Transfer Learning in Neural Networks

Reuse Knowledge, Save Data, Improve Results

Jason Cui

March 19, 2026

Roadmap

1. Big Picture and Core Idea
2. Workflow and Strategies
3. Example and Interpretation
4. Takeaways

What Is Transfer Learning and Why Use It?

One-sentence idea

Transfer learning reuses what a model learned before, so a new task can be solved faster and better.

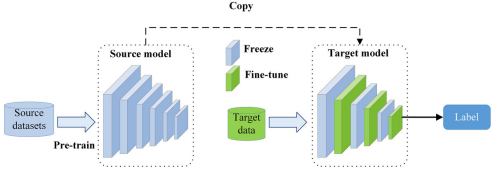
Why it is necessary in practice

- Labeled data is expensive, and annotation quality can vary.
- The dataset may be too small to train a large model from scratch reliably.
- Training from scratch needs much more compute and tuning effort.
- Pretrained weights usually improve optimization stability.

To formalize this setting, we denote source and target data/task pairs as follows:

$$(\mathcal{D}_S, \mathcal{T}_S) \to (\mathcal{D}_T, \mathcal{T}_T), \quad P_S(x, y) \neq P_T(x, y)$$

$\mathcal{D}$: dataset, $\mathcal{T}$: task, S/T: source/target, $P(x, y)$: distribution.



← Source-to-target transfer illustration.

Basic Workflow and the Rationale Behind Each Step

Step 1: Start with a pretrained model

- Use a backbone pretrained on large-scale data (e.g., ImageNet).
- Why: it already learns generic features, so training is faster and more stable.

Step 2: Replace the task head

- Swap the original output layer with a new head for your target labels.
- Why: source and target label spaces are usually different.

Step 3: Adapt on target data

- Train the new head first, then unfreeze upper backbone layers if needed.
- Use data augmentation (e.g., random crop/flip/color jitter) as a training trick to improve generalization.
- Tip: smaller LR for pretrained layers, larger LR for the new head.

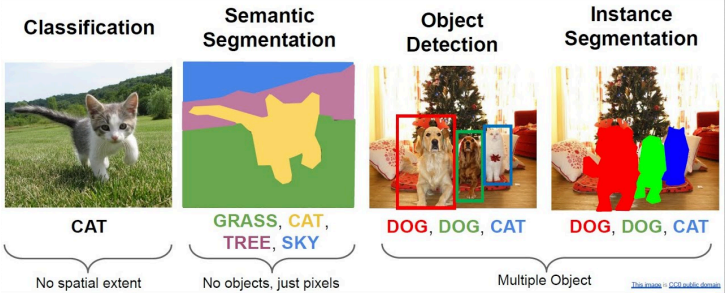
Four Core Vision Tasks and Their Differences

Task definitions

- **Classification**: predict one label for the whole image.
- **Object Detection**: **classification + rough localization** using bounding boxes.
- **Semantic Segmentation**: **per-pixel classification**; same-class objects share one mask.
- **Instance Segmentation**: detection + segmentation; each object instance has its own mask.

Why these tasks matter

- They represent increasing understanding levels: image-level → object-level → pixel-level.
- They support real applications such as autonomous driving, robotics, and medical imaging.



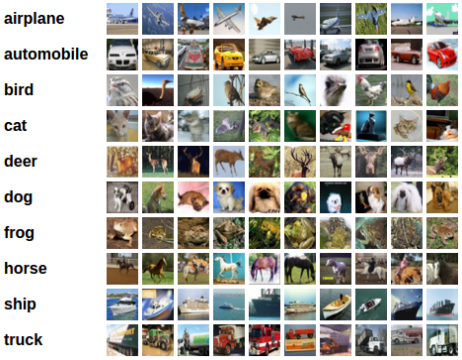
Transfer Learning Example: Classification (CIFAR-10)

This is an example of **image classification** transfer learning.

- Source pretraining: **ImageNet-pretrained** vs **random initialization**.
- Target dataset: **CIFAR-10** (10 classes, 32×32 RGB images).
- Model: ResNet-18.
- Training: same settings and 10 epochs for fair comparison.

Input and output definition

- Input x: one image, $x \in \mathbb{R}^{32\times32\times3}$.
- Output $\hat{y}$: one class label (or a 10-dimensional probability vector).



