

# .NET Conf

2022

Deep Learning in .NET



# Deep Learning in .NET

Kinfey Lo - Microsoft Regional Cloud Advocate

[kinfeylo@microsoft.com](mailto:kinfeylo@microsoft.com)

<https://github.com/Kinfey>

Hi

Kinfey Lo



18 year+



2012-2021



10 year+



2014



7 year+



2018-2021



Since 2021.08  
Microsoft Cloud Advocate

# Today we will Talk

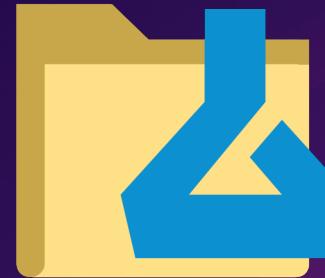
- Machine Learning in .NET
- Deep Learning in .NET
- MLOps in .NET



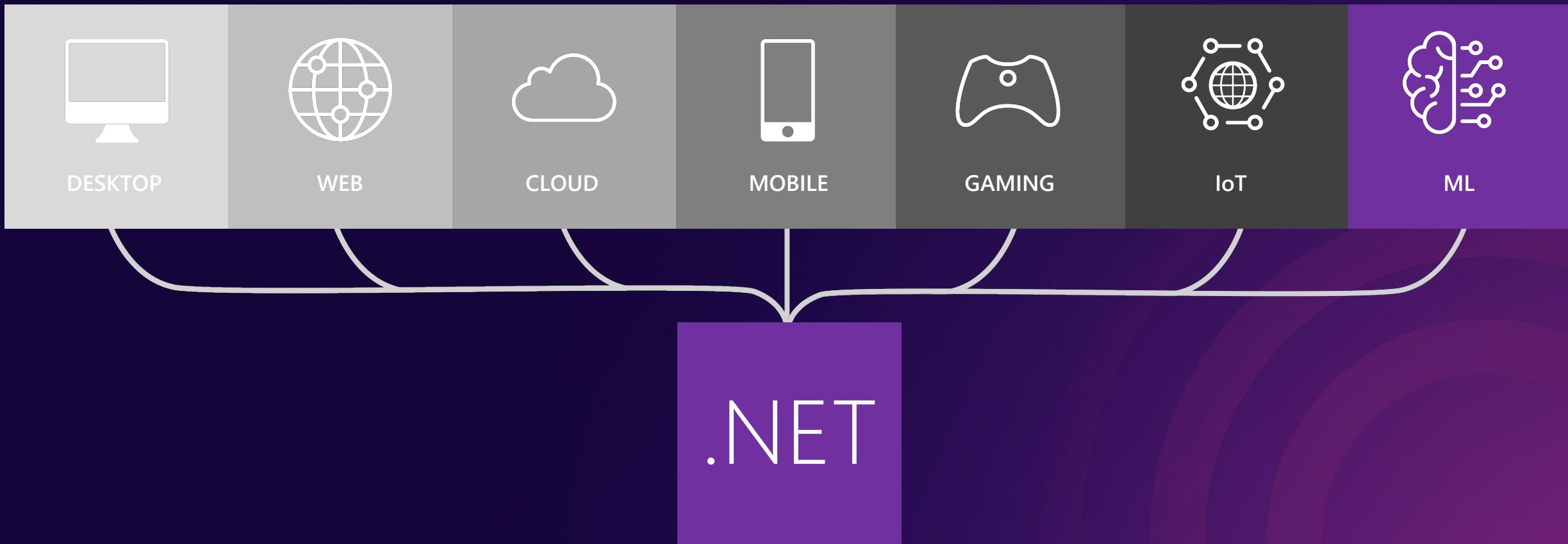
Face



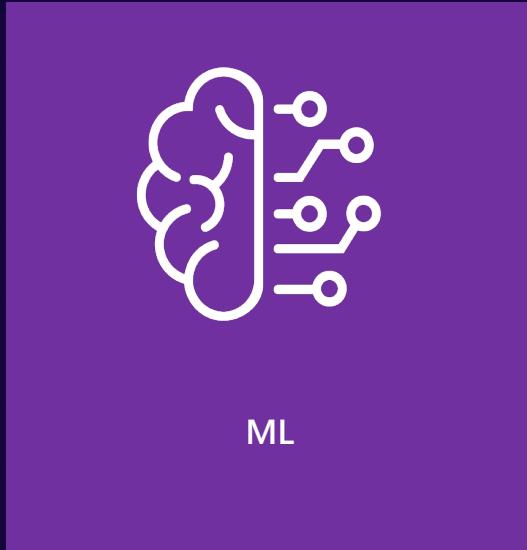
Not a face



# .NET can do everything



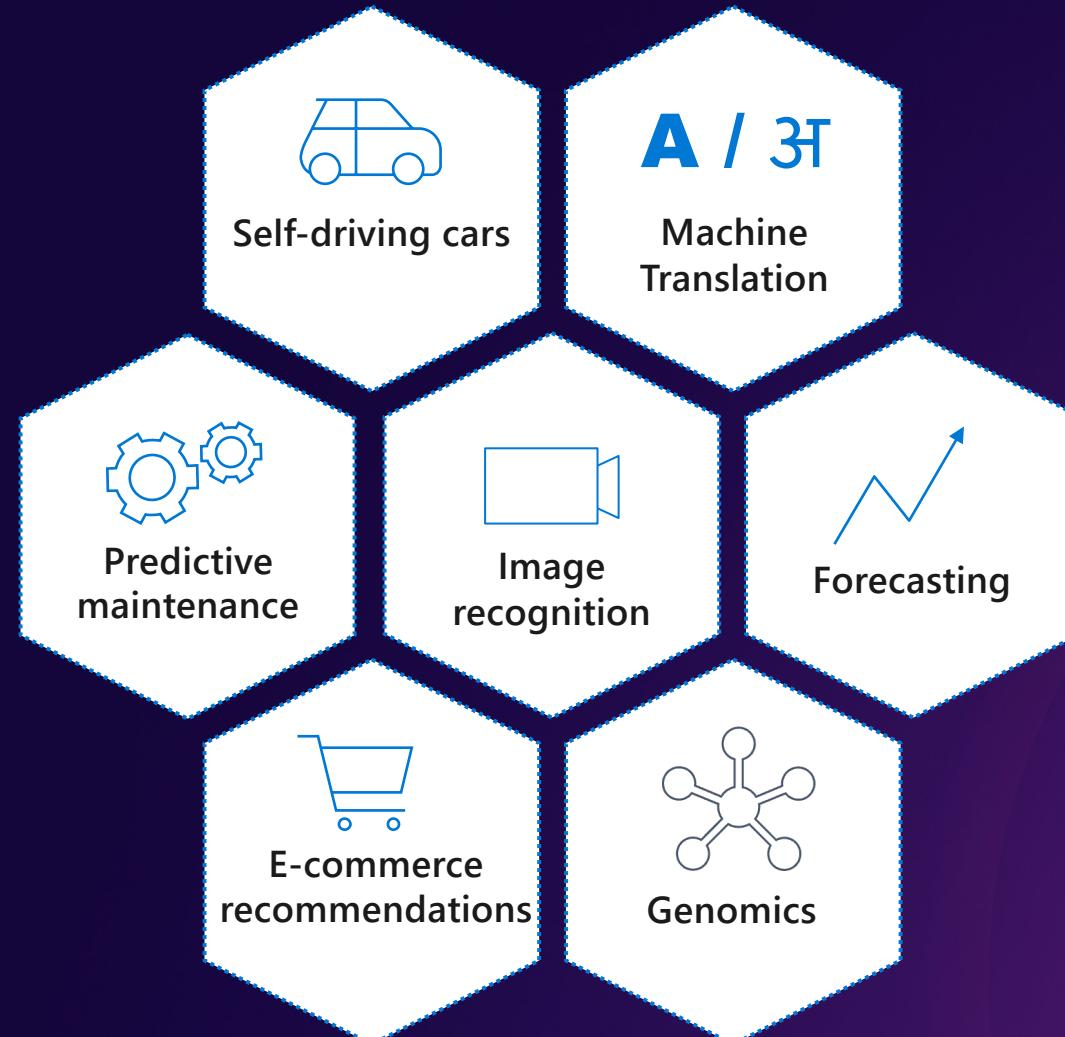
# What's Machine Learning ?



**Machine learning (ML) is a field of inquiry devoted to understanding and building methods that 'learn', that is, methods that leverage data to improve performance on some set of tasks.**

# Machine Learning

## Applications



# ML.NET



An open source and cross-platform  
machine learning framework for .NET

# ML.NET

## ML.NET CLI

```
cesard18@cli-test:~$ minet auto-train --task binary-classification --dataset "yelp_labelled.txt" --label-column-index 1 --has-header false --max-exploration-time 10
Exploring different feature items and settings to find you the best model for ML task: binary-classification
