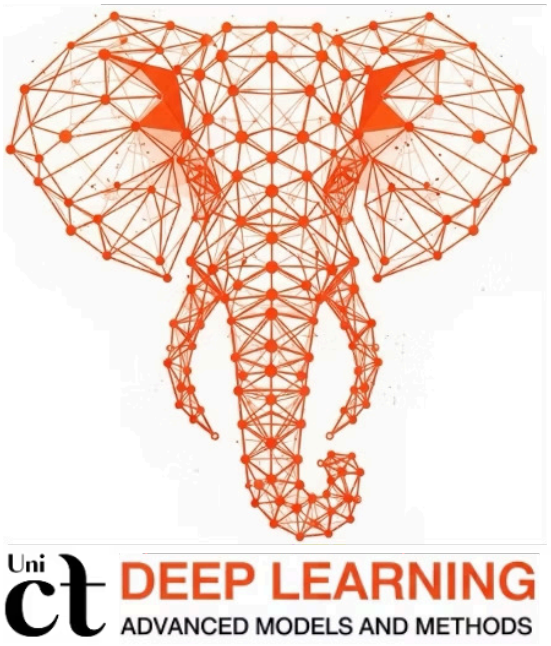


Deep Learning
Advanced Models & Methods

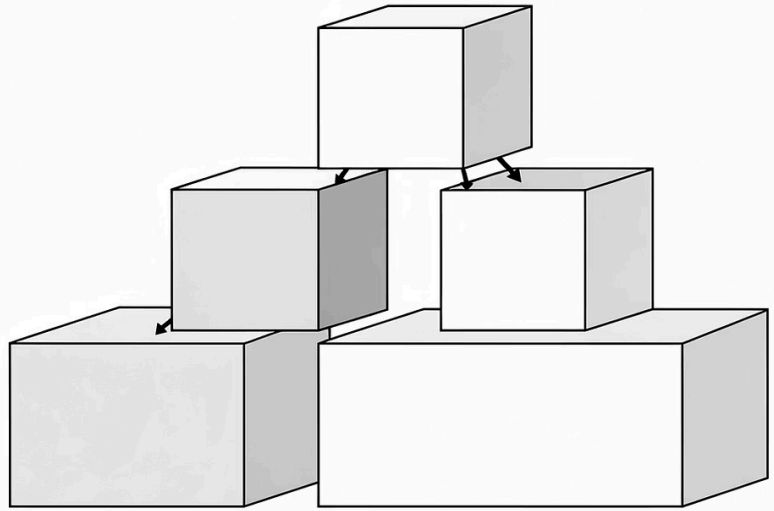
Prof. Antonino Furnari (antonino.furnari@unict.it)
Corso di Laurea Magistrale in Informatica
Dip. di Matematica e Informatica
Università di Catania

Knowledge Distillation

Module 1

Motivation & Conceptual Foundations

Focus: Intuition, Theory, and the "Why"



THE PROBLEM

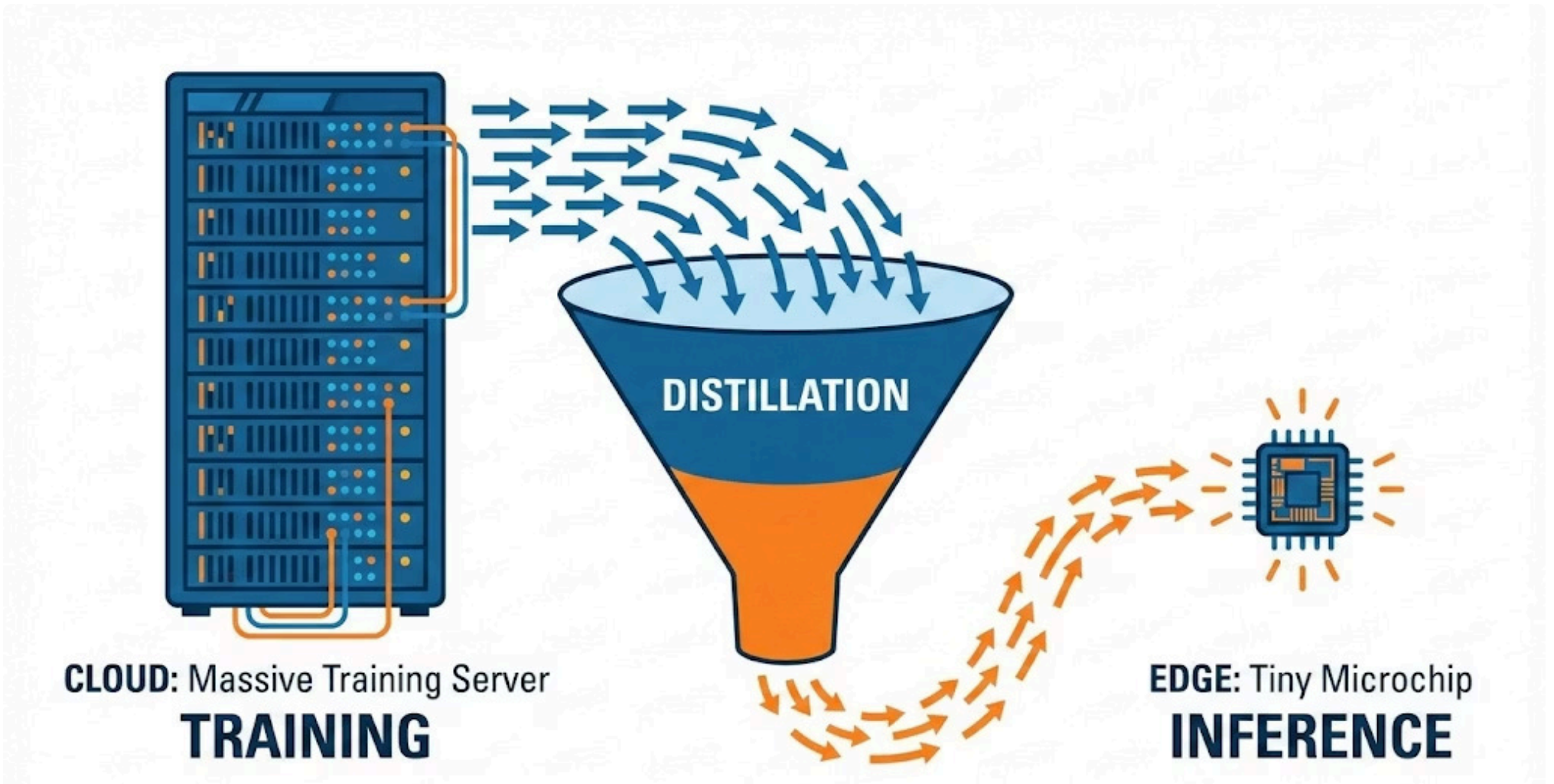
The Inference Gap – Cloud vs. Edge

The Training Reality

Modern deep learning models are trained on massive cloud infrastructure featuring NVIDIA A100 or H100 GPUs, with terabytes of memory, unlimited power budgets, and multi-day training windows. This environment enables large-scale models with billions of parameters.

The Deployment Constraint

However, deployment happens on drastically different hardware: mobile phones, IoT sensors, autonomous vehicles, and edge devices like Jetson Nano. These systems face strict latency requirements (often sub-100ms), limited memory (megabytes, not gigabytes), and severe power constraints.



KD Philosophy: Train a compact "Student" model to function effectively in the inference environment by leveraging the rich probability distributions and internal representations learned by a powerful "Teacher" model during training.

THE PROBLEM

The Overparameterization Paradox

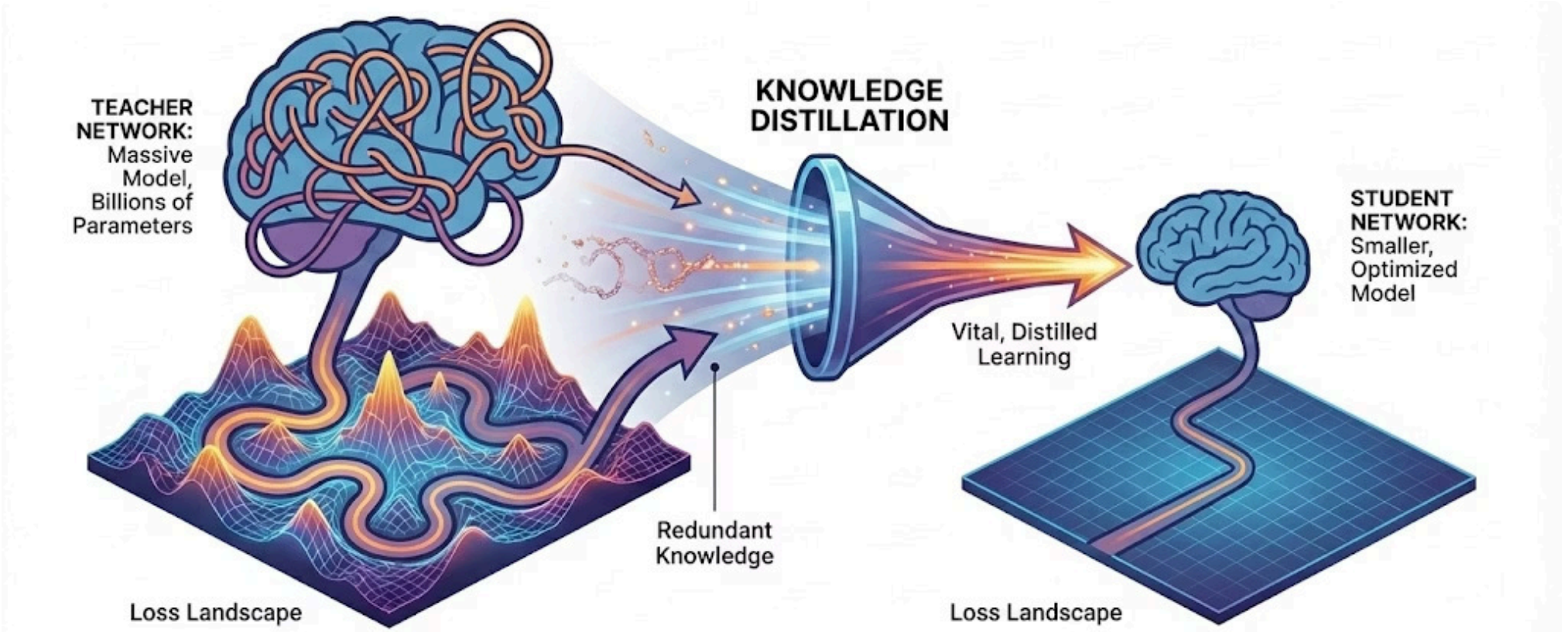
Modern deep learning models are often heavily overparameterized. While this capacity is crucial for navigating complex loss landscapes and finding high-performing solutions, it frequently leads to networks that contain significant redundancy once trained.

Teacher Network

Massive models with billions of parameters efficiently find solutions but often encode redundant, implicit knowledge.

Student Network

Smaller, optimized networks can learn to express the same core solution efficiently by extracting the teacher's 'essence'.



The main hypothesis is that we can distill a large teacher network into a smaller student network. Here, the large model "shows the way" to the small model.

THEORY

Learning Using Privileged Information (LUPI)

Vapnik's LUPI Framework

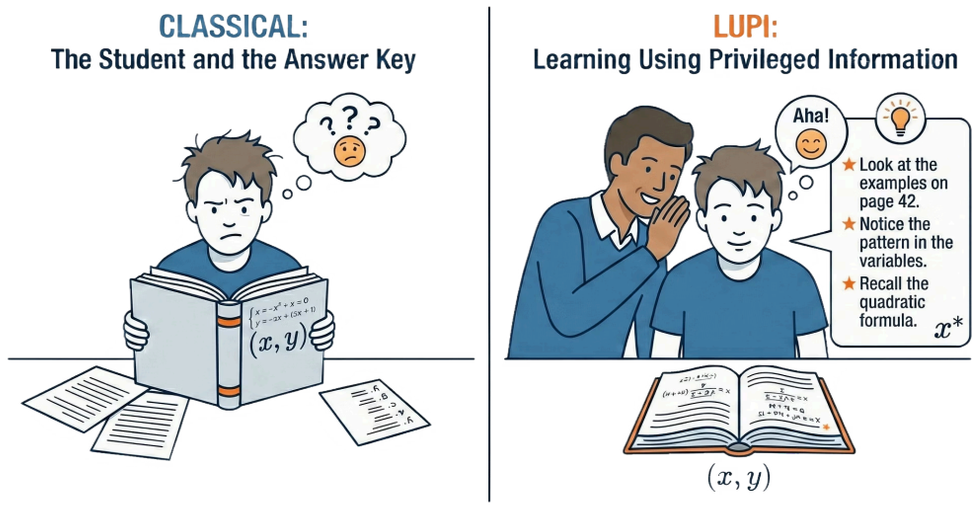
Knowledge Distillation has been proposed in similar forms by Geoffrey Hinton and Vladimir Vapnik in 2015. Vapnik provided a mathematical framework named Learning Using Privileged Information (LUPI).

Classical Learning Paradigm

The student learns from pairs (x, y) where x is the input and y is the ground truth label. The learning rate scales as $O(1/\sqrt{n})$ where n is the number of training samples.

LUPI Paradigm

The student learns from triplets (x, x*, y), where x* represents **Privileged Information**—additional context available only during training.



The Acceleration Mechanism: Vapnik applied this principle to SVM. In this context, the teacher uses the privileged information x* to provide corrective signals about similarity structure (e.g., providing slack value for examples), fundamentally accelerating the learning rate from $O(1/\sqrt{n})$ to $O(1/n)$. This theoretical speedup explains the empirical success of distillation across diverse domains.

Vapnik, V., & Izmailov, R. (2015). "Learning Using Privileged Information: Similarity Control and Knowledge Transfer". JMLR.

Model Compression

This technical approach, proposed by Buciluǎ et al. (2006), outlines a three-step process for distilling a large ensemble's knowledge into a compact neural network. This is a precursor to model Neural Knowledge Distillation.

01

Training the Ensemble

Build a diverse ensemble using methods like bagging, boosting, SVMs, or decision trees. An ensemble contain different models trained independently. It can be used to make predictions on data by averaging individual predictions made by the models.

