Rekognizing patterns

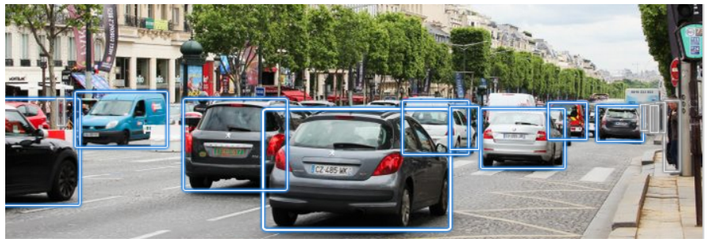
INTRODUCTION TO AWS BOTO IN PYTHON

Maksim Pecherskiy
Instructor

Rekognition

So what is Rekognition anyway?

Detecting Objects in an image



Transportation	99.6 %
Car	99.6 %
Automobile	99.6 %
Person	90.5 %
Human	90.5 %

Extracting Text from Images

Results — US English only

PLEASE
DON'T | FEED | FINGERS
TO | THE | ANIMALS

Request

Response

Why not build our own?

Use Rekognition if:

- Quick but good
- Keep code simple
- Recognize many things

Build a model if:

- Custom requirements
- Security implications
- Large volumes

I am not a computer vision expert

Text in image

Rekognition automatically detects and extracts text in your images. Learn More

Choose a sample image

Use your own image

Image must be .jpeg or .png format and no larger than 5MB. Your image isn't stored.

Upload or drag and drop

Use image URL Go

Done with the demo?

Learn more

Results

US English only

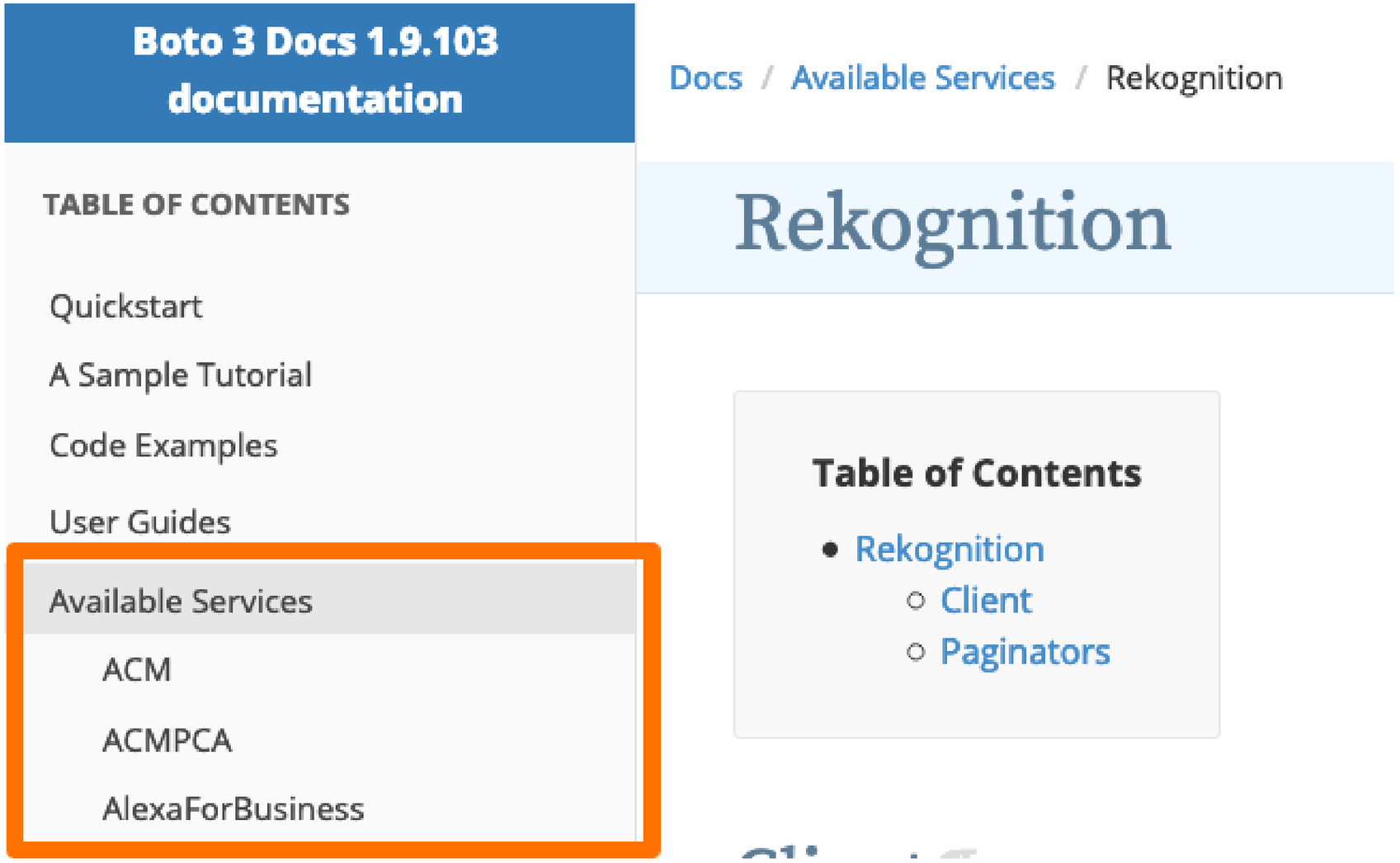
| IT'S |
| MONDAY |
| but | keep |
| Smiling |

Request


```
{
   "Image": {
      "S3Object": {
         "Bucket": "console-sample-images",
         "Name": "coffee_monday.jpg"
      }
   }
}
```


Response


```
{
   "TextDetections": [
      {
         "DetectedText": "IT'S",
         "Type": "LINE",
         "Id": 0,
         "Confidence": 95.83811950683594,
         "Geometry": {
```


Boto 3 Docs 1.9.103 documentation
TABLE OF CONTENTS
Quickstart
A Sample Tutorial
Code Examples
User Guides
Available Services
ACM
ACMPCA
AlexaForBusiness
Docs / Available Services / Rekognition
Rekognition
Table of Contents
Rekognition
Client
Paginators

Client

class **`Rekognition.Client`**

A low-level client representing Amazon Rekognition:

```
import boto3

client = boto3.client('rekognition')
```

These are the available methods:

- `can_paginate()`
- `compare_faces()`
- `create_collection()`
- `create_stream_processor()`

Client

class **Rekognition.Client**

A low-level client representing Amazon Rekognition:

```
import boto3

client = boto3.client('rekognition')
```

These are the available methods:

- **can_paginate()**
- **compare_faces()**
- **create_collection()**
- **create_stream_processor()**

Upload an image to S3

Initialize S3 Client

```
s3 = boto3.client(
  's3', region_name='us-east-1',
  aws_access_key_id=AWS_KEY_ID, aws_secret_access_key=AWS_SECRET
)
```

Upload a file

```
s3.upload_file(
  Filename='report.jpg', Key='report.jpg',
  Bucket='datacamp-img')
```

Object detection

INITIATE THE CLIENT

```
rekog = boto3.client(
  'rekognition',
  region_name='us-east-1',
  aws_access_key_id=AWS_KEY_ID,
  aws_secret_access_key=AWS_SECRET)
```

Object detection

DETECT!

```
response = rekog.detect_labels(
  Image={'S3Object': {
           'Bucket': 'datacamp-img',
           'Name': 'report.jpg'
         },
    MaxLabels=10,
    MinConfidence=95
)
```

```
{
    "LabelModelVersion":"2.0",
    "Labels":[
          {
          "Confidence":99.96617126464844,
          "Instances":[
             {
                "BoundingBox":{
                   "Height":0.2375246733427047,
                   "Left":0.5014267563819885,
                   "Top":0.5496936440467834,
                   "Width":0.12512922286987305
                },
                "Confidence":99.96617126464844
             }
          ],
          "Name":"Bicycle",
          "Parents":[
             { "Name":"Vehicle" },
             { "Name":"Transportation" }
          ]
       },
       {
          "Confidence":99.96617126464844,
          "Instances":[
```