For further learning check: https://aka.ms/mlnet-cli

Best Accuracy: 87.30%, Best Algorithm: LbfgsLogisticRegressionBinary, Last Algorithm: SgdCalibratedBinary
00:00:19

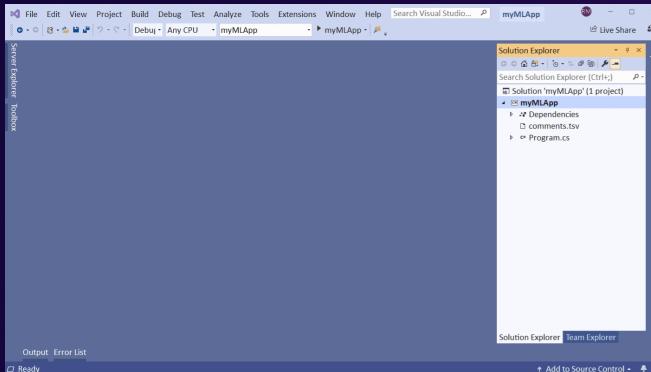
=====
Experiment Results=====

| ML Task: binary-classification
| Dataset: yelp_labelled.txt
| Label : Label
| Total experiment time : 10.50 Secs
| Total number of models explored: 48

| Top 5 models explored
| -----
| Trainer          Accuracy   AUC   AUPRC   F1-score   Duration #Iteration
| 1 LbfgsLogisticRegressionBinary 0.8736 0.9388 0.9272 0.8864 0.1 22
| 2 SgdCalibratedBinary 0.8736 0.9388 0.9272 0.8864 0.1 34
| 3 AveragePerceptronBinary 0.8621 0.9355 0.9181 0.8667 0.9 1
| 4 LbfgsLogisticRegressionBinary 0.8621 0.9388 0.9361 0.8750 0.2 18
| 5 LbfgsLogisticRegressionBinary 0.8621 0.9313 0.9380 0.8700 0.1 24

Generated trained model for consumption: /Users/cesard18/cli-test/SampleBinaryClassification/SampleBinaryClassification.Model/MlModel.zip
Generated C# code for model consumption: /Users/cesard18/cli-test/SampleBinaryClassification/SampleBinaryClassification/ConsoleApp
Check out log file for more information: /Users/cesard18/cli-test/SampleBinaryClassification/log/GoogleLog.txt
cesard18@cli-test:~$
```

