MODULE 5 · STUDY NOTES

Introducing Computer Vision

What computer vision does, the image- and video-analysis tasks it solves, Amazon Rekognition for managed image and video analysis, facial detection and recognition, and training custom models with Rekognition Custom Labels and SageMaker Ground Truth

Computer vision is the automated extraction of information from digital images, and the last decade of gains in computing power and algorithms has turned it from a research curiosity into an everyday, accessible technology. This module builds it up in three moves. First, the field itself: what computer vision does, where it is used, and the two families of tasks it solves — image analysis (classification, detection, segmentation) and video analysis (instance tracking, action recognition, motion estimation). Second, the managed AWS way to do it: Amazon Rekognition, a deep-learning service that adds image and video analysis to an application with no ML expertise, from object and scene detection to facial detection, comparison, and search — used responsibly. Third, what to do when a pre-built model is not enough: train your own domain model with Amazon Rekognition Custom Labels, label the data with Amazon SageMaker Ground Truth, and evaluate the result with a confusion matrix and the precision, recall, and F1 metrics that tell you whether it is ready to use.

LEARNING OBJECTIVES

- Describe the use cases for computer vision
- Describe the Amazon managed machine learning (ML) services for image and video analysis
- List the steps required to prepare a custom dataset for object detection
- Describe how Amazon SageMaker Ground Truth can be used to prepare a custom dataset
- Use Amazon Rekognition to perform facial detection

5.1 What computer vision is and where it is used

Computer vision is the automated extraction of information from digital images. Advances in computing power and algorithms over the last decade have sharply increased its capabilities and made the technology far easier to access. Computer vision enables machines to identify people, places, and things in images with accuracy **at or above human levels**, and with greater speed and efficiency. Usually built with **deep learning** models, it automates the extraction, analysis, classification, and understanding of useful information from a single image or a sequence of images — and the image data can take many forms: single images, video sequences, views from multiple cameras, or three-dimensional data.

PRIMARY COMPUTER VISION USE CASES

- **Public safety and home security** — image and facial recognition helps quickly identify unlawful entries or persons of interest, deterring crime and building safer communities.
- **Authentication and enhanced human-computer interaction** — quick identity-based authentication and sentiment-aware experiences (retail, faster banking) improve customer satisfaction.
- **Content management and analysis** — metadata extraction and image classification handle the millions of images added to media and social channels every day.
- **Autonomous driving** — improved, safer self-driving navigation that helps make autonomous transportation reliable.
- **Medical imaging** — faster, more accurate diagnosis for better treatment outcomes and life expectancy.
- **Manufacturing process control** — vision built into robotics improves quality assurance and operational efficiency for more reliable, cost-effective products.

From a practical standpoint, computer vision divides into two areas. **Image analysis** works on still images and covers **object classification, object detection, and object segmentation**. **Video analysis** works on moving images and covers **instance tracking, action recognition, and motion estimation**. The next two sections take each area in turn.

5.2 Image analysis: classification, detection, and segmentation

Object classification decides which category or categories a picture or object belongs to — no different from any other classification problem in machine learning. Crucially, the answer depends on the **model** you use: a model trained on food types might label an image as *Milk*, *Cookie*, *Orange*, *Hamburger*, *Salad*, and *French Fries*, while a model trained on different images might label the same photo as *Tray*, *Cutlery*, and *Napkin*. Models must be trained, and the training data provides the algorithm with the examples it learns from. When there are more than two categories it is a **multi-class** classification problem; with exactly two, a **binary** classification problem.

Object detection answers a richer question: not just *what* objects are in the image, but *where* they are. Alongside the object categories, detection returns the location of each object as a **bounding box** — typically the **top, left, width, and height** coordinates that surround the object — which your application can act on. Each detected object also comes with a **confidence** number: a percentage indicating how probable it is that the object belongs to a specific class. That confidence matters most when you drive an action from a detection, especially in facial detection.