02

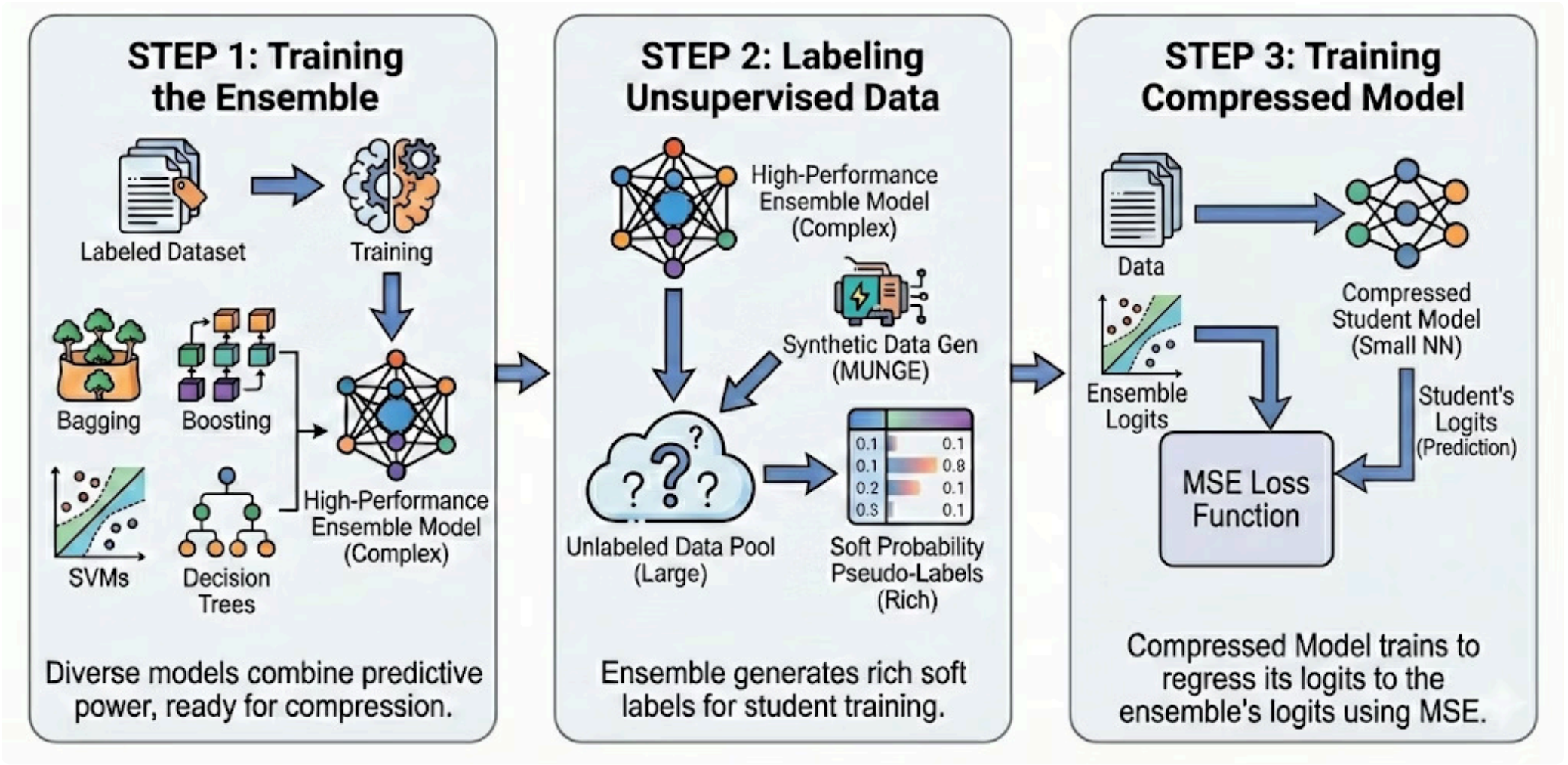
Labeling Unsupervised Data

Leverage the trained ensemble to generate soft probability predictions for a large, unlabeled dataset. These serve as rich pseudo-labels for the student model. If no unlabeled data is available, use synthetic data generation techniques (MUNGE).

03

Training Compressed Model

Train a small neural network (typically 1-3 hidden layers) using the pseudo-labeled data. The loss function is Mean Squared Error (MSE) pushing the small network's prediction with the average logits produced by the ensemble on each example.



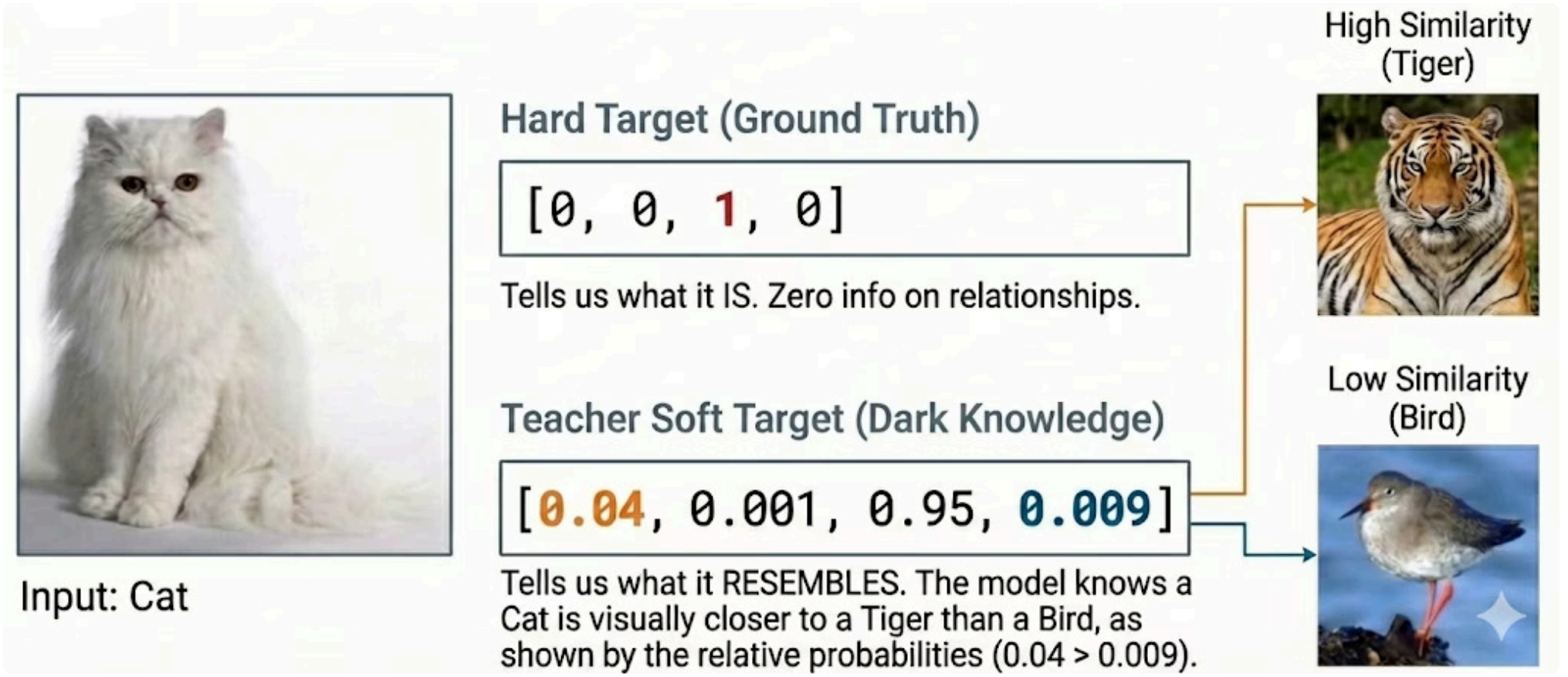
This approach achieved remarkable compression, making the student model 1000× smaller and 1000× faster than the original ensemble. Crucially, it retained approximately 97% of the ensemble's performance improvement, proving effective for deployment on resource-constrained devices like mobile phones and PDAs.

> **Note**: We can see the ensemble's logits as Vapnik's priviledged information in this case. They encode more than simple labels as they implicitly include uncertainty information.

Buciluǎ, C., Caruana, R., & Niculescu-Mizil, A. (2006). "Model Compression". KDD 2006.

Dark Knowledge

In 2015 Hinton formulates the concept of "dark knowledge" residing into the probability estimates of a trained neural network. While hard labels only encode information on the identity of an example, the predicted probabilities also encode "dark knowledge" on the similarity with negative classes.



The Problem with Hard Labels

Traditional training uses one-hot encoded labels like [0, 1, 0], which does not encode the relative similarity information between classes. The model learns nothing about how similar "dog" is to "wolf" versus "car".

Dark Knowledge Revealed

The key insight: the "tail" of the probability distribution—those small non-zero probabilities for incorrect classes—contains crucial structural knowledge. For example, a model might assign 0.8 to "dog", 0.15 to "wolf", and 0.001 to "car", revealing learned similarity.

Using a learned model to extract dark knowledge

A model trained on the target dataset encodes this "dark knowledge" by encoding uncertainty about predictions. Indeed, for a cat a model may predict small non-zero probabilities for similar classes such as "tiger".

Hinton, G., Vinyals, O., & Dean, J. (2015). "Distilling the Knowledge in a Neural Network". NeurIPS Workshop.

The Power of Temperature in Knowledge Distillation

Trained neural networks often produce highly confident predictions, assigning probabilities very close to 1 for the predicted class and near 0 for all others. While good for classification, this extreme certainty hides valuable "dark knowledge" about the subtle relationships between classes. Temperature scaling helps us unveil this hidden information.

Overconfident Predictions

Standard models, after extensive training, output very "sharp" probability distributions, where the correct class gets a near-1 probability and alternatives get near-0. This aggressive confidence masks intricate inter-class similarity.

Softmax's Sharpness

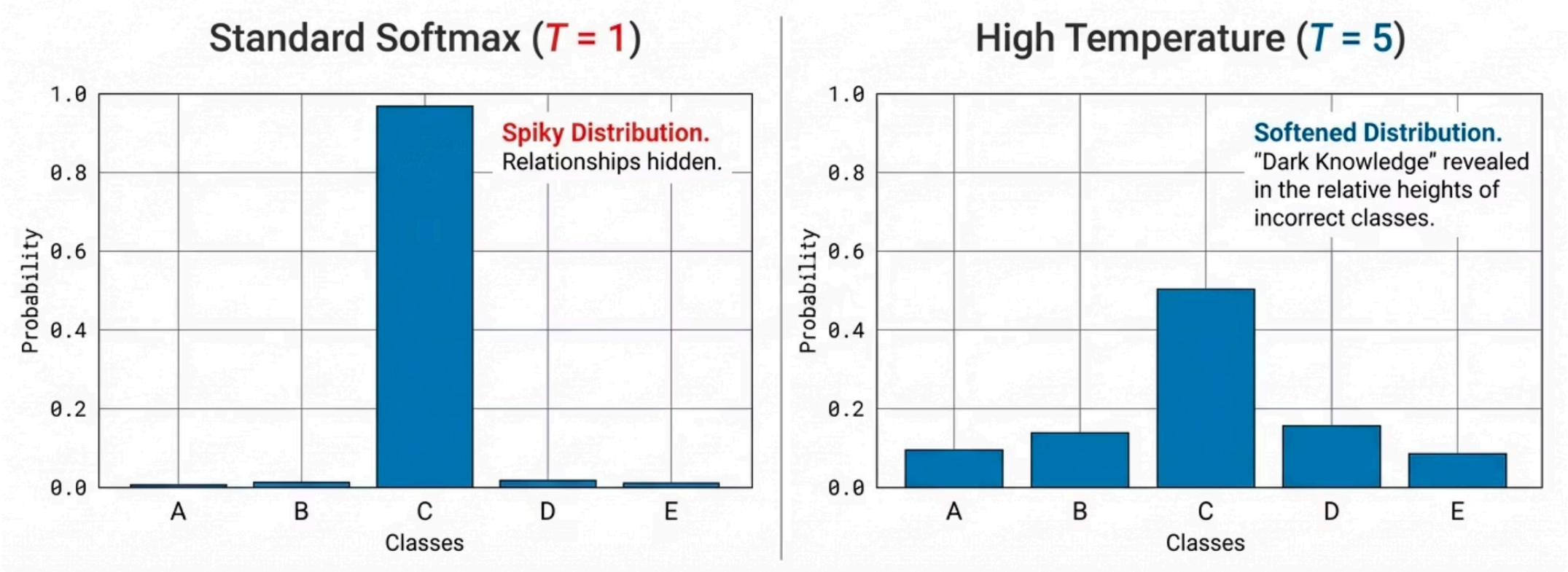
The softmax function, coupled with cross-entropy loss, is designed to maximize the probability of the correct class. This process naturally pushes the logits (raw outputs) for the correct class much higher, leading to highly peaked probability distributions.

Introducing Temperature (T)

Temperature (**T**) is a hyperparameter applied to the logits before the softmax function. By dividing the logits by T, we effectively 'soften' the resulting probability distribution, making it less extreme and more informative.

Scaling Logits

When T > 1, the logits are scaled down, producing a smoother probability distribution that highlights the small, non-zero probabilities for incorrect but similar classes. A higher T value yields softer probabilities, while T=1 reverts to the standard softmax.



> **Note**: this "dark knowledge" can be seen as Vapnik's privileged information, provided by an "intelligent teacher" to a "student" model.

Hinton, G., Vinyals, O., & Dean, J. (2015). "Distilling the Knowledge in a Neural Network". NeurIPS Workshop.

Crucial Distinction – KD vs. Label Smoothing

The Misconception

"Why do we need a complex teacher model? Can't we just manually set soft labels to [0.9, 0.05, 0.05] for regularization?"

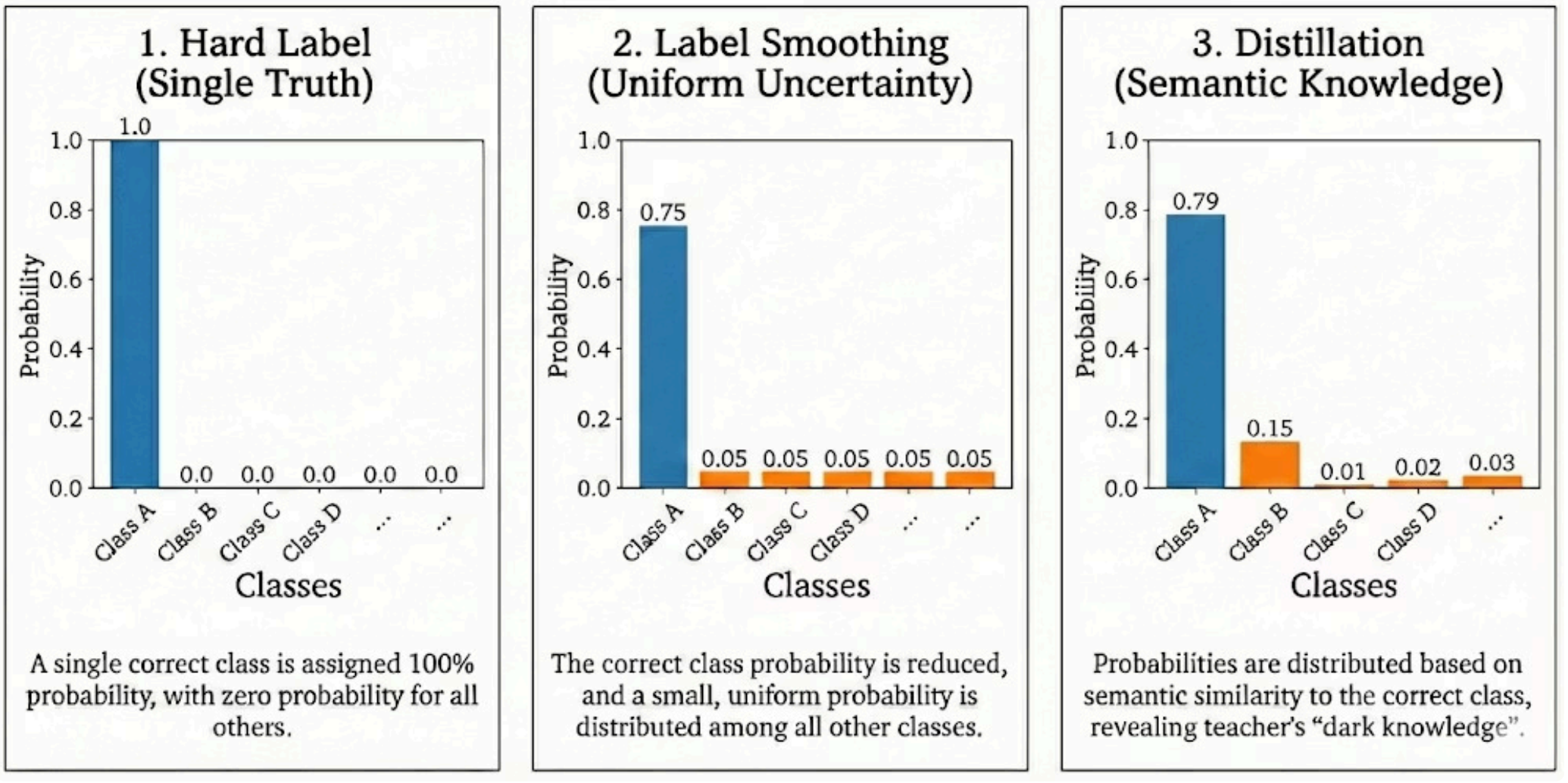
This question conflates two fundamentally different mechanisms with different objectives and outcomes.