## Model Builder



## AutoML

Can run an AutoML experiment on a given dataset to iterate over different data featurizations, ML algorithms, and hyperparameters to **select the best model**

# All .NET Developers

Can use  
existing C# and  
F# skills to  
integrate ML  
into .NET apps

No data  
science or ML  
experience  
required

Tools and features to help developers easily build, train, and deploy high-quality custom ML models both locally and in Azure

\*Goal is not to convert Python or R developers to .NET but instead to target 100% of .NET developers and enable them to add ML to their .NET apps

# Developer friendly API for Machine Learning

## ML Model Training & Consumption

Data	Training ML Tasks	Model Consumption & Evaluation	Extensions & Tools
IDataView (Dataset)	Classical ML Classification Regression Anomaly Detection Recommendation Time Series Ranking Clustering	Consumption Model Prediction Engine Prediction Engine Pool ONNX Export	ONNX Consumption
File Loaders	Computer Vision Image Classification Object Detection	Evaluation Model Evaluators Quality Metrics	TensorFlow Consumption
Database Loader	NLP Text Classification Sentence Similar		AutoML
Image Loader			CLI (Command Line Interface)
Data Transforms			Model Builder in VS
			Jupyter Notebooks



What about  
Deep Learning in .NET ?

# Deep Learning in .NET



## TensorFlow.NET

<https://github.com/SciSharp/TensorFlow.NET>

**TensorFlow.NET (TF.NET)** provides a .NET Standard binding for TensorFlow. It aims to implement the complete Tensorflow API in C# which allows .NET developers to develop, train and deploy Machine Learning models with the cross-platform .NET Standard framework. TensorFlow.NET has built-in Keras high-level interface and is released as an independent package TensorFlow.Keras.

```
### install tensorflow C#/F# binding
PM> Install-Package TensorFlow.NET
### install keras for tensorflow
PM> Install-Package TensorFlow.Keras

### Install tensorflow binary
### For CPU version
PM> Install-Package SciSharp.TensorFlow.Redist

### For GPU version (CUDA and cuDNN are required)
PM> Install-Package SciSharp.TensorFlow.Redist-Windows-GPU
```

# Deep Learning in .NET



TorchSharp is a .NET library that provides access to the library that powers PyTorch. It is part of the .NET Foundation.