Label Name

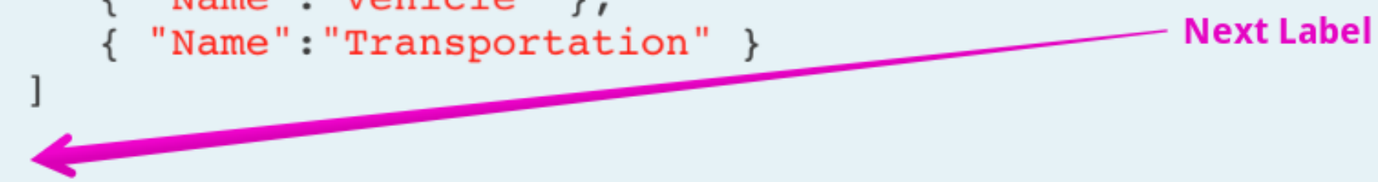
```
{
    "LabelModelVersion":"2.0",
    "Labels":[
          {
          "Confidence":99.96617126464844,
          "Instances":[
             {
                "BoundingBox":{
                   "Height":0.2375246733427047,
                   "Left":0.5014267563819885,
                   "Top":0.5496936440467834,
                   "Width":0.12512922286987305
                },
                "Confidence":99.96617126464844
             }
          ],
          "Name":"Bicycle",
          "Parents":[
             { "Name":"Vehicle" },
             { "Name":"Transportation" }
          ]
       },
       {
          "Confidence":99.96617126464844,
          "Instances":[
```

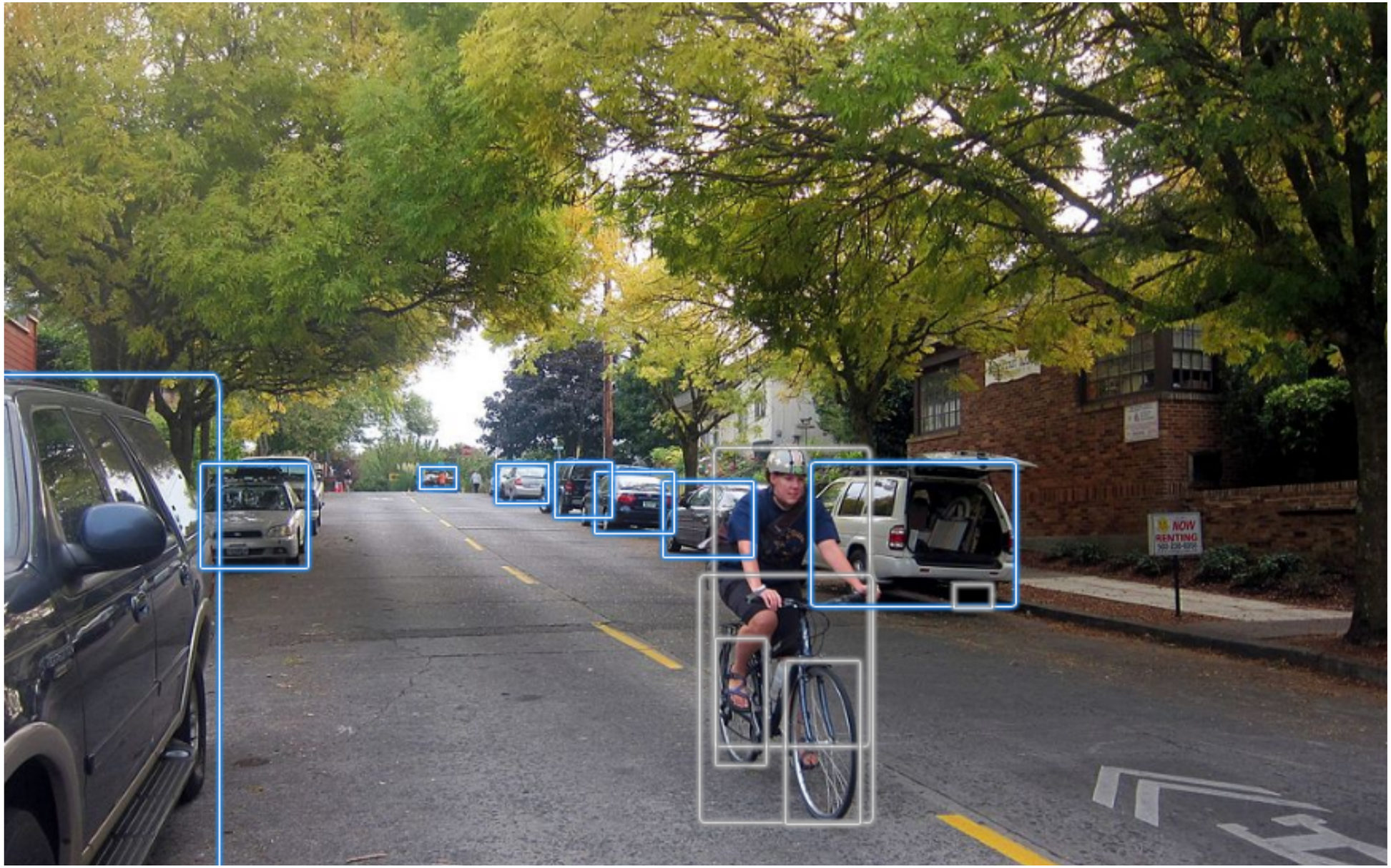

Label Exists Confidence %

```
{
   "LabelModelVersion":"2.0",
   "Labels":[
         {
         "Confidence":99.96617126464844,
         "Instances":[
            {
               "BoundingBox":{
                  "Height":0.2375246733427047,
                  "Left":0.5014267563819885,
                  "Top":0.5496936440467834,
                  "Width":0.12512922286987305
               },
               "Confidence":99.96617126464844
            }
         ],
         "Name":"Bicycle",
         "Parents":[
            { "Name":"Vehicle" },
            { "Name":"Transportation" }
         ]
      },
      {
         "Confidence":99.96617126464844,
         "Instances":[
```


← List of Instances

```
{
    "LabelModelVersion":"2.0",
    "Labels":[
          {
          "Confidence":99.96617126464844,
          "Instances":[
             {
                "BoundingBox":{
                   "Height":0.2375246733427047,
                   "Left":0.5014267563819885,
                   "Top":0.5496936440467834,
                   "Width":0.12512922286987305
                },
                "Confidence":99.96617126464844
             }
          ],
          "Name":"Bicycle",
          "Parents":[
             { "Name":"Vehicle" },
             { "Name":"Transportation" }
          ]
       },
       {
          "Confidence":99.96617126464844,
          "Instances":[
```


NOW
RENTING

Text detection

PERFORM DETECTION

```
response = rekog.detect_text(
  Image={'S3Object':
          {
            'Bucket': 'datacamp-img',
            'Name': 'report.jpg'
          }
        }
)
```

```
{
    "TextDetections":[
        {
            "DetectedText":"STREET PROHIBITED",
            "Type":"LINE",
            "Id":0,
            "Confidence":99.88934326171875,
            "Geometry":{
                "BoundingBox":{
                    "Width":0.342500239610672,
                    "Height":0.0641483291983604 4,
                    "Left":0.4973224401473999,
                    "Top":0.2827266752719879
                },
                "Polygon":[
                    {"X":0.4675000011920929, "Y":0.4453125},
                    {"X":0.8650000095367432, "Y":0.4453125},
                    {"X":0.8650000095367432, "Y":0.512499988079071},
                    {"X":0.4675000011920929, "Y":0.5140625238418579},
                ]
            }
        },
        {
            "DetectedText":"STREET",
            "Type":"WORD",
            "Id":14,
```


Line Detection

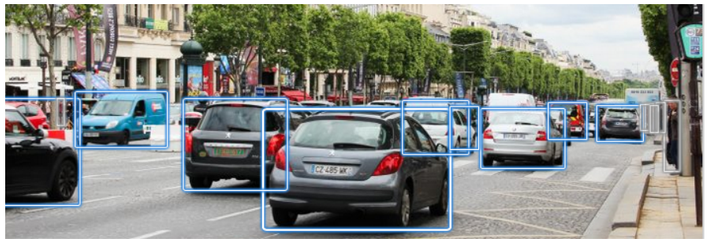
```
            "Polygon":[
                {"X":0.4675000011920929, "Y":0.4453125},
                {"X":0.8650000095367432, "Y":0.4453125},
                {"X":0.8650000095367432, "Y":0.512499988079071},
                {"X":0.4675000011920929, "Y":0.5140625238418579},
            ]
        }
    },
    {
        "DetectedText":"STREET",
        "Type":"WORD",
        "Id":14,
        "ParentId":2,
        "Confidence":99.97589111328125,
        "Geometry":{
```



NO
STREET
PARKING
ANY TIME
TOW AWAY
ZONE
ON-STREET
PARKING IS
PROHIBITED
Violators will be
Towed Away at the
Vehicle Owner's
Expense

Summary

- Detect objects in an image
- Count instances
- Learn a new AWS Service



Transportation	99.6 %
Car	99.6 %
Automobile	99.6 %
Person	90.5 %
Human	90.5 %

```
.detect_labels()
```

- Recognize text in an image
- Word vs line detections
- When to build our own

Results — US English only

PLEASE
DON'T | FEED | FINGERS
TO | THE | ANIMALS

Request

Response

```
.detect_text()
```

Let's do some computer vision!

