Information Technology Course

Module Software Engineering

by Damir Dobric / Andreas Pech

# *Implementation of Sequence Learning of*

# *SP/TM Layer in Microsoft Azure Cloud*

Ghulam Mustafa

ghulam.mustafa@stud.fra-usa.de

Muhammad Mubashir Ali Khan

muhammad.khan2@stud.fra-uas.de

Abdul Samad

abdul.samad@stud.fra-uas.de

Treesa Maria Thomas

[treesa.thomas@stud.fra-uas.de](mailto:treesa.thomas@stud.fra-uas.de)

Diab elmehdi

el.diab@stud.fra-uas.de

**Abstract— Cloud computing is the recent trend and its pay per usage pricing of IT services is conquering the World Wide Web and leading it to a new era. Rather than purchasing and managing the physical data centers and servers, this allows the access of technology services, such as computing power, storage, and databases, typically on a sufficiently needed basis from a cloud provider service like Microsoft Azure. The major goal here is to deploy our project of Sequence Learning of SP/TM Layer in Microsoft Azure Cloud.**

**Keywords— *Cloud computing, Microsoft azure cloud, Docker container***

1. **INTRODUCTION**

Various organizations including all types, size and sectors are using the cloud for variety of applications such as data backup, disaster recovery, email, virtual desktops, software development and testing, big data analytics and web applications facing customers. For instance, Health Care providers are making use of cloud for more personalized treatment of patients. Account firms are using such services for real-time fraud detection and prevention. And the game developers use this to offer online games to thousands of players around the world. The major benefits of cloud include agility, elasticity, cost savings and faster deployment.

The cloud brings you easy access to a wide variety of technology, and you can help build more efficiently and develop practically anything you picture. The cloud brings you easy access to a wide variety of technology, and you can help build more efficiently and develop practically anything you picture. Cloud computing can accommodate peak level of business operation in the future without the need to over – provision the resources. It also allows you to scale up or down the resources as per the need of the hour. Cost saving is another major advantage of this. Additionally the variable costs are much smaller than what you would pay to do it yourself. The cloud helps you to grow to new geographic regions and deploy in minutes globally. This enormously brings the applications more closely to the end-users and eliminates latency and enhances their user experience.

To test the Microsoft ML-based Sequence learner in Microsoft Azure Cloud the knowledge and understanding of following components are necessary.

# 1 *Azure Container Instances*

# Just as shipping and parcel services use physical containers to separate various loads, software development technologies are increasingly using a method called containerization. A standard software package, known as a container, wraps the code of an application along with the associated configuration files and libraries and the dependencies necessary to run the program. This enables apps to be implemented smoothly across all environments by developers and IT pros.

# *2 Azure Storage*

# In Microsoft Azure, there are five storage forms and, by their nature, they can be divided into two classes. One group is designed for file storage, scalability, and collaboration and can be accessed through the REST API. The other group is designed to improve the Microsoft Azure Virtual Machine environment functionality and to be accessed exclusively from VMs. Fig 1.1 illustrates the hierarchy of storage.

# 

# *Fig. 1.1. Hierarchy of Blob storage*

# *2a Azure Blob storage*

# When you upload a file in blob, it is broken down into parts called blocks and only after the upload is complete it is put together in blob. You can have access to the files, but inside the storage, you can't alter them. Block Blobs are suitable for backups and user file storage and is the cheapest.

# *2b Azure Table storage*

# As surmised table storage will hold tables. The storage is highly versatile and, at the same time, inexpensive for data preservation. However, as you view files regularly, it becomes more costly

# *2c Azure Queue Storage*

# In general, Azure Queue storage is a service for storing large numbers of messages which is accessible via authenticated HTTP or HTTPS requests from everywhere in the world. The size of a single queue is up to 64 kb. With Azure Queues, you have a third player that connects the two components and acts as both a buffer and a mediator.

# 

# ( this picture is from git wiki I am not sure if we can use this instead of 1.1. Kindly advice)

# GENERAL WORKFLOW

# The major goal of our project is to run the already implemented code of Sequence Learning of SP/TM Layer in Microsoft Azure Cloud.