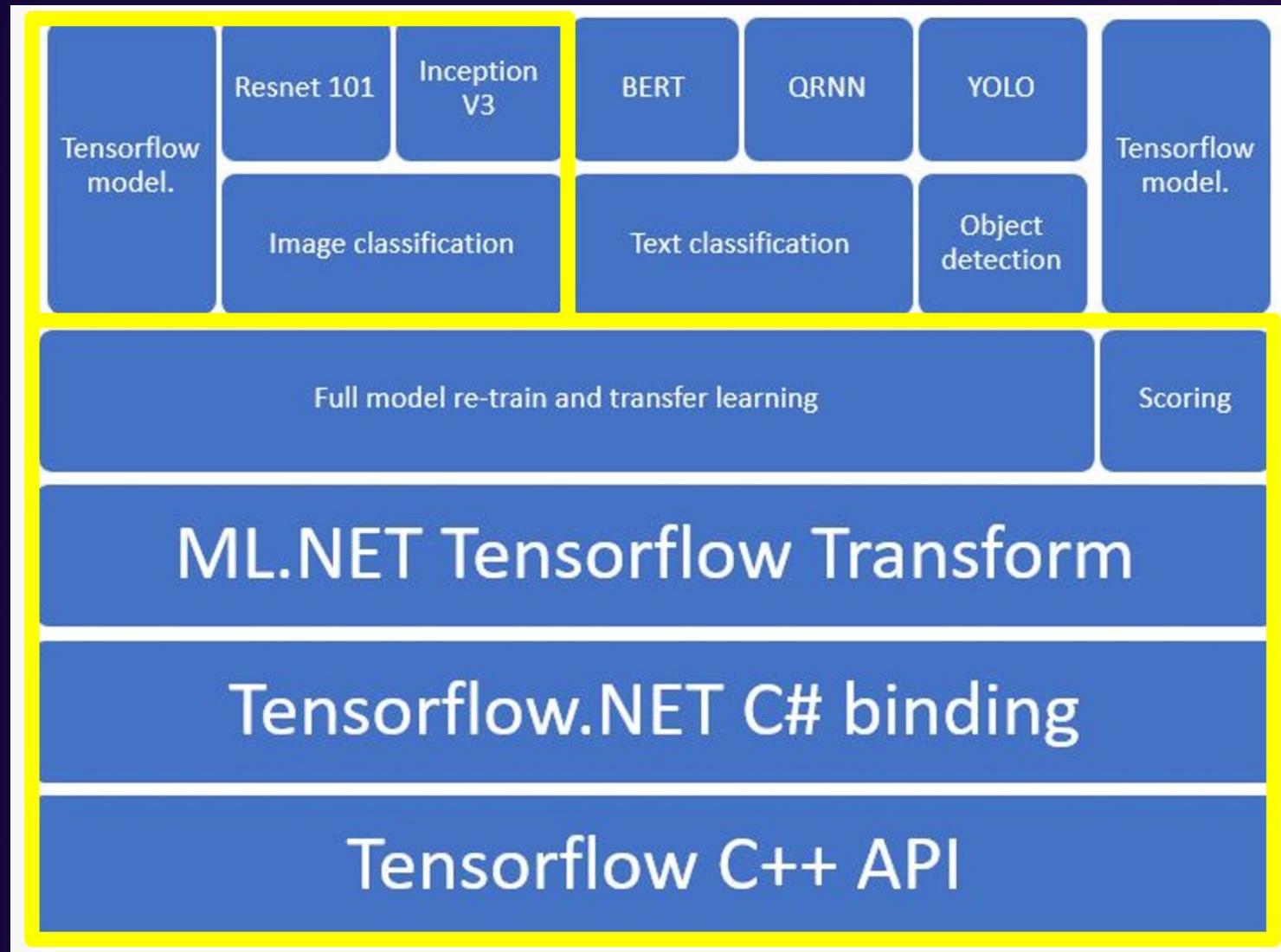
<https://github.com/dotnet/TorchSharp>

- `libtorch-cpu-linux-x64` (CPU, Linux)
- `libtorch-cpu-win-x64` (CPU, Windows)
- `libtorch-cpu-osx-x64` (CPU, OSX)
- `libtorch-cpu` (CPU, references all three, larger download but simpler)
- `libtorch-cuda-11.7-linux-x64` (CPU/CUDA 11.3, Linux)

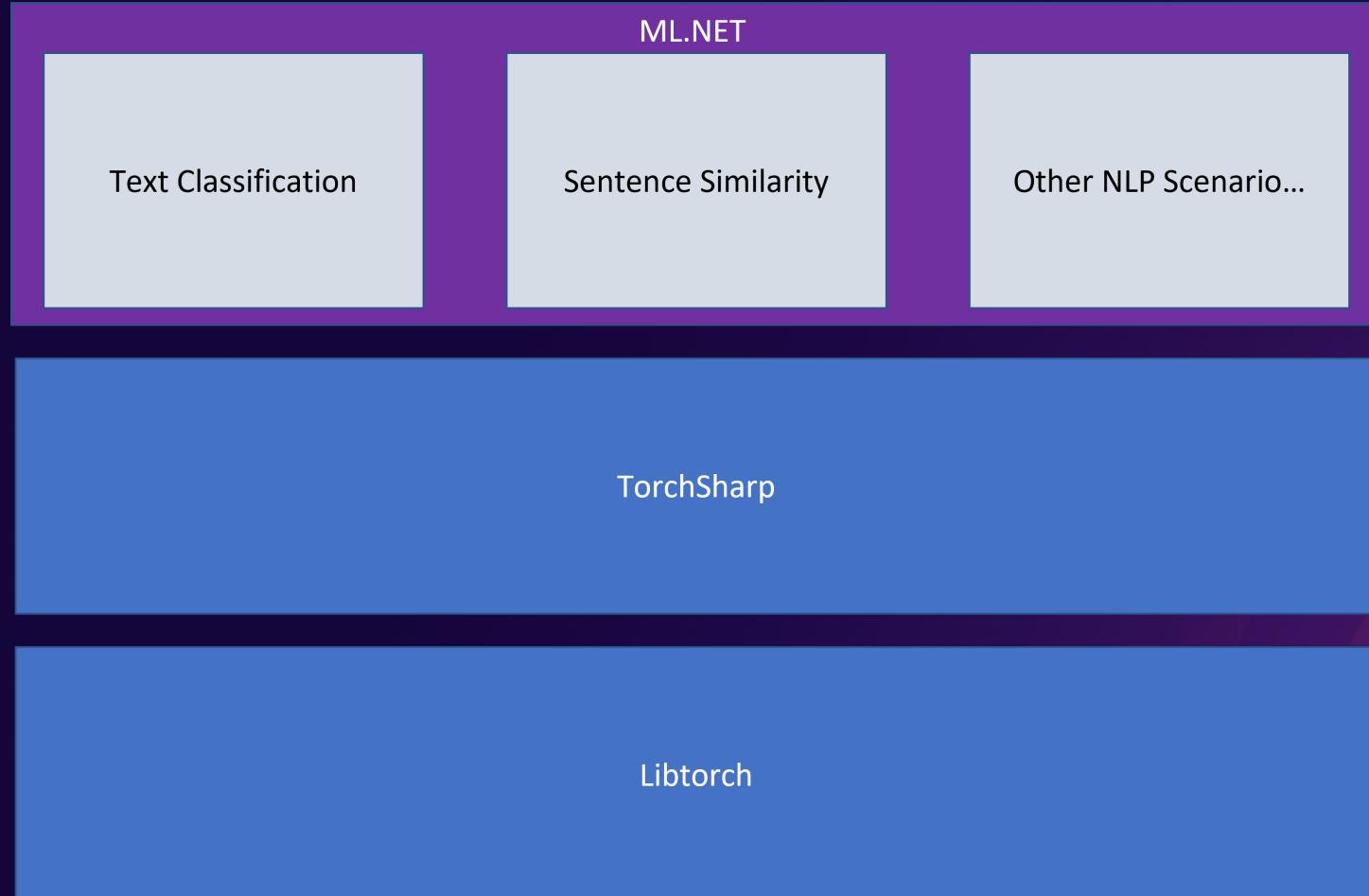
NOTE: Due to the presence of very large native binaries, using the `libtorch-cuda-11.7-linux-x64` package requires .NET 6, e.g. .NET SDK version `6.0.100-preview.5.21302.13` or greater.