Label Smoothing: Uniform Regularization

Label smoothing assigns *uniform* probability mass to all incorrect classes. It tells the model "Don't be overconfident," preventing overfitting. However, it treats all errors equally: confusing "Dog" with "Wolf" receives the same penalty as confusing "Dog" with "Car".

Knowledge Distillation: Structural Transfer

KD assigns *structured* probability based on learned similarity. It tells the model "This German Shepherd looks 12% like a Wolf, 8% like a Fox, but only 0.001% like a Car." This preserves the geometric structure of the data manifold.



Müller et al. (2019) demonstrated that label smoothing helps primarily when the model capacity matches the task complexity, while KD consistently improves student performance regardless of capacity by transferring genuine structural knowledge.

Müller, R., Kornblith, S., & Hinton, G. E. (2019). "When Does Label Smoothing Help?". NeurIPS.

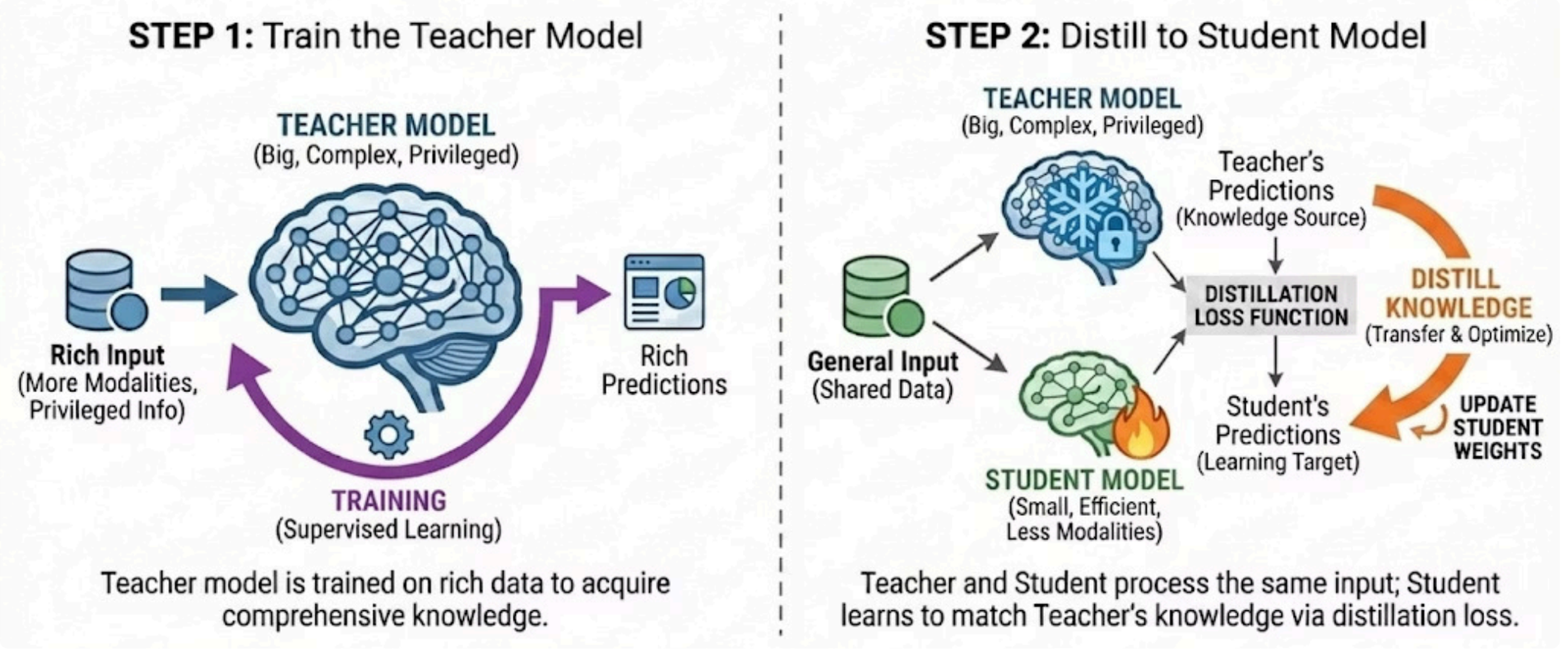
The General Approach

1. Train the Teacher Model

This model is typically large, having access to more modalities or privileged information, capturing deep insights from the data.

2. Distill to Student

A smaller, more efficient student model is trained. Both teacher and student process the same input in parallel, with knowledge transferred from the teacher to guide the student's learning.



Hinton, G., Vinyals, O., & Dean, J. (2015). "Distilling the Knowledge in a Neural Network". NIPS Workshop.

The Taxonomy of Knowledge Distillation

Response-Based Distillation

Focuses on mimicking the teacher's final output predictions—the probability distributions over classes.
Best for: Classification tasks where output structure matters most.

Feature-Based Distillation

Targets intermediate layer representations, forcing the student's internal features to match the teacher's.
Best for: Tasks requiring rich internal representations like detection and segmentation.

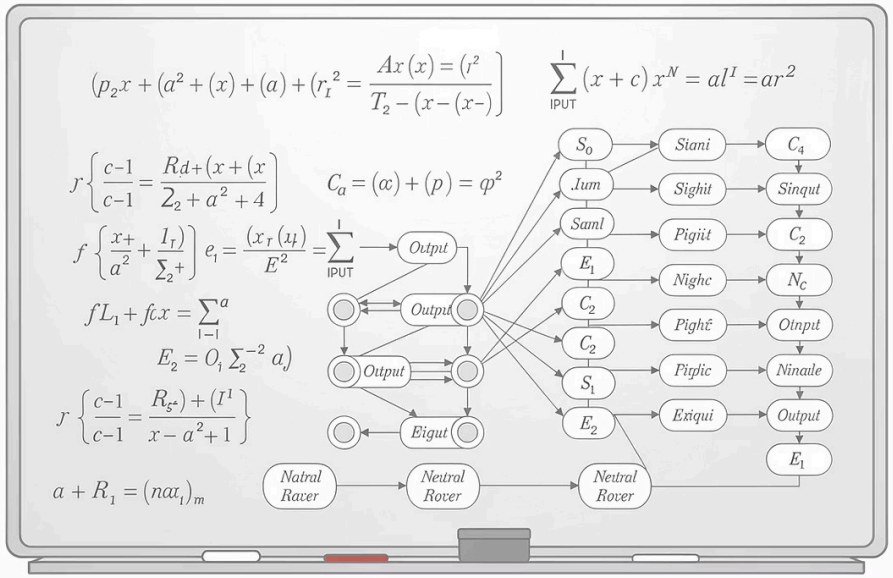
Relation-Based Distillation

Captures structural relationships between data samples or feature maps, preserving the geometric manifold.
Best for: Few-shot learning and transfer learning scenarios.

Gou, J., Yu, B., Maybank, S. J., & Tao, D. (2021). "Knowledge Distillation: A Survey". International Journal of Computer Vision.

Module 2

Methodologies by Taxonomy



Response-Based I – Vanilla Knowledge Distillation

This is Knowledge Distillation approach, as originally introduced by Hinton.

1	2	3

Teacher

A large model trained in a standard supervised way (e.g., cross-entropy for classification) on the training set.

Student

A smaller model. Knowledge is distilled from the teacher to the student during training to allow the small student achieve good performance guided by the teacher.

Training Paradigm

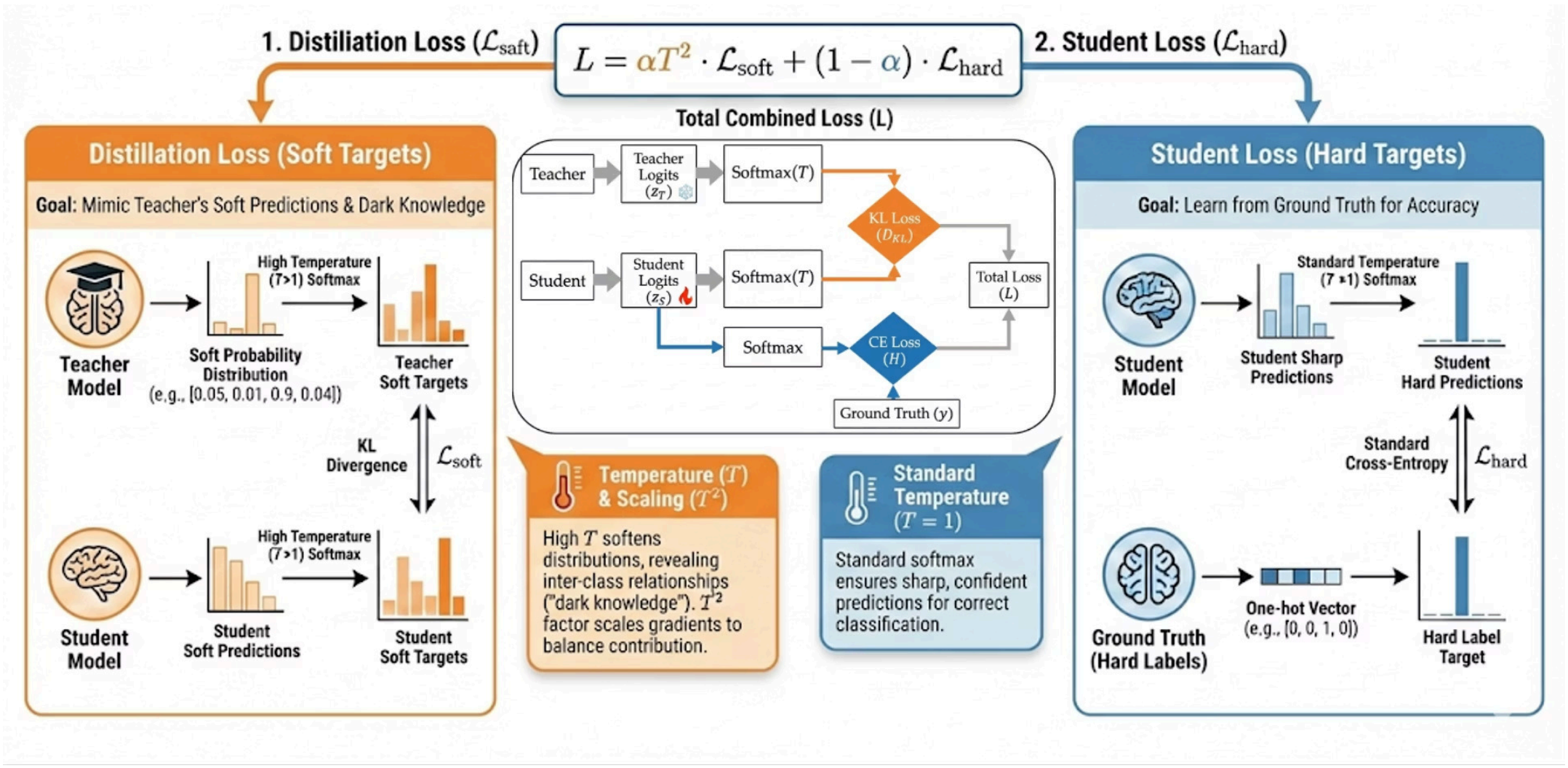
- The teacher is pre-trained, then frozen;
- The student is trained with standard cross entropy with hard labels ($\mathcal{L}_{hard}$) + a Kullback-Leibler (KL) loss minimizing the divergence between predicted and soft target probabilities ($\mathcal{L}_{soft}$).
- Temperature is added only for the KL loss (T set to 1 for standard CE)

The $\mathcal{L}_{soft}$ loss function is based on the Kullback-Leibler (KL) divergence, which quantifies the cost of using probability q instead of p:

$$D_{KL}(p||q) = \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)}$$

In the context of Knowledge Distillation, q is the probability distribution predicted by the student , while p is the target teacher probability.

The total loss function is described in the following:



Critical Implementation Detail: The KL divergence gradient must be scaled by T^2 to maintain effective learning signals. This scaling compensates for the temperature-induced smoothing of the distribution and ensures gradients remain properly balanced. The parameter α balances the contribution of the two loss functions. Typical values are in the 0.7 - 0.9 range.

Hinton, G., Vinyals, O., & Dean, J. (2015). "Distilling the Knowledge in a Neural Network". NIPS Deep Learning Workshop.

Response-Based II – Decoupled Knowledge Distillation

The Core Insight

Vanilla KD unknowingly entangles two distinct learning signals:

1. **Target Class Knowledge (TCKD):** How confident should the model be? This reflects sample difficulty.
2. **Non-Target Class Knowledge (NCKD):** How should probability distribute among wrong classes? This captures structural similarity—the true "dark knowledge".

Indeed, given a training sample from ground truth class t, it can be shown that the standard KL loss can be seen as:

$$KD = KL(\mathbf{p}^{\mathcal{T}}||\mathbf{p}^{\mathcal{S}}) = \underbrace{KL(\mathbf{b}^{\mathcal{T}}||\mathbf{b}^{\mathcal{S}})}_{TCKD} + (1 - p_t^{\mathcal{T}})\underbrace{KL(\hat{\mathbf{p}}^{\mathcal{T}}||\hat{\mathbf{p}}^{\mathcal{S}})}_{NCKD}$$

where:

- $\mathbf{p}^{\mathcal{T}}$ and $\mathbf{p}^{\mathcal{S}}$ are the teacher and student probability distributions respectively
- $\mathbf{b}$ is a binary probability distribution of the ground truth class, constructed as $\mathbf{b} = [p_t, p_{\setminus t}]$, such that:

$$p_t = \frac{\exp(z_t)}{\sum_{j=1}^{C} \exp(z_j)} \qquad p_{\setminus t} = \frac{\sum_{k=1, k\neq t}^{C} \exp(z_k)}{\sum_{j=1}^{C} \exp(z_j)}$$

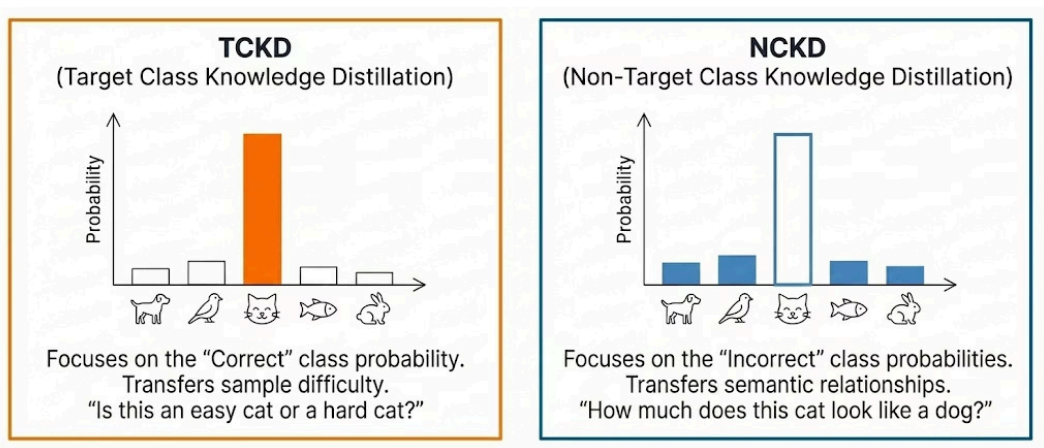
  where z_t are logits and $\mathbf{b}^{\mathcal{T}}$ and $\mathbf{b}^{\mathcal{S}}$ denote teacher and student distributions respectively. For example $\mathbf{b}^{\mathcal{T}} = [0.8, 0.2]$.