# 

# *Fig. 1.1. Workflow*

# Develop your code as an azure function and publish it as a Docker container ( explain according to our project )

# User uploads the training data files as a ZIP File to the Blob storage. ( explain according to our project )

# User sends a message to trigger the training process. explain according to our project )

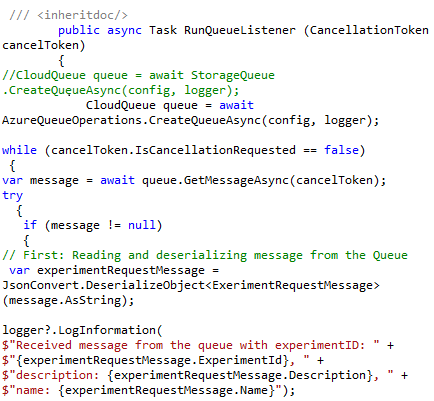
# Download and extract the ZIP file with your training data in it. ( explain according to our project )

# Your code as an Azure function inn Docker container hosted in the App service. ( explain according to our project )

# 3.CODE DESCRIPTION

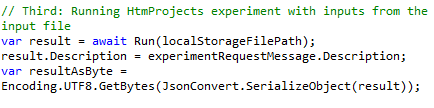
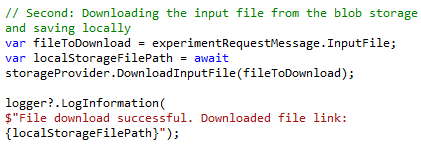
The first Three components are determined to be used in the code as seen below.

1. storageProvider: Concerns blob storage connections, i.e. upload and transfer blob information. b) logger: concerns the instant appendage of standardized messages through the code-named logs. Logger: c) config: reflects the project's specified configuration, container name relations, queue and Group ID.
2. The layout for the class shall be declared as shown after the object declarations. Note that the variable set to connect with configSection above is configSection. The configSection claim uses an IConfiguarationSection (descending configuration) that uses the InitHelper class internally to read the configuration or retrieve data from the project appconfig.json format. The next step is to construct a directory using the Path.combine analogy to concatenate the address using the two parameters Environment. SpecialFolder as defined in the config file appconfig.json. This guarantees an acceptable file route.

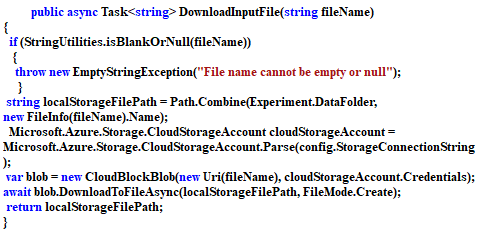


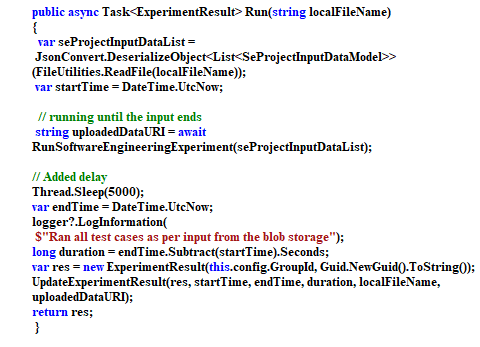
The process RunQueueListener is before the builder. It is expected to hear the queue message for the execution of the program as seen in the following text. This explains how the protocol works: the method 's arguments are canceled with the token form CancellationToken after a further execution. A queue is first generated using the user's storage account information which includes the software configuration to be run. The token status is then reviewed. Afterwards. In a cancelation condition (under pressing of the key from the user), the operation is halted and a file file is entered in. The process approaches 6 phases in non-cancelation.

Phase I: As shown by the above code the message is read at the queue. The message will be deserialized or translated from byte strike to entity in memory for a non-null message (which means that the program will run). A registration log will then be signed into the database, which contains information like groupID, name, definition, etc.

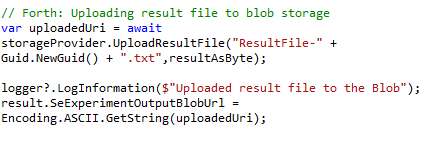
Phase II: You retrieve the name of the training data file from the queue post, and save the local path of the returned file in the localStorageFilePath vector. The DownloadInputFile function in AzureBlobOperation.cs is used to update the input file as seen in the following code snippet. Its solution is simple, as seen in the following code. The downloadable file name is provided in the claims. First you build the route for the local repository and add the file name. Next a blob instance will be configured using the Storage Account link chain, using the account

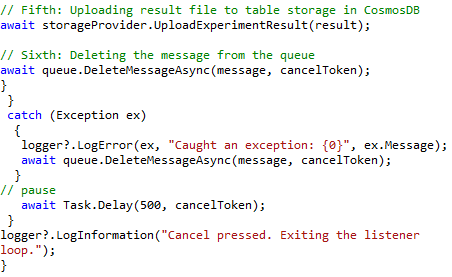
credentials and the file name URI in the database blob. The blob content is then downloaded into the file using the DownloadToFileAsync case.

