Building Smart Applications with
Azure Cognitive Services

Jeff Prosise

jeffpro@wintellect.com

@jprosise

Wintellect®

Wintellect NOW

The AI Landscape

1950s – Researchers begin pondering whether computers can think like humans and devise **Symbolic AI** – e.g. chess programs

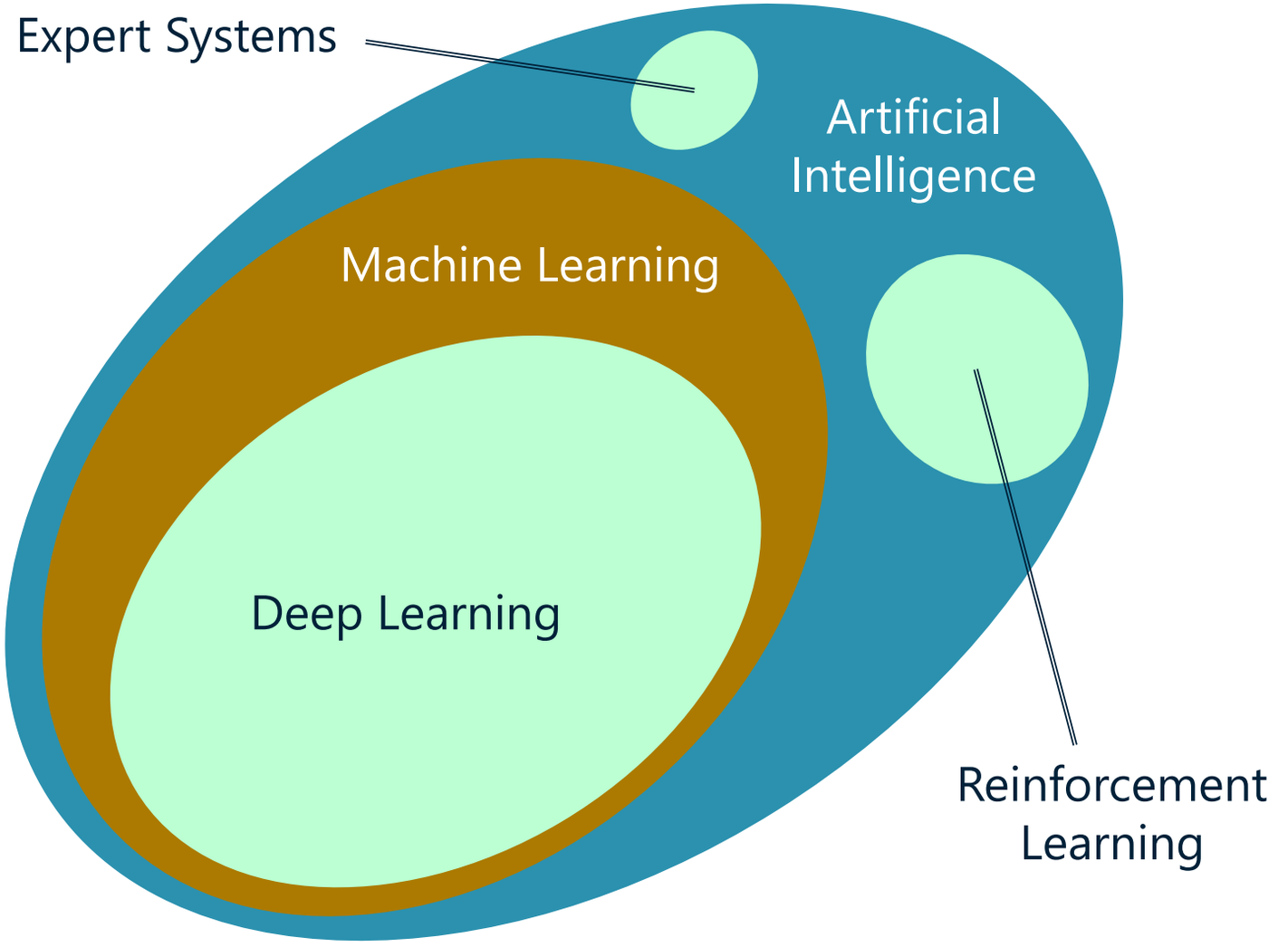
1960s – Early **neural networks** are devised, but training is impractical due to lack of computing power and access to compute resources

1980s – Researchers improve **back-propagation algorithms** for training neural networks and apply them to convolutional neural networks

1990s – Researchers at Bell Labs invent the modern **Support Vector Machine** (SVM) algorithm for discovering decision boundaries

2000s – Algorithmic advances in machine learning give rise to **decision trees, random forests**, **gradient-boosting machines**, and more

2010s – ML and AI explode due to faster computing hardware (especially **GPUs**), expanded availability of data, and increased research funding



