Université Abdelmalek Essaadi
Faculté des Sciences et Techniques de Tanger
Département Génie Informatique

Module : Deep Learning

End of Module Project: Arabic Automated short answers grading system and Smart Assistance for islamic education

Encadré par :

Prof.Lotfi ELAACHAK

Réalisé par :

EL OUATILY Ouafae

DARKAOUI Mohamed

I. Introduction:

Arabic Automated Short Answers Grading System and Smart Assistance for Islamic Education, is all about making school learning easier. We want to create a special system that gives grades to students' short answers automatically. The best part is, it works in Arabic, and students get the total score from 0 to 20.

To build this system, we are gathering information from different places like Arabic websites using a clever technique called scraping. This helps us have a lot of different examples for the computer to learn from.

But it's not just about grades. We also want to help students in a smart way. So, as they answer questions, the system will give them personalized hints to help them do better.

The primary objective is to create an automated grading system operating in the Arabic language, assigning a grade: **0**, **1** or **2** for each question based on students' answer. And beside grading, the system endeavors to provide intelligent assistance to students by generating customized guidelines during the evaluation process, designed to align with their individual progression. This project represents a significant step toward enhancing the efficiency and effectiveness of educational evaluation and support in the context of Arabic language processing.

This project represents a significant step toward enhancing the efficiency and effectiveness of educational evaluation and support in the context of Arabic language processing. We're excited about making education more effective and supportive in the Arabic language.

II. Methodology and Technologies:

1. Data Collection:

we have leveraged web scraping techniques to collect data from various sources, including Arabic websites and Arabic datasets, where we have collected approximately 200KB, and manually add some meaningful answers to ultimately obtain homogeneous data, we have associated each answer with a grade that evaluate its quality (**True**: **2**, **Almost True**: **1**, **Wrong Answer**: **0**).
After eliminating missing data and duplicates, the resulting dataset is as follows:

	id_question	answer	grade
0	1	كان عمره ست سنوات و أمه دفنت في الأبواء	2
1	1	كان عمره ست سنوات	1
2	1	سنوات	0
3	1	...توفيت و الرسول مُحمَّد ابن ست سنوات و دُفنت في	2
4	1	...في سن الست سنوات و توفيت والدته و دُفِنَت في ا	2
...	...	...	...
1440	10	يعمل في رعاية الأغنام و التجارة	2
1441	10	كان النبي تاجر و يعمل في رعاية الأغنام	2
1442	10	.كانت إحدى أنشطة النبي المشاركة في التجارة	1
1443	10	كان النبي يعمل في الحداده	0
1444	10	كان النبي يعمل في الحداده و التجارة	1

1445 rows × 3 columns

➔ **Tools:** Scrapy, BeautifulSoup, Pandas

2. Arabic Natural Language Processing pipeline:

2.1 Remove punctuation:

Unlike in sentiment analysis, punctuation is not crucial for our task. Therefore, we have chosen to remove it. Our focus is on extracting the meaning from the answers to evaluate their quality and assign a score based on that meaning.

2.2 Tokenization:

We utilized NLTK for tokenization to process our answers systematically. Tokenization allows us to transform each answer into a vector of words, breaking it down into individual units.

Tokens
[كان, عمره, ست, سنوات, و, أمه, دفنت, في, الأبواء]
[كان, عمره, ست, سنوات]
...,توفيت, و, الرسول, مُحمَّد, ابن, ست, سنوات, و]
...,في, سن, الست, سنوات, و, توفيت, والدته, و, دُف]
... ,في, عامه, السادس, توفيت, والدته, و, وُضِعت]

2.3 Remove stop words:

We used an Arabic.txt file that contains stop words for Arabic language and we removed stop words from the tokens.