Phase III: The software is run by supplying the direction of the stored file by means of the training file in the local repository. The product of the returned program is serialized for further processing or translated into a byte stream. When running the Run test, as outlined in the following code, the data from the downloaded file is first deterred after running the RunSoftware Engineering Experiment using RunHtmProjects, and the results are documented with time and length.

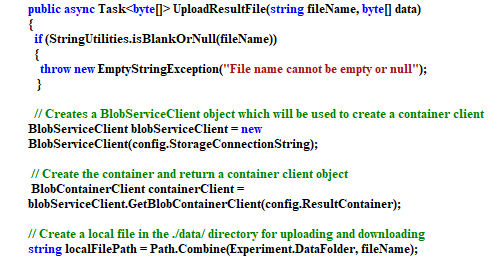


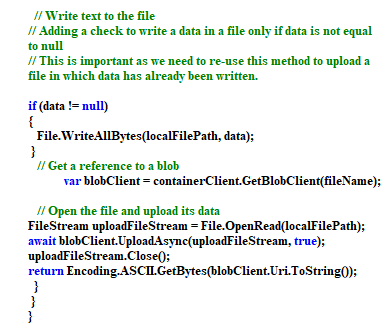
The above code indicates the loop with which input data is interpreted by calling various methods in the SDR classifier. The RunSoftwareEngineeringExperiment. The respective findings would then be reported in a file. The code snippet below illustrates how the output file of SDRClassifer is downloaded in blob using the UploadResultFile process.

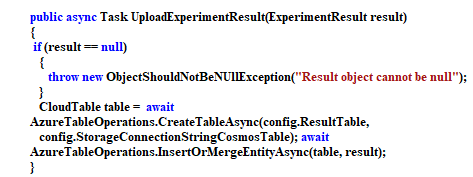


****

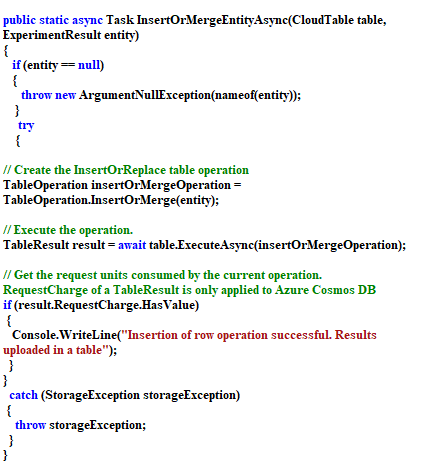
Phase IV: Through the UploadResultFile method in AzureBlobOperations.cs, the outcome of the software is loaded to blob storage. A fitting log is added, and the resultant blob URI is preserved. The method of UploadResultFile uses a basic methodology, as seen in the following code snippet, to construct a container client item. The result file name is added to a suitable local road.

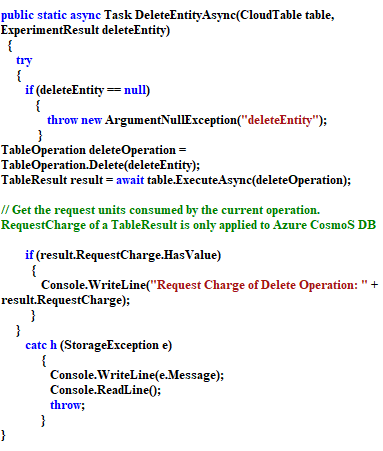
**** The blob data file will be uploaded and the URI of the file in blob will be returned as seen.

Phase V: Using the UploadExperimentResult in the AzureBlobOperation.cs below, the test is uploaded in a collection table.



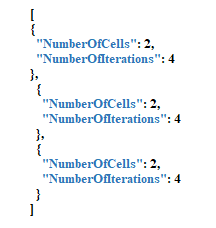
A table is generated by using the CreateTableAsync storage relation string in the AzureTableOperations class and if there's no such entity then the result is mixed. This is achieved by AzureTableOperations.cs InsertOrMergeEntityAsync process.

Phase VI: The queue message is eliminated. In the DeleteEntityAsync process, you transfer the cancelation token and post. This is to observe the token of cancelation of the deletion. The deletion would not occur if the token signals interrupt the operation.

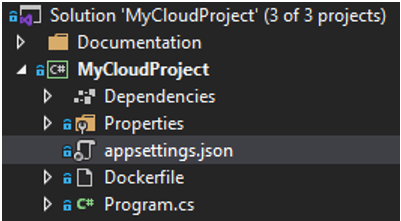


# 3.How to run experiment

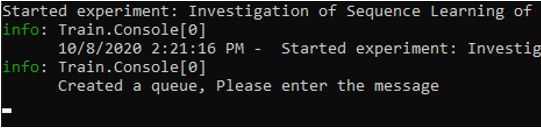
Phase 1: Two blob containers, respectively input and result files, are created. As seen below, a json file is then uploaded into the container of input data.