Azure Cognitive Services – AI as a Service

Vision

Computer Vision API | Custom Vision Service | Face API | Form Recognizer API | Ink Recognizer API | Video Indexer

Speech

Speech to Text API | Text to Speech API | Speaker Recognition API | Speech Translation API

Language

Immersive Reader API | Language Understanding | QnA Maker | Text Analytics API | Translator API

Decision

Anomaly Detector API | Content Moderator API | Personalizer API

Web Search

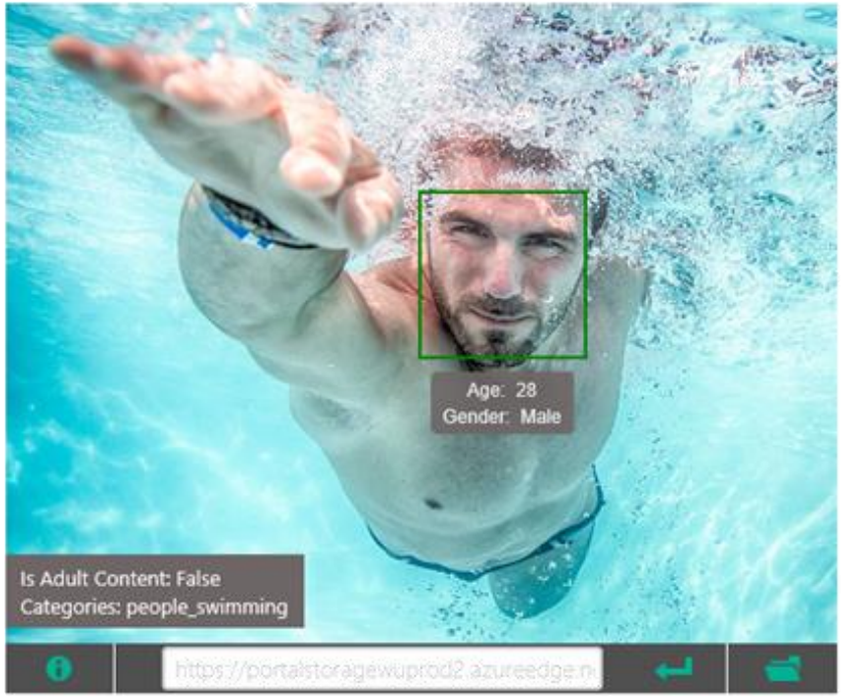
Bing Web Search API | Bing Custom Search API | Bing Video Search API | Bing Image Search API | Bing Spell Check API | Bing Visual Search API | Bing Entity Search API | Bing News Search API | Bing Autosuggest API

Three Ways to Authenticate

- Subscription keys (single-service or multi-service)
  - Obtained from Azure portal and passed in Ocp-Apim-Subscription-Key headers
  - Multi-service keys supported by most, but not all, Cognitive Services
- Azure Active Directory (AAD)
  - Apply role-based security using AAD service principals
  - Or use managed identities for function apps and apps running in VMs
- Authentication tokens
  - Created from subscription keys and valid for 10 minutes
  - Text Translation API, Speech to Text API, and Text to Speech API only

Computer Vision API

- Identify objects in photographs, caption images, detect adult content, generate smart thumbnails, OCR text, and more



Features:

Feature Name	Value
Description	{ "type": 0, "captions": [{ "text": "a man swimming in a pool of water", "confidence": 0.7850108693093019 }] }
Tags	[{ "name": "water", "confidence": 0.9996442794799805 }, { "name": "sport", "confidence": 0.9504992365837097 }, { "name": "swimming", "confidence": 0.9062818288803101, "hint": "sport" }, { "name": "pool", "confidence": 0.8787588477134705 }, { "name": "water sport", "confidence": 0.631849467754364, "hint": "sport" }]
Image Format	jpeg
Image Dimensions	1500 x 1155
Clip Art Type	0 Non-clipart
Line Drawing Type	0 Non-LineDrawing
Black & White Image	False

Analyzing an Image for Adult Content

```
POST /vision/v3.0/analyze?visualFeatures=Adult HTTP/1.1
Content-Type: application/json
Content-Length: •••
Host: westus.api.cognitive.microsoft.com:443
Ocp-Apim-Subscription-Key: ••••••••••••••••••••••••••••••••
```

```
{"url":"https://intellipix.blob.core.windows.net/photos/racy.jpg"}
```

JSON Output


```
{
  adult: {
    isAdultContent: false,
    isRacyContent: true,
    adultScore: 0.0017618900164961815,
    racyScore: 0.991065263748169
  },
    ...
}
```