filtered_tokens
[عمره, ست, سنوات, أمه, دفنت, الأبواء]
[عمره, ست, سنوات]
[توفيت, الرسول, مُحمَّد, ابن, ست, سنوات, دُفنت...
[سن, الست, سنوات, توفيت, والدته, دُفِنَت, الأب...
[عامه, السادس, توفيت, والدته, وُضِعت, الأبواء]

2.4 Word2Vec:

We chose Word2Vec as a robust tool in our arabic answers processing approach for its capacity to represent words in a continuous vector space. This popular word embedding technique captures semantic relationships by mapping words to high-dimensional vectors. Overall, our use of Word2Vec aligns with our goal of enriching word representations, facilitating a contextually aware and semantically meaningful analysis.
This step transforms each tokenized sentence to a fixed-size vector

➔ **Tools**: NLTK, Word2Vec, Pandas

3. Model Training:

3.1 RNN:

The Recurrent Neural Network model underwent training over ten epochs, aiming to learn patterns in sequential data. Throughout the training, the model's performance was evaluated, with validation loss and accuracy tracked after each epoch. Despite the training effort, the model demonstrated consistent validation accuracy at approximately 40.74%. This may suggest that the model faced challenges in capturing intricate patterns within the data, leading to limited improvement in accuracy. The overall results indicate a need for further exploration or consideration of alternative architectures to enhance the model's ability to capture and understand the sequential patterns in the given dataset, and that’s why we look forward another solution.

```
Epoch [1/10], Training Loss: 1.0713952779769897, Validation Loss: 1.0892646312713623, Validation Accuracy: 0.40740740299224854
Epoch [2/10], Training Loss: 1.0686914920806885, Validation Loss: 1.0878198146820068, Validation Accuracy: 0.40740740299224854
Epoch [3/10], Training Loss: 1.0701733827590942, Validation Loss: 1.0859315395355225, Validation Accuracy: 0.40740740299224854
Epoch [4/10], Training Loss: 1.0598347187042236, Validation Loss: 1.0921968221664429, Validation Accuracy: 0.40740740299224854
Epoch [5/10], Training Loss: 1.052068829536438, Validation Loss: 1.0908136367797852, Validation Accuracy: 0.40740740299224854
Epoch [6/10], Training Loss: 1.0458682775497437, Validation Loss: 1.0892748832702637, Validation Accuracy: 0.40740740299224854
Epoch [7/10], Training Loss: 1.0579593181610107, Validation Loss: 1.0933361053466797, Validation Accuracy: 0.40740740299224854
Epoch [8/10], Training Loss: 1.0518063306808472, Validation Loss: 1.0919899940490723, Validation Accuracy: 0.40740740299224854
Epoch [9/10], Training Loss: 1.0580165386199951, Validation Loss: 1.0939068794250488, Validation Accuracy: 0.40740740299224854
Epoch [10/10], Training Loss: 1.0615689754486084, Validation Loss: 1.0934714078903198, Validation Accuracy: 0.40740740299224854
1/1 [==============================] - 0s 336ms/step
Accuracy: 0.4074
F1 Score: 0.2359
ROC AUC Score: 0.4949
BLEU Score: 0.0000
```

