 Amazon Rekognition

*This document offers a brief outline of Amazon Rekognition as well as a simple Java implementation of the Rekognition API.*

# What It is

* [**Rekognition**](https://aws.amazon.com/rekognition/)is Amazon’s proprietary image and video analysis service. By harnessing heavily-developed **deep learning** technology, Rekognition is capable of detecting, searching, and labelling data on a granular level based on image and video input.
* First introduced to the public in 2016, Rekognition service currently comes in two flavors: Rekognition Image and Rekognition Video (stored and streaming).
* As with most AWS technologies, Rekognition can be **integrated into numerous types of workflows**, utilizing a plethora of different services on AWS and other platforms.

(More on this in section 3. Integration Options below.

# Features and example use cases

* **Abilities and Functions**:
  1. Faces

- Includes: *Detect in scene*  • *Search & Match in Collection or Private Repository* • *Verification • Add Facial Attribute Tags (expression, demographic data) • Recognize Celebrity* )

Use Case: Police departments are currently using detection capabilities on images and videos to check faces against public mugshot collections.

Figure a. - Mugshot match-finding

* 1. Objects (labels)

Figure b. - No cats found :(

Use Case**:** Disallow video upload to cat video site if label detection for cat-related terms returns 0 detected instances in file.

* 1. Text

Use Case**:** Filtering out personally identifiable info from images.

* 1. Pathing (Video only)

Figure e. - Analysis of customer browsing paths

Use case**:** Identify most common customer routes taken in a store in order to make smart adjustments to section layout, endcap presentation, and so on.

* 1. Unsuitable Content

Use case**:** Disallow very young users from accessing videos that have been deemed by way of analysis to contain people wearing risqué clothing.

# Integration Options and Sample Implementation (using AWS Java SDK)

* Sample Label Detection implementation:

Sample project can be found in /src of this repo ([direct link](https://github.com/zcmarcus/RekognitionTest/blob/master/src/main/java/org/zcmarcus/VideoLabelDetection.java)).

***Note****: The label detection example contained in this repo is written in Java and uses Maven; for implementations in other languages see:* [*AWS Dev Build Tools*](https://docs.aws.amazon.com/rekognition/latest/dg/setting-up.html)

***Pre-requisite****: it assumed that the user already has an Amazon AWS account and at least one IAM User set up for account management. For more information, see* [*this section of the Rek**ognition “Getting Started” guide.*](https://docs.aws.amazon.com/rekognition/latest/dg/setting-up.html)

# Controversies and Potential Social Ramifications

* Controversy surrounding Amazon pitching Rekognition to law-enforcement agencies including the CIA, [ICE](https://www.washingtonpost.com/news/the-switch/wp/2018/06/22/amazon-employees-demand-company-cut-ties-with-ice/), and [police departments](https://www.oregonlive.com/washingtoncounty/2019/05/amazons-facial-recognition-technology-is-supercharging-washington-county-police.html) around the country.
* [Racial biases persist in face detection technology](https://www.youtube.com/watch?v=N-Lxw5rcfZg) (presentation by Joy Buolamwini, Youtube – Wired UK).

Can we trust that new facial analysis techniques will not repeat mistakes of the past?

* **Q:** Given the ubiquity of Amazon and AWS. Are we as a society okay with these kinds of advanced technology invading our day-to-day lives?
* **Q:** Are we as developers feel comfortable harnessing this kind of power?

# Additional Related Links

### [Deep Learning (Wikipedia)](https://en.wikipedia.org/wiki/Deep_learning)

### [Amazon Simple Notification Service](https://aws.amazon.com/sns/getting-started/)

### [Amazon Simple Queue Service](https://aws.amazon.com/sqs/getting-started/)

### [Rekognition Developer Guide - Available API Operations](https://docs.aws.amazon.com/rekognition/latest/dg/API_Operations.html)