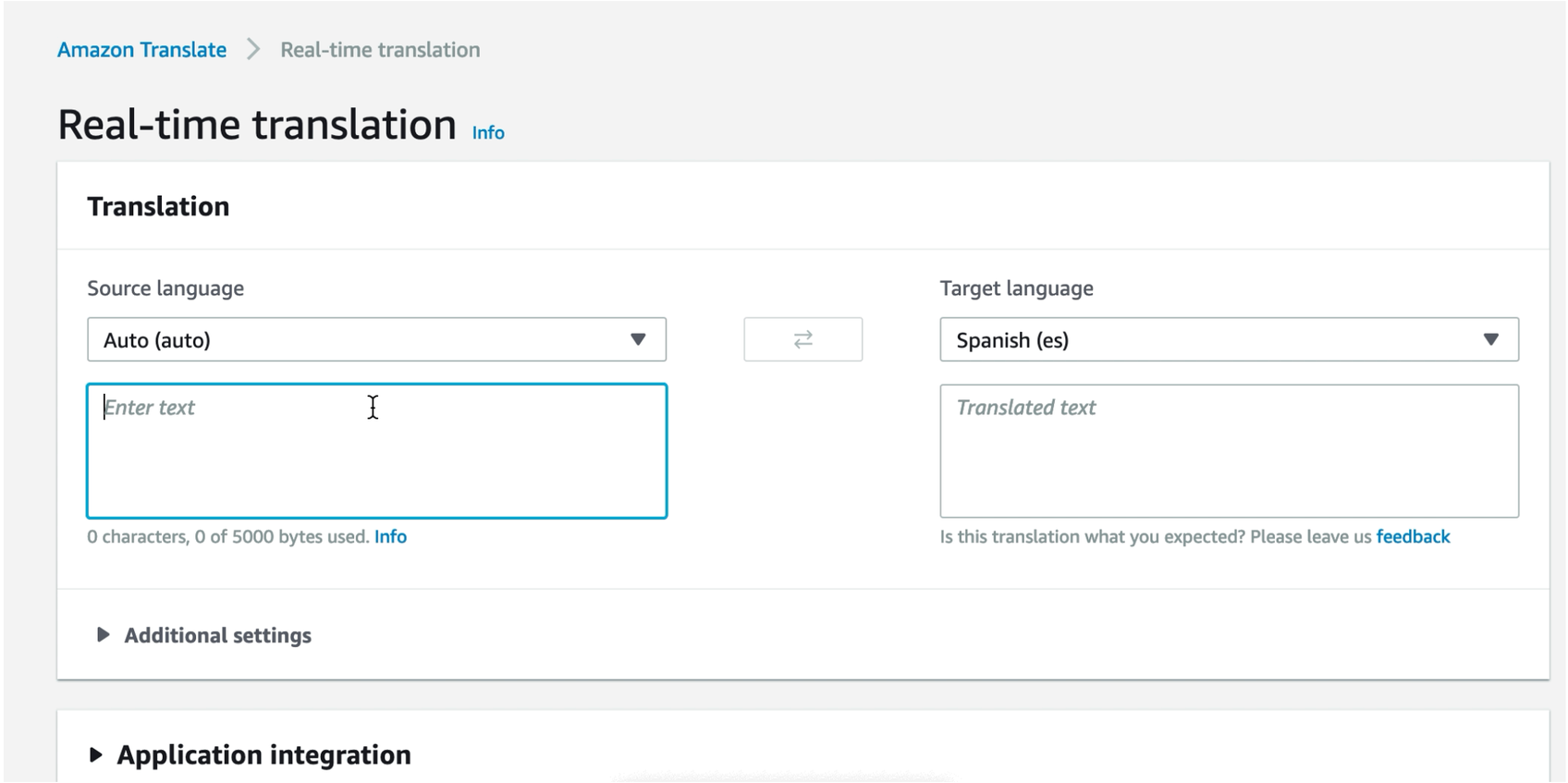
INTRODUCTION TO AWS BOTO IN PYTHON

Comprehending text

INTRODUCTION TO AWS BOTO IN PYTHON

Maksim Pecherskiy
Data engineer

AWS Translate console



Translating text

Initialize client

```
translate = boto3.client('translate',
    region_name='us-east-1',
    aws_access_key_id=AWS_KEY_ID, aws_secret_access_key=AWS_SECRET)
```

Translate text

```
response = translate.translate_text(
    Text='Hello, how are you?',
    SourceLanguageCode='auto',
    TargetLanguageCode='es')
```

Translating text

```
{'TranslatedText': 'Hola, ¿cómo estás?',
 'SourceLanguageCode': 'en',
 'TargetLanguageCode': 'es',
 'ResponseMetadata': {'RequestId': 'e7cdb3a2-91d5-11e9-bf94-7f3fb6911ee5',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'cache-control': 'no-cache',
   'content-type': 'application/x-amz-json-1.1',
   'date': 'Tue, 18 Jun 2019 14:32:31 GMT',
   'x-amzn-requestid': 'e7cdb3a2-91d5-11e9-bf94-7f3fb6911ee5',
   'content-length': '94',
   'connection': 'keep-alive'},
  'RetryAttempts': 0}}
```

Translating text

```
translated_text = translate.translate_text(
    Text='Hello, how are you?',
    SourceLanguageCode='auto',
    TargetLanguageCode='es')['TranslatedText']
```

Detecting language

Initialize boto3 Comprehend client

```
comprehend = boto3.client('comprehend',
    region_name='us-east-1',
    aws_access_key_id=AWS_KEY_ID, aws_secret_access_key=AWS_SECRET)
```