- $\hat{\mathbf{p}}$ is the distribution of non-target classes, defined by re-normalizing logits associated to non-target classes: $\hat{\mathbf{p}} = [\hat{p_1}, \ldots, \hat{p_{t-1}}, \hat{p_{t+1}}, \ldots, \hat{p_C}]$ such that:

$$\hat{p}_i = \frac{\exp(z_i)}{\sum_{j=1, j\neq t}^{C} \exp\left(z_j\right)}$$

In the formula above:

- TCKD - Target Class Knowledge Distillation: distills binary probabilities of target classes. Teacher targets encode the "hardness" of an example. A large teacher probability means that the example is a hard one (even the teacher is struggling).
- NCKD - Non-target Class Knowledge Distillation: distills probability distributions of non-target classes. Teacher targets here encode the relationship between this example and non-target classes (what does this example look like, beyond the target class). Experiments show that this is the most effective term.



In this view, vanilla Knowledge Distillation entangles the two by weighing NCKD by the hardness of the example $(1 - p_t^{\mathcal{T}})$. In practice, if the example is an easy one, NCKD is not enforced at all.

The DKD Solution

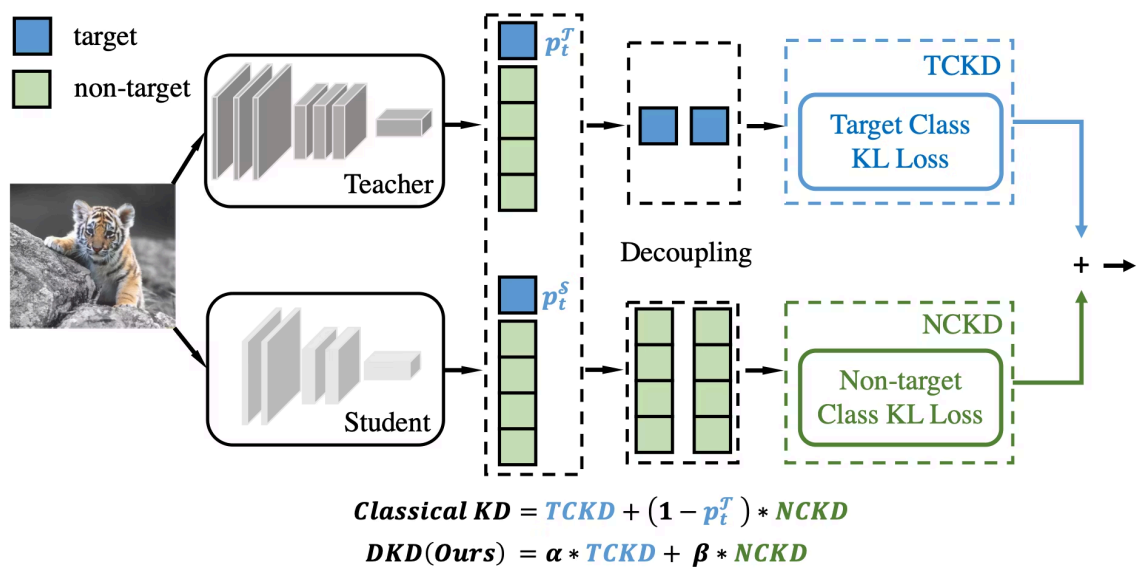
Mathematically decouple the KL divergence into the two TCKD and NCKD terms and use uneven weights to put more emphasis on NCKD with the following loss:

$$DKD = \alpha \cdot TCKD + \beta \cdot NCKD$$

In practice, setting

$$\alpha = 1, \qquad \beta = 8$$

leads to best results, validating that NCKD is more important.



Zhao, B., Cui, Q., Song, R., Qiu, Y., & Liang, J. (2022). "Decoupled Knowledge Distillation". CVPR.

FEATURES

Feature-Based I – FitNets

Output logits capture only the final decision boundary. The teacher's intermediate layers contain rich hierarchical features—from edge detectors in early layers to semantic object parts in deep layers. FitNets pioneered transferring this intermediate knowledge through "hint-based training."

01

Select Hint Layers

Choose strategic intermediate layers from the teacher network. Typically the middle layers where semantic features crystallize before the final classification head.

02

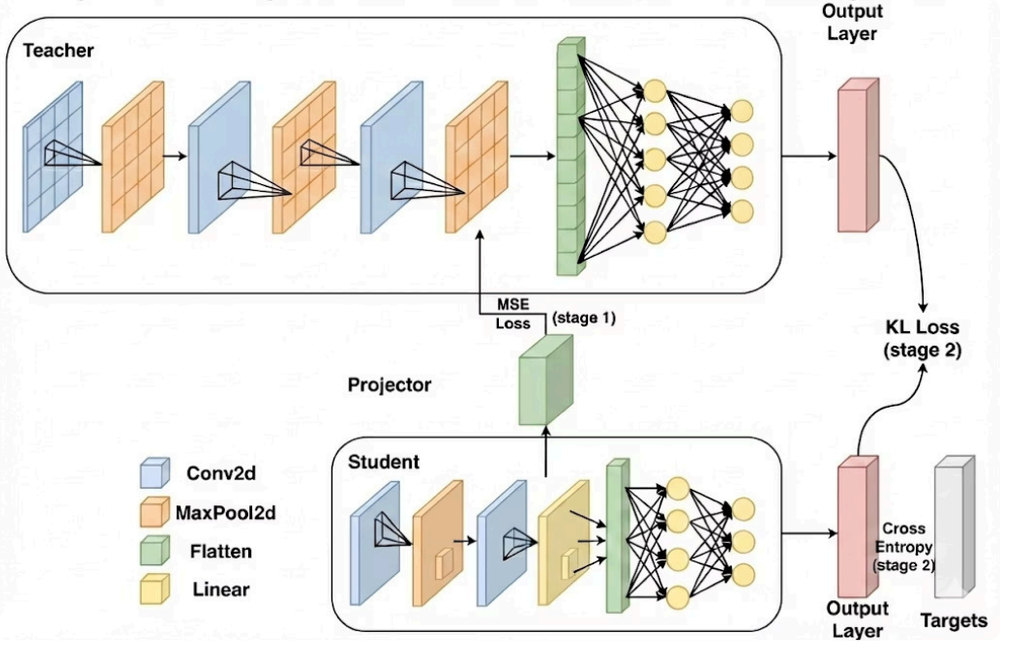
Handle Dimension Mismatch

Teacher and student have different channel dimensions (CT ≠ CS). Solution: Insert a learnable 1×1 convolutional "regressor" layer that projects student features into teacher feature space.

03

Two-Stage Training

Stage 1: Train the student to match teacher's hint layer using L2 loss.
Stage 2: Fine-tune end-to-end with standard classification loss plus knowledge distillation.



Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., & Bengio, Y. (2015). "FitNets: Hints for Thin Deep Nets". ICLR.

SPATIAL

Feature-Based II – Attention Transfer

Focus: Transferring spatial attention patterns—teaching the student *where* to look rather than *what* to see.

The Core Philosophy

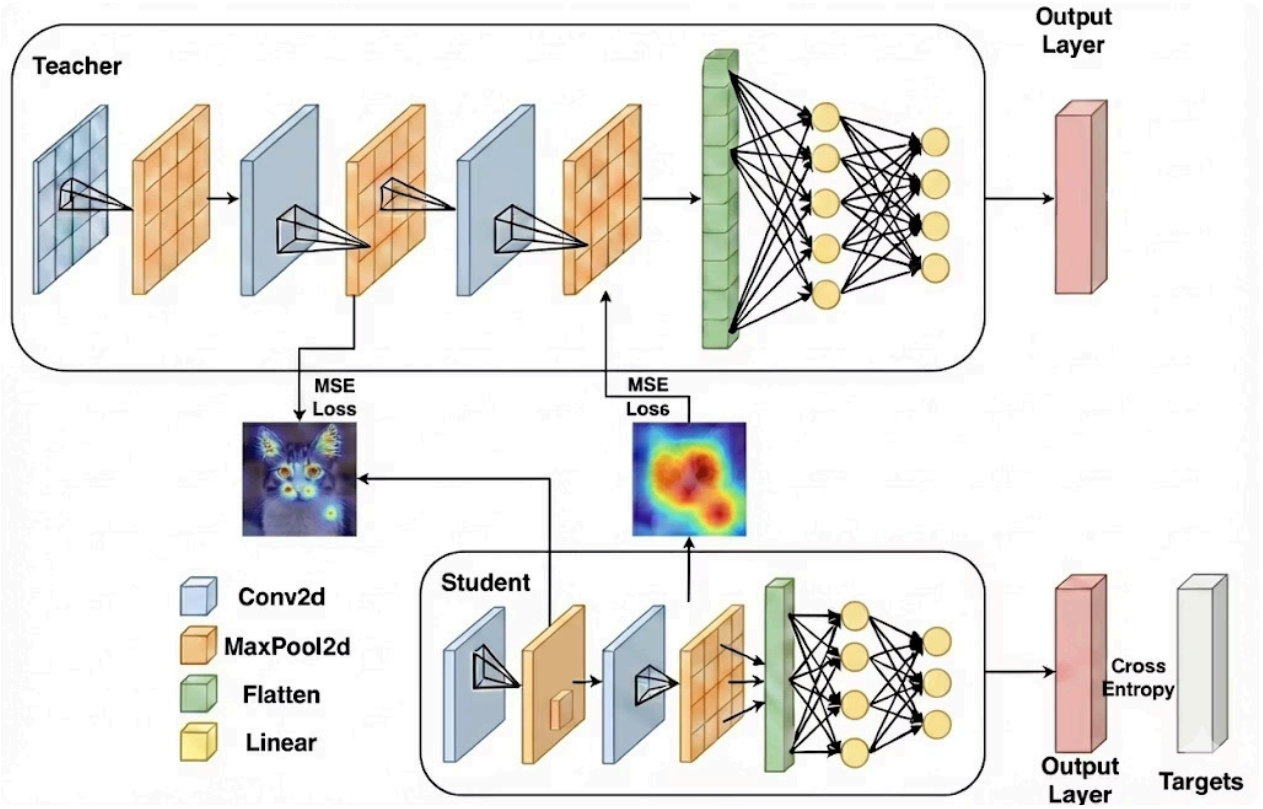
Instead of matching raw feature values (which can be brittle), Attention Transfer (AT) matches the spatial activation patterns. The key question: *Which regions of the image does the teacher consider important?*

Mechanism

For each feature map F with dimensions [C, H, W], collapse the channel dimension using the L2 norm:

$$A(x) = \sum_c |F_c(x)|^2$$

This produces a 2D attention map showing spatial importance.



The model is trained with the loss:

$$\mathcal{L}_{AT} = \mathcal{L}(\mathbf{W}_S, x) + \frac{\beta}{2} \sum_{j \in \mathcal{I}} \left\| \frac{Q_S^j}{\|Q_S^j\|_2} - \frac{Q_T^j}{\|Q_T^j\|_2} \right\|_2,$$

where $Q_S^j = vec(F(A_S^j))$ and $Q_T^j = vec(F(A_T^j))$ are the $j-th$ pair of student and teacher attention maps, L2-normalized and in vectorized form. $\mathcal{L}(\mathbf{W}_s, x)$ is standard cross entropy ($\mathbf{W}_S$ are student's models and x is the input-label pair), β is a hyperparameter regulating the trade-off between the two loss functions.

Zagoruyko, S., & Komodakis, N. (2017). "Paying More Attention to Attention: Improving the Performance of CNNs via Attention Transfer". ICLR.

SPATIAL

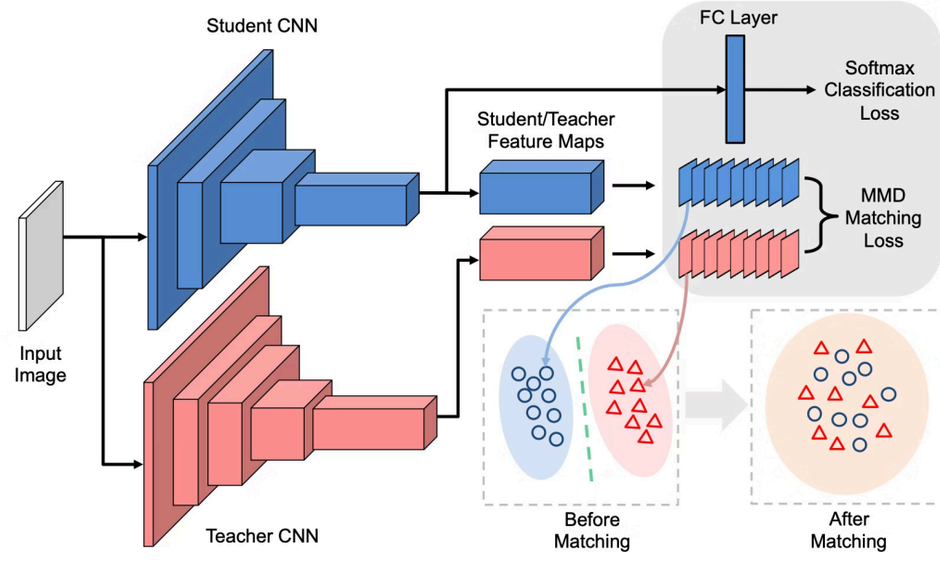
Feature-Based III – Neuron Selectivity Transfer

Building on spatial attention, **Neuron Selectivity Transfer (NST)** moves beyond full feature maps or attention summaries to distill the teacher's neuron selectivity patterns—how neurons respond across different spatial regions and samples.

Core Insight: Selectivity Distribution

Instead of rigid feature matching, NST frames knowledge transfer as a distribution matching problem. It aligns the statistical properties of teacher and student neuron activations using **Maximum Mean Discrepancy (MMD)**.

- Each neuron's activations across an image form a distribution.
- MMD captures where neurons cluster their activations and how regions are statistically similar.
- This approach is less sensitive to noise and sample density than direct feature matching.



Loss Function: MMD with Polynomial Kernel

NST combines standard cross-entropy ($\mathcal{L}_{hard}$) with an MMD loss term:

$$\mathcal{L}_{NST} = \mathcal{L}_{hard} + \lambda \cdot \text{MMD}^2(F^T, F^S)$$

Where F^T and F^S are the L2 normalized feature maps of a selected layer of the teacher and student respectively. Different kernels can be used for MMD. It is shown that choosing a linear kernel ($K(x, y) = x \cdot y$) is equivalent to attention transfer.

Huang, Z., & Wang, N. (2017). "Like What You Like: Knowledge Distill via Neuron Selectivity Transfer". ICCV.

Relation-Based I: Matching the Flow of Solution Procedure (FSP)

The Problem: Direct Feature Matching

FitNet and similar approaches force the student network to match the teacher's intermediate feature maps exactly. This method, while intuitive, proves to be:

- **Too restrictive:** It imposes identical internal representations, limiting the student's ability to discover its own optimal features.
- **Harder to optimize:** The stringent constraints make the training process more difficult and less flexible.