Captioning an Image

```
POST /vision/v3.0/analyze?visualFeatures=Description HTTP/1.1
Content-Type: application/json
Content-Length: •••
Host: westus.api.cognitive.microsoft.com:443
Ocp-Apim-Subscription-Key: ••••••••••••••••••••••••••••••••
```

```
{"url":"https://intellipix.blob.core.windows.net/photos/dubai.jpg"}
```

JSON Output


```
{
  description: {
    tags: [ "man", "dune", "riding", "board", "hill", ..., "sand" ],
    captions: [
      {
        text: "a man riding a skateboard in the sand",
        confidence: 0.66107721083049154
      }
    ]
  },
    ...
}
```

Captioning an Image (C#)

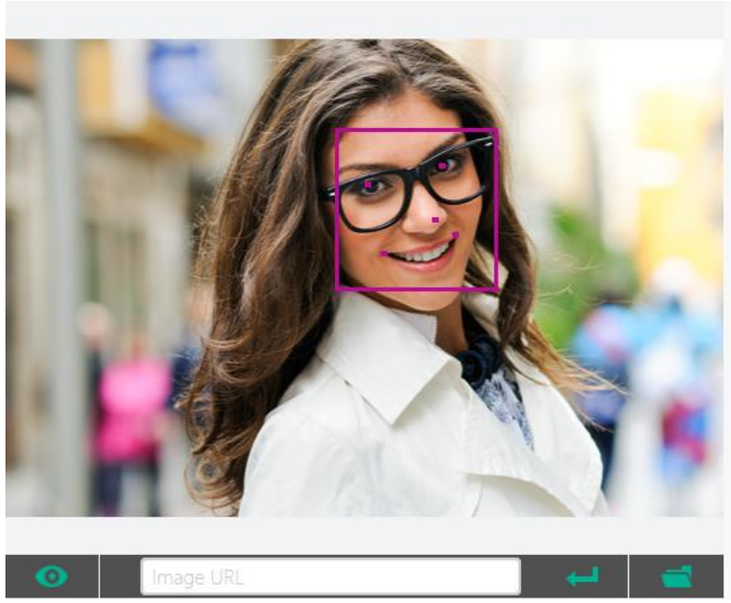
```
ComputerVisionClient client = new ComputerVisionClient(
    new ApiKeyServiceClientCredentials("SUBSCRIPTION_KEY"),
    new System.Net.Http.DelegatingHandler[] { }
);

client.Endpoint = "VISION_SERVICE_URL";
var features = new VisualFeatureTypes[] { VisualFeatureTypes.Description };
ImageAnalysis result = await client.AnalyzeImageAsync("IMAGE_URL", features);

string caption = result.Description.Captions[0].Text;
foreach (string tag in result.Description.Tags)
{
    // tag holds descriptive tag for image (e.g., "sand")
}
```

Demo

Computer Vision API

Face API

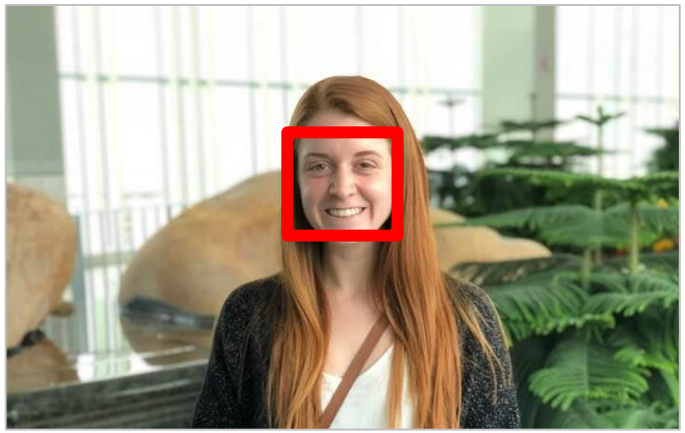
- Detect and identify faces and facial features in photos
- Identify age, gender, emotion, facial landmarks, and more
- Perform face verification by comparing two photos




```
Detection Result:
JSON:
[
  {
    "faceId": "d543ccd7-9988-4794-9c94-697dafef65f4",
    "faceRectangle": {
      "width": 227,
      "height": 227,
      "left": 459,
      "top": 124
    },
    "faceLandmarks": {
      "pupilLeft": {
        "x": 504.4,
        "y": 202.8
      },
      "pupilRight": {
        "x": 607.7,
        "y": 175.9
      },
      "noseTip": {
        "x": 598.5,
        "y": 250.9
```