- `libtorch-cuda-11.7-win-x64` (CPU/CUDA 11.3, Windows)

# ML.NET Image Classification



# ML.NET Image Classification



Maybe we need to more real Machine Learning scenario

# Azure ML

1. Snapshot folder and send to experiment



My Computer



Experiment

6. Stream  
stdout, logs,  
metrics

7. Copy over outputs

2. Create docker image



Docker Image

5. Launch script



Compute Target

Azure ML  
Workspace



3. Deploy  
docker and  
snapshot to  
compute



Data Store

4. Mount  
datastore to  
compute

# Prepare : Upload your data to Azure ML Data

Data

Data assets   Datastores   Dataset monitors (preview)

Data assets are references to your data. You can create data assets from datastores, local files, public URLs, or Open Datasets. Data assets can be versioned and easily referenced and reused for machine learning tasks. [Learn more about data assets](#)

+ Create   Refresh   Archive   Edit columns   Reset view

Search   All filters   Clear all

Showing 1-4 of 4 data assets   Page size: 25

Name	Version	Data source	Created on	Modified on	Type	Properties	Created by
starwar_data	1	workspaceblobstore	Nov 8, 2022 10:19 AM	Nov 8, 2022 10:19 AM	Folder		Lo Kinfey
flower_data	1	workspaceblobstore	Nov 2, 2022 5:55 PM	Nov 2, 2022 5:55 PM	Folder		Lo Kinfey
news_data	1	workspaceblobstore	Oct 31, 2022 5:38 PM	Oct 31, 2022 5:38 PM	Folder		Lo Kinfey
song_popularity	1	workspaceblobstore	Oct 28, 2022 3:32 PM	Oct 28, 2022 3:32 PM	File		Lo Kinfey

# Prepare : Set Azure ML Compute

Compute

Compute instances   Compute clusters   Inference clusters   Attached computers

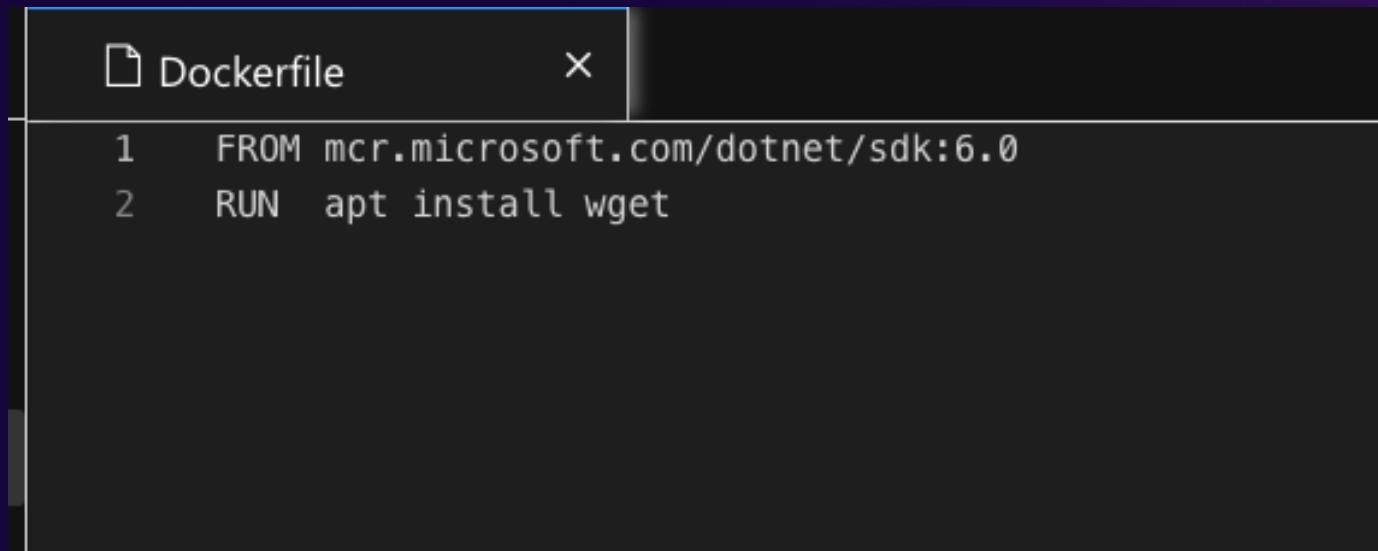
+ New   Refresh   Delete   Edit columns   Reset view   |   View quota

Search   State   Location   All filters   Clear all

Name	☆	State	Size	Location	Created on	Active runs	Idle nodes	Busy nodes	U
GPUCluster		<span>✓ Succeeded (0 nodes)</span>	STANDARD_NC6	westus2	Nov 3, 2022 8:16 PM	0	0	0	0

# Prepare : Create a docker and yml

```
FROM mcr.microsoft.com/dotnet/sdk:6.0
RUN dotnet tool install --global mlnet-linux-x64
ENV PATH="$PATH:/root/.dotnet/tools"
```



A screenshot of a code editor window titled "Dockerfile". The window contains the following two lines of Dockerfile code:

```
1 FROM mcr.microsoft.com/dotnet/sdk:6.0
2 RUN apt install wget
```

# Prepare : Create a docker and yml

```
$schema: https://azuremlschemas.azureedge.net/latest/commandJob.schema.json
command: |
    FILENAME=libtensorflow-gpu-linux-x86_64-2.5.0.tar.gz
    wget -q --no-check-certificate https://storage.googleapis.com/tensorflow/libtensorflow/${FILENAME}
    tar -C /usr/local -xzf ${FILENAME}
    ldconfig /usr/local/lib
    export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/cuda-11.3/targets/x86_64-linux/lib
    cd code
    dotnet restore
    dotnet build
    dotnet run --dataPath=${{inputs.data_dir}} --outputPath="outputs"
code: .
inputs:
  data_dir:
    type: uri_folder
    path: azureml:flower_data:1
experiment_name: tf-image-training
environment:
  build:
    path: .
    dockerfile_path: Dockerfile
compute: azureml:GPUCluster
```

# Prepare : Create a docker and yml

```
$schema: https://azuremlschemas.azureedge.net/latest/commandJob.schema.json
command: mlnet image-classification --dataset ${inputs.data_dir} --output outputs --name "StarWarML"
code: .
inputs:
  data_dir:
    type: uri_folder
    path: azureml:starwar_data:1
experiment_name: mlnet-imageclassification-training
environment:
  build:
    path: .
    dockerfile_path: Dockerfile
compute: azureml:GPUCluster
```

# 1 : Snapshot folder and send to experiment

The ml extension to the Azure CLI is the enhanced interface for Azure Machine Learning. It enables you to train and deploy models from the command line, with features that accelerate scaling data science up and out while tracking the model lifecycle.