3.2 LSTM:

The LSTM model exhibits a progressive improvement in both training and validation accuracy throughout the epochs, demonstrating its ability to learn and generalize patterns effectively. This suggests that the model is well-suited for the task at hand, achieving high accuracy on both the training and validation datasets. The decreasing trend in training and validation loss indicates effective learning and generalization.

```
Epoch 1/10
28/28 [==============================] - 3s 33ms/step - loss: 1.0508 - accuracy: 0.4595 - val_loss: 0.9659 - val_accuracy: 0.5000
Epoch 2/10
28/28 [==============================] - 0s 13ms/step - loss: 0.8427 - accuracy: 0.6937 - val_loss: 0.7119 - val_accuracy: 0.6071
Epoch 3/10
28/28 [==============================] - 0s 14ms/step - loss: 0.4532 - accuracy: 0.8108 - val_loss: 0.3825 - val_accuracy: 0.8571
Epoch 4/10
28/28 [==============================] - 0s 14ms/step - loss: 0.3689 - accuracy: 0.8559 - val_loss: 0.4343 - val_accuracy: 0.8214
Epoch 5/10
28/28 [==============================] - 0s 15ms/step - loss: 0.1674 - accuracy: 0.9279 - val_loss: 0.2940 - val_accuracy: 0.8571
Epoch 6/10
28/28 [==============================] - 0s 13ms/step - loss: 0.1061 - accuracy: 0.9550 - val_loss: 0.3400 - val_accuracy: 0.8571
Epoch 7/10
28/28 [==============================] - 0s 14ms/step - loss: 0.0944 - accuracy: 0.9730 - val_loss: 0.1385 - val_accuracy: 0.9286
Epoch 8/10
28/28 [==============================] - 0s 13ms/step - loss: 0.0420 - accuracy: 0.9820 - val_loss: 0.3605 - val_accuracy: 0.8571
Epoch 9/10
28/28 [==============================] - 0s 12ms/step - loss: 0.0858 - accuracy: 0.9730 - val_loss: 0.3485 - val_accuracy: 0.8571
Epoch 10/10
28/28 [==============================] - 0s 12ms/step - loss: 0.0489 - accuracy: 0.9910 - val_loss: 0.1026 - val_accuracy: 0.9643
```

3.2 Transformer:

The Transformer model demonstrates a robust learning capability, achieving an overall accuracy of 85.19% on the evaluation dataset. The classification report further reveals high precision, recall, and F1-score values. These results suggest that the Transformer model effectively captures complex patterns within the data, showcasing its proficiency in tasks requiring sequential data processing. The fluctuations in training loss values indicate the model's adaptability and continuous learning throughout the training epochs.

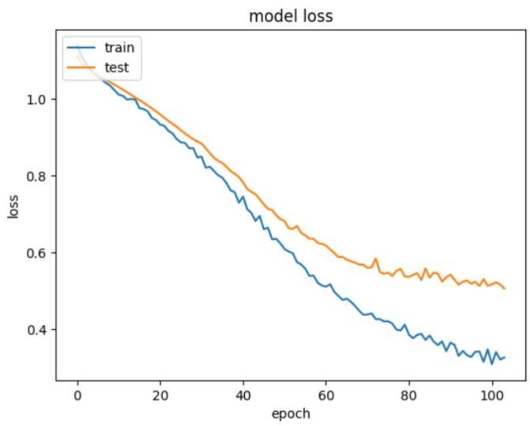
Accuracy: 85.19%
Classification Report:

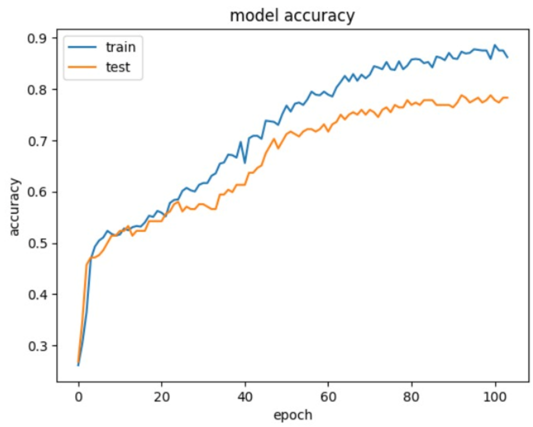
	precision	recall	f1-score	support
0	0.83	0.91	0.87	11
1	1.00	0.70	0.82	10
2	0.75	1.00	0.86	6
accuracy			0.85	27
macro avg	0.86	0.87	0.85	27
weighted avg	0.88	0.85	0.85	27

