|  |  |
| --- | --- |
| Image Analysis | ARCHITECTURE EXPLAINED |

Version/revision: 1.0

Date: 29/08/2017

Authors: Pablo Doval, Jose Vizoso, Francisco Martínez, Daniel Escribano

Change Control

| DATE | VER./REV. | DESCRIPTION |
| --- | --- | --- |
| 29/08/2017 | **1.0** | Initial Version. |

Contents

[1. Introduction 4](#_Toc491802460)

[2. Proposed system 4](#_Toc491802461)

[2.1. Data Ingestion 5](#_Toc491802462)

[2.2. Model Training and Retraining 5](#_Toc491802463)

[2.3. Output 7](#_Toc491802464)

[2.4. Evaluation 8](#_Toc491802465)

# Introduction

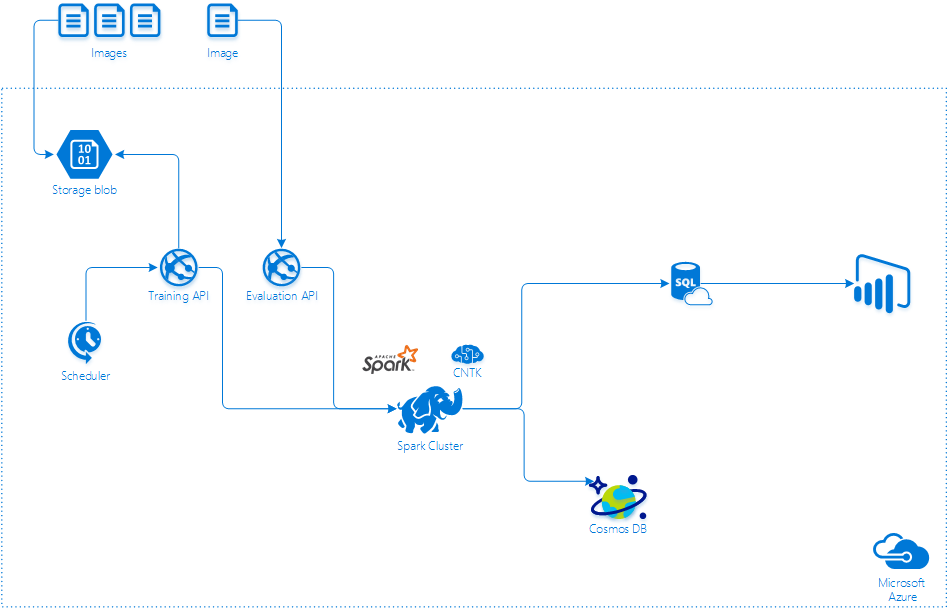
This document will walk through the architecture of the Image Analysis piece of Energy Demo, providing details about the proposed scenario and the different components that are part of it.

The scenario proposes an architecture to provide automatized help to technicians in the analysis of aerial imagery of power lines. A set of drones will survey the power lines in regular intervals, taking pictures of the structures along the way and submitting them to the system, where they’ll be stored in a blob storage account. A machine learning model will be trained with those images, using CNTK and FastRCNN on top of an HDInsight cluster, to identify the components of the image so a technician can easily evaluate the status of the structures. This model will be periodically retrained with new images to improve the precision and accuracy.

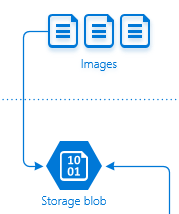
Results from the machine learning model training will be pushed to a SQL database (from where they’ll feed a PowerBI dashboard) and a Cosmos DB (to be consumed by client applications). An evaluation API will also be available to client apps to call, providing a single image and obtaining the position of the identified structures.

# Proposed system

An overview of the proposed architecture for this demo scenario can be seen in the diagram below:



* A set of aerial imagery will be stored in Azure Blob Storage. In a real-world scenario, these images will be obtained from a number of drones flying over the power lines, but for demo purposes we’ll have a set of them already stored. Since they’ll be used for training purposes, they need to be tagged with the identified classes and boundaries.
* The training Python script will perform all the required steps to train (and retrain) the model including downloading and classifying the images as test and train datasets, download the previous model and training checkpoint (if they exist), run the training and output the results to both SQL Server and CosmosDB. All these steps will be executed seamlessly in the HDInsight cluster, using PySpark and CNTK.
* From the training results stored in SQL Server, a PowerBI dashboard will be fed with data to showcase the improvements in precision and accuracy over subsequent trainings.
* An evaluation WebAPI will be available, from where an image can be submitted for evaluation, and the results will be returned in JSON form. In a real-world scenario, this would could be used in real time to analyze the images a drone is capturing.
  1. Data Ingestion



**Blob storage**

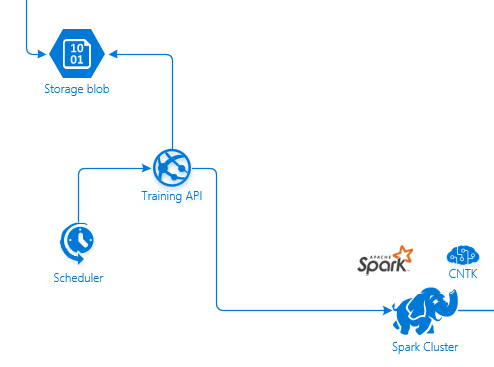
The images used for training the model will be stored in an Azure Blob storage account, so they can also be used by other applications and it’s easy for anyone that generates these images to make them available for the machine learning experiment. Within the boundaries of this demo, it will also be used to enable the re-training of the model, as explained in the next section.