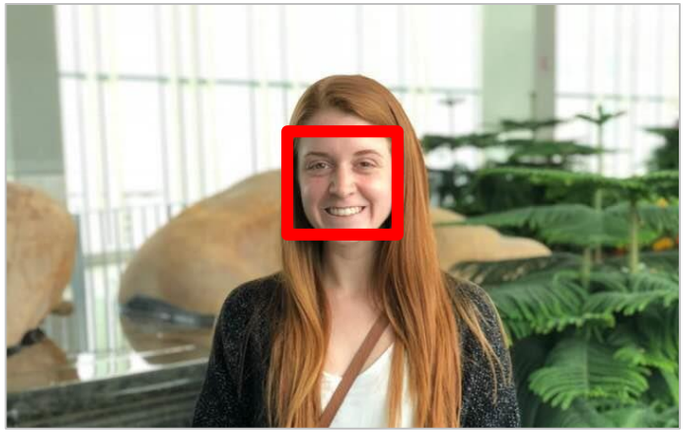
Detecting Faces in a Photo

```
FaceClient client = new FaceClient(
    new ApiKeyServiceClientCredentials("SUBSCRIPTION_KEY"),
    new System.Net.Http.DelegatingHandler[] { }
);

client.Endpoint = "FACE_SERVICE_URL";
IList<DetectedFace> faces = await client.Face.DetectWithUrlAsync("IMAGE_URL");

foreach (DetectedFace face in faces)
{
    FaceRectangle rect = face.FaceRectangle;
}
```



Getting Information About Faces

```
IList<FaceAttributeType> attributes = new FaceAttributeType[]
{
    FaceAttributeType.Gender, FaceAttributeType.Age
};

IList<DetectedFace> faces =
    await client.Face.DetectWithUrlAsync("IMAGE_URL", true, false, attributes);

foreach (DetectedFace face in faces)
{
    FaceRectangle rect = face.FaceRectangle;
    Gender gender = (Gender)face.FaceAttributes.Gender;
    double age = (double)face.FaceAttributes.Age;
}
```



Comparing Two Faces

```
DetectedFace face1 = (await client.Face.DetectWithUrlAsync("IMAGE_URL"))[0];
DetectedFace face2 = (await client.Face.DetectWithUrlAsync("IMAGE_URL"))[0];

Guid id1 = (Guid)face1.FaceId;
Guid id2 = (Guid)face2.FaceId;

VerifyResult result = await client.Face.VerifyFaceToFaceAsync(id1, id2);
double confidence = result.Confidence;
bool identical = result.IsIdentical;
```

Creating and Training a Person Group

```
string id = "myfamily"; // Lowercase characters only!
await client.PersonGroup.CreateAsync(id, "My Family");

Person jeff = await client.PersonGroupPerson.CreateAsync(id, "Jeff");
await client.PersonGroupPerson.AddFaceFromUrlAsync(id, jeff.PersonId, "IMAGE_URL");
await client.PersonGroupPerson.AddFaceFromUrlAsync(id, jeff.PersonId, "IMAGE_URL");

Person lori = await client.PersonGroupPerson.CreateAsync(id, "Lori");
await client.PersonGroupPerson.AddFaceFromUrlAsync(id, lori.PersonId, "IMAGE_URL");
await client.PersonGroupPerson.AddFaceFromUrlAsync(id, lori.PersonId, "IMAGE_URL");

await client.PersonGroup.TrainAsync(id);
```

Identifying People in a Photo

```
string id = "myfamily"; // Lowercase characters only!
IList<DetectedFace> faces = await client.Face.DetectWithUrlAsync("IMAGE_URL");
IList<Guid> ids = faces.Where(face => face.FaceId != null)
    .Select(face => face.FaceId).Cast<Guid>().ToList();
IList<IdentifyResult> results = await client.Face.IdentifyAsync(ids, id);

foreach (var result in results)
{
    if (result.Candidates.Count > 0)
    {
        Person person =
            await client.PersonGroupPerson.GetAsync(id, result.Candidates[0].PersonId);
        string name = person.Name;
        double confidence = result.Candidates[0].Confidence;
    }
}
```