Classification Metrics

Overall accuracy (OA)

$$\mathrm{OA} = \frac{\sum_{c=1}^{C} TP_c}{N}$$

Fraction of all samples predicted correctly.

Average accuracy (AA)

$$\mathrm{AA} = \frac{1}{C} \sum_{c=1}^{C} \frac{TP_c}{TP_c + FN_c}$$

Mean per-class recall; reduces majority-class dominance.

Kappa coefficient

$$\kappa = \frac{p_o - p_e}{1 - p_e}, \quad p_o = \mathrm{OA}, \quad p_e = \sum_{c=1}^{C} \left(\frac{n_{c,\mathrm{true}}}{N} \cdot \frac{n_{c,\mathrm{pred}}}{N} \right)$$

Here $n_{c,\mathrm{true}}$ and $n_{c,\mathrm{pred}}$ are true/predicted counts of class c.

Per-class precision and recall

$$\mathrm{Precision}_c = \frac{TP_c}{TP_c + FP_c}, \qquad \mathrm{Recall}_c = \frac{TP_c}{TP_c + FN_c}$$

Precision: how reliable positive predictions are; recall: how many true samples are found.

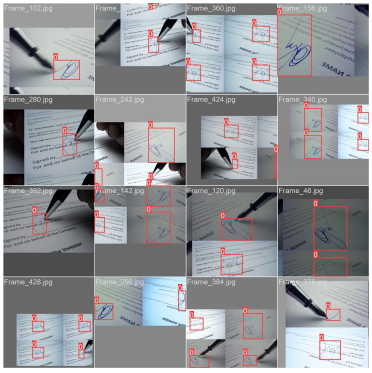
Transfer Learning Example: Object Detection (Signature)

This is an example of **object detection** transfer learning.

- Source pretraining: **COCO-pretrained** vs **random initialization**.
- Target dataset: **Signature** dataset.
- Model: YOLO11n.
- Training: same settings and 10 epochs for fair comparison.

Input and output definition

- Input x: one document image.
- Output $\hat{y}$: bounding boxes + confidence scores for signature regions.



Detection Metrics

IoU for one prediction

$$\mathrm{IoU} = \frac{|B_{pred} \cap B_{gt}|}{|B_{pred} \cup B_{gt}|}$$

A detection is a TP when class is correct and IoU exceeds a threshold.

Single-class AP metrics

TP at threshold t : $\mathrm{IoU}(B_{pred}, B_{gt}) \geq t$ and class is correct, $\quad \mathrm{AP}_t = \int_0^1 P_t(r)\, dr, \quad \mathrm{AP}_{50} = \mathrm{AP}_t\big|_{t=0.50},$

r is recall, $P_t(r)$ is the precision–recall curve built using IoU threshold t; $\mathrm{AP}_{50:95}$ averages stricter IoU thresholds and is more comprehensive.

Multi-class mAP

$$\mathrm{mAP}_{50} = \frac{1}{C}\sum_{c=1}^{C} \mathrm{AP}_{50}^{(c)}, \qquad \mathrm{mAP}_{95} = \frac{1}{C}\sum_{c=1}^{C} \mathrm{AP}_{95}^{(c)}, \qquad \mathrm{mAP}_{50:95} = \frac{1}{C}\sum_{c=1}^{C} \mathrm{AP}_{50:95}^{(c)}$$

Also report precision/recall and F1 at a chosen confidence threshold for deployment.

When It Works, Fails, and What To Do

When transfer works well

- Source and target are semantically related.
- Rationale: shared structure means reusable features.

When it can fail

- Large domain gap or mismatched objectives.
- Rationale: source features may bias learning to wrong cues.

What to do

- Start with feature extractor as baseline.
- Unfreeze gradually from top layers downward.
- Use smaller LR for pretrained layers than for new head.
- For small batch sizes, freeze BatchNorm statistics (or use GroupNorm) to improve stability.
- Track validation curve and early-stop when needed.

Summary + Common Mistakes

Key takeaways

1. Transfer learning is a practical paradigm, not just a trick.
2. Key tradeoff: stability vs adaptability.
3. Backbone + head decomposition enables cross-task reuse.
4. Good engineering (LR, unfreezing, validation) matters.

Common mistakes

- Forgetting to replace the final classification layer.
- Using too large LR for pretrained parameters.
- Evaluating only on training data.
- Reporting one run without variance.

Q&A

Questions?