The Core Insight: Learning Solution Flow

What if the teacher didn't just transfer final answers, but rather the **solution procedure itself**? Instead of demanding that the student produces the same features, the FSP method proposes to match **how features relate to each other**.

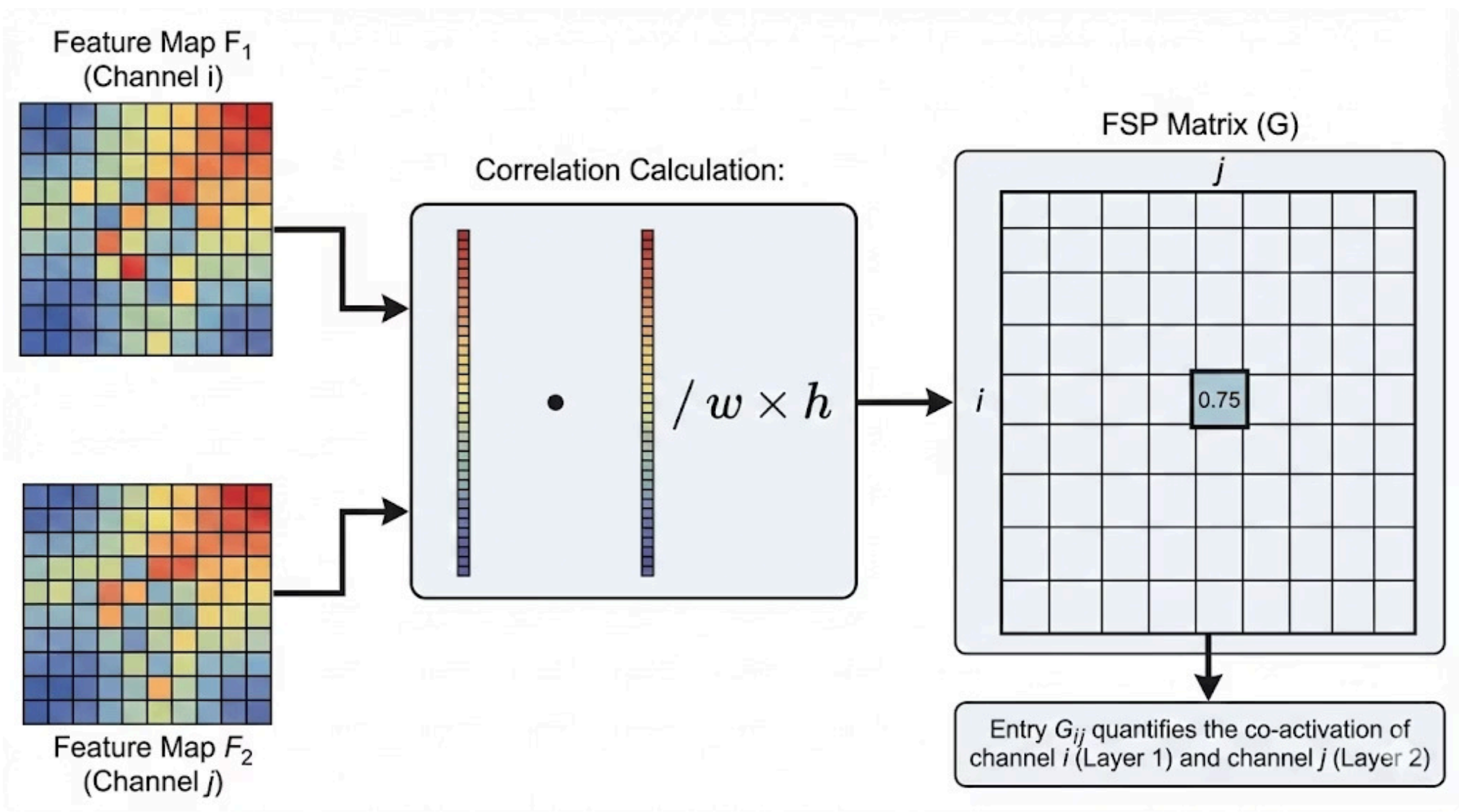
> Match the dynamic interplay between features, not their static values.

FSP Matrix: Knowledge Representation

The FSP (Flow of Solution Procedure) matrix captures the relationships between feature channels across consecutive layers. Given two layers with feature maps $F_1 \in \mathbb{R}^{h \times w \times m}$ and $F_2 \in \mathbb{R}^{h \times w \times n}$, the FSP matrix G is defined as:

$$G_{i,j}(x; W) = \frac{1}{h \times w} \sum_{s=1}^{h} \sum_{t=1}^{w} F_1^{s,t,i}(x; W) \cdot F_2^{s,t,j}(x; W)$$

In practice, given two channels i and j, we compute the dot product between feature map i and feature map j and divide by $h \times w$.



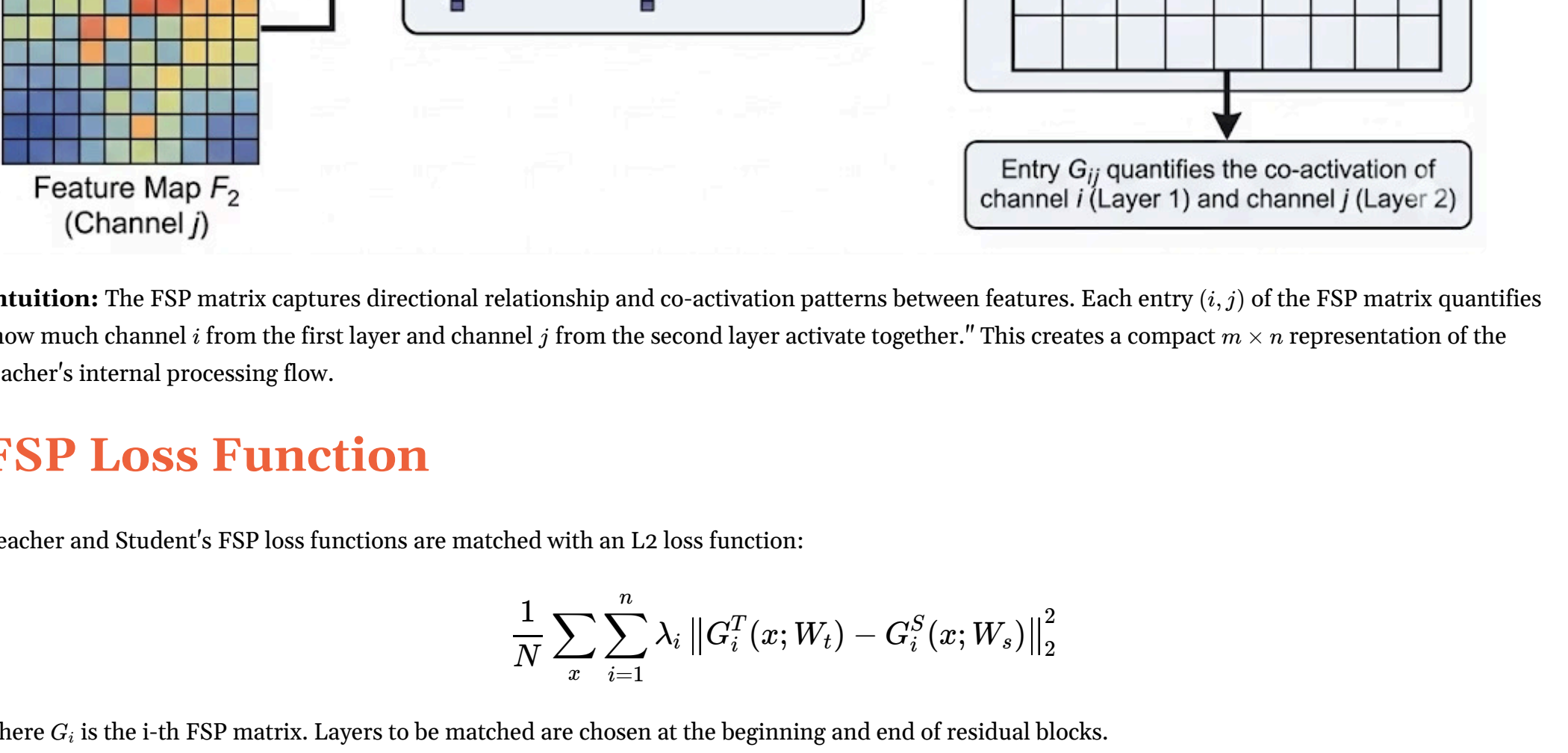
Intuition: The FSP matrix captures directional relationship and co-activation patterns between features. Each entry (i, j) of the FSP matrix quantifies "how much channel i from the first layer and channel j from the second layer activate together." This creates a compact $m \times n$ representation of the teacher's internal processing flow.

FSP Loss Function

Teacher and Student's FSP loss functions are matched with an L2 loss function:

$$\frac{1}{N} \sum_{x} \sum_{i=1}^{n} \lambda_i \left\| G_i^T(x; W_t) - G_i^S(x; W_s) \right\|_2^2$$

where G_i is the i-th FSP matrix. Layers to be matched are chosen at the beginning and end of residual blocks.



Two-Stage Training

Training happens in two stages:

- The student is distilled from the pre-trained teacher using the L2 loss on FSP matrices;
- The student is fine-tuned using standard cross entropy.

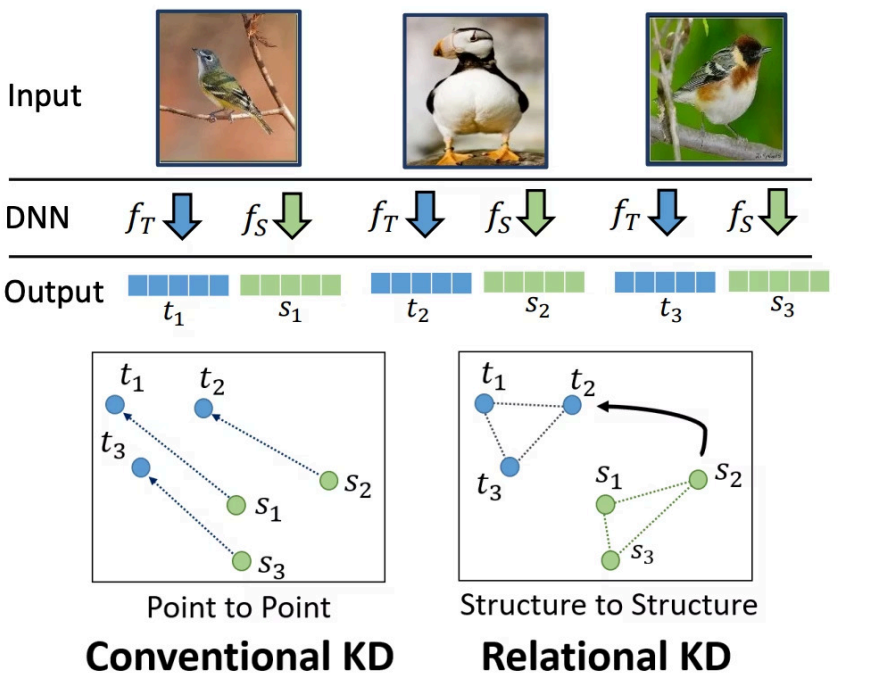
Yim, J., Joo, D., Bae, J., & Kim, J. (2017). A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4133-4141).

Relation-Based II: Relational Knowledge Distillation (RKD)

Relational Knowledge Distillation (RKD) shifts the focus from individual feature activations to the **mutual relationships between data examples** within the embedding space.

Instead of enforcing the student to mimic the teacher's response to single inputs, RKD guides the student to preserve the structural relationships that the teacher learns across a set of inputs.

This is expressed in terms of relative distances and angles of triplets of points embedded by the teacher and student models.

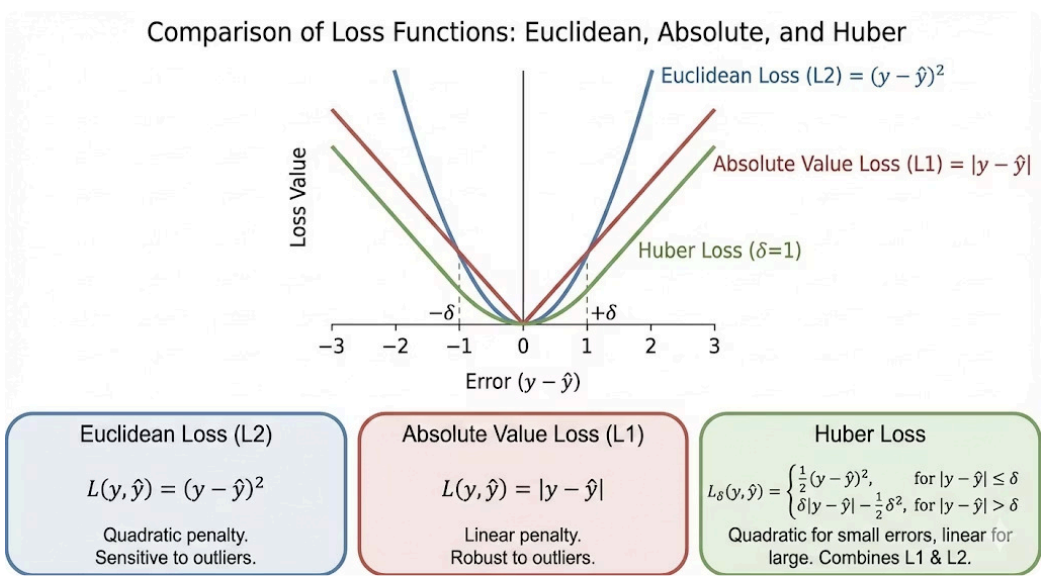


Framework: Beyond Instance-Wise Matching

RKD introduces a relational loss that evaluates how well the student preserves the complex relationships between multiple data examples. This is achieved by comparing 'potential functions' (ψ) that describe these relationships in the teacher's and student's embedding spaces:

$$\mathcal{L}_{\text{RKD}} = \sum_{(x_1,\ldots,x_n)} \ell_\delta(\psi(t_1,\ldots,t_n), \psi(s_1,\ldots,s_n))$$

where t_i and s_i are the teacher and student input data embeddings (extracted from the final feature layer) and ℓ_δ is the Huber loss (to mitigate the effect of outliers).



