

# Training Supervised CV Models from Foundational Models with Autodistill

Daniel Bank

<https://github.com/danielbank/label-with-autodistill>

# Scope of This Talk

- Brief review of Computer Vision
- Manually label image data for a supervised model using Vertex AI
- Automatically label image data using a foundational model and Autodistill
- ~~— Train the supervised model~~
- Examine the limitations of Autodistill



WARNING  
I love dice

[histogramo.com](https://histogramo.com)

# A Brief Review of Computer Vision



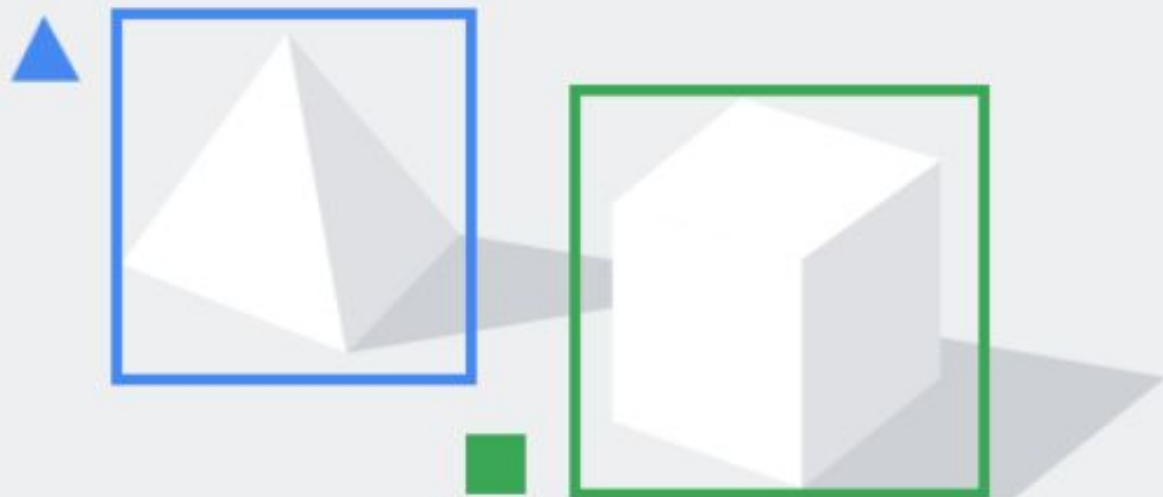
☐ **Image classification (Single-label)**

Predict the one correct label that you want assigned to an image.



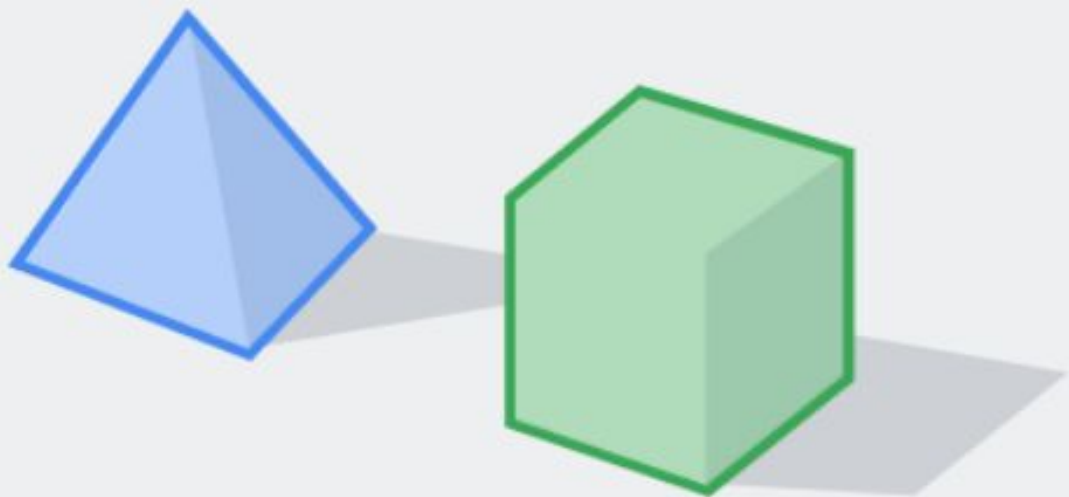
☐ **Image classification (Multi-label)**

Predict all the correct labels that you want assigned to an image.