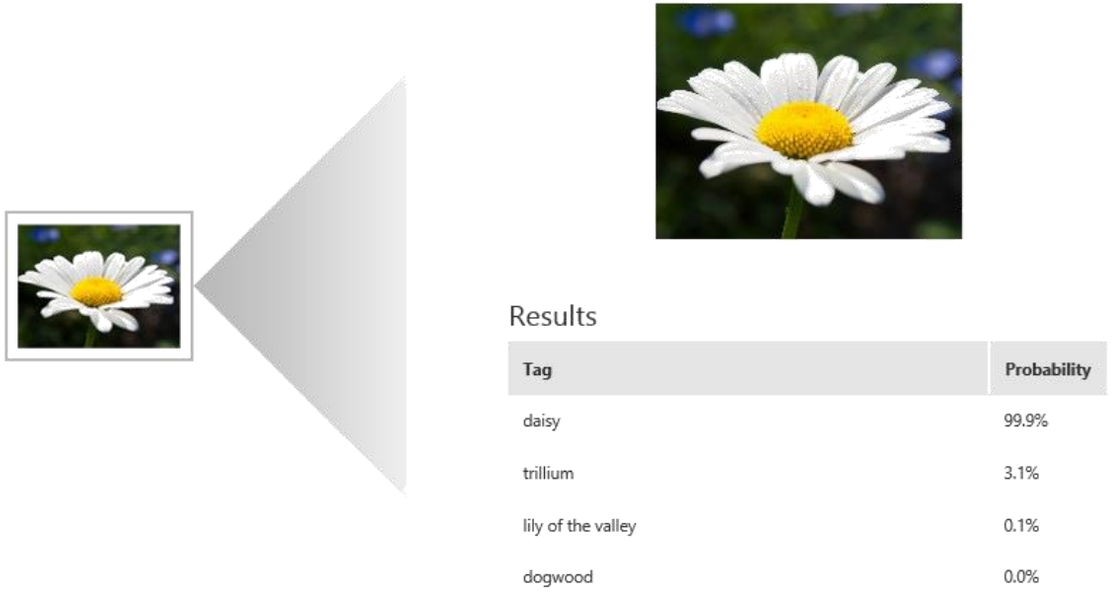
Demo

Face API

Custom Vision Service

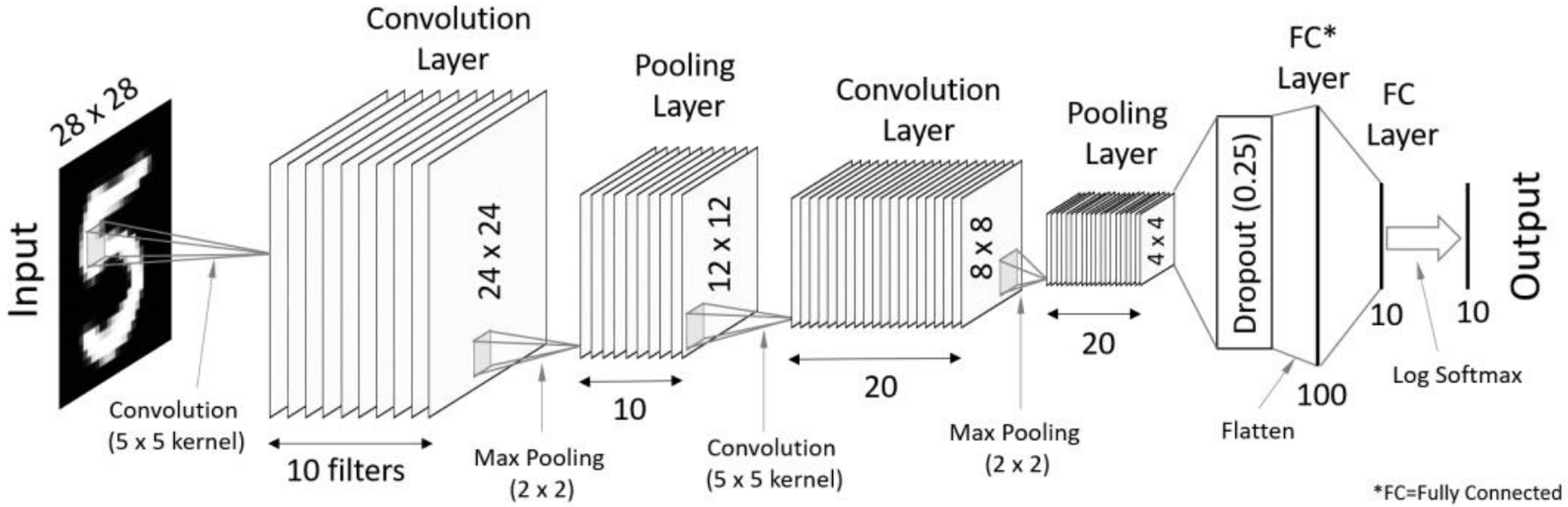
- Build image-classification models backed by convolutional neural networks
  - Get acceptable accuracy with as few as 50 to 100 training images
- Build intelligent apps that invoke models using API
- Optionally export to iOS (CoreML) or Android (TensorFlow) and run locally

