

AI Factory Austria AI:AT



Modern Computer Vision & Knowledge Distillation Day 2: Hands-On Lab

Shaping the Future of AI

Dejan Đukić
Senior AI/ML Engineer, TetraScience

11.03.2026

Day 1 Recap: The Four Pillars



The Four Pillars (Recap)

1. CLIP

- Text and images in the same embedding space. 400M image-caption pairs trained two encoders to agree.

2. Vision Transformers

- Chop image into patches. Every patch talks to every other patch in one step. No more drinking straw & narrow receptive field.

3. Open Vocabulary Detection

- Dense grid of vectors over the image. Match your text query at every position. Text becomes bounding boxes.

4. Distillation

- Big slow teacher labels data. Small fast student learns from those labels (OR, probability distributions).



CLIP - Contrastive Image-Language Pretraining

UNDERSTANDING CONTRASTIVE VISION-LANGUAGE PRE-TRAINING WITH VISUAL PATCHES

STAGE 1: INPUT AND ENCODING

Golden Retriever in a park



IMAGE

1. VISUAL PATCH EXTRACTION



1. VISUAL PATCH EXTRACTION

Image is divided into a grid of N fixed-size visual patches (e.g., 16x16 pixels).

2. TEXT TOKENIZATION

[CLS] A golden retriever sitting in green grass

TEXT PROMPT:
"A golden retriever sitting in green grass."

TEXT

3. IMAGE ENCODER (e.g., ViT - Vision Transformer)

Transformer layers process patches, learning spatial dependencies and global context.

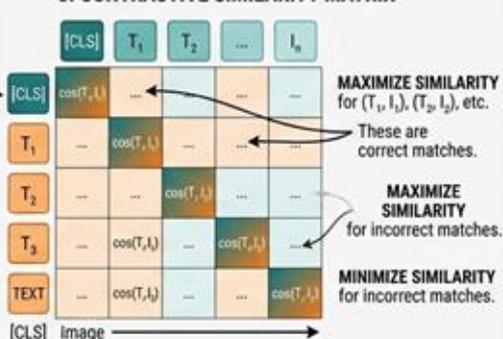
4. TEXT ENCODER
(e.g., BERT, Transformer)

Transformer layers capture semantic meaning of the text.

[CLS]

STAGE 2: CONTRASTIVE LOSS

5. CONTRASTIVE SIMILARITY MATRIX



$$\text{Contrastive Loss} = \sum_{i,j} \begin{cases} (T_i, I_j) \cos(T_i, I_j) & \text{if } i, j \text{ is a positive pair} \\ -\cos(T_i, I_j) & \text{if } i, j \text{ is a negative pair} \end{cases}$$



Zero-Shot Image Classification
Compare image embedding with class text embeddings.



Image Retrieval
Query text embedding is matched to visual patch embeddings.

3. APPLICATIONS AND LEARNING



LEARNED SHARED EMBEDDING SPACE

The model learns to project visual patches and corresponding text into a joint embedding space. Patches of "grass" are close to the text token "grass" and the overall image 'dog in grass' is close to its description.