Object segmentation — also called **semantic segmentation** — is similar to detection but goes into far more detail to get **fine boundaries** for each detected object. It is a fine-grained inference that predicts a class for **every pixel** in the image. Applications that need this precision include autonomous vehicles and advanced human-computer interaction. Segmentation is **not covered** further in this course, but it rounds out the ladder of image-analysis tasks.

TABLE 5.1 Image-analysis tasks compared

TASK	WHAT IT PRODUCES	NOTES
Object classification	The category or categories an image or object belongs to	Binary = two classes; multi-class = more. The output depends on how the model was trained
Object detection	Categories **plus** each object's location and confidence	Bounding box = top, left, width, height; confidence drives actions, especially in facial detection
Object segmentation	Fine, per-pixel boundaries for each detected object	Also called semantic segmentation; used in autonomous vehicles; not covered in this course

COMMON PITFALL

Classification, detection, and segmentation are a ladder of increasing detail, not synonyms. Classification only names the category; detection adds *where* (a bounding box) and a confidence; segmentation predicts a class for every pixel. On scenario questions, match the task to how much spatial detail the problem actually needs.

5.3 Video analysis: tracking, action recognition, and motion estimation

Video analysis applies computer vision to moving images. Amazon Rekognition Video is the AWS engine behind these tasks, and they group into three kinds of question: **where is each subject moving** (instance tracking), **what behavior is occurring over time** (action recognition), and **what activity does the motion itself reveal** (motion estimation).

THE THREE VIDEO-ANALYSIS TASKS

- **Instance tracking (pathing)** — capture the path each person follows through a scene. Rekognition Video's `TrackPersons` operation detects people and how they move even when the camera is in motion, and re-attributes motion to the same person when a face is blocked or leaves and re-enters the frame, returning **time segments** and **confidence scores**.
- **Action recognition** — study behavior over time: analyze shopper density and the path each person takes, and use face analysis to read **average age ranges, gender distribution, and emotions without identifying anyone**.
- **Motion estimation** — Rekognition Video identifies **thousands of objects, scenes, and activities** and relies on **motion** to recognize complex activities such as blowing out a candle or extinguishing a fire.

WORKED EXAMPLE – SPORTS ANALYTICS FROM MOTION

Point Rekognition Video at a baseball game and motion cues let it capture the type of pitch (slow ball, slider, and others), the batter's accuracy, and performance against a specific pitcher — data coaches use to guide players and to make in-game decisions. Modeled ball speed and trajectory off the bat can even trigger timed, automated events: an audio or visual warning about a possible foul ball into the crowd, or a preemptive cue that a hit has a high probability of being a home run so the follow-up (music, fireworks) is well-timed.

5.4 Amazon Rekognition: a managed vision service

Amazon Rekognition is a computer-vision service based on **deep learning** that adds image and video analysis to your applications. Because it is a **managed service**, Rekognition hosts the machine-learning models, maintains an API, and

scales to meet demand for you — and its models continuously learn and improve. It integrates with other AWS services such as **Amazon S3** for storage and **AWS IAM** for authentication and authorization, so you focus on building applications that call the API and, optionally, on training the service for your unique requirements. Rekognition offers APIs, SDKs, and AWS CLI commands, with SDKs for JavaScript, Python, PHP, .NET, Ruby, Java, Go, Node.js, and C++.

WHAT AMAZON REKOGNITION CAN ANALYZE

- **Searchable image and video libraries** — discover the objects and scenes that appear in your media.
- **Face-based user verification** — confirm a user's identity by comparing a live image with a reference image.
- **Sentiment and demographic analysis** — interpret expressions such as happy, sad, or surprised, and demographic details such as gender.
- **Unsafe content detection** — detect inappropriate content in images and in stored videos.
- **Text detection** — recognize and extract text from images (Amazon Rekognition Text in Image).
- Security and compliance are a **shared responsibility** between AWS and the customer; check whether your application falls under any regulatory restrictions for your field or country.