Distance-Wise Loss (RKD-D): Preserving Pairwise Distances

RKD-D aims to match the pairwise Euclidean distances between feature embeddings in the teacher and student networks. For any two examples t_i and t_j from the teacher's embedding space (and s_i, s_j for the student), the potential function normalizes their distance by the mean distance across all pairs:

$$\psi_D(t_i, t_j) = \frac{\|t_i - t_j\|_2}{\mu}, \quad \mu = \mathbb{E}_{(i,j)}\|t_i - t_j\|_2$$

The loss function then minimizes the difference between these normalized pairwise distances:

$$\mathcal{L}_{\text{RKD-D}} = \sum_{i,j} \ell_\delta(\psi_D(t_i, t_j), \psi_D(s_i, s_j))$$

Angle-Wise Loss (RKD-A): Preserving Triplet Angles

RKD-A extends this concept to third-order relationships by matching the angles formed by triplets of feature embeddings. For any three examples t_i, t_j, t_k, the potential function computes the cosine similarity between the vectors connecting them:

$$\psi_A(t_i, t_j, t_k) = \frac{\langle t_i - t_j, t_k - t_j \rangle}{\|t_i - t_j\|_2 \|t_k - t_j\|_2}$$

Similar to RKD-D, the loss for RKD-A uses a the Huber loss to compare these triplet angles between teacher and student embeddings:

$$\mathcal{L}_{\text{RKD-A}} = \sum_{i,j,k} \ell_\delta(\psi_T^A(t_i, t_j, t_k), \psi_S^A(s_i, s_j, s_k))$$

Training for Metric Learning

Teacher is trained with the metric loss, then the student is trained only with RKD:

$$\mathcal{L}_{\text{student}} = \lambda_{\text{RKD-D}}\,\mathcal{L}_{\text{RKD-D}} + \lambda_{\text{RKD-A}}\,\mathcal{L}_{\text{RKD-A}}$$

where

$$\lambda_{\text{RKD-D}} = 1, \qquad \lambda_{\text{RKD-A}} = 2$$

Classification

Teacher is trained with cross-entropy, then the student is trained with cross-entropy plus RKD on the final embeddings:

$$\mathcal{L}_{\text{student}} = \mathcal{L}_{\text{CE}} + \lambda_{\text{RKD-D}}\,\mathcal{L}_{\text{RKD-D}} + \lambda_{\text{RKD-A}}\,\mathcal{L}_{\text{RKD-A}},$$

with:

$$\lambda_{\text{RKD-D}} = 25, \qquad \lambda_{\text{RKD-A}} = 50$$

Key Insight: By transferring the teacher's embedding **geometry** (pairwise distances and triplet angles), the student learns a more generalizable metric structure. This allows the student to not only match the teacher's performance but, in some cases, even **outperform the teacher** on downstream tasks by learning richer, more robust representations.

Park, W., Kim, D., Lu, Y. and Cho, M., 2019. Relational knowledge distillation. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 3967-3976).

Contrastive Representation Distillation (CRD)

CRD trains the student to match the teacher's representation using a contrastive objective. This maximizes **mutual information** between penultimate-layer features, ensuring the student learns the teacher's relational structure, not just individual predictions.

The Contrastive Loss Framework

Inputs x are given as input to the teacher and student models, deriving teacher and student embeddings T and S.

CRD defines a **critic** $h(T, S)$ that estimates the probability that a teacher–student feature pair comes from the **same input** (positive) rather than from **different inputs** (negative). This critic is used in a contrastive NCE-style loss that pulls positive pairs together and pushes negative pairs apart in the embedding space, and this objective maximizes a lower bound on the mutual information $I(T; S)$ between teacher and student representations.

Teacher and student features are first mapped into a common embedding space:

$$z_T = g_T(T), \quad z_S = g_S(S),$$

where g_T and g_S are small linear heads followed by L_2 normalization, and τ is a temperature.

The critic has the form

$$h(T, S) = \frac{\exp\left(z_T^\top z_S / \tau\right)}{\exp\left(z_T^\top z_S / \tau\right) + N/M}$$

where N is the number of negatives and M is the dataset size.

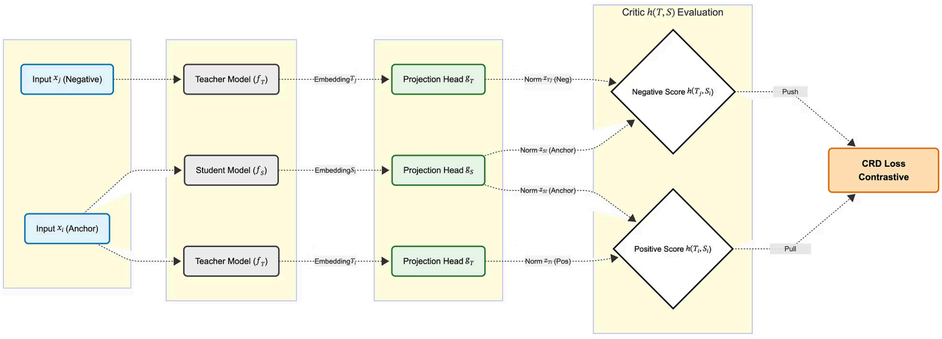
Given positive pairs (T_i, S_i) (same input) and negative pairs (T_j, S_i) (different inputs, often with different labels), the **CRD contrastive loss** is the binary log-likelihood:

$$\mathcal{L}_{\text{CRD}} = \mathbb{E}_{\text{pos}}\left[\log h(T_i, S_i)\right] + N\,\mathbb{E}_{\text{neg}}\left[\log\left(1 - h(T_j, S_i)\right)\right].$$

Intuitively:

- T_i: teacher representation of an input.
- S_i: student representation of the same input (positive pair).
- Negatives are formed by pairing S_i with teacher features from other inputs T_j.
- (g_T, g_S): linear heads projecting into a shared, normalized space used by the critic.

By training both the student f_S and the critic h to maximize $\mathcal{L}_{\text{CRD}}$, CRD learns student features that align with teacher features for the same input and differ for other inputs, thereby maximizing a mutual-information lower bound $I(T; S)$.



Training

Optimization is **single-stage**: Student trained jointly with contrastive distillation loss and supervised cross-entropy (when labels exist). Three cases are considered.

Model Compression

Teacher is pretrained and frozen; only student (and projection/critic) are trained.

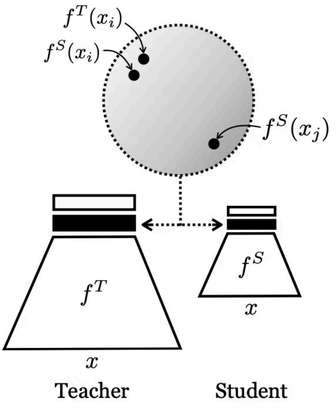
Total loss combines:

- Cross-entropy with ground-truth labels: $H(y, y_S)$
- CRD contrastive loss: $\mathcal{L}_{\text{CRD}}$ between teacher-student representations

Loss function:

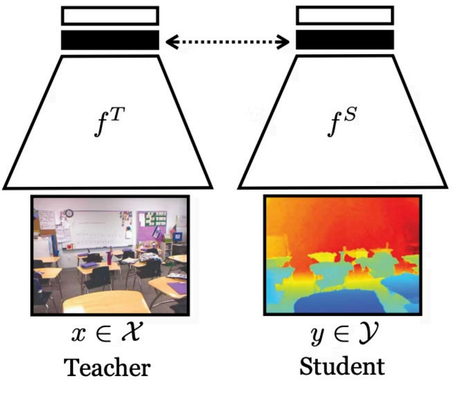
$$\mathcal{L} = H(y, y_S) + \beta\mathcal{L}_{\text{CRD}}$$

β is a hyper-parameter.



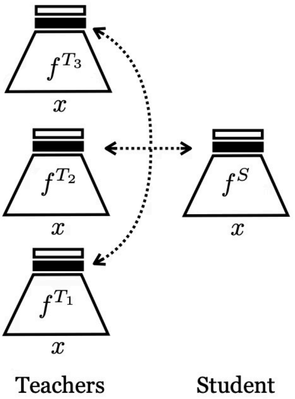
Cross-Modal Transfer

No labels available → no cross-entropy term; only CRD on paired unlabeled data.



Ensemble Distillation

CRD summed over all teacher-student pairs combined with cross-entropy in single-stage training.



Tian, Y., Krishnan, D., & Isola, P. (2020). "Contrastive Representation Distillation". ICLR.

Cross-Modal & Multimodal Knowledge Distillation

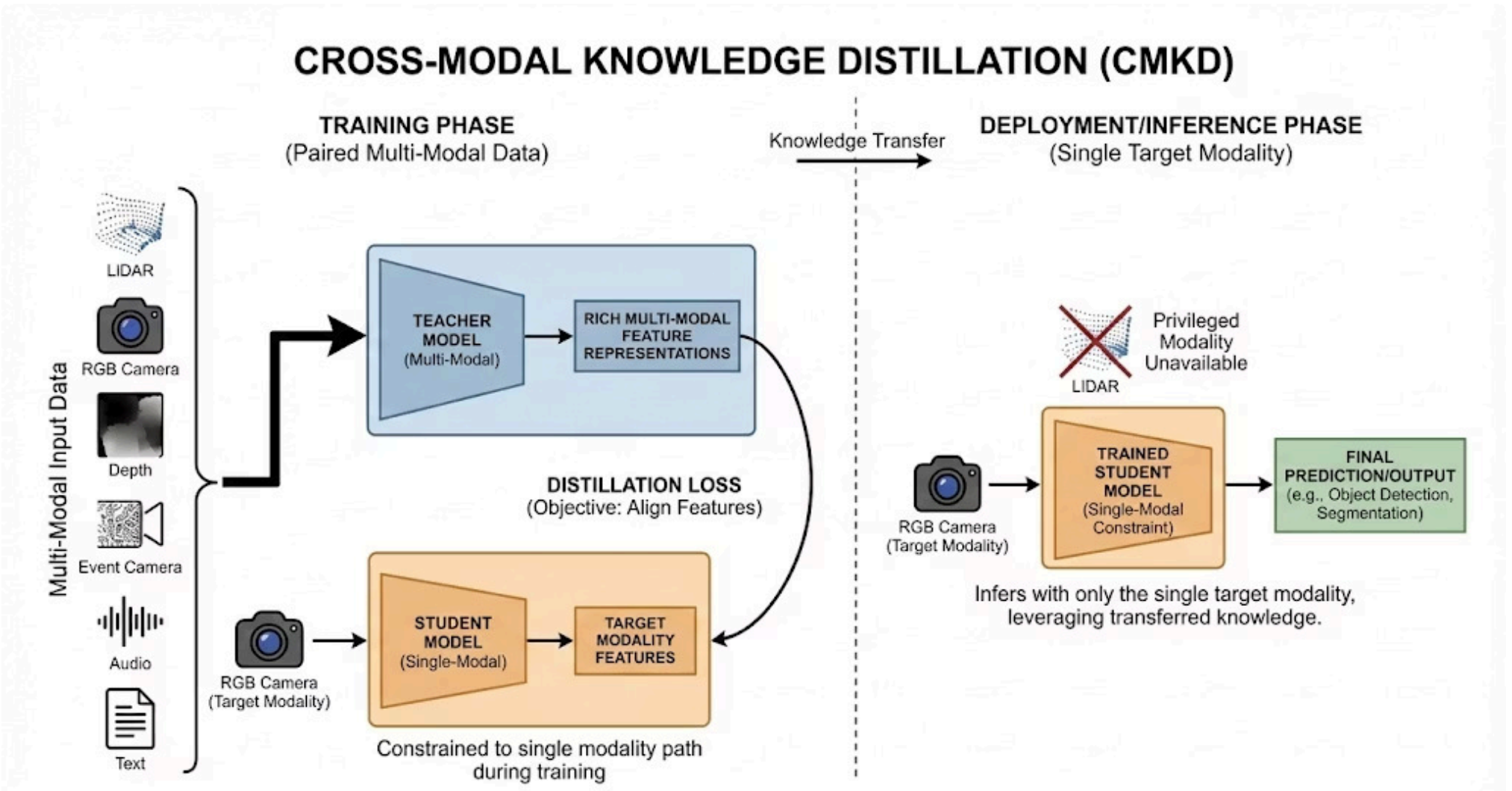
Cross-Modal/Multimodal Knowledge Distillation (CMKD)

Transferring knowledge from a well-annotated source modality to a less-annotated target modality.

The teacher model uses rich, often privileged, modalities during training, while the student is constrained to a single, target modality for deployment.

- **Modalities:** RGB, Depth, LiDAR, Event cameras, Text, Audio, Optical Flow
- **Process:** Trained with paired multi-modal data, but infers with only the single target modality.
- **Example:** Teacher trained on LiDAR+Camera data, student deployed with camera-only input.

The teacher may also process unified multimodal representations (e.g., image + text as in CLIP).



Key Applications

3D Perception (Autonomous Driving)

Distilling LiDAR-based 3D object detection into multi-camera Bird's-Eye-View (BEV) detection for cost-effective systems.

RGB-D Vision

Transferring depth estimation capabilities from RGB-D sensors to RGB-only camera systems.

Event-based Vision

Enabling high-speed frame capture and motion understanding via efficient event cameras.

Vision-Language Integration

Facilitating text-guided image understanding in vision-only models, inspired by models like CLIP.

Module 3

Other Paradigms

Deep Mutual Learning – Distillation Without a Teacher

Breaking the Teacher Dependency

Traditional KD requires training a large teacher first, then distilling to a student—a two-stage, computationally expensive process. Deep Mutual Learning (DML) asks: *Can multiple small models teach each other simultaneously?*

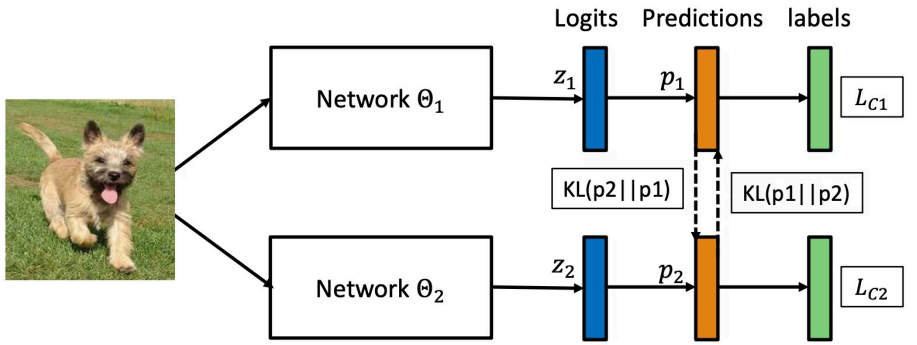
The Mechanism

Train two (or more) student networks in parallel from scratch. Each network learns from:

1. Its own predictions on the ground truth (standard supervised learning)
2. The other network's soft predictions (mutual distillation)

Loss for Network A:

$$L_A = CE(y, p_A) + KL(p_B||p_A)$$



Even though both networks start from random initialization, they explore different regions of the solution space due to random weight initialization. This diversity creates complementary knowledge that can be mutually transferred, often resulting in both networks outperforming individually trained models.

Zhang, Y., Xiang, T., Hospedales, T. M., & Lu, H. (2018). "Deep Mutual Learning". CVPR.

Data-Free Knowledge Distillation

GDPR and privacy laws often mandate data deletion (or limited sharing) after model training. Medical, financial, and personal data cannot be shared. Yet we still want to distill large models into efficient students. How do we distill without any training data?

Generator Initialization

Initialize a generative model (GAN or Diffusion Model) with random weights. This generator will "dream" synthetic data that activates the teacher.

Activation Maximization

Generate synthetic images by maximizing the teacher's class-specific activations. The generator learns to create images that the teacher confidently classifies, even if these images don't look perfectly realistic.

Knowledge Transfer

Use the synthetic images and teacher's soft predictions as pseudo-training data for the student. The student learns the teacher's decision boundaries without ever seeing real data.

Iterative Refinement

Alternate between improving the generator (creating better synthetic data) and training the student (learning from synthetic data) until convergence.

H. Yin et al., "Dreaming to Distill: Data-Free Knowledge Transfer via DeepInversion," CVPR 2020.

G. Fang et al., "Up to 100× Faster Data-free Knowledge Distillation," AAAI 2022.

Xie, et al. (2025). "DiffDFKD: Diffusion-Based Data-Free Knowledge Distillation for Privacy-Preserving Model Compression". CVPR.