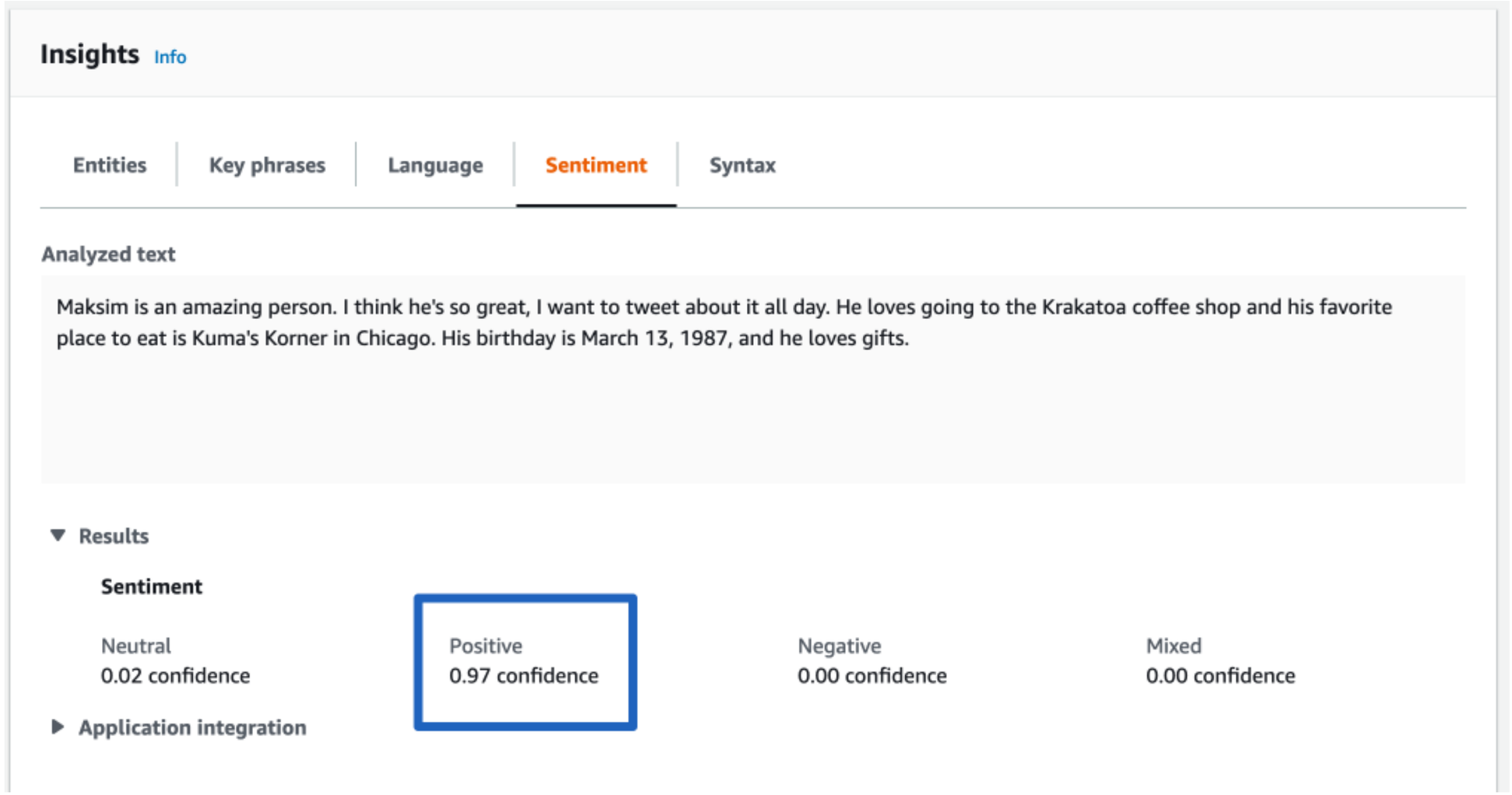
Detect dominant language

```
response = comprehend.detect_dominant_language(
  Text="Hay basura por todas partes a lo largo de la carretera.")
```

Detecting language

```
{'Languages': [{'LanguageCode': 'es', 'Score': 0.9935020208358765}],
 'ResponseMetadata': {'RequestId': '9c895635-9217-11e9-beed-a767c995e190',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'x-amzn-requestid': '9c895635-9217-11e9-beed-a767c995e190',
   'content-type': 'application/x-amz-json-1.1',
   'content-length': '64',
   'date': 'Tue, 18 Jun 2019 22:22:51 GMT'},
  'RetryAttempts': 0}}
```

Understanding sentiment



Understanding sentiment

Detect text sentiment

```
response = comprehend.detect_sentiment(
  Text="DataCamp students are amazing.",
  LanguageCode='en')
```

Understanding sentiment

```
{'Sentiment': 'POSITIVE',
 'SentimentScore': {'Positive': 0.9827685952186584
  'Negative': 0.0006051281234249473,
  'Neutral': 0.012124153785407543,
  'Mixed': 0.004502176772803068},
 'ResponseMetadata': {'RequestId': 'ab001220-91d9-11e9-b992-63ec3d2de699',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'x-amzn-requestid': 'ab001220-91d9-11e9-b992-63ec3d2de699',
   'content-type': 'application/x-amz-json-1.1',
   'content-length': '165',
   'date': 'Tue, 18 Jun 2019 14:59:26 GMT'},
  'RetryAttempts': 0}}
```

Understanding sentiment

```
sentiment = comprehend.detect_sentiment(
  Text='Maksim is amazing.',
  LanguageCode='en')['Sentiment']
```

Review

Initialize client

```
translate = boto3.client('translate',
    region_name='us-east-1',
    aws_access_key_id=AWS_KEY_ID, aws_secret_access_key=AWS_SECRET)
```

Translate text

```
response = translate.translate_text(
    Text='Hello, how are you?',
    SourceLanguageCode='auto',
    TargetLanguageCode='es')
```

Review

Initialize boto3 Comprehend client

```
comprehend = boto3.client('comprehend',
    region_name='us-east-1',
    aws_access_key_id=AWS_KEY_ID, aws_secret_access_key=AWS_SECRET)
```

Detect dominant language

```
response = comprehend.detect_dominant_language(
  Text="Hay basura por todas partes a lo largo de la carretera.")
```

Review

Detect text sentiment

```
response = comprehend.detect_sentiment(
  Text="Maksim is amazing.",
  LanguageCode='en')
```

Let's practice!

INTRODUCTION TO AWS BOTO IN PYTHON

Case Study: Scooting Around!

INTRODUCTION TO AWS BOTO IN PYTHON

Maksim Pecherskiy
Data Engineer

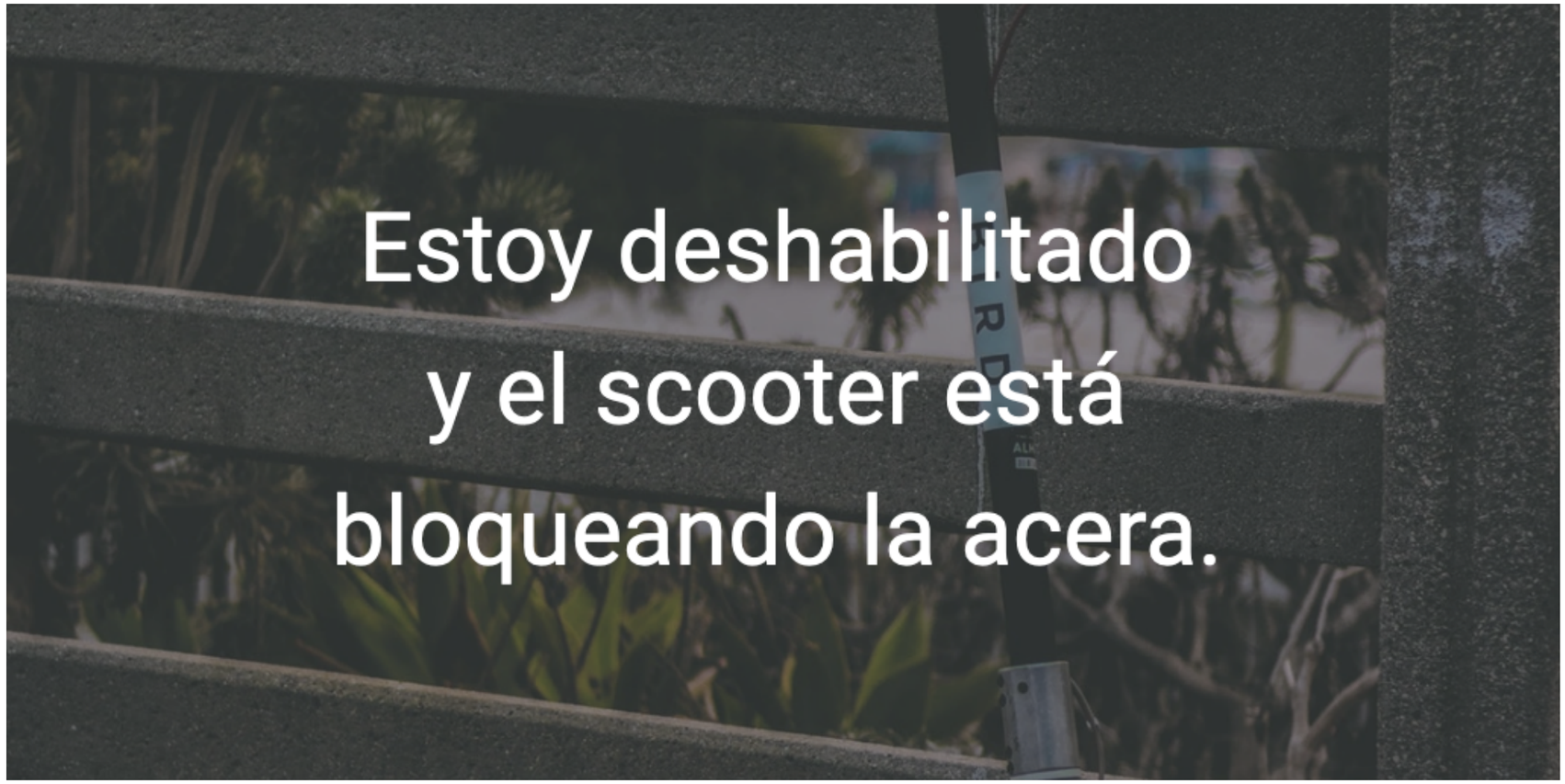
The Quandary

The Quandary

The Data

service_request_id	image	lat	long	public_description
93494	report_113439.jpg	32.723138	-117.128237	Hay un scooter electrico en sidewalk
101502	report_134938839.jpg	32.7077658	-117.1281408	This scooter helped me move a mattress!
101520	report_272819.jpg	32.77605567	-117.100004	There is a scooter blocking the sidewalk
101576	report_3722938.jpg	32.68899358	-117.0584723	I tripped on a stupid scooter

Final Product



Final Product

Final Product

Final Product

Final Product

Initialize boto3 service clients.