<https://learn.microsoft.com/en-us/azure/machine-learning/how-to-configure-cli>

```
az configure --defaults group=Your Resource Group workspace=Azure ML Workspace Name
```

```
az ml job create --file Your.yml
```

```
~/L/CloudStorage/OneDrive-Microsoft/Microsoft/FY23/GitHubRepo/dotNETMLinAzureML on main
az configure --defaults group=CUDAWSGroup workspace=CUDAWS
```

```
~/L/CloudStorage/OneDrive-Microsoft/Microsoft/FY23/GitHubRepo/dotNETMLinAzureML on main
az ml job create --file AzureTrain.yml
```

# 2 : Experiment

happy\_box\_6rpsq5nf1d 🖊️ ⚡️ ✅ Completed

Overview Metrics Images Child jobs Outputs + logs Code Explanations (preview) Fairness (preview) Monitoring (preview)

⟳ Refresh ⚡️ Connect to compute 🖊️ Edit and submit + Register model ⏹ Cancel 🗑️ Delete | ⏴ Download all ⏵ Enable log streaming ⏵ Word wrap

🔍 <> std.log.txt ✎ 20\_image\_build\_log.txt

azureml-logs  
20\_image\_build\_log.txt  
system\_logs  
user\_logs  
std\_log.txt

```
128 2022/11/03 05:41:19 Populating digests for step ID: acb_step_0...
129 2022/11/03 05:41:21 Successfully populated digests for step ID: acb_step_0
130 2022/11/03 05:41:21 Step ID: acb_step_1 marked as successful (elapsed time in seconds: 28.011481)
131 2022/11/03 05:41:21 Step ID: acb_step_2 marked as successful (elapsed time in seconds: 2.055775)
132 2022/11/03 05:41:21 The following dependencies were found:
133 2022/11/03 05:41:21
134 - image:
135   registry: cde4c13a8285455380d8260c9975c937.azurecr.io
136   repository: azureml/azureml_bc7358ba4e6b7e0cffd51cf5c7d34fab
137   tag: latest
138   digest: sha256:615c99cb033d002a5598e15ef50124711e5453867eda0d901ab2979b6595c51c
139   runtime-dependency:
140     registry: mcr.microsoft.com
141     repository: dotnet/sdk
142     tag: "6.0"
143     digest: sha256:ce977e0ce71ce4aecde3917f3abf0dcffcf952e9ca138704b63e1a838b4700c
144     git: {}
145 - image:
146   registry: cde4c13a8285455380d8260c9975c937.azurecr.io
147   repository: azureml/azureml_bc7358ba4e6b7e0cffd51cf5c7d34fab
148   tag: "1"
149   digest: sha256:615c99cb033d002a5598e15ef50124711e5453867eda0d901ab2979b6595c51c
150   runtime-dependency:
151     registry: mcr.microsoft.com
152     repository: dotnet/sdk
153     tag: "6.0"
154     digest: sha256:ce977e0ce71ce4aecde3917f3abf0dcffcf952e9ca138704b63e1a838b4700c
155     git: {}
156
157 Run ID: cc17 was successful after 51s
```

# 2 : Experiment

The screenshot shows a file explorer interface with a dark theme. On the left, there is a tree view of log files:

- azureml-logs/
  - 20\_image\_build\_log.txt
- outputs/
  - models.h5
- system\_logs/
  - cs\_capability
  - data\_capability
  - hosttools\_capability
  - lifecycler
  - metrics\_capability
  - snapshot\_capability
- user\_logs/
  - std\_log.txt