You call Rekognition through the API and SDKs. API operations cover detecting **labels**, detecting **faces**, recognizing **celebrities**, and detecting **unsafe** images. To run a prediction, hand the service an image either as an object in **Amazon S3** or as an uploaded **byte stream** — images must be **JPEG or PNG** — and Rekognition processes it, makes the prediction, and returns a **JSON** object with the results.

For **object and scene detection**, Rekognition often returns **multiple labels**, each with its own **confidence** that the label was found, and labels can be arranged in **hierarchies**. When it finds an instance of an object, the result includes a **bounding box** — a starting coordinate (**Top, Left**) and box dimensions (**Width, Height**), expressed as a ratio of the overall image size — that tells you where the object sits in the image. Every finding carries a **confidence score**, and you tune your application's response to it: the higher the score, the more likely the object was labeled correctly.

TABLE 5.2 Reference architectures – Rekognition wired into AWS

USE CASE	HOW REKOGNITION IS WIRED IN
Searchable image library	Mobile photo → **Amazon S3** (write event) → **AWS Lambda** calls Rekognition via the SDK → labels and confidence stored in **Amazon Elasticsearch** for later property lookup
Image moderation	Upload → **S3** write event → **Lambda** → Rekognition checks for inappropriate content → content approved, or routed to manual inspection with the user notified if it is rejected
Sentiment analysis (video)	In-store camera → **Amazon Kinesis** video → Rekognition extracts sentiment, age, and gender → Kinesis → **Lambda** → **S3** → **Amazon Redshift** → reports in **Amazon QuickSight**

COMMON PITFALL

Rekognition returns a **confidence score** on every finding for a reason — it is a probability, not a certainty. Don't hard-wire an application to act on raw detections; choose a confidence threshold appropriate to the stakes of the decision — a content-moderation filter and a door-unlock check warrant very different thresholds.

5.5 Detecting, comparing, and searching faces

Facial detection uses a model tuned specifically to find faces and facial features. For each face, Rekognition returns a **bounding box** (with a confidence that the box actually contains a face), **facial landmarks**, **facial attributes**, **quality**,

pose, and **emotions** — and, as always, a **confidence** on each detected feature. Every prediction here is based **only on visual observation** of the image.

KEY TERMS

Bounding box
Coordinates of the box that surrounds the face, plus a confidence that it contains a face

Facial landmarks
An array of feature points (left eye, right eye, mouth), each with x and y coordinates

Facial attributes
Values such as beard or sunglasses (Boolean) or gender (string) — most with their own confidence

Quality
Describes the brightness and the sharpness of the face

Pose
Describes the rotation of the face within the image

Emotions
A set of emotions, each with a confidence in the analysis

Facial recognition compares two images to decide whether they contain the **same person**, and it always needs both a **source** and a **target** image. The results describe every face found: **matched faces** (bounding box and confidence, a **similarity score**, and facial-landmark locations), **source face information**, and any **unmatched faces**. As everywhere in Rekognition, confidence scores indicate how likely each prediction is.

To search for **known** faces, you first train the model by building a **collection** — create the collection, then add faces to it. Rekognition performs facial recognition on the images you provide and returns the usual bounding-box coordinates and confidence scores. You associate each face with an image ID through the `ExternalImageId` parameter (a filename or an ID you create), and later call the `SearchFacesByImage` operation to find matches; the response is an array of matched faces with bounding boxes, confidence scores, and the `ExternalImageId` value you can use to link back to the source image.

> COMMON PITFALL
>
> Read Rekognition's face outputs for exactly what they are. **Gender is inferred from the image, not from identity**, and **emotion is inferred from physical appearance** and may not reflect how the person actually feels. Facial recognition should **never** be used in ways that violate an individual's rights — including the right to privacy — or to make **autonomous decisions** in scenarios that require human analysis. In public-safety and law-enforcement contexts, use it only to **narrow the field of potential matches** for a human to review.