Initialize rekognition client

```
rekog = boto3.client('rekognition',
                     region_name='us-east-1',
                     aws_access_key_id=AWS_KEY_ID,
                     aws_secret_access_key=AWS_SECRET)
```

Initialize comprehend client

```
comprehend = boto3.client('comprehend',
                     region_name='us-east-1',
                     aws_access_key_id=AWS_KEY_ID,
                     aws_secret_access_key=AWS_SECRET)
```

Initialize boto3 service clients

Initialize translate client

```
translate = boto3.client('translate',
                         region_name='us-east-1',
                         aws_access_key_id=AWS_KEY_ID,
                         aws_secret_access_key=AWS_SECRET)
```

Translate all descriptions to English

```
for index, row in df.iterrows():
    desc = df.loc[index, 'public_description']
    if desc != '':
        resp = translate_fake.translate_text(
          Text=desc,
          SourceLanguageCode='auto',
          TargetLanguageCode='en')
        df.loc[index, 'public_description'] = resp['TranslatedText']
```

Translate all descriptions to English

service_request_id	image	lat	long	public_description
93494	report_113439.jpg	32.723138	-117.128237	Electric scooter on sidewalk
101502	report_134938839.jpg	32.7077658	-117.1281408	This scooter helped me move a mattress!
101520	report_272819.jpg	32.77605567	-117.100004	There is a scooter blocking the sidewalk
101576	report_3722938.jpg	32.68899358	-117.0584723	I tripped on a stupid scooter

Detect text sentiment

```
for index, row in df.iterrows():
    desc = df.loc[index, 'public_description']
    if desc != '':
        resp = comprehend.detect_sentiment(
          Text=desc,
          LanguageCode='en')
        df.loc[index, 'sentiment'] = resp['Sentiment']
```

Detect text sentiment

service_request_id	image	lat	long	sentiment	public_desc
93494	report_113439.jpg	32.723138	-117.128237	NEGATIVE	Electric scoo sidewalk
101502	report_134938839.jpg	32.7077658	-117.1281408	POSITIVE	This scooter helped me m mattress!
101520	report_272819.jpg	32.77605567	-117.100004	NEGATIVE	There is a sc blocking the sidewalk
101576	report_3722938.jpg	32.68899358	-117.0584723	NEGATIVE	I tripped on

Detect scooter in image

```
df['img_scooter'] = 0
for index, row in df.iterrows():
    image = df.loc[index, 'image']
    response = rekog.detect_labels(
      # Specify the image as an S3Object
      Image={'S3Object': {'Bucket': 'gid-images', 'Name': image}}
    )
    for label in response['Labels']:
        if label['Name'] == 'Scooter':
        df.loc[index, 'img_scooter'] = 1
        break
```

Detect scooter in image

service_request_id	image	img_scooter	sentiment	lat	long
93494	report_113439.jpg	1	NEGATIVE	32.723138	-117.128237
101502	report_134938839.jpg	1	POSITIVE	32.7077658	-117.128140
101520	report_272819.jpg	0	NEGATIVE	32.77605567	-117.100004
101576	report_3722938.jpg	1	NEGATIVE	32.68899358	-117.058472

Final count!

Select only rows where there was a scooter image and that have negative sentiment

```
pickups = df[((df.img_scooter == 1) & (df.sentiment == 'NEGATIVE'))]


num_pickups = len(pickups)
```

332 Scooters!

Let's practice!