Computer Vision as a Service



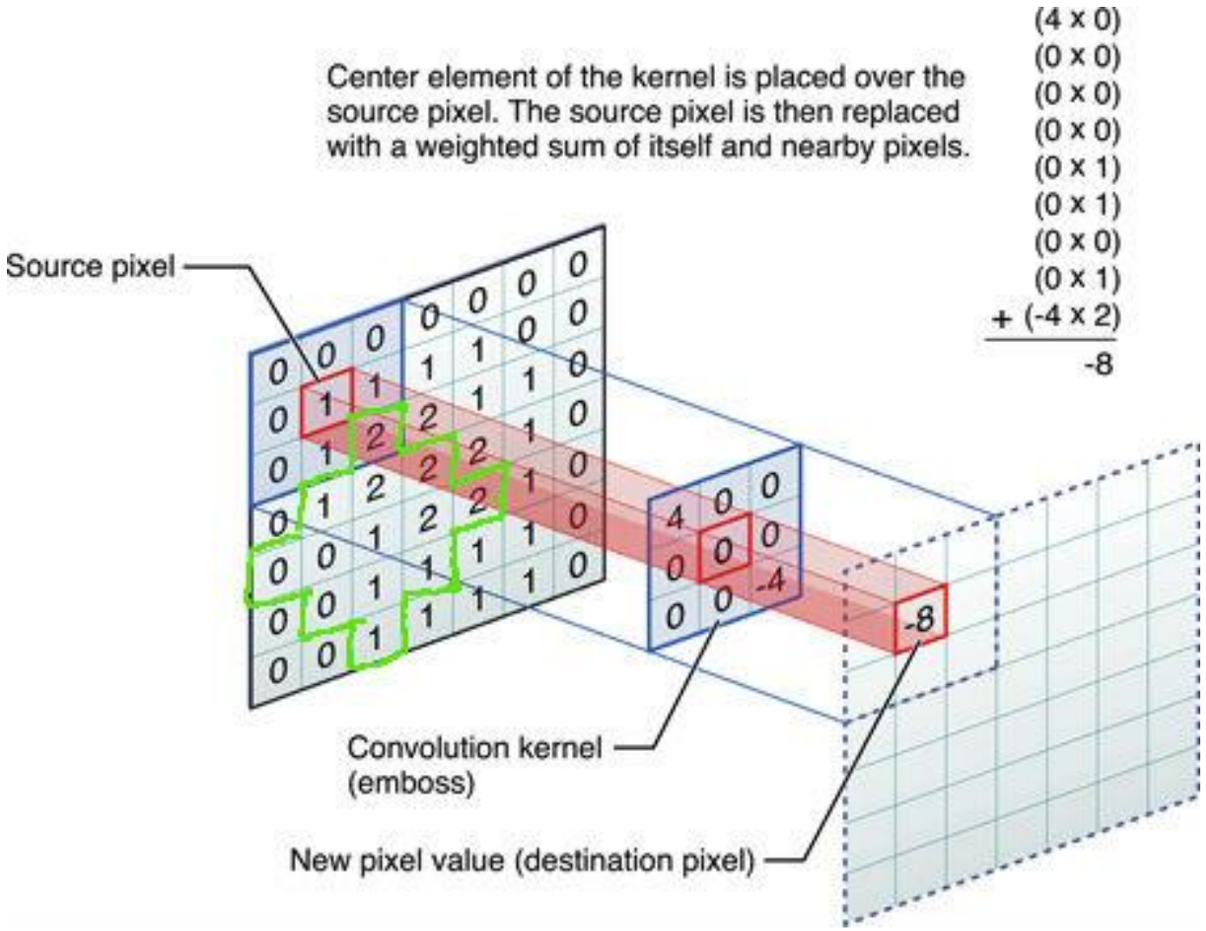
Convolutional Neural Networks

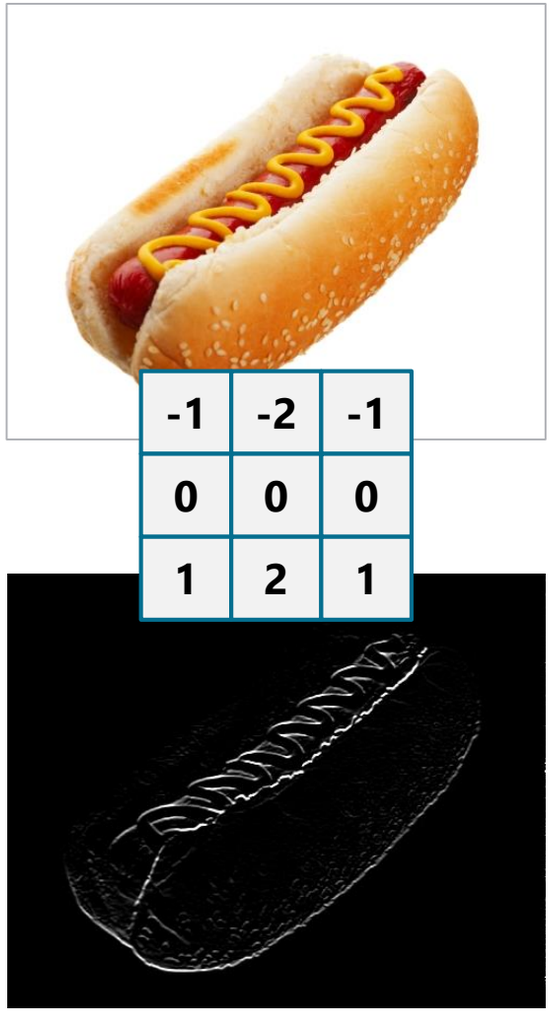
- Excel at tasks involving computer vision and sequence processing
- Use convolution layers and convolution kernels to create feature maps
- Use pooling layers to subsample feature maps and generalize features