The "std\_log.txt" file is selected and highlighted with a blue border. The content of the file is displayed in a large text area on the right:

```
1555 Epoch: 010/010, Step: 0127/0147, loss: 1.648712, accuracy: 0.173244
1556 Epoch: 010/010, Step: 0128/0147, loss: 1.648889, accuracy: 0.172670
1557 Epoch: 010/010, Step: 0129/0147, loss: 1.648503, accuracy: 0.172494
1558 Epoch: 010/010, Step: 0130/0147, loss: 1.647967, accuracy: 0.173092
1559 Epoch: 010/010, Step: 0131/0147, loss: 1.647576, accuracy: 0.173298
1560 Epoch: 010/010, Step: 0132/0147, loss: 1.647555, accuracy: 0.172741
1561 Epoch: 010/010, Step: 0133/0147, loss: 1.647415, accuracy: 0.172570
1562 Epoch: 010/010, Step: 0134/0147, loss: 1.647400, accuracy: 0.172775
1563 Epoch: 010/010, Step: 0135/0147, loss: 1.647714, accuracy: 0.172606
1564 Epoch: 010/010, Step: 0136/0147, loss: 1.648350, accuracy: 0.171702
1565 Epoch: 010/010, Step: 0137/0147, loss: 1.648853, accuracy: 0.171178
1566 Epoch: 010/010, Step: 0138/0147, loss: 1.648233, accuracy: 0.171750
1567 Epoch: 010/010, Step: 0139/0147, loss: 1.648779, accuracy: 0.171233
1568 Epoch: 010/010, Step: 0140/0147, loss: 1.648586, accuracy: 0.172155
1569 Epoch: 010/010, Step: 0141/0147, loss: 1.648377, accuracy: 0.171997
1570 Epoch: 010/010, Step: 0142/0147, loss: 1.648831, accuracy: 0.171842
1571 Epoch: 010/010, Step: 0143/0147, loss: 1.648998, accuracy: 0.171689
1572 Epoch: 010/010, Step: 0144/0147, loss: 1.649805, accuracy: 0.170842
1573 Epoch: 010/010, Step: 0145/0147, loss: 1.649572, accuracy: 0.172080
1574 Epoch: 010/010, Step: 0146/0147, loss: 1.649600, accuracy: 0.171929
1575 Epoch: 010/010, Step: 0147/0147, loss: 1.649328, accuracy: 0.171438
1576 123:/mnt/azureml/cr/j/6767acdac1334190bc919637e7ad132a/exe/wd
1577 2022-11-03 06:45:37.149685: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1733] Found device 0 with properties:
1578 pciBusID: 32f6:00:00.0 name: Tesla K80 computeCapability: 3.7
1579 coreClock: 0.8235GHz coreCount: 13 deviceMemorySize: 11.17GiB deviceMemoryBandwidth: 223.96GiB/s
1580 2022-11-03 06:45:37.149719: W tensorflow/core/common_runtime/gpu/gpu_device.cc:1766] Cannot dlopen some GPU libraries. Pla
1581 Skipping registering GPU devices...
1582 2022-11-03 06:45:37.149736: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1258] Device interconnect StreamExecutor w:
1583 2022-11-03 06:45:37.149744: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1264]      0
1584 2022-11-03 06:45:37.149750: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1277] 0: N
1585
```

# 2 : Experiment

Jobs

All experiments All jobs

⟳ Refresh ⌂ Archive experiment 📈 Edit columns ↻ Reset view

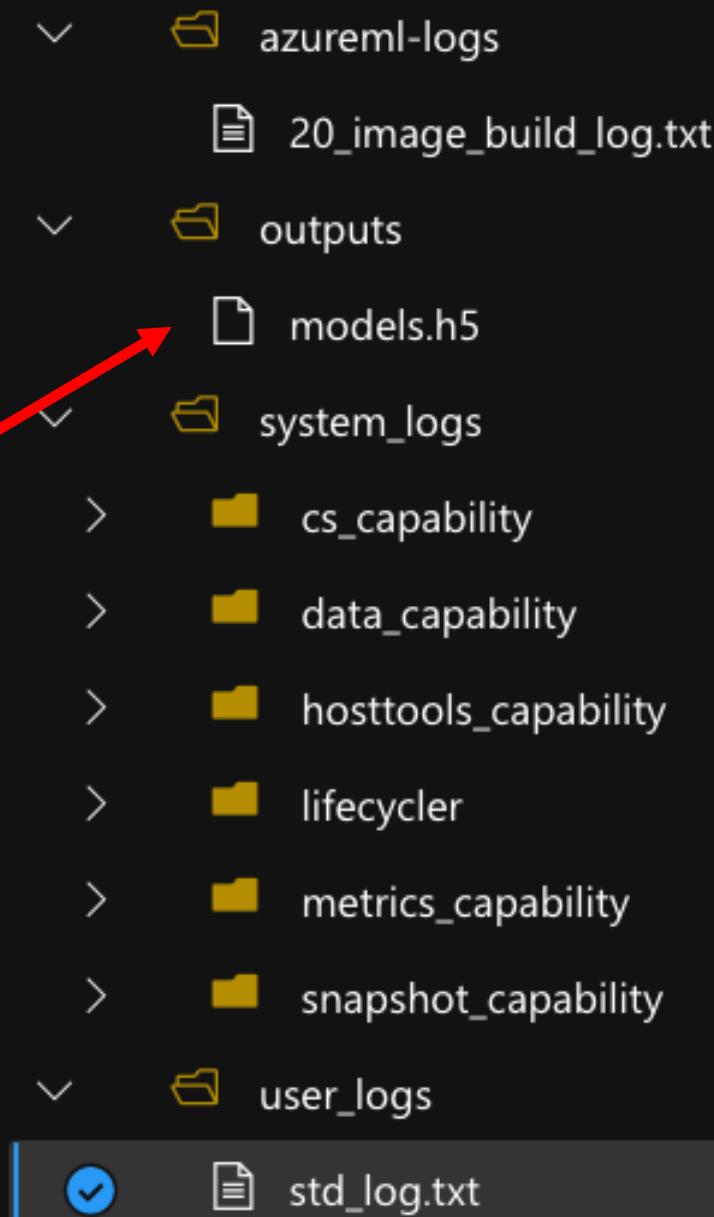
🔍 Search  View archived experiments

Showing 1-5 experiments

Experiment	⭐	Latest job	Last submitted ↓	Created	Created
tf-image-training		mango_cart_z3syldpp	Nov 10, 2022 10:16 AM	Nov 2, 2022 6:01 PM	Lo Ki
mlnet-imageclassification-training		sincere_needle_gh2ptwy5	Nov 8, 2022 11:11 AM	Nov 8, 2022 10:46 AM	Lo Ki
torchsharp-training	★	funny_pear_wxkwfpr2	Nov 2, 2022 6:29 PM	Nov 1, 2022 7:36 AM	Lo Ki
mldotnet-training		heroic_window_99jksv15	Nov 1, 2022 7:49 AM	Oct 28, 2022 4:13 PM	Lo Ki
dotnet-training		mango_balloon_6wvt1xkwqr	Oct 28, 2022 7:26 PM	Oct 28, 2022 7:26 PM	Lo Ki

# 3 : Output Model

You can see the model



# 4 : Register Model

Model List

+ Register ⏪ Refresh 🗑 Delete 📁 Archive ⏪ Deploy ⏪ Compare (preview) ⏪ Edit columns ⏪ Reset view |  Show latest versions only  Include archived

Search

Showing 1-1 of 1 models

Name	☆	Version	Experiment	Job (Run ID)	Created on ↓	Tags	Properties
demo		1		tough_shoe_2dvs7v6d53	Nov 3, 2022 5:29 PM		...