INTRODUCTION TO AWS BOTO IN PYTHON

Wrap up

INTRODUCTION TO AWS BOTO IN PYTHON

Maksim Pecherskiy
Data Engineer

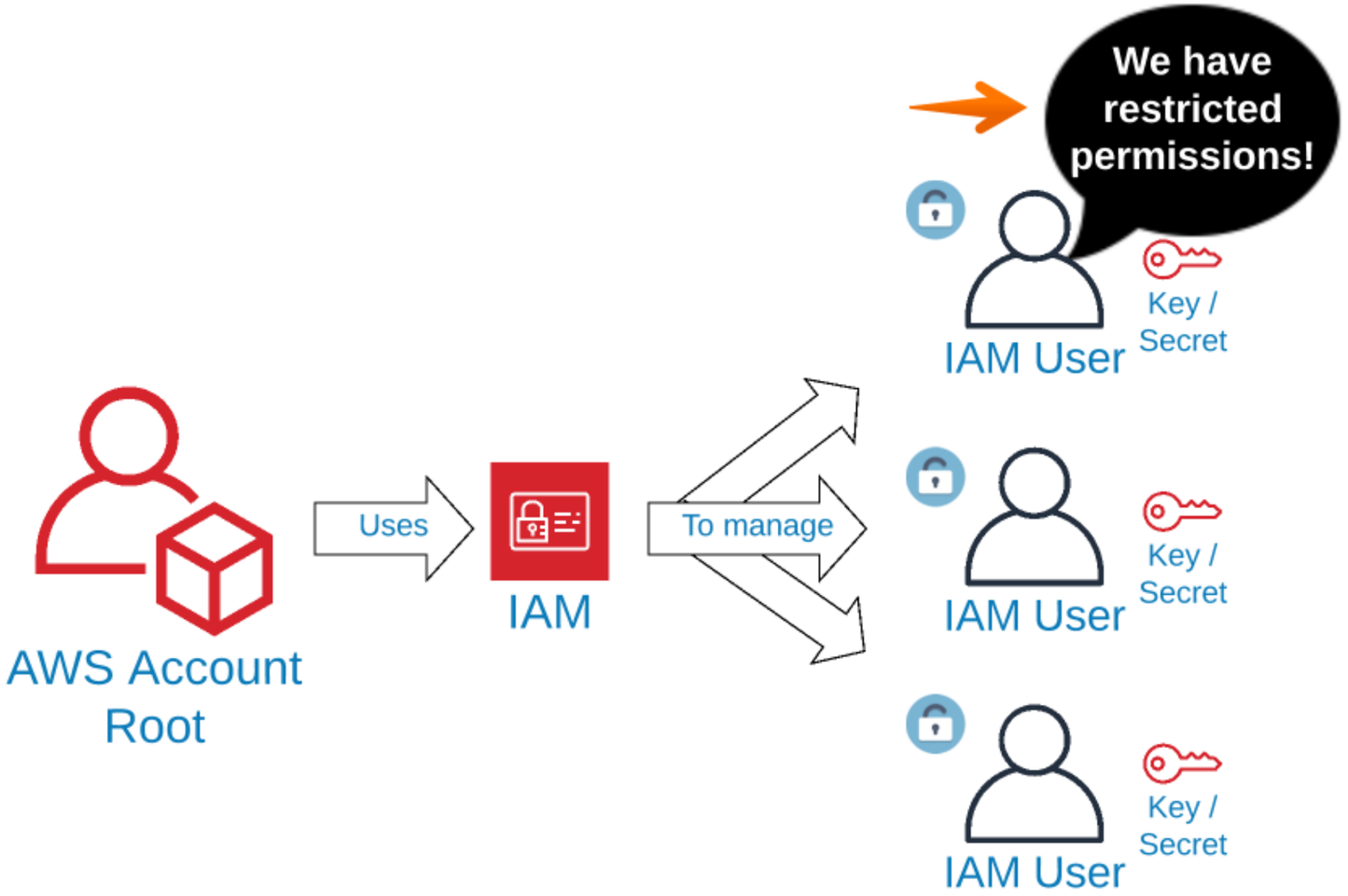
AWS Services

Boto3

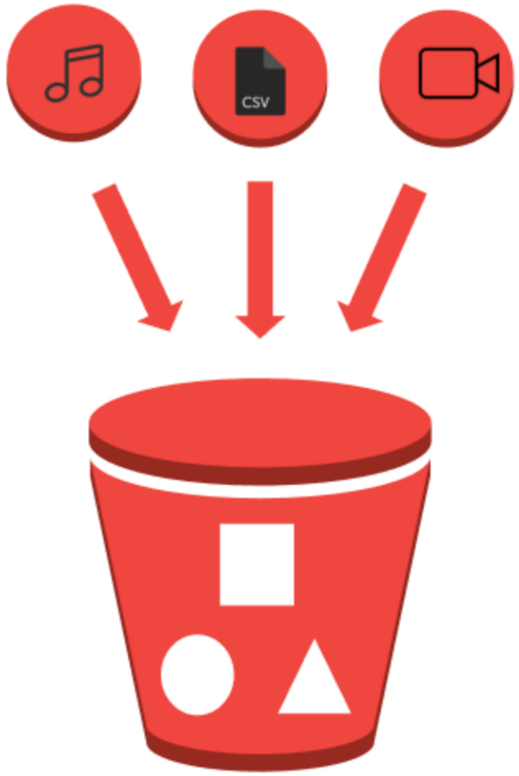
```
s3 = boto3.client('s3',
    region_name='us-east-1',
    aws_access_key_id=AWS_KEY_ID,
    aws_secret_access_key=AWS_SECRET)
```

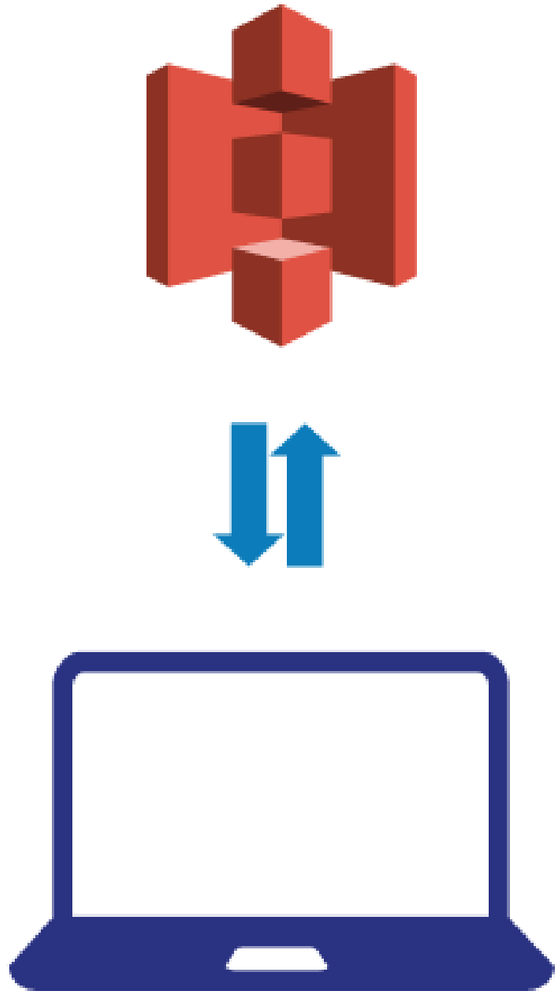
```
buckets = s3.list_buckets()
```

IAM

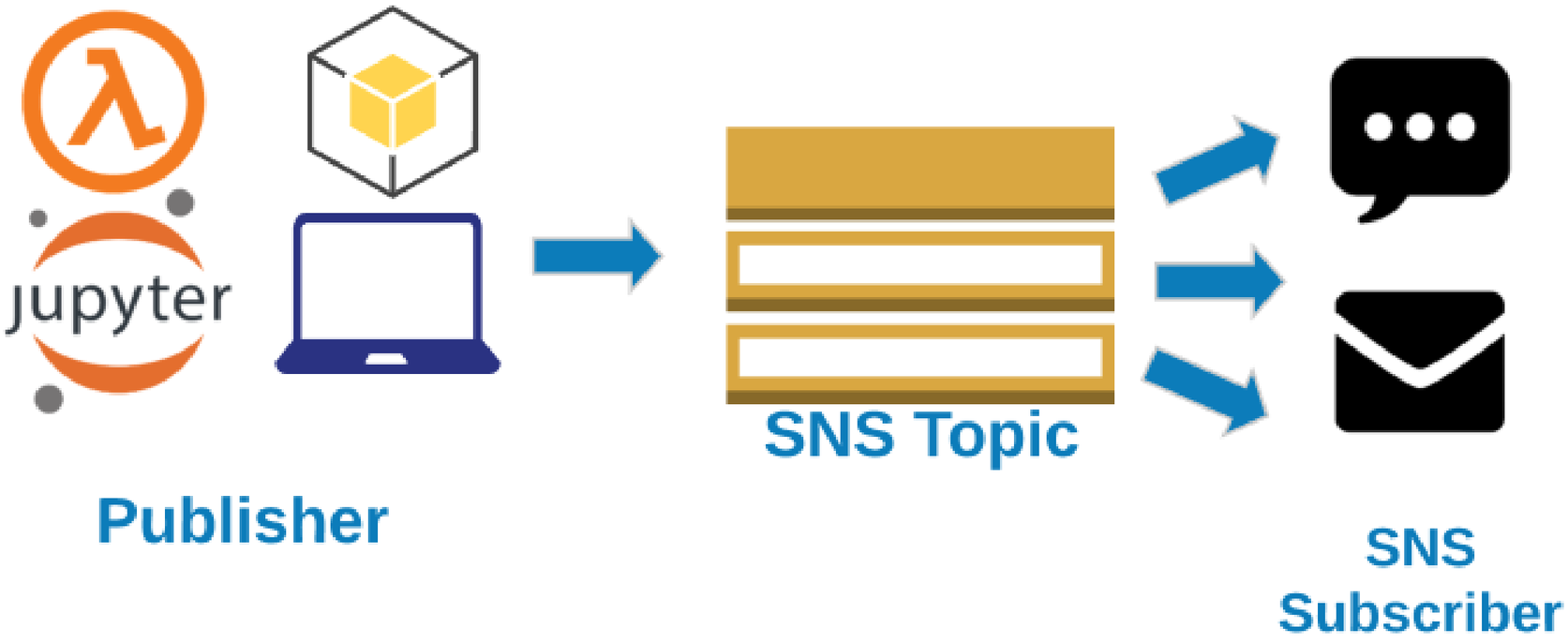

We have restricted permissions!
Key / Secret
IAM User
Uses
IAM
To manage
AWS Account Root
Key / Secret
IAM User
Key / Secret
IAM User