## Image object detection

Predict all the locations of objects that you're interested in.



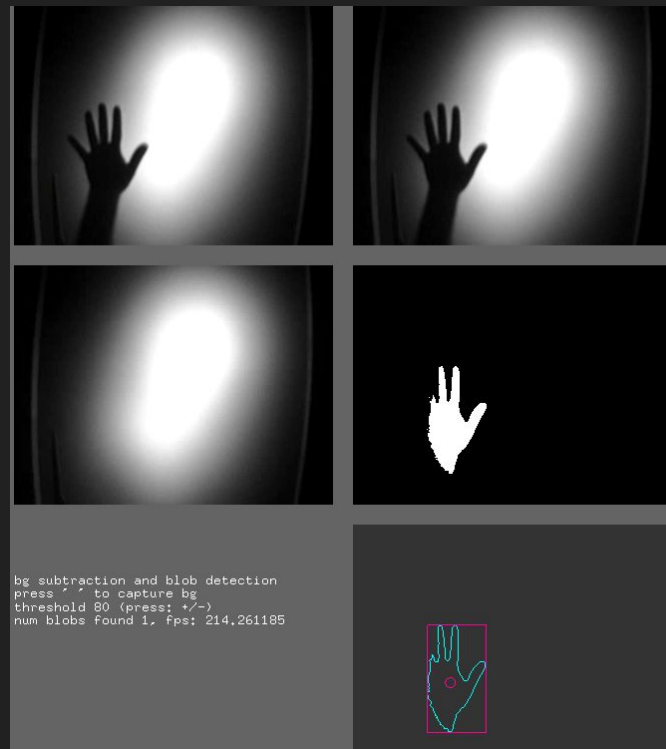
☐ **Image segmentation**

Predict per-pixel areas of an image with a label.