Source: https://codetolight.wordpress.com/2017/11/29/getting-started-with-pytorch-for-deep-learning-part-3-neural-network-basics/

Feature Mapping with Convolution Kernels





Source: https://stats.stackexchange.com/questions/235032/any-use-of-non-rectangular-shaped-kernels-in-convolutional-neural-networks-espe

Evolution of CNNs

2015 ResNet (ILSVRC'15) 3.57

Year	Codename	Error (percent)	99.9% Conf Int
2014	**GoogLeNet**	**6.66**	**6.40 - 6.92**
2014	VGG	7.32	7.05 - 7.60
2014	MSRA	8.06	7.78 - 8.34
2014	AHoward	8.11	7.83 - 8.39
2014	DeeperVision	9.51	9.21 - 9.82
2013	Clarifai†	11.20	10.87 - 11.53
2014	CASIAWS†	11.36	11.03 - 11.69
2014	Trimps†	11.46	11.13 - 11.80
2014	Adobe†	11.58	11.25 - 11.91
2013	**Clarifai**	**11.74**	**11.41 - 12.08**
2013	NUS	12.95	12.60 - 13.30
2013	ZF	13.51	13.14 - 13.87
2013	AHoward	13.55	13.20 - 13.91
2013	OverFeat	14.18	13.83 - 14.54
2014	Orange†	14.80	14.43 - 15.17
2012	SuperVision†	15.32	14.94 - 15.69
2012	**SuperVision**	**16.42**	**16.04 - 16.80**
2012	ISI	26.17	25.71 - 26.65
2012	VGG	26.98	26.53 - 27.43
2012	XRCE	27.06	26.60 - 27.52
2012	UvA	29.58	29.09 - 30.04

Microsoft ResNet, a 152 layers network

GoogLeNet, 22 layers network

U. of Toronto, SuperVision, a 7 layers network

human error is around *5.1%* on a subset

Calling the Custom Vision Service

```
CustomVisionPredictionClient client = new CustomVisionPredictionClient()
{
    ApiKey = "PREDICTION_KEY",
    Endpoint = "PREDICTION_URL"
};

ImagePrediction result = await client.ClassifyImageUrlAsync(Guid.Parse("PROJECT_ID"),
    "ITERATION_NAME", new ImageUrl("IMAGE_URL"));

foreach (var prediction in result.Predictions)
{
    string tag = prediction.TagName;
    double probability = prediction.Probability;
}
```

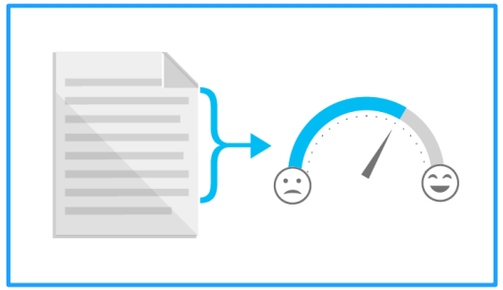
Demo

Custom Vision Service

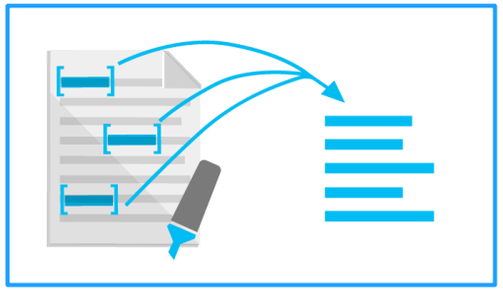
Text Analytics API

- Measure sentiment in Twitter feeds and other textual data, extract topics and key phrases, and detect language(s) used

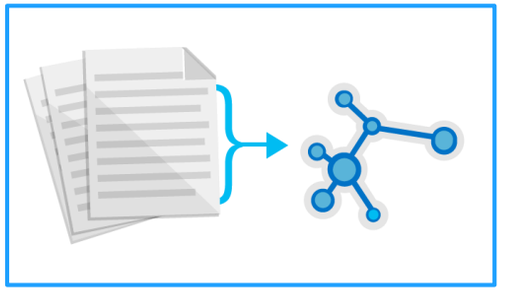
Sentiment Analysis



Key Phrase Extraction



Topic Detection



Language Detection

"Thanks to Text Analytics…we are able to incorporate guest sentiment into our actionable guest feedback platform that delivers a comprehensive view of guest satisfaction and server performance."

