Министерство образования Республики Беларусь

Учреждение образования

"Белорусский Государственный университет информатики

и радиоэлектроники"

Лабораторная работа

по учебной дисциплине “Модели и методы обработки больших объемов данных”

|  |  |
| --- | --- |
| Выполнил: | Студент гр. 956241 Дубовик Н.О. |
|  |  |
| Проверил: | Стержанов М.В. |

Минск 2019

Целью данной лабораторной работы было создать приложение, способное собрать датасеты фотографий с людьми и определить эмоции людей на этих фотографиях.

Для достижения поставленной цели был использован Azure Cognitive Service и, в частности, такие его возможности как Bing Image Search API и Face API.

Само приложение написано на .NET Core 2.0.

Перед началом работы необходимо заполнить следующие переменные значения из Azure Portal:

static string SubscriptionKeyFace = "52bd7b2e2b704f98ba875709424f6d6e";

static string SubscriptionKeySearch = "670fd9da678c45bc863d686d402b32cb";

static string Endpoint = "https://mimobodface.cognitiveservices.azure.com/";

Входной точкой в приложение является

static void Main(string[] args)

{

GetImages();

var client = Authenticate(Endpoint, SubscriptionKeyFace);

// Analyze an image to get features and other properties.

DetectFaceExtract(client).Wait();

}

Метод GetImages использует Bing Image Search API и static string searchTerm = "" для того, чтобы скачать необходимые картинки и сохранить их на локальную файловую систему. Само API не возвращает картинки, у которых точность ответа на заданный запрос составляет меньше, чем 0.95. При этом можно задать сколько именно необходимо найти картинок с помощью аргумента count.

public static void GetImages()

{

var client = new ImageSearchClient(new Microsoft.Azure.CognitiveServices.Search.ImageSearch.ApiKeyServiceClientCredentials(SubscriptionKeySearch));

// make the search request to the Bing Image API, and get the results

var imageResults = client.Images.SearchAsync(query: searchTerm, count: 10).Result; //search query

if (imageResults != null)

{

int i = 0;

foreach (var image in imageResults.Value)

{

using (WebClient webClient = new WebClient())

{

webClient.DownloadFileAsync(new Uri($"{image.ContentUrl}"), $@"Path\image\image{i}.{image.EncodingFormat}");

i++;

}

}

}

}

Данный процесс можно представить с помощью блок-схемы, рисунок 1.

A screenshot of a social media post

Description automatically generated

Рисунок 1 – Алгоритм получения изображений

После того, как все изображения скачаются, можно приступать к их анализу с помощью метода DetectFaceExtract. Данный метод пробегается по всем файлам в указанной папке, делает их них stream с помощью метода GetImageAsByteArray и его передает в Face API, указывая, что именно необходимо найти на фотографии, в данном случае эмоции. Затем по ответу от сервиса он определяет какая именно эмоция у каждого человека, найденного на фотографии.

Программа определяет эмоции человека из следующего списка:

1. злость;
2. пренебрежение;
3. отвращение, раздражение;
4. страх;
5. счастье;
6. нейтральное;
7. печаль;
8. удивление.

public static async Task DetectFaceExtract(IFaceClient client)

{

Console.WriteLine("========DETECT FACES========");

Console.WriteLine();

// Create a list of images

string[] filePaths = Directory.GetFiles(@"Path\image\", "\*", SearchOption.AllDirectories);

foreach (var filePath in filePaths)

{

try

{

byte[] byteData = GetImageAsByteArray(filePath);

using (var stream = new MemoryStream(byteData))

{

IList<DetectedFace> detectedFaces;

// Detect faces with all attributes from image url.

detectedFaces = await client.Face.DetectWithStreamAsync(stream,

returnFaceAttributes: new List<FaceAttributeType>

{

FaceAttributeType.Emotion

});

Console.WriteLine($"{detectedFaces.Count} face(s) detected from image `{filePath}`.");

foreach (var face in detectedFaces)

{

string emotionType = string.Empty;

double emotionValue = 0.0;

Emotion emotion = face.FaceAttributes.Emotion;

if (emotion.Anger > emotionValue) { emotionValue = emotion.Anger; emotionType = "Anger"; }

if (emotion.Contempt > emotionValue) { emotionValue = emotion.Contempt; emotionType = "Contempt"; }

if (emotion.Disgust > emotionValue) { emotionValue = emotion.Disgust; emotionType = "Disgust"; }

if (emotion.Fear > emotionValue) { emotionValue = emotion.Fear; emotionType = "Fear"; }

if (emotion.Happiness > emotionValue) { emotionValue = emotion.Happiness; emotionType = "Happiness"; }

if (emotion.Neutral > emotionValue) { emotionValue = emotion.Neutral; emotionType = "Neutral"; }

if (emotion.Sadness > emotionValue) { emotionValue = emotion.Sadness; emotionType = "Sadness"; }

if (emotion.Surprise > emotionValue) { emotionType = "Surprise"; }

Console.WriteLine($"Emotion : {emotionType}");

}

}

}

catch (Exception)

{

Console.WriteLine("--------------------------------------");

}

}

}

static byte[] GetImageAsByteArray(string imageFilePath)

{

FileStream fileStream = new FileStream(imageFilePath, FileMode.Open, FileAccess.Read);

BinaryReader binaryReader = new BinaryReader(fileStream);

return binaryReader.ReadBytes((int)fileStream.Length);

}

Само распознавание лиц и их эмоций происходит на основании 27 точек, изображенных на рисунке 2.

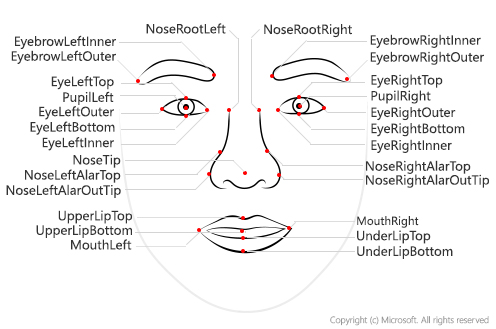


Рисунок 2 – Точки лица для распознавания

Таким образом, в ходе лабораторной работы было разработано приложение, которое имеет возможность получить необходимые изображения из интернета, а затем провести их анализ на эмоции человека на изображениях.