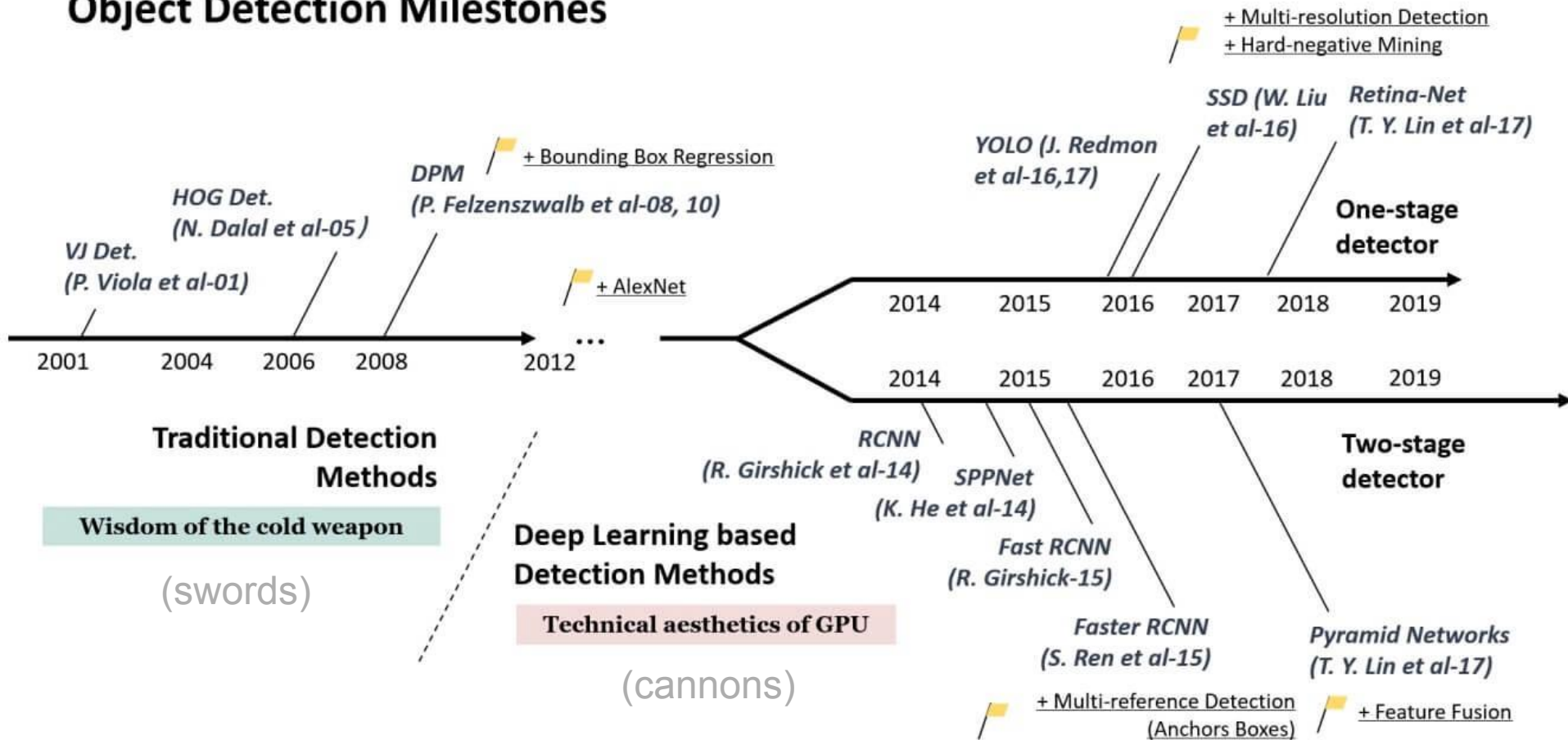


# OpenCV

- C++ library initial released in **2000**
- Algorithms for image processing / analysis
  - [Convolution and Filtering](#)
  - [Geometric Transformations](#), etc.
- DNN Module (Deep Neural Network) in **2015**