➔ **Tools:** PyTorch, Sklearn, Numpy, Seaborn, Keras, BERT

During the course of our project, a noteworthy observation emerged, revealing a potential challenge in the system's evaluation process. We identified that when a student provides a correct answer intended for one question in the response field of another question, the system erroneously assigns a favorable grade, even though the answer is unrelated to the specific question being evaluated. Recognizing this issue, the solution that we found out was incorporating the question id alongside the corresponding answer during model training. This adjustment aimed to enhance the model's contextual understanding and mitigate the risk of misattributed grades. Remarkably, implementing this approach yielded positive results, indicating an improvement in the system's accuracy.
Furthermore, in our pursuit of optimizing model performance, we explored an alternative method. We divided the dataset into subsets, each dedicated to a specific question and comprising a mix of correct, nearly correct, and incorrect answers. Training the model on each of these specialized datasets yielded favorable outcomes, showcasing enhanced performance.

And finally, we opted for the LSTM model in our application due to its consistent and superior overall performance.





4. Text generation for student assistance:

For the assistance part in our application, we utilized the **Alpaca** pre-trained model. This advanced language model played a crucial role in generating valuable guidelines and support for students during the evaluation process. By integrating Alpaca, our system benefits from its sophisticated natural language understanding, enabling it to provide contextually relevant and insightful assistance to students in Arabic, aiding them throughout the evaluation process.

➔ Prompt template:

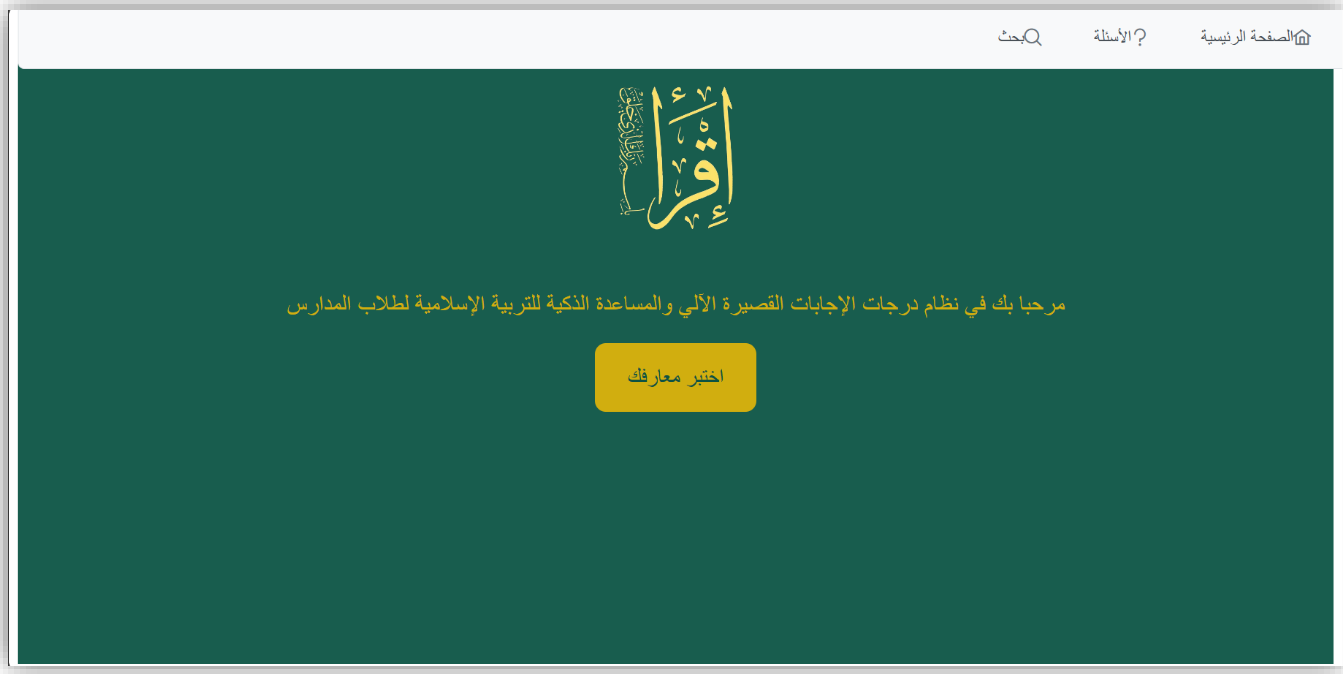
```
from langchain import PromptTemplate, LLMChain

template = """You are an adept Islamic expert assistant,
proficient in providing subtle guidance for intricate questions,
 assisting students in navigating towards the intended answers without outright disclosure.
  Below is the question the student is tasked to answer along with their response.
  Your role is to craft a response that effectively guides the student to a more comprehensive and refined answer,
  maintaining the essence of critical thinking and analytical exploration.
### Question :
{qst}
### Student answer:
{answer}

Hint:"""

prompt = PromptTemplate(template=template, input_variables=["answer","qst"])
```