5.6 Processing stored and streaming video

Amazon Rekognition can process both **stored videos** and **streaming videos**, and the two follow different patterns. Stored-video analysis is **asynchronous** — you start a job and retrieve results when it finishes — while streaming works on a live feed in near real time. Detection results for video include the same labels as image detection, plus a **timestamp** marking where in the video the label was found (in milliseconds from the start).

STORED VIDEO	STREAMING VIDEO
asynchronous, results retrieved later	*near real time on a live feed*
▪ Store the video in an **Amazon S3** bucket ▪ Each detection type has its own **Start** operation (people, faces, labels, celebrities, text, unsafe content) ▪ Completion is published to an **Amazon SNS** topic; best practice routes it to an **Amazon SQS** queue for durability, which your app monitors ▪ A matching **Get** operation returns the labels plus a timestamp (ms from the start)	▪ Send the feed to **Amazon Kinesis Video Streams** ▪ A Rekognition Video **stream processor** (`CreateStreamProcessor`) manages the analysis ▪ Analysis is written to a **Kinesis data stream** that your client application reads ▪ To recognize a face you must create a **collection**; a JSON frame record is emitted per analyzed frame (not every frame is analyzed)

TABLE 5.3 Stored-video Start/Get operation pairs

DETECTION	START / GET OPERATIONS
People	`StartPersonTracking` / `GetPersonTracking`
Faces	`StartFaceDetection` / `GetFaceDetection`
Labels	`StartLabelDetection` / `GetLabelDetection`
Celebrities	`StartCelebrityRecognition` / `GetCelebrityRecognition`
Unsafe content	`StartContentModeration` / `GetContentModeration`

5.7 Training on your domain: Rekognition Custom Labels and Ground Truth

A pre-built model is powerful but bounded: it finds only what it was trained to find. **Amazon Rekognition** was trained on tens of millions of images across many categories, yet it cannot label objects it never saw. Run an *eight of hearts* playing card through it and you get labels like *wood*, *paper*, and *text* — never *playing card* or *eight of hearts*. To detect objects in your problem domain, you must train the model on your own images, and the process is similar for other pre-trained models.

Amazon Rekognition Custom Labels finds the objects and scenes unique to your business — searching for logos, identifying products or machine parts, or telling healthy plants from infected ones. It can **classify** images (image-level predictions) or **detect** objects (object- or bounding-box-level predictions), and it builds on Rekognition's existing training so you supply far fewer images than training from scratch would demand. Its three benefits are **simplified data labeling** (a UI for labels and bounding boxes), **automated machine learning** (it loads and inspects the data, selects the algorithm, trains, and reports metrics), and **simplified evaluation, inference, and feedback** (side-by-side prediction-versus-label review, then retrain with more images to improve).

> **COMMON PITFALL**
>
> Almost every practical vision solution starts from an existing model. Training a computer-vision algorithm from scratch needs a very large labeled dataset — often thousands to tens of thousands of hand-labeled images and months of effort — which is impractical for most organizations. Custom Labels exists so you can adapt a pretrained model with only a few hundred domain images instead.

TABLE 5.4 The custom-labeling workflow

STEP	WHAT YOU DO
1. Collect images	Gather a few hundred domain images (or fewer); use images similar to production with varied lighting, backgrounds, and resolutions; build a separate model per domain
2. Create training dataset	Upload and label images (console or Ground Truth), or import a Ground Truth **.manifest** file; a dataset holds images, labels, and bounding boxes — use at least 2 labels and 10 images per label
3. Create test dataset	Supply your own test set, reuse an existing one, or let Custom Labels split training **80/20** (80% train, 20% test)
4. Train the model	Custom Labels automatically loads and inspects the data, selects the algorithm, trains, and reports metrics; you are billed for training time
5. Evaluate	Review precision, recall, and F1 against the test set; improve by adding better or more data and adjusting labels
6. Use the model	Start the model and run inferences through an API or CLI operation that returns labels, bounding boxes, and confidence