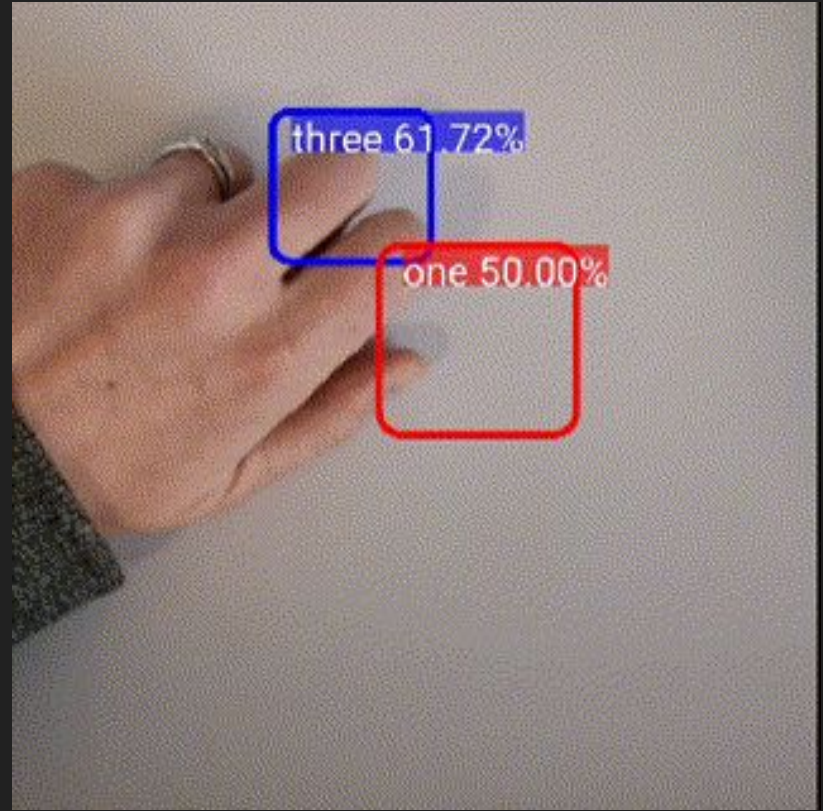
# Object Detection Milestones



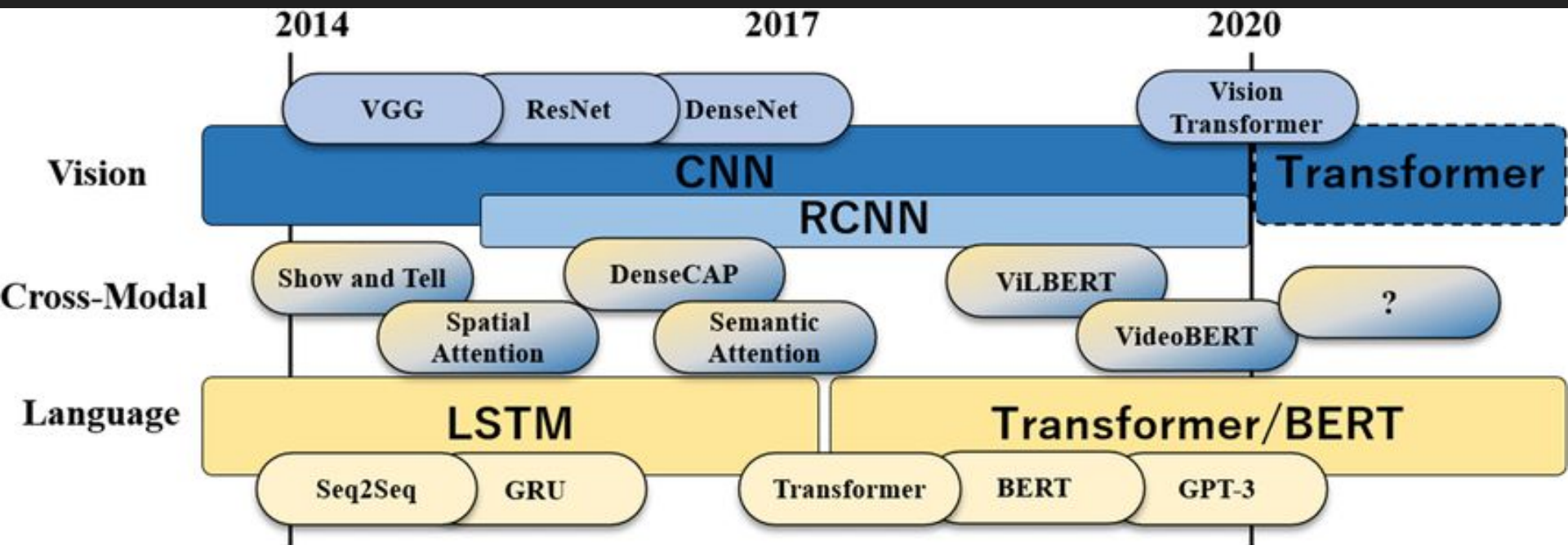
Source [Object Detection in 20 Years: A Survey](#)

# Supervised Models

- **“Knows a lot about a little”**
  - Only knows the classes it was trained on
- **Small (model size ~ 100KB-10MB)**
  - Can run on your phone or a microcontroller
- **Fast (inference time ~ 1-10ms)**
  - Real Time Inferencing



# SOTA in 2024: Vision Transformers



Examples: OWL ViT, DETR, Grounded SAM

# Foundational Models

- **“Knows a lot about a lot”**
  - No predefined classes
- **Large** (model size ~ 100GB-1TB)
  - Cannot run on your phone... yet
- **Slow** (inference time ~ 1-10s)
  - Not Real Time



What do you see?



I see an image of several dice scattered inside a rectangular container or tray. The dice come in different colors and styles. Some of the colors visible include yellow, red, green, purple, white, and black. The dice have a variety of dot patterns representing the numbers 1 through 6. There's also

## Describe the two white dice



The two white dice in the image have the following characteristics:

### 1. **White Die #1** (Top-left):

- Positioned towards the bottom left of the tray.
- It has black dots on its visible face, showing the number **1**.
- This die appears plain, with standard black pips against a white background.

### 2. **White Die #2** (Upper-right):

- Located near the upper right part of the tray.
- This die shows the number **6** on its visible face, with black dots.
- Similar to the first one, it has a classic design with black dots on a white surface.

Both dice have standard designs, differing only in the numbers shown.