Azure Blob Storage is a specialized storage solution for storing unstructured data offering great durability, availability, scalability, and performance. It offers two different access tiers:

* **Hot:** for frequently accessed data
* **Cool:** for less frequently accessed data, but with a considerable cost reduction in comparison with the hot storage.

Azure Hot Blob Storage will be used, since it provides better performance and availability.

* 1. Model Training and Retraining



There are two important pieces here: the initial training of the model, and the incremental training to improve precision and accuracy. In a real scenario, the training will be performed with all the images available, and then improved upon regularly with new images obtained by the drones. In our demo, we’ll work with two sets of images that will be placed in blob storage.

The user will be in charge of executing the script that trains the model, and can observe the process if desired. After training, the model and the training checkpoint will be serialized and stored in Azure Storage. Any subsequent executions will require the image set to be replaced with the retrain one, and will perform an incremental training.

**Spark cluster**

The aerial images obtained by the drones and stored in Azure Blob Storage will be processed by an application running in the Spark cluster. This application will be created in Python, and will use both CNTK and FastRCNN to train a model that is able to identify and classify the objects in the provided images.

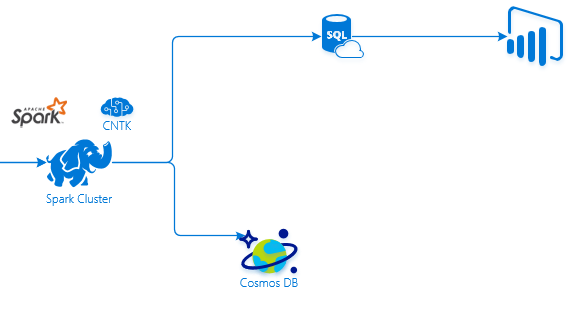
**CNTK**

The Microsoft Cognitive Toolkit (<https://cntk.ai>), is a unified deep-learning toolkit that describes neural networks as a series of computational steps via a directed graph, where leaf nodes represent input values or network parameters, while other nodes represent matrix operations upon their inputs. CNTK allows to easily realize and combine popular model types such as feed-forward DNNs, convolutional nets (CNNs), and recurrent networks (RNNs/LSTMs). We’ll combine CNTK with FastRCNN to achieve our image detection experiment,

**FastRCNN**

FastRCNN is an object detection algorythm proposed by Ross Girshick in 2015. It builds on uses deep convolutional networks to classify object proposals efficiently, by employing a region of interest pooling scheme (in our demo, based on the Selective Search algorithm) that allows training to be single stage, with a multi-task loss. For this demo, and to simplify the steps, we’ll use a pre-trained model of AlexNet that will be adapted to work with our power lines aerial imagery.

* 1. Output



**Power BI dashboard**

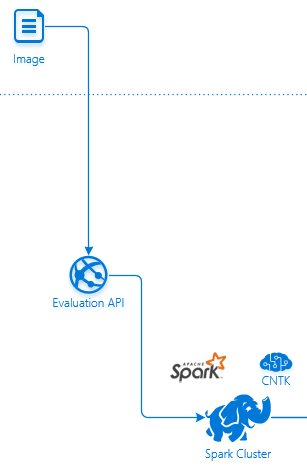
The results of training the model will be pushed to an SQL Server and a Cosmos DB instance. From this SQL Server, a Power BI dashboard will consume data to show the results of the training, and allow users to see how the model improves with each training performed.

**Azure Cosmos DB**

Azure Cosmos DB is Microsoft’s globally-distributed, multi-model database service "for managing data at planet-scale" launched in May 2017. It builds upon and extends the earlier Azure DocumentDB, which was released in 2014. It is schema-less and generally classified as a NoSQL database.

Data for the training will be pushed to a Cosmos DB instance so they’re available by clients that will be built in the future, such as Xamarin apps.

* 1. Evaluation



The evaluation WebAPI allows a user to evaluate a single image using the trained model. It will check periodically if there is a new version of model available to evaluate data against. In that case, it will be downloaded and used for the evaluation.

Each call to the API needs to provide an image, which will be evaluated. The results of the evaluation will be returned as a JSON response, containing the identified objects and their positions. In a real-world scenario, this API could be called by any client.