IMAGE-LEVEL LABELS	OBJECT-LEVEL LABELS
scenes and concepts ▪ Applied to the whole image ▪ Best for detecting a scene or concept (for example, a *beach*) ▪ Manifest stores the label / class-name plus labeling metadata	*objects with bounding boxes* ▪ Applied to a region of the image (localization information) ▪ A bounding box marks where the object is (for example, an *Echo Dot*) ▪ Manifest stores each image's bounding boxes and the matching label

AMAZON SAGEMAKER GROUND TRUTH

- Builds **high-quality labeled datasets** using workers from **Amazon Mechanical Turk**, a **vendor company**, or your **own internal workforce**.
- **Active learning** (called automated data labeling) labels part of the data automatically to cut cost and time — but incurs SageMaker training and inference charges.
- Recommended for **large** datasets: minimum **1,250** objects, with **5,000+** strongly suggested, because the active-learning network needs enough data to reach the required accuracy.
- Output can train your own models or feed an Amazon Rekognition Custom Labels dataset; already-labeled images must be converted into a Ground Truth **manifest file**.

5.8 Evaluating and improving a custom model

When training finishes, you evaluate the model. During testing, Custom Labels predicts whether each test image contains a custom label, and the **confidence score** quantifies how certain that prediction is. Because this is a **classification** problem, results map onto a **confusion matrix** whose four outcomes drive every metric — and the same definitions apply at the **bounding-box level**, where metrics are computed over each box in each test image.

TABLE 5.5 The confusion matrix (example label: cat)

OUTCOME	WHAT IT MEANS
True positive (TP)	Model correctly predicts the label is present — the predicted label is also a ground-truth label (a cat is present, and it says cat)
False positive (FP)	Model predicts the label is present, but it is not in the ground truth (says cat, no cat)
False negative (FN)	Model misses a label the ground truth says is present (a cat is present, but it does not say cat)
True negative (TN)	Model correctly predicts the label is absent (no cat, and it does not say cat)

THE THREE EVALUATION METRICS

- **Precision** — the proportion of positive results that were correctly classified.
- **Recall** — the fraction of your test set's labels that were correctly classified.
- **F1 score** — combines precision and recall into a single number; useful when there is class imbalance but you want to keep precision and recall balanced. A higher value means better performance on both.

REDUCE FALSE POSITIVES → PRECISION	REDUCE FALSE NEGATIVES → RECALL
the model over-predicts your label	*the model misses your label*
▪ **Raise** the confidence threshold to keep correct predictions while dropping false ones (with diminishing returns) ▪ Add the confusing class as its own label so the model learns B, not A ▪ Check for and fix mislabeled training or test images	▪ **Lower** the confidence threshold ▪ Use better, more representative training examples ▪ Split one label into easier-to-learn classes (for example, good / burnt / broken cookies)

WORKED EXAMPLE – RUNNING AN INFERENCE WITH A CUSTOM MODEL

Once you are satisfied, start the model and call it from the AWS CLI or an SDK. The `DetectCustomLabels` operation takes the model's **ARN** and an image supplied as an **Amazon S3 object** or a base64-encoded **byte array**, and returns an array of custom labels — each with a **label**, a **bounding box** for objects (coordinates given as a ratio of the image size), and a **confidence**. During training the model calculates a threshold; by default results below it are hidden, so pass a `MinConfidence` above that threshold to filter, or `0` to return everything. `MaxResults` caps how many labels come back, sorted highest-confidence first.

COMMON PITFALL

Mind the direction of the confidence threshold. **Raising** it removes false positives and improves **precision**; **lowering** it recovers missed detections and improves **recall**. Because precision and recall trade off, pushing the threshold too far in either direction brings diminishing returns — tune it against your use case rather than chasing a single metric.