# Labelling a Dataset Manually ([Vertex AI](#))

# Six-sided Dice Dataset on Kaggle

- <https://www.kaggle.com/nellbyler/d6-dice>
- 250 images
- 1-25 six-sided dice per image
- Images and annotations (YOLO format)



## Vertex AI



Dashboard



Datasets



Feature Store



Labeling tasks



Workbench



Pipelines



Training



Experiments



Model Registry



Endpoints



Batch predictions



Metadata



Matching Engine



Marketplace



## Create dataset

☐ Image classification (Single-label)

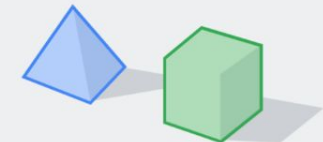
Predict the one correct label that you want assigned to an image.

☐ Image classification (Multi-label)

Predict all the correct labels that you want assigned to an image.

☒ Image object detection

Predict all the locations of objects that you're interested in.

☐ Image segmentation

Predict per-pixel areas of an image with a label.

Region

us-central1 (Iowa) ▾ ?

## ▼ ADVANCED OPTIONS

You can use this dataset for other image-based objectives later by creating an annotation set. [Learn more](#)

## CREATE LABELING TASK

Vertex AI

Dashboard

Datasets

Feature Store

Labeling tasks

Workbench

Pipelines

Training

Experiments

Model Registry

Endpoints

Batch predictions

Metadata

Matching Engine

Marketplace

Item 5 of many

Data split  
Default ▾

OBJECTS

DETAILS

Filter

Filter labels

+

1 (1)

1 1

4 (1)

4 1

2 (1)

2 1

6 (3)

6 1

6 2

6 3

5 (3)

5 1

5 2

ADD LABEL

SAVE

<

>

↶

↷

🔍

🔍

📄

⚙️ ▾

🔄

🖱️

📐 ▾

⋮

Vertex AI

Dashboard

Datasets

Feature Store

Labeling tasks

Workbench

Pipelines

Training

Experiments

Model Registry

Endpoints

Batch predictions

Metadata

Matching Engine

Marketplace

Item 5 of many

Data split  
Default

OBJECTS

DETAILS

Filter

Filter labels

+

1 (1)

1 1

4 (1)

4 1

2 (1)

2 1

6 (4)

6 1

6 2

6 3


6 4

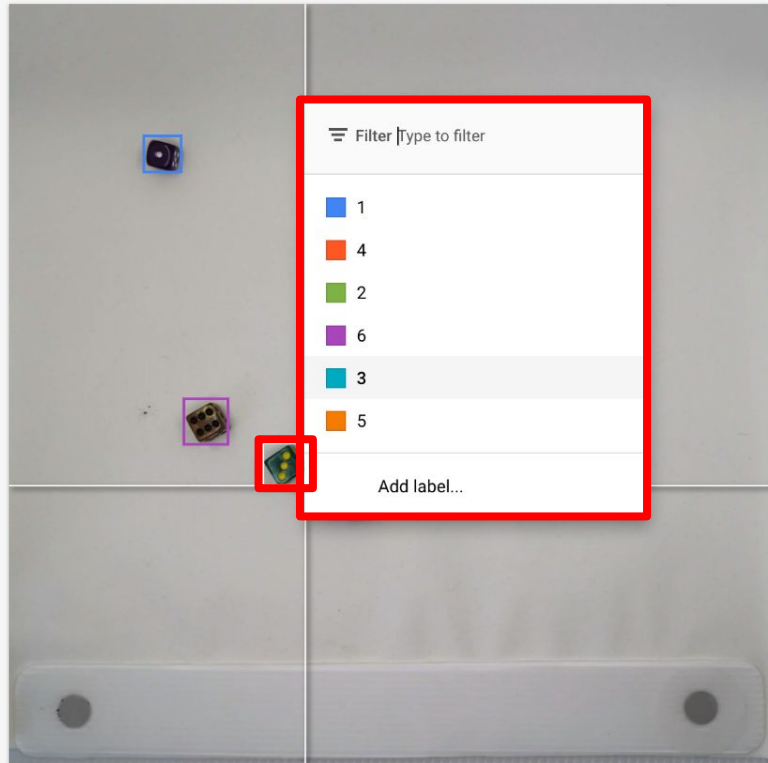
5 (3)