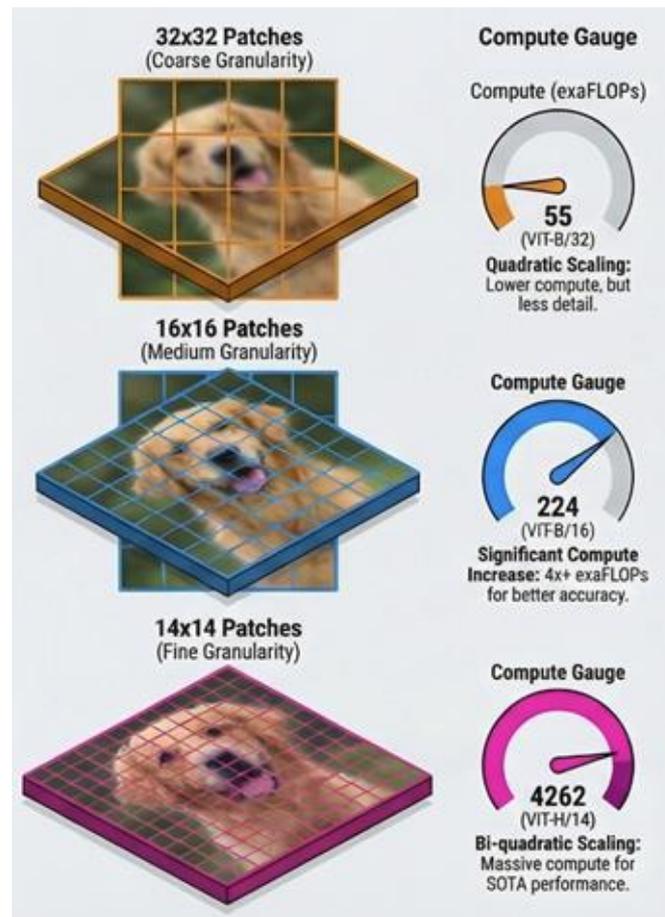
VISUAL PATCHES TEXT PROMPT JOINT EMBEDDING ZERO-SHOT