5. Application and model deployment:

5.1 Application:

In developing our frontend application, we chose **Angular** for its robust features, creating a dynamic interface. We integrated **Primeng** for enhanced aesthetics.
On the backend, we used **Flask** for its simplicity, building a resilient server foundation.
This combination facilitates a seamless user experience, enriched by a well-designed API for enhanced communication and data exchange.

- **Home page:**



- **Questions page:**


الصفحة الرئيسية
الأسئلة
بحث
إقرأ
اختبار التربية الإسلامية
يرجى الإجابة على الأسئلة باللغة العربية وبطريقتك الخاصة
2. من الذي أنزل الوحي على النبي صلى الله عليه وسلم؟
الملك جبريل
السؤال التالي

- **Result page:**


الصفحة الرئيسية
الأسئلة
بحث
إقرأ
نتائج الاختبار
المجموع الكلي: 10/20
أعد الاختبار
1. كم كان عمر النبي صلى الله عليه وسلم عندما توفيت أمه وأين دفنت؟
إجابتك تكاد تكون صحيحة كان عمر النبي صلى الله عليه وسلم ست سنوات
2. من الذي أنزل الوحي على النبي صلى الله عليه وسلم؟
إجابتك صحيحة الملك جبريل
3. ما أول ما نزل من القرآن الكريم على الرسول صلى الله عليه وسلم؟
إجابتك صحيحة سورة العلق

الصفحة الرئيسية الأسئلة بحث

نتائج الاختبار

المجموع الكلي: 10/20

أعد الاختبار

4. من أوّل من آمن بالنَّبيّ وصدّقه؟

إجابتك خاطئة عائشة

5. لماذا أخلى النبي صلى الله عليه وسلم مكة؟

إجابتك تكاد تكون صحيحة الهجرة الى المدينة

6. ما اسم الصّلاة التي يصلّى فيها ركوعين وسجودين من كلّ ركعةٍ؟

إجابتك خاطئة صلاة الاستخارة

الصفحة الرئيسية الأسئلة بحث

نتائج الاختبار

المجموع الكلي: 10/20

أعد الاختبار

7. ماذا لُقِّب عثمان بن عفان رضي الله عنه؟ ولماذا؟

إجابتك تكاد تكون صحيحة لقب عثمان بن عفان ذي النورين

8. ما هي أسباب غزوة بدر؟

إجابتك صحيحة تعاون الأنصار مع النبي و خوف قريش على عاداتها وآلهتها

9. ما هي أسباب وقوع غزوة أحد؟

إجابتك خاطئة وقوع

10. ما هي مهنة النَّبيّ صلّى الله عليه وسلّم قبل البعثة؟

إجابتك تكاد تكون صحيحة يرعى الغنم

➢ Assistance:

5.2 Application deployment:

For streamlined deployment, scalability, and efficient orchestration, we employed **Docker** containers and **Kubernetes** to deploy our application. Leveraging Docker's containerization technology, we encapsulated our frontend, backend built with Flask, and any required dependencies into self-contained units. Furthermore, we utilized Kubernetes, with Minikube for local testing, to manage and orchestrate these containers.