5 1

ADD LABEL

SAVE





Filter

Type to filter

1

4

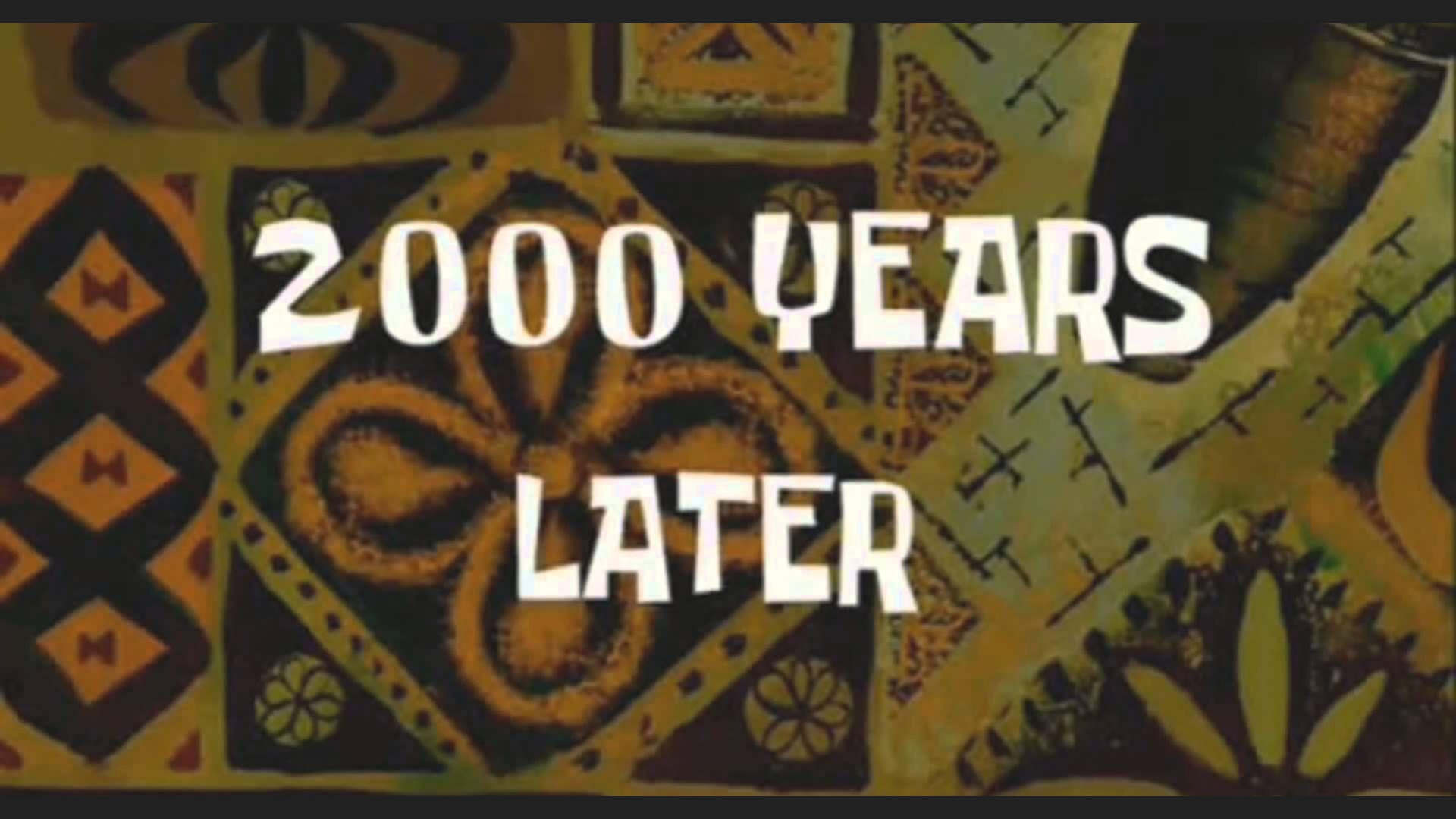
2

6

3

5

Add label...