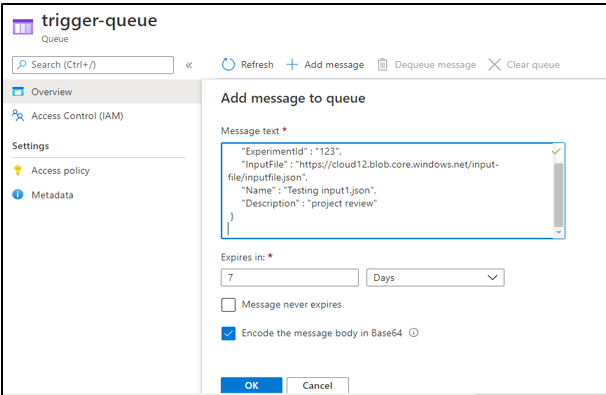
Phase 2: The connection string in the file appsettings.json is updated



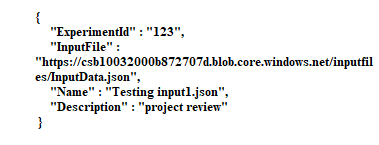
****Phase 3: These connection strings can be found in the setting of Storage Account and Cosmos DB respectively. After making these changes make two containers in the storage account one for input “training-files” and other for output “result-files”.After running the project, it will create a queue and wait for our message.



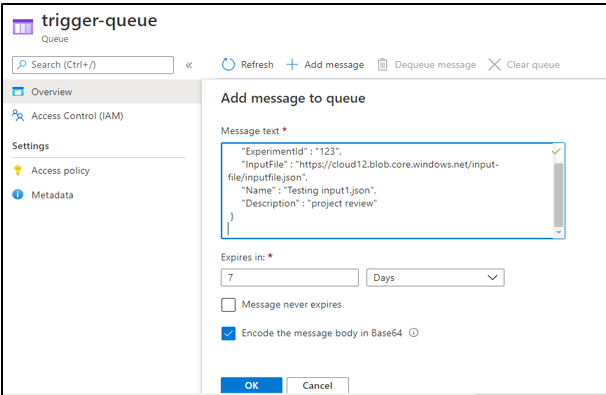
The queue can be found in the storage account.



Phase 4: The experiment is started and consecutively a queue is observed to be created in azure with the name; triggered queue. below data is copied and pasted in the queue message. Noted as url of the InputFile should be updated according to the individual container location



Phase 5: After uploading the input file to the container, we need to copy and paste the link in the queue message which can found under “MyExperiment/ProcessingData/QueueMessage.txt”. Now add this message to the queue.



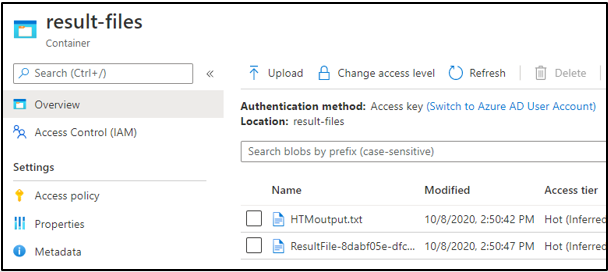
Now, it will run the experiment as per specifications and save the result files in the result file container.



# 4. RESULT

The results are validated with the presence of result-files in the respective azure blob container and table storage. Also the time stamp of the files helps in showcasing the latest execution of the experiment.

The uploaded files can be found in the result-files folder



# The first file HTMoutput.txt is the main result file for the experiment it has the result in terms of average cycles that the experiment will take to get 100 % prediction for given width, input bits, and iterations

# 

# The other file “ResultFile-……” will have different information like the name of experiment, start time, end time, duration etc. Most of this data and the links to output file and blob are also uploaded to Cosmos DB. It can be seen in the data explorer.

# 

REFERENCES

[1] Hawkins, J., & Ahmad, S. (2016). Why neurons have thousands of synapses, a theory of sequence memory in neocortex. Frontiers in neural circuits, 10.

[2] S. Ahmad, M. Lewis, (2017).“Temporal memory algorithm”, Technical Report Version 0.5, Numenta Inc.

[3] Zyarah, A. M., & Kudithipudi, D. (2019). Neuromorphic architecture for the hierarchical temporal memory. IEEE Transactions on Emerging Topics in *Computational Intelligence*, *3*(1), 4-14.