Page size: 25

Create deployment

Endpoint  
 Model  
 Deployment  
 Environment  
 Compute  
 Traffic  
 Review

**Create endpoint**

An endpoint is used to deploy and score your models. [Learn more](#)

**Endpoint name \***

**Description**

**Compute type** ⓘ  
 Managed  Kubernetes

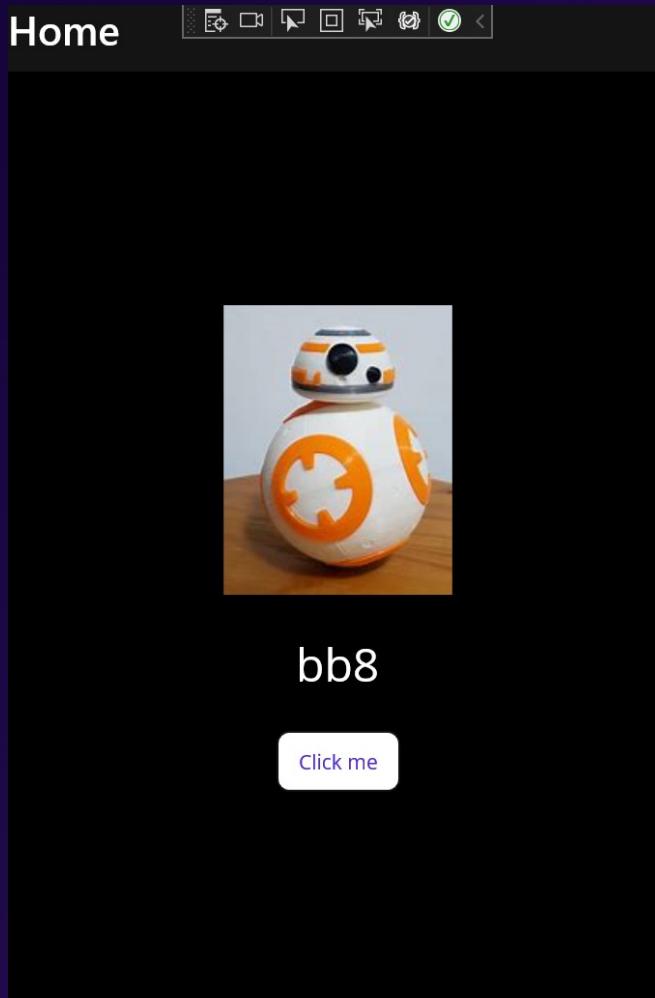
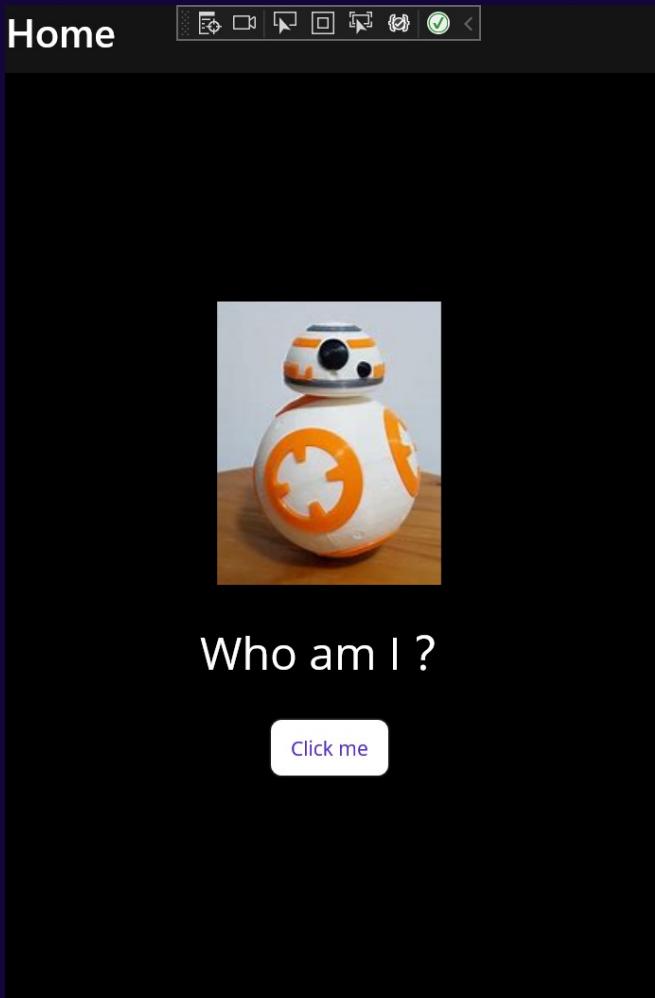
**Authentication type** ⓘ  
 Token-based authentication  Key-based authentication

**Public network access**

Enable this option if you want to allow scoring requests to your endpoint from the internet. Disable this option if you want to allow scoring requests to your endpoint only from your resources in your virtual network. [Learn more](#)

Enabled

# 5 : Download Model and use it in your .NET applications



We can use ML.NET in any .NET solutions

# Thank you

<https://github.com/kinfey/dotNETMLinAzureML>