**2000 YEARS  
LATER**





# Generating Labels Automatically ([Autodistill](#))

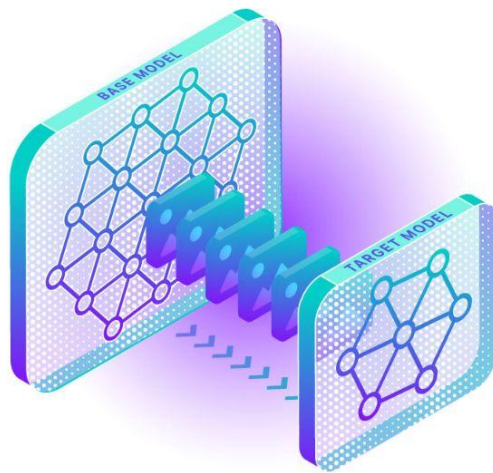
# Autodistill ([github.com/autodistill/autodistill](https://github.com/autodistill/autodistill))

- Human labeling is one of the biggest barriers to broad adoption of CV
- Takes 1000's of hrs to craft a dataset suitable for training a production model
- Traditional human labeling is just another form of distillation from an extremely capable Base Model (the human brain 🧠)
- Foundation models know a lot about a lot, but for production we need models that know a lot about a little
- For many production use-cases, we need models that can run:
  - Cheaply
  - In real time
  - At the edge