Module summary

- Computer vision is the automated extraction of information from digital images; it divides into image analysis and video analysis, is usually built on deep learning, and works on single images, video, multi-camera views, or 3D data.
- Image-analysis tasks form a ladder: classification (which category), detection (category plus a bounding box and confidence), and segmentation (per-pixel boundaries; not covered in this course).
- Video analysis covers instance tracking (pathing), action recognition, and motion estimation; Rekognition Video tracks people even when the camera moves or a face is briefly blocked.
- Amazon Rekognition is a managed, deep-learning service (no ML expertise required) for image and video analysis — faces, sentiment and demographics, text, unsafe content, and searchable libraries — integrating with S3, IAM, Lambda, and Kinesis; it takes JPEG/PNG and returns JSON with labels and confidence.
- Every finding carries a confidence score; choose a threshold for the stakes. Facial gender and emotion are inferred from the image, not identity, and facial recognition must only narrow matches for human review — never decide autonomously or violate privacy.
- Face features: detection returns a bounding box, landmarks, attributes, quality, pose, emotions, and confidence; recognition compares a source and target for a similarity score; searching known faces uses a collection and the SearchFacesByImage operation.
- Stored video uses asynchronous Start/Get operations with completion via SNS then SQS and video in S3; streaming video uses Kinesis Video Streams in and a Kinesis data stream out through a Rekognition Video stream processor.
- Pre-built models detect only what they were trained on; Rekognition Custom Labels trains a domain model from a few hundred images (at least 2 labels, at least 10 images per label) via a 6-step workflow with automated ML; SageMaker Ground Truth builds labeled datasets with a workforce and active learning (minimum 1,250 objects, 5,000+ recommended).
- Evaluate with a confusion matrix (TP, FP, FN, TN): precision, recall, and F1. Raise the confidence threshold and add classes to cut false positives (precision); lower the threshold and improve data to cut false negatives (recall).

Review questions

1. **Define computer vision and name its two practical areas.**

   Answer: Computer vision is the automated extraction of information from digital images; its two areas are image analysis and video analysis.

2. **Distinguish object classification, object detection, and object segmentation.**

   Answer: Classification names the category or categories; detection adds each object's location as a bounding box plus a confidence; segmentation predicts fine, per-pixel boundaries for each object (semantic segmentation).

3. **What does Amazon Rekognition provide, and what inputs and integrations does it use?**

   Answer: Managed deep-learning image and video analysis (faces, sentiment and demographics, text, unsafe content, library search) with no ML expertise required; it integrates with S3, IAM, Lambda, and Kinesis, accepts JPEG or PNG images as an S3 object or byte stream, and returns JSON with labels and confidence.

4. Why should Rekognition's gender and emotion outputs be treated cautiously, and how should facial recognition be used responsibly?

Answer: Gender and emotion are inferred from the image only — gender is not identity, and emotion may not match the person's true feelings. Facial recognition should only narrow the field of potential matches for human review, never make autonomous decisions where human analysis is required, and never violate privacy.

5. Contrast processing stored video versus streaming video with Rekognition.

Answer: Stored: the video sits in Amazon S3, each detection type has its own asynchronous Start/Get operation, and completion is published to SNS and routed to SQS for durability. Streaming: the feed goes to Kinesis Video Streams, a Rekognition Video stream processor analyzes it, and results are written to a Kinesis data stream that the client application reads.

6. List the six steps to build a Rekognition Custom Labels model and the minimum data it needs.

Answer: Collect images, create a training dataset, create a test dataset, train the model, evaluate, and use the model; you need at least 2 labels and at least 10 images per label, typically a few hundred images total.

7. Explain precision, recall, and F1, and how to reduce false positives versus false negatives.

Answer: Precision is the proportion of positive predictions that were correct; recall is the fraction of test-set labels correctly classified; F1 combines both into one number. Reduce false positives by raising the confidence threshold and adding classes (better precision); reduce false negatives by lowering the threshold and using better data or more precise classes (better recall).

8. What is Amazon SageMaker Ground Truth, and when is automated data labeling appropriate?

Answer: A service that builds high-quality labeled training datasets using a workforce (Amazon Mechanical Turk, a vendor, or an internal team) with active learning. Automated data labeling suits large datasets — minimum 1,250 objects, 5,000+ recommended — and incurs SageMaker training and inference costs.