2018.
- [33] M. Abadi *et al.*, "TensorFlow: A system for large-scale machine learning," in *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, May 2016, pp. 265–283.
- [34] HDF Group, "HDF5, hierarchical data format, version 5," *HDF Group*, 2022. Accessed: Jun. 11, 2023. [Online], Available: <https://www.loc.gov/preservation/digital/formats/fdd/fdd000229.shtml>
- [35] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes, "Supervised learning of universal sentence representations from natural language inference data," *arXiv preprint arXiv:1705.02364*, May 2017.
- [36] T. Liu, W. Ding, Z. Wang, J. Tang, G. Y. Huang, and Z. Liu, "Automatic short answer grading via multiway attention networks," *Artificial Intelligence in Education: 20th International Conference, AIED 2019, Chicago, IL, USA, June 25-29, 2019, Proceedings, Part II 20*, pp. 169–173, Sep. 2019.
- [37] Z. Wang, A. S. Lan, A. E. Waters, P. Grimaldi, and R. G. Baraniuk, "A meta-learning augmented bidirectional transformer model for automatic short answer grading," *EDM*, 2019.

BIOGRAPHIES OF AUTHORS






Younes Alaoui Soulimani     received a master's degree from Ecole Polytechnique Paris, France, in 1989. He is a seasoned engineer in information technology. He is a Ph.D. student at the University of Abdelmalek Essaadi, Tangier Morocco in the field of serious games, machine learning, and NLP for education. He can be contacted at email: younes.alaoui@amana.ac.ma.



Lotfi El Achaak    is an assistant professor and doctor at the Faculty of Sciences and Technologies, University Abdelmalek Essaadi, Tanger. His recent research and policy interests concentrate broadly on the areas of serious games, augmented reality, reinforcement learning, machine learning or deep learning, and NLP for education. He can be contacted at email: lelaachak@uae.ac.ma.



Mohammed Bouhorma    is an experienced academic who has more than 25 years of teaching and tutoring experience in the area of Information Technology at Abdelmalek Essaadi University. He received his M.S. and Ph.D. degrees in Electronics and Telecommunications from INPT in France. He has held a visiting professor position at many universities (France, Spain, Egypt, and Saudi Arabia). His research interests include IoT, big data analytics, AI, smart cities technology, and serious games. He can be contacted at email: mbouhorma@uae.ac.ma.