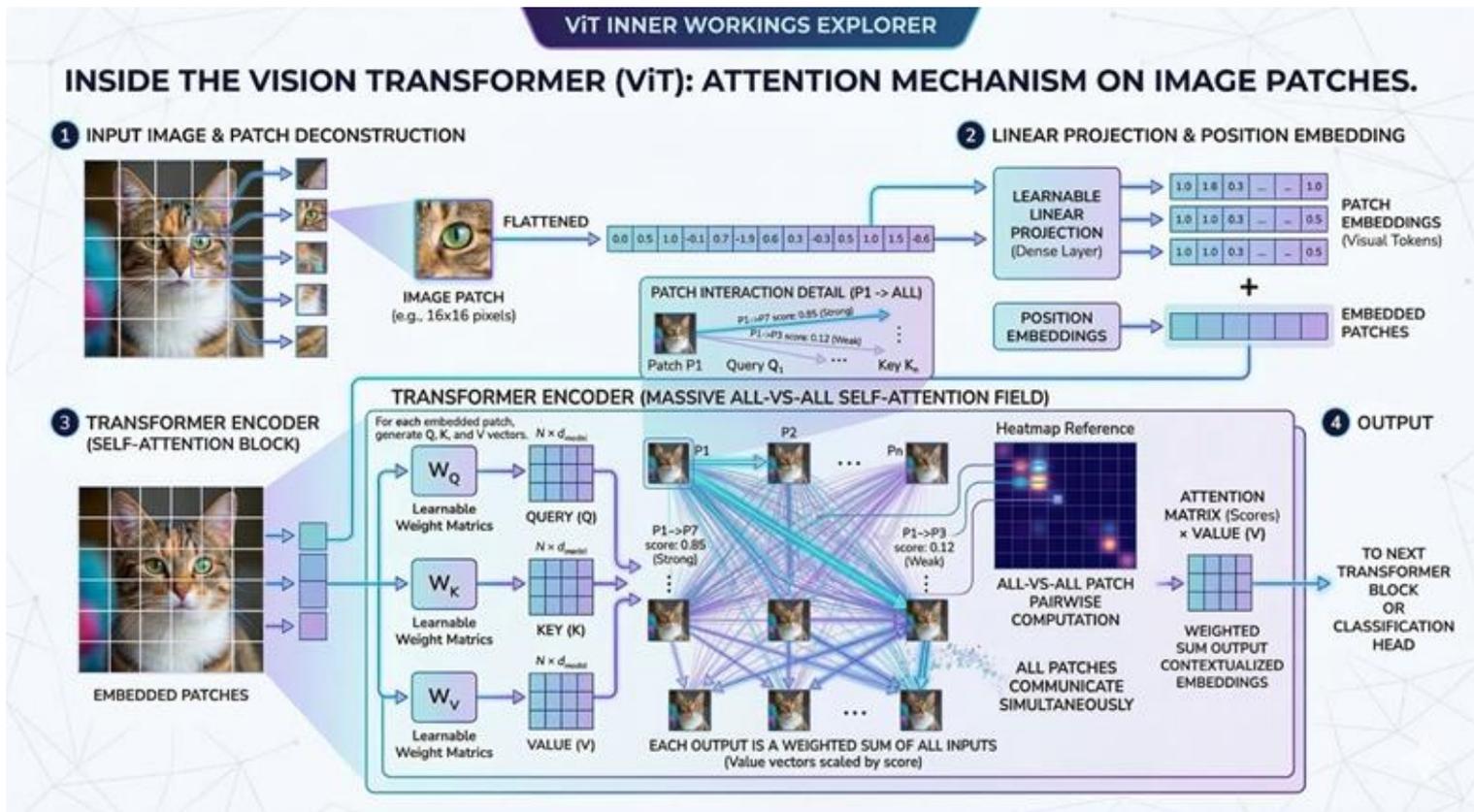
CNN vs ViT in Practice

"Is this a bottle or a cap?"

- CNN approach:
 - pixels \rightarrow edges \rightarrow shapes \rightarrow cylinder \rightarrow bottle \rightarrow yes
 - 50 layers of local processing
- ViT approach:
 - [bottle patch] \leftarrow attention \rightarrow [cap patch]
 - One attention step. Sees both at once.
- The tradeoff:
 - ViTs are data-hungry. No built-in spatial bias.
 - CLIP's 400M pairs was the breakthrough.



CNN vs ViT in Practice



From "What" to "Where"

How SAM3 works: text prompt becomes bounding box

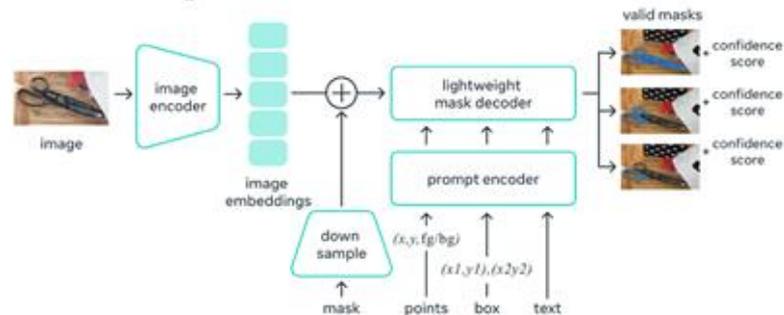
The problem:

- CLIP tells me WHAT is in the image.
- But I need WHERE. I need bounding boxes.

The solution:

- Instead of ONE vector per image...
- create a DENSE GRID of vectors. (basically just the ViT approach)
- One vector at every spatial position.

Universal segmentation model



Like a chessboard - every square has its own fingerprint.

Embedding similarity is the key: <https://huggingface.co/spaces/webml-community/dinov3-web>