S3

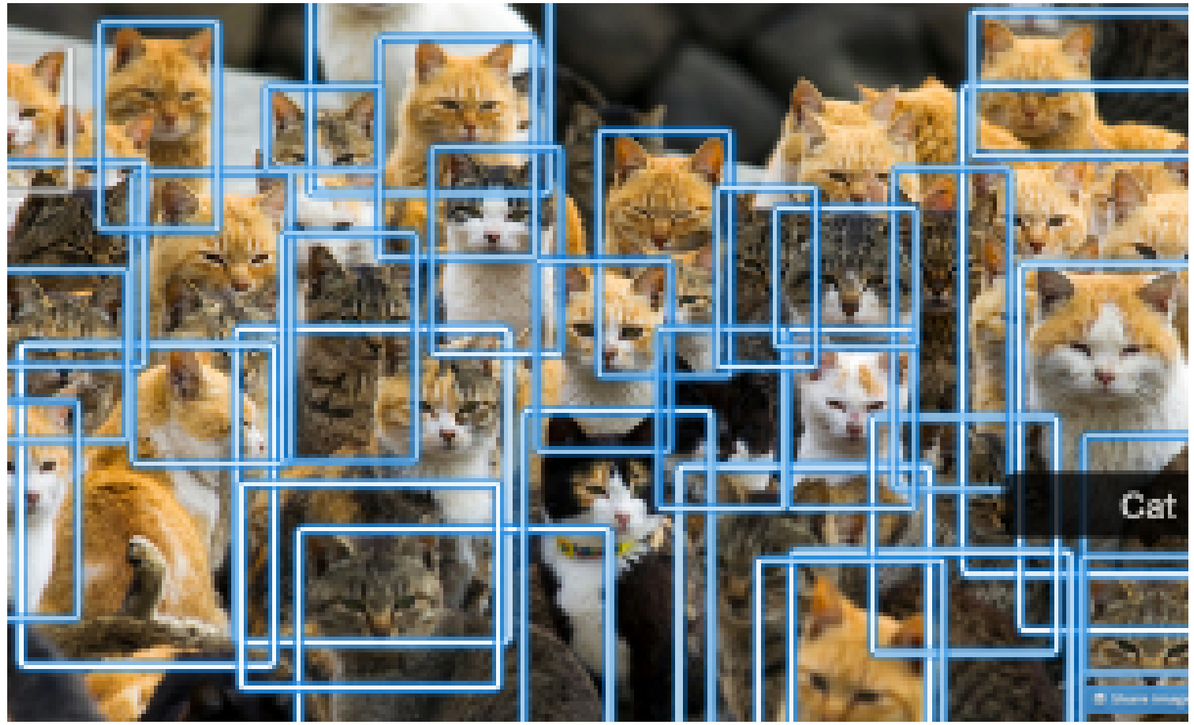




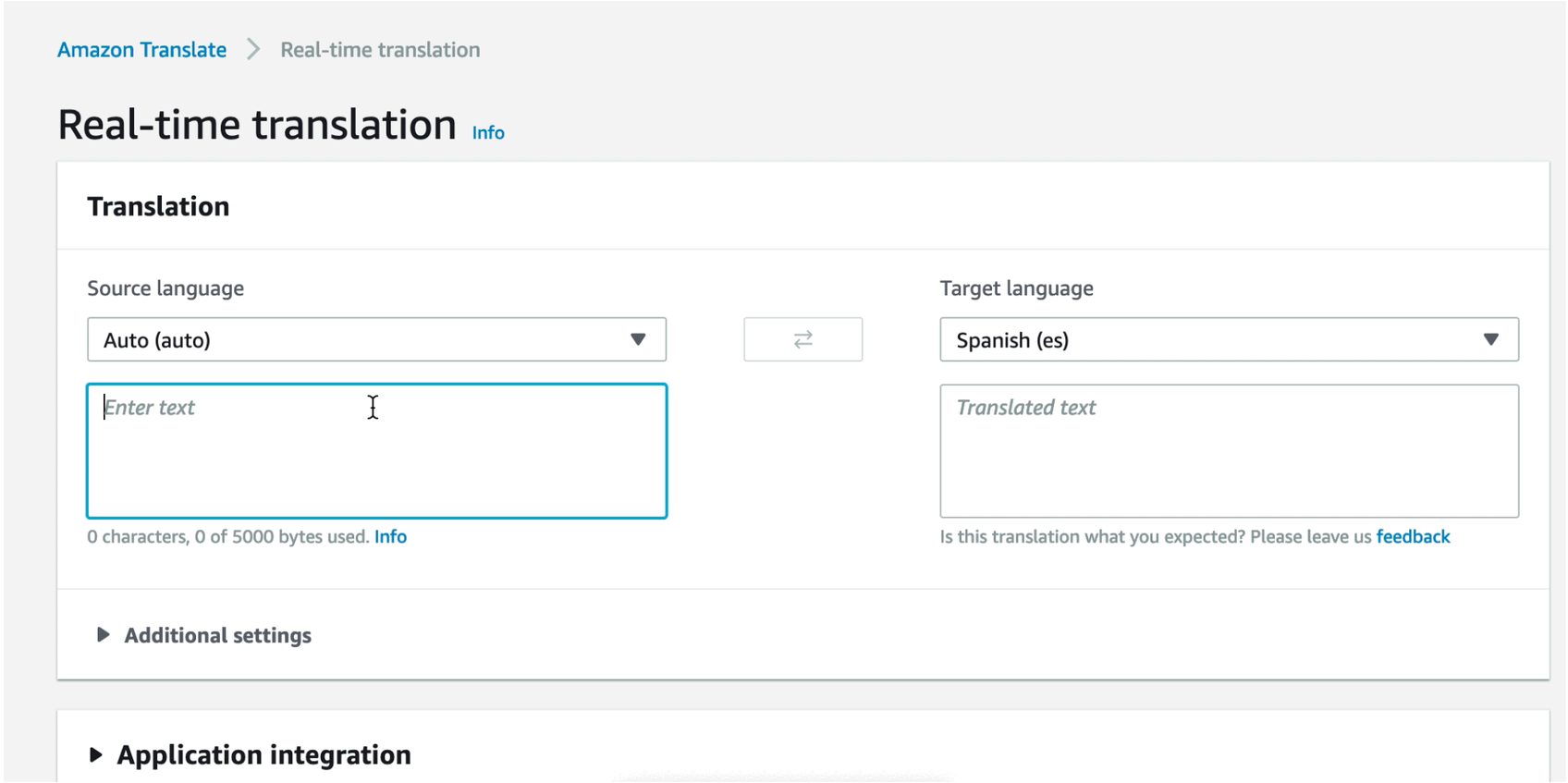
SNS



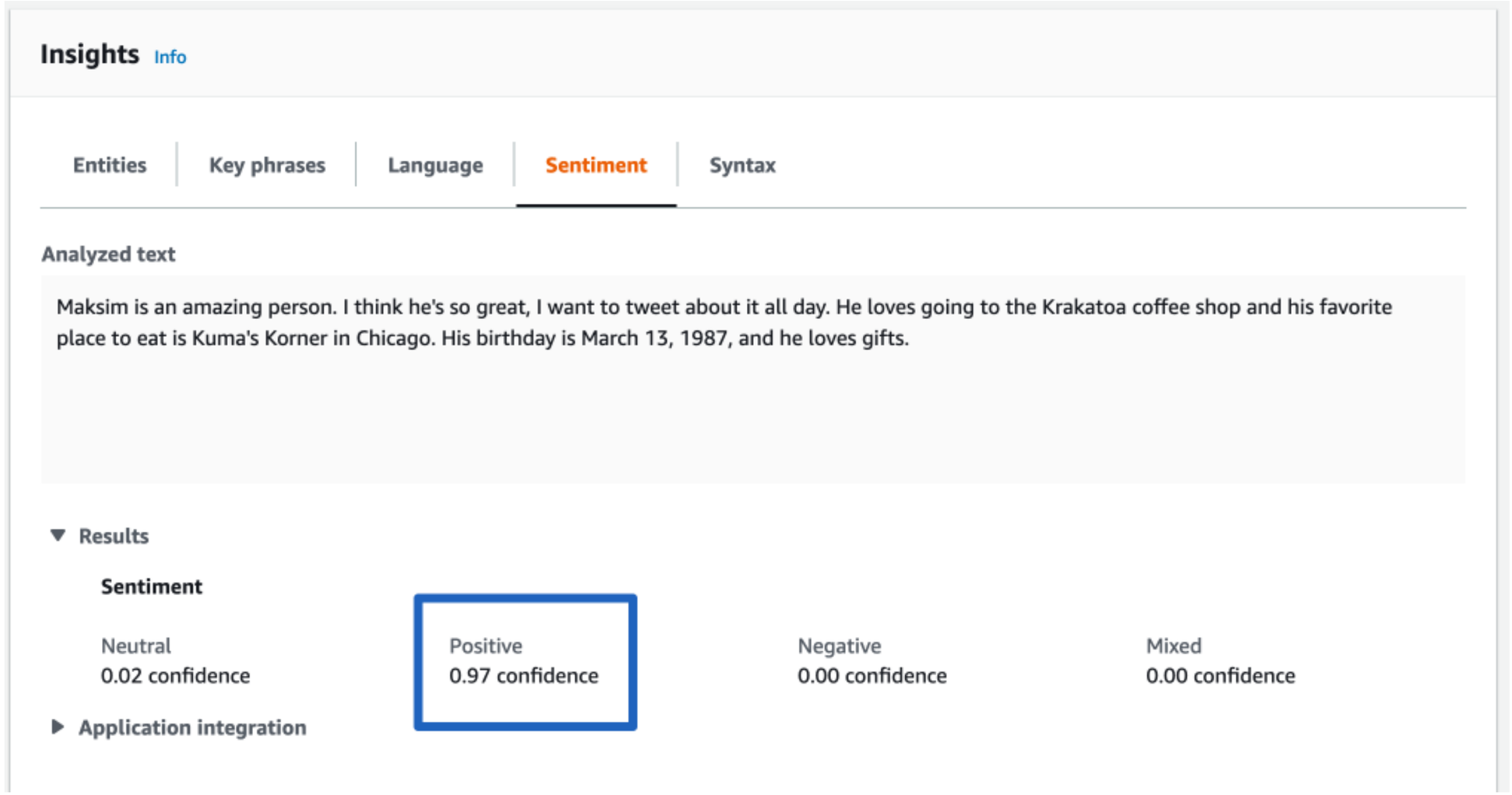
Rekognition



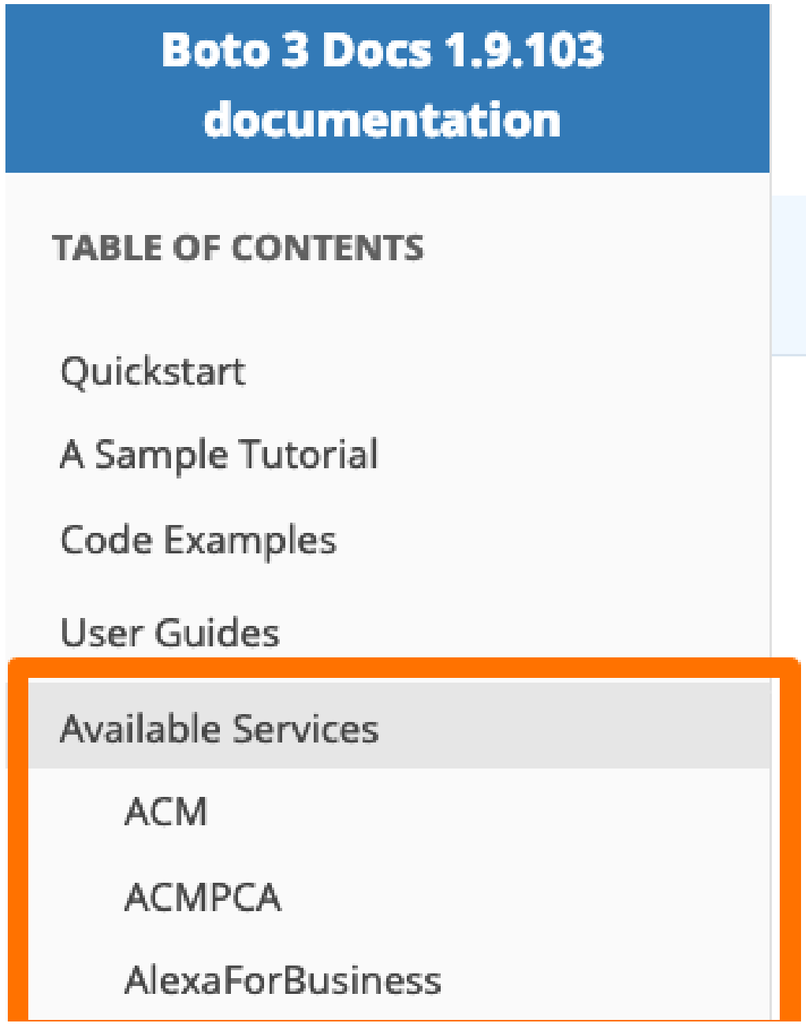
AWS Translate

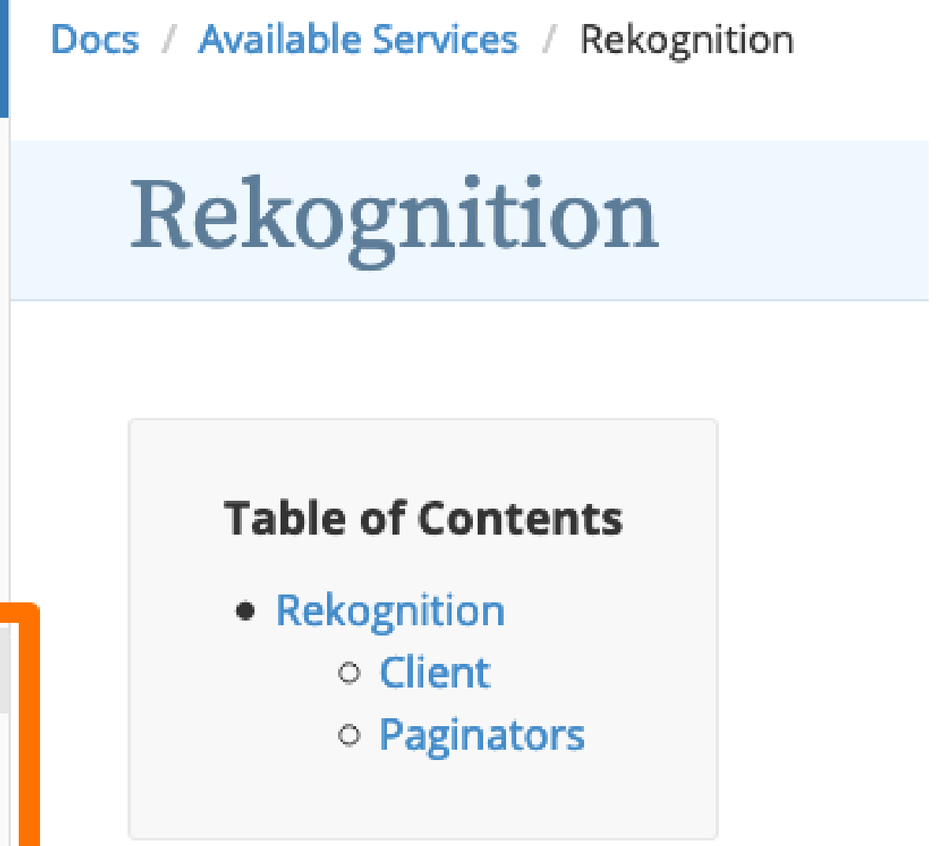


Comprehend

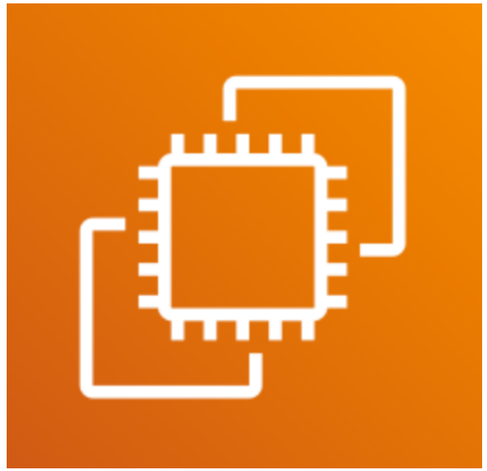

Insights Info
Entities
Key phrases
Language
Sentiment
Syntax
Analyzed text
Maksim is an amazing person. I think he's so great, I want to tweet about it all day. He loves going to the Krakatoa coffee shop and his favorite place to eat is Kuma's Korner in Chicago. His birthday is March 13, 1987, and he loves gifts.
Results
Sentiment
Neutral
0.02 confidence
Positive
0.97 confidence
Negative
0.00 confidence
Mixed
0.00 confidence
Application integration

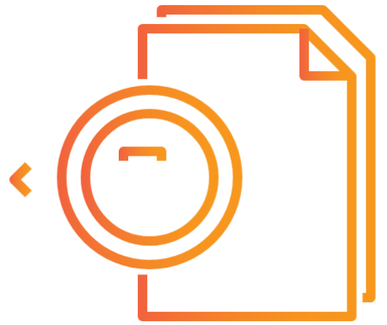
Use the Docs!


Boto 3 Docs 1.9.103 documentation
TABLE OF CONTENTS
Quickstart
A Sample Tutorial
Code Examples
User Guides
Available Services
ACM
ACMPCA
AlexaForBusiness


Docs / Available Services / Rekognition
Rekognition
Table of Contents
Rekognition
Client
Paginators

Other AWS Services





Thank you!

See you in the clouds!

INTRODUCTION TO AWS BOTO IN PYTHON