# Autodistill ([docs.autodistill.com](https://docs.autodistill.com))



PROVIDE IMAGES



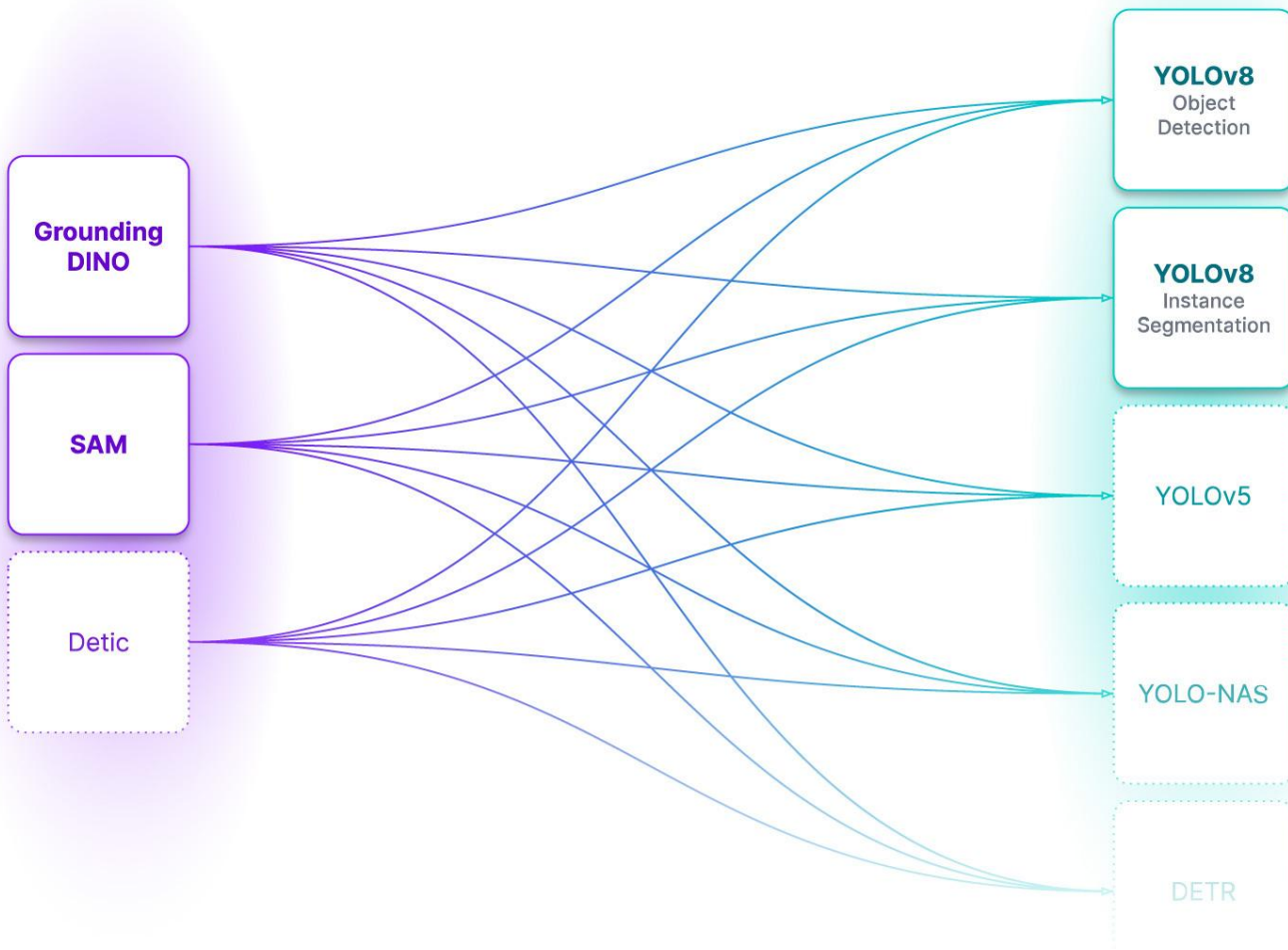
RUN AUTODISTILL



EDGE-READY MODEL

## BASE MODEL

Powerful, often too big  
for edge deployment



## TARGET MODEL

Smaller, faster to run for  
edge deployment

# Demo Notebook

Link to Demo Notebook

<https://github.com/danielbank/label-with-autodistill/blob/main/autodistill.ipynb>

This is basically taken verbatim from Roboflow's notebook "How to Auto Train YOLOv8 Model with Autodistill":

[https://colab.research.google.com/github/roboflow-ai/notebooks/blob/main/notebooks/how-to-auto-train-yolov8-model-with-autodistill.ipynb#scrollTo=c\\_nGQ9Ps842E](https://colab.research.google.com/github/roboflow-ai/notebooks/blob/main/notebooks/how-to-auto-train-yolov8-model-with-autodistill.ipynb#scrollTo=c_nGQ9Ps842E)

# Limitations with Autodistill - Complex Ontologies

- More complex ontologies consume all System RAM until the runtime crashes
- { “dice showing the number 1”: “one”, ..., “dice showing the number 6”: “six” }

```
trying to load grounding dino directly
downloading dino model weights
... torch.meshgrid: in an upcoming release, it will be required to pass the indexing argument. (Tri
final_text_encoder_type: bert-base-uncased

tokenizer_config.json: 100% ██████████ 48.0/48.0 [00:00<00:00, 3.64kB/s]

config.json: 100% ██████████ 570/570 [00:00<00:00, 46.1kB/s]

vocab.txt: 100% ██████████ 232k/232k [00:00<00:00, 9.77MB/s]

tokenizer.json: 100% ██████████ 466k/466k [00:00<00:00, 2.16MB/s]
`clean_up_tokenization_spaces` was not set. It will be set to `True` by default. This behavior
model.safetensors: 100% ██████████ 440M/440M [00:01<00:00, 435MB/s]

You are using `torch.load` with `weights_only=False` (the current default value), which uses th
You are using `torch.load` with `weights_only=False` (the current default value), which uses th
Labeling /content/images/IMG_20191208_111718.jpg: 0%|██████████| 0/250 [00:00<?, ?it/s]The `c
torch.utils.checkpoint: the use_reentrant parameter should be passed explicitly. In version 2.4
None of the inputs have requires_grad=True. Gradients will be None
`torch.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args..
The `device` argument is deprecated and will be removed in v5 of Transformers.
torch.utils.checkpoint: the use_reentrant parameter should be passed explicitly. In version 2.4
None of the inputs have requires_grad=True. Gradients will be None
`torch.cuda.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cuda', args..
Labeling /content/images/IMG_20191208_112501.jpg: 71%|██████████| 178/250 [10:58<04:25, 3.68s
```

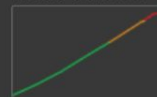
## Resources X

You are subscribed to Colab Pro. [Learn more](#)  
Available: 75.68 compute units  
Usage rate: approximately 3 per hour  
You have 1 active session.

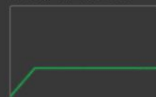
[Manage sessions](#)

Python 3 Google Compute Engine backend (GPU)  
Showing resources from 7:46 PM to 7:59 PM

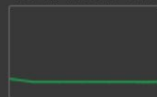
System RAM  
49.5 / 53.0 GB



GPU RAM  
7.3 / 22.5 GB



Disk  
40.4 / 235.7 GB



# Limitations with Autodistill

- { “dice”, “background” } gets the segmentation right but we still need to manually change labels on each class in each image
- { “one”: “one”, ..., “six”: “six” } does not get the segments right









Thank you!