The Key Insight

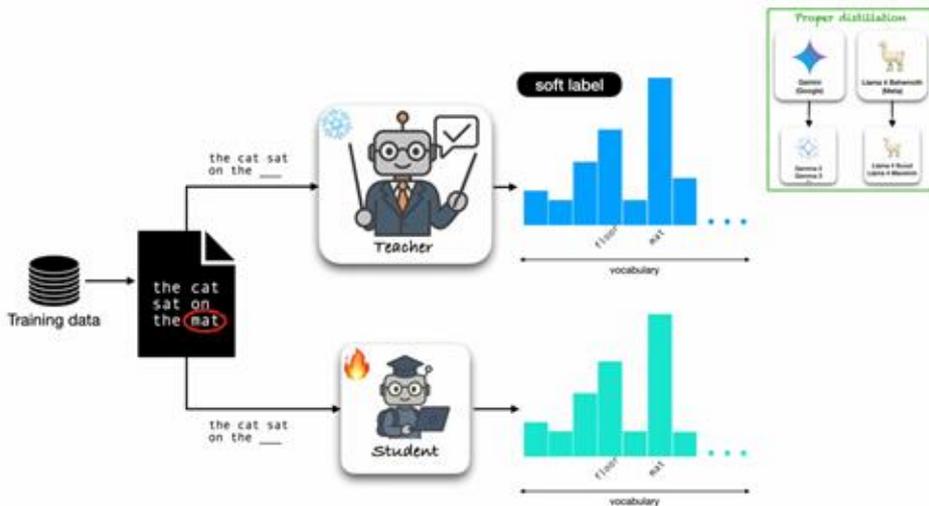
Annotation IS Distillation

Manual annotation:

- Human looks at image, draws box, writes label
- The human brain is the teacher model

Foundation model annotation:

- SAM3 looks at image, draws box, writes label
- 400M image-text pairs is the teacher's education



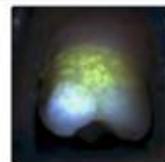
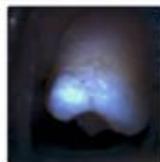
What Actually Goes Wrong (And How We Fix It Today)



The Data Problem Nobody Budgeted For

Pitfall 1: The Annotation Treadmill

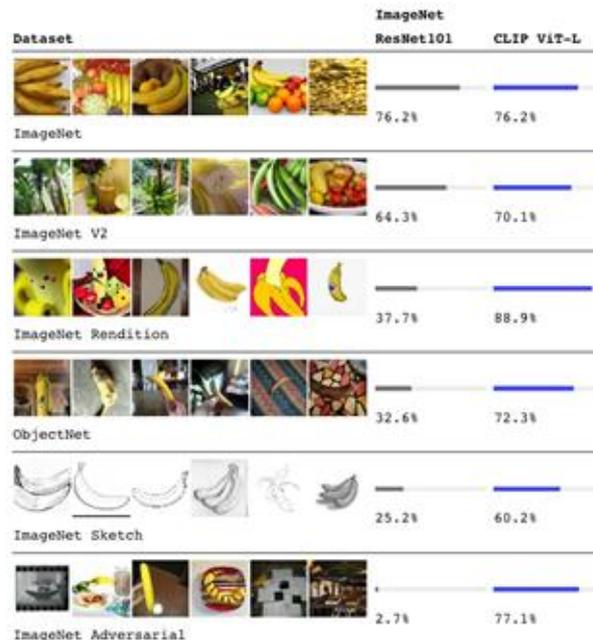
- Hardware gen 1 → Collect → Annotate → Train → Ship ✓
 - Hardware gen 2 → New sensor → 40% labels stale → Re-annotate
 - Hardware gen 3 → New optics → 60% labels stale → Re-annotate
 - Hardware gen 4 → ...
-
- Each cycle: \$50K+ and 3-4 months
 - The model was fine.
 - The data pipeline was the bottleneck.



The Data Problem Nobody Budgeted For

Pitfall 1: The Annotation Treadmill

- Each cycle: \$50K+ and 3-4 months
- The model was fine.
- The data pipeline was the bottleneck.
- **Fix:**
 - **Use heavy data augmentation to inject priors**
 - **Generate synthetic data**
 - **Try fine tuning foundation-level models for more stable features, allowing for faster re-annotation cycles**
 - Reminder from Day 1 about better of open vocabulary models generalizability →



"Just Use a Better Model"

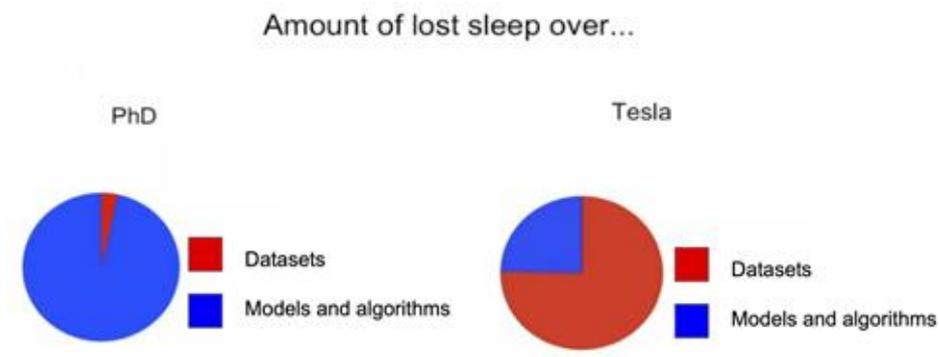
Pitfall 2: Model-Centric Thinking

The instinct when accuracy drops:

- → Try YOLOv26 instead of v11
- → Larger model variant
- → Tune hyperparameters for 2 weeks

What actually moves the needle:

- → Fix 200 mislabeled images +3 mAP
- → Remove ambiguous objects +2 mAP
- → 500 targeted images for hard class +5 mAP
- → Right prompt for auto-labeling +15 mAP



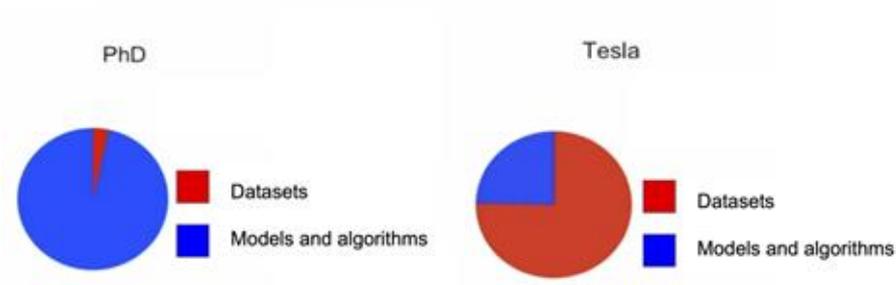
"Just Use a Better Model"

Pitfall 2: Model-Centric Thinking

What actually moves the needle:

- → Fix 200 mislabeled images +3 mAP
- → Remove ambiguous objects +2 mAP
- → 500 targeted images for hard class +5 mAP
- → Right prompt for auto-labeling +15 mAP

Amount of lost sleep over...



Fix:

- **No shortcuts, look at the data**
- **Uniqueness / Mistakeness metrics help**
- **Visualize your failures and your good cases, think why they occur**

"Works in the Notebook"

Pitfall 3: The Deployment Gap

Your dev setup:

- A100 GPU • 80GB VRAM • unlimited power

Your deployment target:

- Jetson Orin Nano • 8GB • 15W • <50ms latency

- SAM3: 2.9B params | ~200ms on A40
- YOLO26: 2.4M params | ~1.7ms on T4

- That's 1,200× smaller.
- You don't deploy the teacher. You distill it.



"Works in the Notebook"

Pitfall 3: The Deployment Gap

Your dev setup:

- A100 GPU • 80GB VRAM • unlimited power

Your deployment target:

- Jetson Orin Nano • 8GB • 15W • <50ms latency

Fix:

- **Requirements, requirements, requirements**
- **...or is it?**



"Works in the Notebook"

Pitfall 3: The Deployment Gap

Your dev setup:

- A100 GPU • 80GB VRAM • unlimited power

Your deployment target:

- Jetson Orin Nano • 8GB • 15W • <50ms latency

Fix:

- **Requirements, requirements, requirements**
- **...or is it?**
- **Thinking outside the box, stakeholder management, holistic usecase & business thinking**



Today's Challenge



Your challenge today:

You have been contracted to develop a system evaluating the worksite safety compliance of workers, i.e. What is the percentage of workers wearing hard hats?

1. Generate the synthetic data
 - a. Done
2. Inspect it, think about it, try out different prompts, strategies of how you will approach the problem
 - a. Experiment with SAM3, Qwen3.5, YOLOE26
3. Auto-annotate the data
 - a. Inspect the annotations, think about the fail cases, are you happy
4. Train a model & evaluate its performance
 - a. Bonus points: deploy / measure throughput / latency



Let's Build