➔ Enable Kubernetes:

➔ start a local Kubernetes cluster using Minikube:

```
PS C:\Users\ouafae> minikube start
* minikube v1.32.0 on Microsoft Windows 10 Pro 10.0.19045.3803 Build 19045.3803
* Using the docker driver based on existing profile
* Starting control plane node minikube in cluster minikube
* Pulling base image ...
* Restarting existing docker container for "minikube" ...
* Preparing Kubernetes v1.28.3 on Docker 24.0.7 ...
* Configuring bridge CNI (Container Networking Interface) ...
* Verifying Kubernetes components...
  - Using image gcr.io/k8s-minikube/storage-provisioner:v5
* Enabled addons: storage-provisioner, default-storageclass
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

Minikube now is running:

```
PS C:\Users\ouafae> minikube status -p minikube
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured
```

➔ Build Docker images:

```
PS C:\Users\ouafae> cd C:\Users\ouafae\Documents\Server-ShortAnswers
PS C:\Users\ouafae\Documents\Server-ShortAnswers> docker context use default
default
Current context is now "default"
PS C:\Users\ouafae\Documents\Server-ShortAnswers> minikube -p minikube docker-env --shell powershell | Invoke-Expression
PS C:\Users\ouafae\Documents\Server-ShortAnswers> docker build -t ouafae1004/flask .
[+] Building 54.9s (5/10)                                                                                 docker:default
 => [internal] load .dockerignore                                                                                   0.4s
 => => transferring context: 2B                                                                                     0.0s
 => [internal] load build definition from Dockerfile                                                                0.4s
 => => transferring dockerfile: 181B                                                                                0.0s
 => [internal] load metadata for docker.io/library/python:3.8                                                       4.5s
 => [auth] library/python:pull token for registry-1.docker.io                                                       0.0s
 => [1/5] FROM docker.io/library/python:3.8@sha256:3377bcab6b2a6fa83bb91d40c290b779ca76344e3d7d36a3553b338d3f128be2 49.9s
 => => resolve docker.io/library/python:3.8@sha256:3377bcab6b2a6fa83bb91d40c290b779ca76344e3d7d36a3553b338d3f128be2  0.2s
 => => sha256:3377bcab6b2a6fa83bb91d40c290b779ca76344e3d7d36a3553b338d3f128be2 1.86kB / 1.86kB                      0.0s
 => => sha256:d9cf0b2737d4a090ca50573519e5f02077788f40b790c55a9d4be899ecf874a4 2.01kB / 2.01kB                      0.0s
 => => sha256:415775fb4b1f611fd55155a8ac56082f6c599c295f5dd2b604e910ffbe3921b0 7.56kB / 7.56kB                      0.0s
 => => sha256:b5de22c0f5cd2ea2bb6c0524478db95bff5a294c99419ccd4a9d3ccc9873bad9 7.34MB / 24.05MB                    49.7s
 => => sha256:917ee5330e73737d6095a802333d311648959399ff2c067150890162e720f863 6.29MB / 64.13MB                    49.7s
 => => sha256:bc0734b949dcdcabe5bfdf0c8b9f44491e0fce04cb10c9c6e76282b9f6abdf01 13.63MB / 49.56MB                   49.7s
 => [internal] load build context                                                                                   0.5s
 => => transferring context: 13.54MB                                                                                0.3s
```

III. Conclusion

In summary, our project, the Arabic Automated Short Answers Grading System and Smart Assistance for Islamic Education, is designed to deliver precise grades to students based on their answers in Arabic. Our system employs advanced NLP techniques and LSTM model for accurate grading and provides valuable assistance to students during the evaluation process.