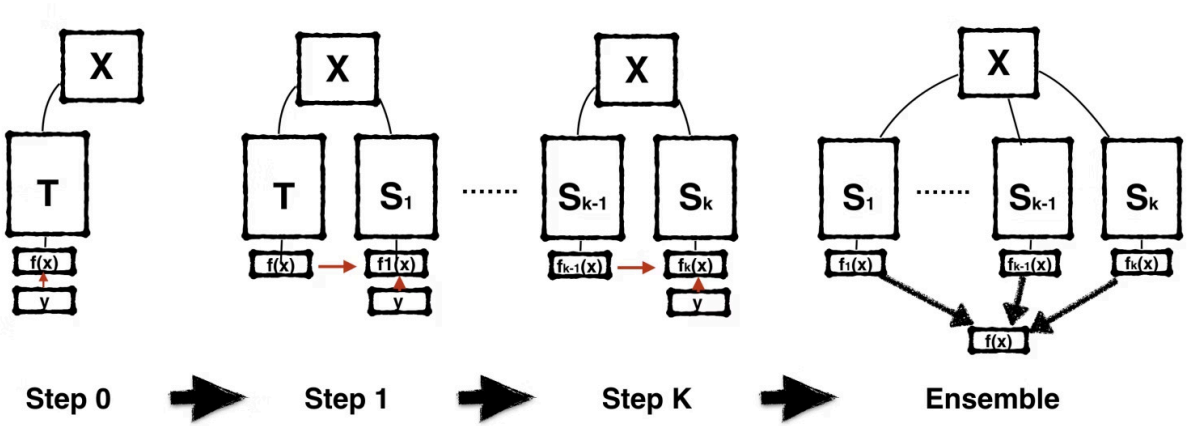
Born-Again Networks – Iterative Self-Improvement

The Counterintuitive Result

Born-Again Networks (BAN) discovered something surprising: using a trained model as a teacher to train an identically-sized student from scratch often *improves* performance.

The Mechanism

1. Train Model A normally on ground truth labels
2. Use Model A as teacher to train Model B (same architecture) using KD
3. Use Model B as teacher to train Model C
4. Repeat for a number of generations
5. Distill the ensemble to a single network



Furlanello, T., Lipton, Z., Tschannen, M., Itti, L., & Anandkumar, A. (2018). "Born-Again Neural Networks". ICML.

Bridging the Gap – Teacher Assistant

The Problem: When the teacher is extremely large (e.g., ResNet152) and the student is extremely small (e.g., MobileNetV2), direct distillation often fails. The capacity gap is too large—the student cannot capture the teacher's complex decision boundaries, leading to poor knowledge transfer.

Example

Student (Tiny)

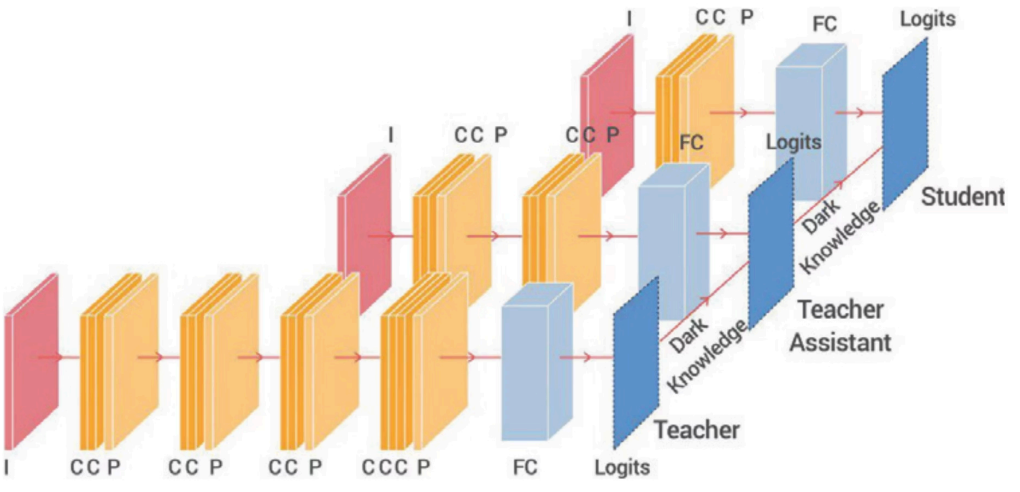
MobileNetV2 (3.5M parameters) - Final deployment target for edge devices

Teacher Assistant (Medium)

ResNet50 (25M parameters) - Intermediate bridge network for progressive distillation

Teacher (Large)

ResNet152 (60M parameters) - Original powerful model with peak performance



Multi-Step Distillation

Instead of Teacher → Student directly, use an intermediate "Teacher Assistant" (TA):

Step 1: Teacher → TA distillation

Step 2: TA → Student distillation

The TA acts as a bridge, making the knowledge more accessible to the student by reducing the representation complexity at each step.

Selecting the TA

The ideal TA should be:

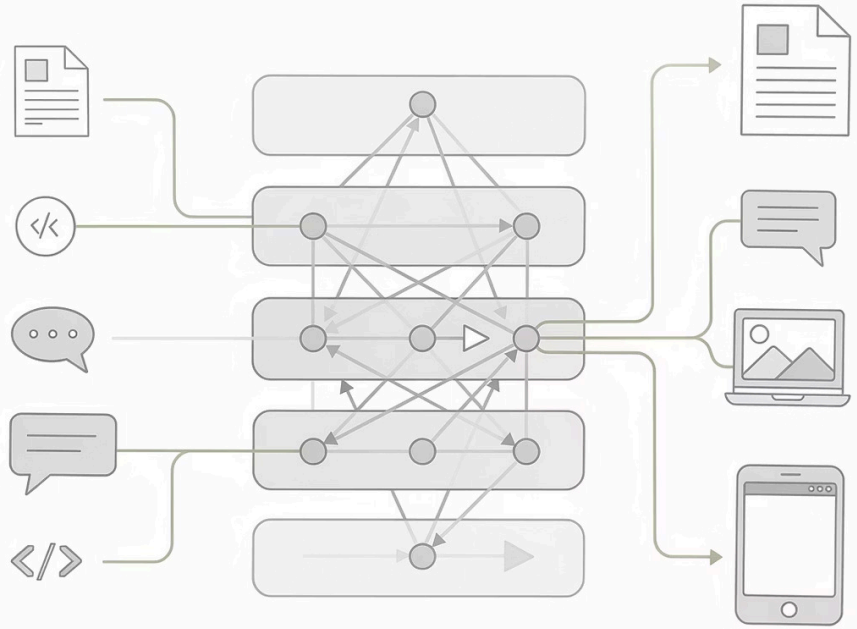
- Medium capacity often near to average accuracy between teacher and student in that architecture family
- Similar architecture family to student for compatibility
- Capable enough to capture teacher's knowledge but simple enough to transfer to student

Mirzadeh, S. I., Farajtabar, M., Li, A., Levine, N., Matsukawa, A., & Ghasemzadeh, H. (2020). "Improved Knowledge Distillation via Teacher Assistant". AAAI.

Module 4

Applications & Large Language Models

Focus: GenAI, NLP, and Future Trends



Beyond Classification – Detection & Segmentation

> **The Challenge:** Classification KD works on a single global output vector (class probabilities). Object detection and segmentation require predicting *structured outputs*—bounding boxes, objectness scores, instance masks, and semantic labels—for every spatial location. How do we distill this complex spatial knowledge?

Feature Map Distillation

Match intermediate feature pyramids (FPN outputs) between teacher and student using spatial attention. The teacher's multi-scale features guide the student's region proposals and feature extraction.

Localization Distillation

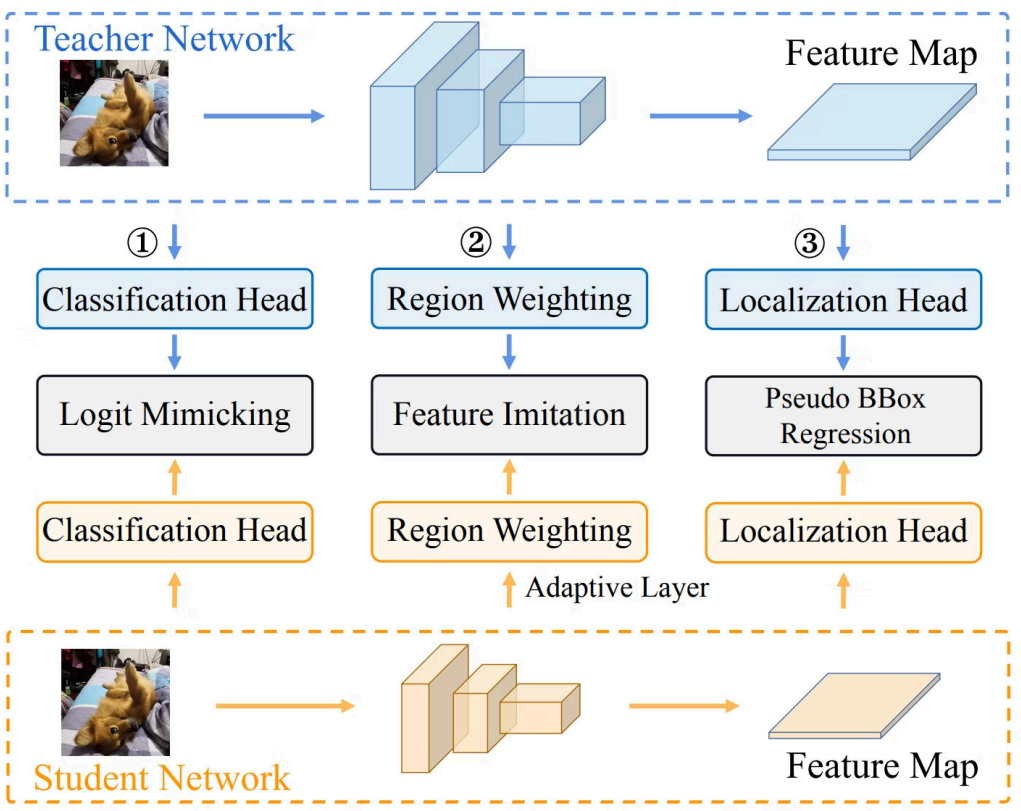
For bounding box regression, distill the probability distributions over box edges. The teacher provides "localization quality" scores indicating confidence in box coordinates, not just class predictions.

Foreground-Background Balance

Weight distillation loss differently for foreground (object) vs background regions. Teacher's foreground predictions contain more valuable knowledge than background, which is often trivial.

Localization Distillation Details

- Teacher produces both classification logits *and* localization quality (IoU) scores for each box
- Student learns to predict both, weighted by importance
- Addresses the misalignment between classification confidence and localization accuracy



Zheng, Z., Ye, R., Wang, P., Ren, D., Zuo, W., Hou, Q., & Cheng, M. M. (2022). "Localization Distillation for Dense Object Detection". CVPR.

TRANSFORMERS

Token-Level vs. Sequence-Level KD

Token-Level Distillation (TinyBERT)

Treats sequence generation as independent classification at each position. For each token position t, minimize KL divergence between teacher and student probability distributions:

$L_{token} = \Sigma_t KL(P_T(\cdot|x<t) \,||\, P_S(\cdot|x<t))$

Best for: Masked language modeling (BERT), where tokens are predicted independently given context.

- Simple and efficient to implement
- Works well for understanding tasks (classification, NER, QA)
- Ignores token dependencies in generation

Sequence-Level Distillation

Considers the probability of the *entire output sequence* as a unit. Uses beam search or sampling to generate complete sequences from the teacher, then trains the student to maximize the probability of these sequences.

$L_{seq} = -\log P_S(y_{1:T}|x)$ where $y_{1:T} \sim$ Teacher

Best for: Machine translation, summarization, dialogue—tasks where word choice depends on future context.

- Captures inter-token dependencies
- Produces more coherent outputs
- Computationally expensive (requires generation)

Kim, Y., & Rush, A. M. (2016). "Sequence-Level Knowledge Distillation". EMNLP.

Token-Level vs. Sequence-Level Distillation

Token-Level Distillation (Word-Level KD)

This approach treats sequence generation as a sum of local classification losses at each position. For each token position *t*, the goal is to minimize the KL divergence (or cross-entropy) between the teacher and student probability distributions for that specific token.

$$L_{\text{token}} = \sum_t \mathrm{KL}\big(P_T(\cdot \mid x, y_{<t}) \,\|\, P_S(\cdot \mid x, y_{<t})\big)$$

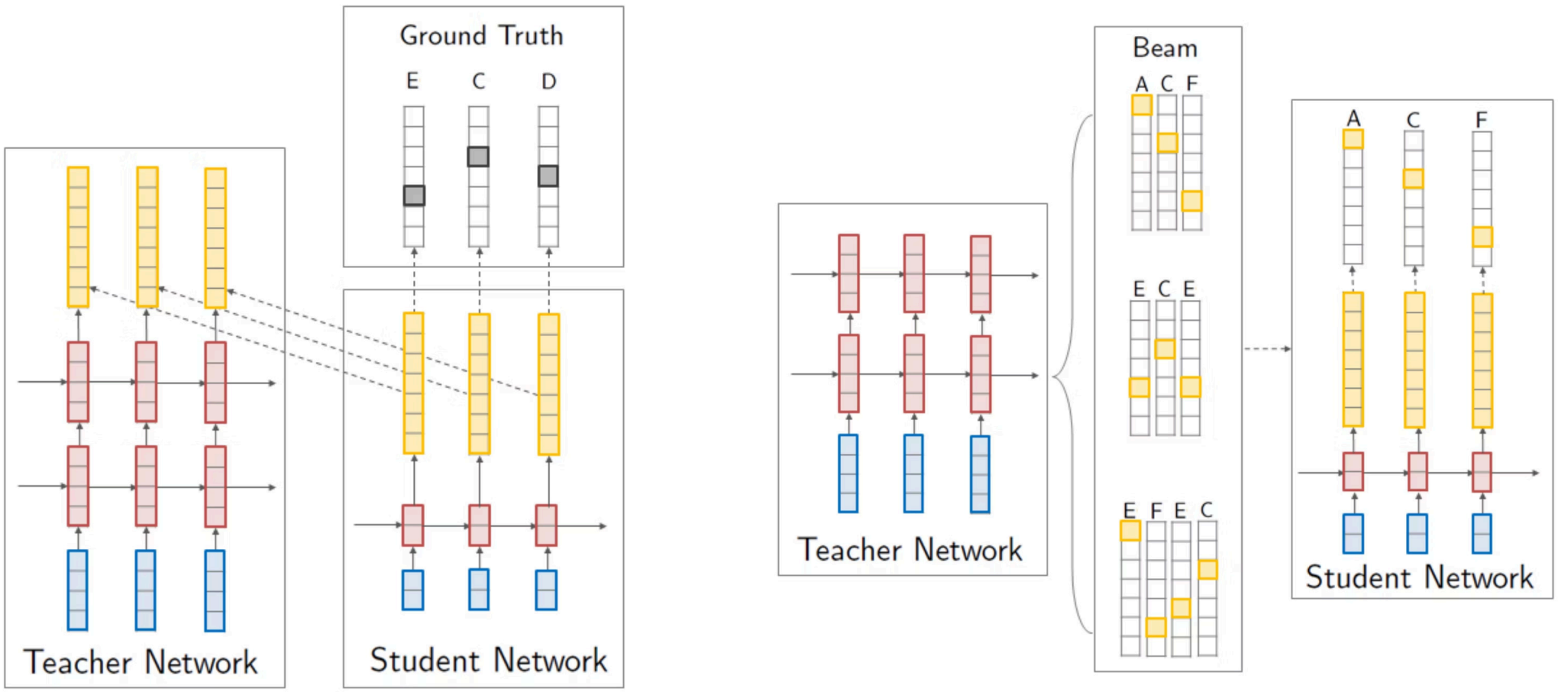
- Each token distribution still depends on previous tokens $y_{<t}$ through the autoregressive decoder.
- Commonly used in sequence-to-sequence KD and also in BERT-style distillation, but not exclusive to TinyBERT.
- Simple and efficient to implement; works well for many understanding and generation tasks.
- Matches the teacher at the level of per-token distributions, but does not directly model the full sequence distribution $P_T(y_{1:T} \mid x)$.