— Al Pappa, Head of Business Intelligence, Ziosk

Analyzing Text for Sentiment

```
TextAnalyticsClient client = new TextAnalyticsClient(
    new ApiKeyServiceClientCredentials("SUBSCRIPTION_KEY"),
    new System.Net.Http.DelegatingHandler[] { }
);

client.Endpoint = "TEXT_ANALYTICS_SERVICE_URL";
string text = "I am mostly happy today, but coding sometimes makes me angry";

SentimentResult result = await client.SentimentAsync(text);
double score = (double)result.Score; // 0.06383291
```

Analyzing Text in Batches

```
List<MultiLanguageInput> input = new List<MultiLanguageInput>();
input.Add(new MultiLanguageInput("1", "I am mostly happy today"));
input.Add(new MultiLanguageInput("2", "Coding sometimes makes me angry"));
MultiLanguageBatchInput batch = new MultiLanguageBatchInput(input);

SentimentBatchResult result = await client.SentimentBatchAsync(batch);

foreach (var document in result.Documents)
{
    double score = (double)document.Score;
}
```

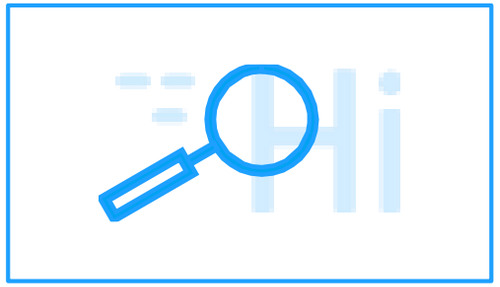
Demo

Text Analytics API

Translator API

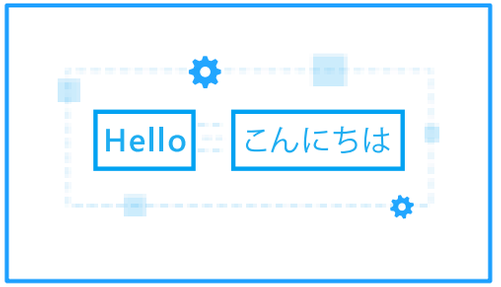
- Translate text between more than 60 languages, break text into sentences, automatically mark or delete profanity, and more

Detect Language



Translate Text

Transliterate



Translate Offline

"By using the Microsoft Translator API to automate SQL Server data translation into English, we are able to present senior leaders with universally usable data that supports better informed decisions."

— Mark Hutcheson, IT Specialist, U.S. Army

Converting English to French

```
HttpClient client = new HttpClient();
client.DefaultRequestHeaders.Add("Ocp-Apim-Subscription-Key", "TRANSLATOR_KEY");
string body = JsonConvert.SerializeObject(new object[] { new { Text = "Hello world" } });
StringContent content = new StringContent(body, Encoding.UTF8, "application/json");
string url = "https://api.cognitive.microsofttranslator.com/translate?api-version=3.0&to=fr";
HttpResponseMessage response = await client.PostAsync(url, content);

if (response.IsSuccessStatusCode)
{
    dynamic results = JsonConvert.DeserializeObject(await response.Content.ReadAsStringAsync());

    foreach (var result in results)
    {
        foreach (var translation in result.translations)
        {
            string language = translation.to; // "fr"
            string text = translation.text; // "Salut tout le monde"
        }
    }
}
```

Speech-to-Text and Text-to-Speech APIs

- Convert speech to text in 20 languages
  - Input WAV audio and receive JSON text
  - REST API limited to 15-second snippets
  - Client SDKs add support for longer clips and also feature LUIS integration
- Convert text to speech in 20 languages
  - Input text (SSML) and receive MP3 audio in body of response
- Audio uses PCM Mono 16000 format

Converting Text to Speech

```
POST /synthesize HTTP/1.1
Host: speech.platform.bing.com
Authorization: Bearer ••••••••••••••••••••••••••••••••
X-Microsoft-OutputFormat: audio-16khz-64kbitrate-mono-mp3
Content-Type: application/ssml+xml
Content-Length: •••

<speak version='1.0' xml:lang='en-US'>
  <voice xml:lang='en-US'
    xml:gender='Female'
    name='Microsoft Server Speech Text to Speech Voice (en-US, ZiraRUS)'>
    Convert this text into audio for me
  </voice>
</speak>
```

Speech Translation API

- Uses WebSockets to translate speech in real time
- Uses deep neural networks to translate speech, remove disfluencies, and identify profanity
- Uses silence detection to intelligently frame "utterances"
- Supports WAV and MP3 formats

Demo

Speech Translation API

Get the Code

Download the code from GitHub:

https://github.com/jeffprosise/Cognitive-Services

Jeff Prosise
https://www.wintellect.com
jeffpro@wintellect.com
@jprosise