Sequence-Level Distillation

Instead of focusing on individual tokens, this method considers the probability of the entire output sequence as a single unit. It involves using beam search or sampling to generate complete teacher sequences, then training the student to maximize the probability of these generated sequences.

$$y_{1:T}^{(T)} \sim \text{Teacher}(x), \quad L_{\text{seq}} = -\log P_S\big(y_{1:T}^{(T)} \mid x\big)$$

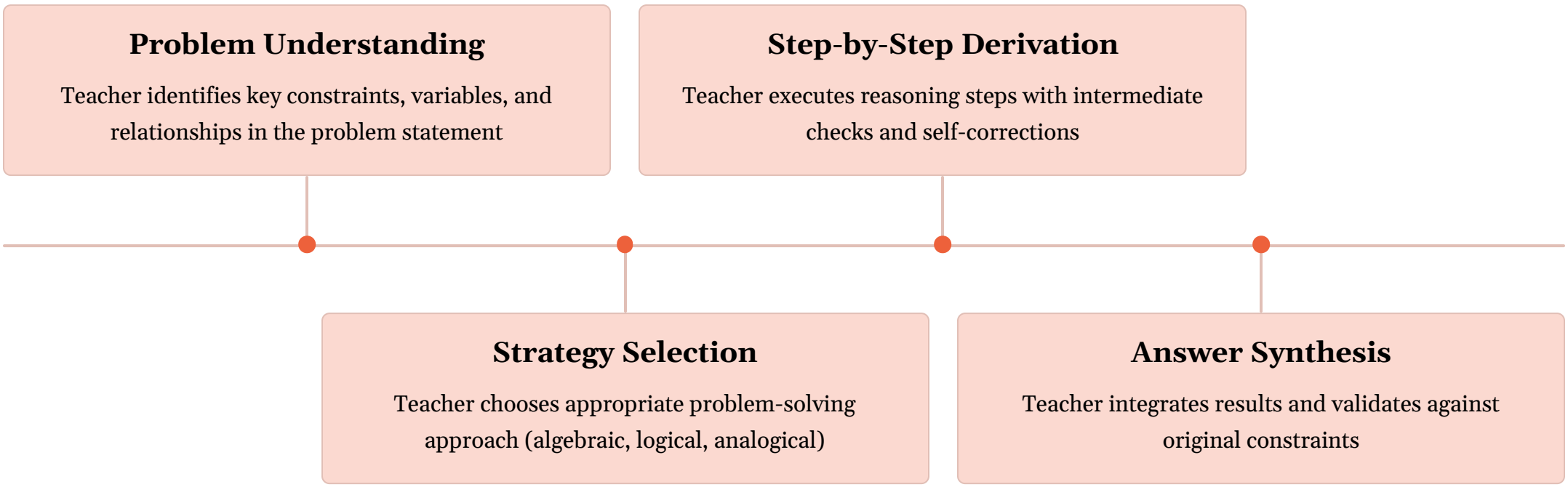
- Often instantiated by taking the teacher's beam-search mode $\hat{y} = \arg\max_y P_T(y \mid x)$ as a single target sequence.
- Particularly useful for tasks like machine translation, summarization, and dialogue, where global sequence quality and coherence matter.
- Captures inter-token dependencies at the sequence level and can yield more coherent outputs, at the cost of running generation with the teacher.



Kim, Y., & Rush, A. M. (2016). "Sequence-Level Knowledge Distillation". EMNLP.

DeepSeek & Reasoning Distillation

Modern LLMs like DeepSeek-R1 (671B parameters) don't just produce answers—they generate detailed reasoning traces showing step-by-step problem decomposition. Distilling this capability requires more than matching final outputs; we must transfer the entire *reasoning trajectory*.



The Trajectory Matching Paradigm

Unlike standard KD that matches only outputs, reasoning distillation requires the student (7B model) to replicate the teacher's *complete reasoning chain*:

- **Step Alignment:** Student must generate same intermediate reasoning steps in same order
- **Verification Matching:** Student must perform same self-checks and corrections
- **Explanation Quality:** Student's reasoning must be equally clear and pedagogical

Reasoning Collapse

The critical failure mode: if the student skips reasoning steps or takes shortcuts, the entire knowledge transfer fails. The student might produce correct answers on training distribution but fails on novel problems requiring genuine reasoning.

Prevention: Explicit supervision at each reasoning step, not just the final answer. Use both trajectory matching loss and reinforcement learning from reasoning quality feedback.

DeepSeek-R1 Result: Successfully distilled 671B reasoning model → 7B student while retaining 95% of reasoning capability on mathematical problem-solving benchmarks. The key: treating each reasoning step as a distillation target, not just the conclusion.

DeepSeek-R1 Technical Report (2025). "Distilling Reasoning Capabilities in Large Language Models".

DeepSeek-R1: Reasoning via Reinforcement Learning

DeepSeek-R1 used distillation to transfer complex problem-solving trajectories to smaller models through Supervised Fine-Tuning (SFT)-based distillation.

Approach: Extracting & Fine-tuning Reasoning Traces

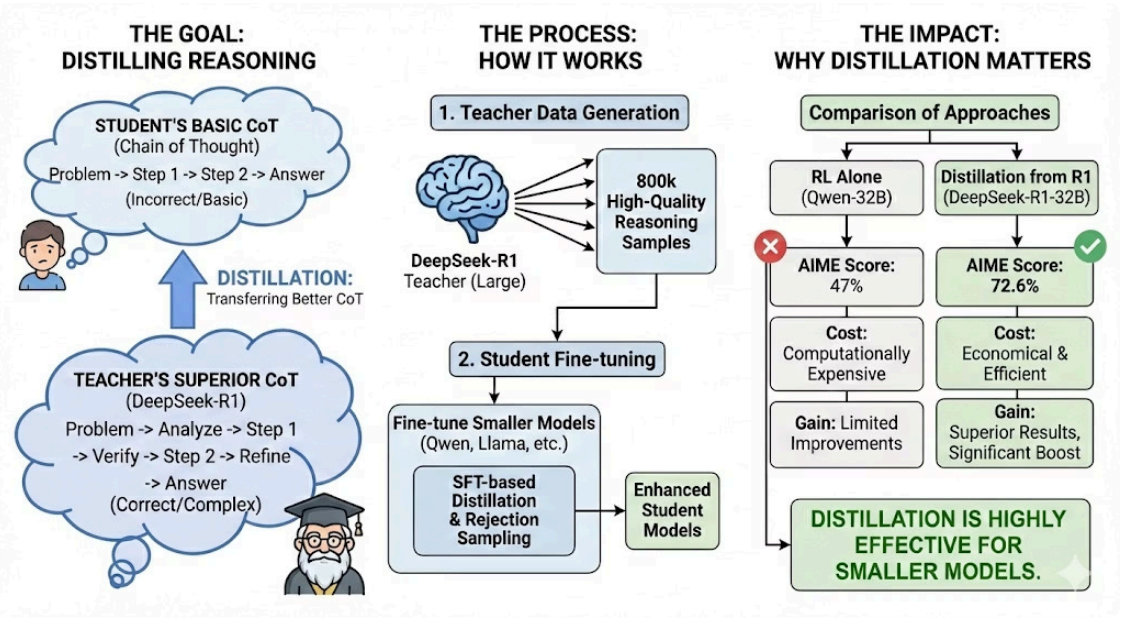
The distillation process involved two key stages:

1. Teacher Data Generation

- Extracted approximately 800,000 high-quality reasoning samples from the powerful DeepSeek-R1 (teacher) model.
- Each sample included detailed intermediate steps and final solutions.

2. Student Fine-tuning

- Fine-tuned smaller base models (e.g., Qwen2.5, Llama family) using these extracted reasoning samples (standard cross-entropy).
- Enhanced data quality via rejection sampling of generated reasoning steps.



Distillation vs. Reinforcement Learning on Small Models

RL Alone (Qwen-32B)

- **Result:** 47% AIME
- **Cost:** Computationally expensive
- **Gain:** Limited improvements in reasoning capability.

Distillation from R1 (DeepSeek-R1-32B)

- **Result:** 72.6% AIME
- **Cost:** Economical and efficient.
- **Gain:** Superior results, significant boost in reasoning.

This comparison highlights that while reasoning discovery requires powerful base models, distilling these complex reasoning patterns is a highly effective and efficient strategy for smaller models.

Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., ... & He, Y. (2025). Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. arXiv preprint arXiv:2501.12948.

Distilling Diffusion Models

Diffusion models can get big and usually require many denoising steps to generate images. This approach focuses on drastically reducing the number of sampling steps, making them practical for real-time applications without sacrificing image quality.

Teacher: High-Step Sampling

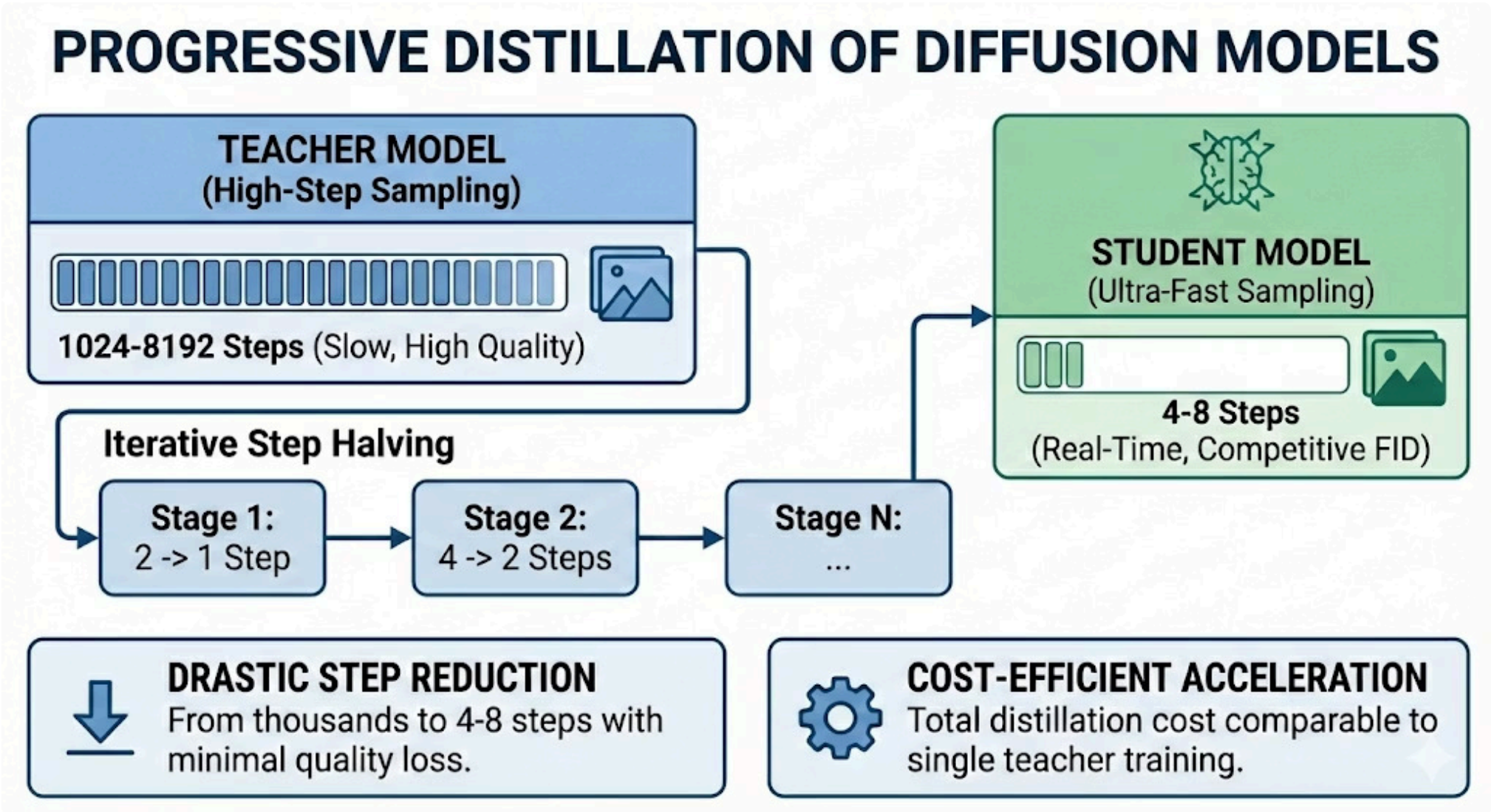
A pretrained diffusion model uses 1024–8192 steps to achieve near-optimal FID on various benchmarks like CIFAR-10 and ImageNet.

Iterative Step Halving

Each distillation stage trains a student (same U-Net architecture) to match two teacher denoising steps with a single denoising step, effectively halving the required sampling steps progressively.

Ultra-Fast 4-Step Sampling

After several distillation stages, the resulting student models can sample in as few as 4 steps while retaining competitive FID, such as FID 3.0 on CIFAR-10.



Key Results of Progressive Distillation

Drastic Step Reduction

Achieved sampling step reduction from thousands to just 4–8 steps, with minimal loss in perceptual quality across diverse datasets.

Cost-Efficient Acceleration

The total computational cost for progressive distillation to reach 4 steps is comparable to training the original teacher model just once, making it an efficient acceleration method.

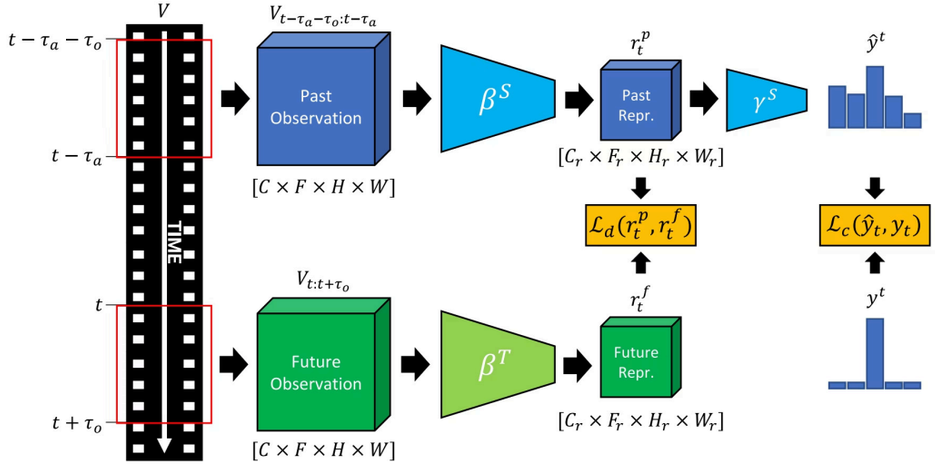
First-Hand Experience with Knowledge Distillation for Egocentric Action Anticipation

In real-world applications like augmented reality or assistive robotics, predicting user actions in real-time from a first-person perspective (egocentric view) is crucial. This domain presents unique challenges, especially regarding computational latency and the need for timely predictions.

This approach highlights how knowledge distillation can transform computationally intensive tasks like video anticipation, making them viable for real-time, resource-constrained environments.

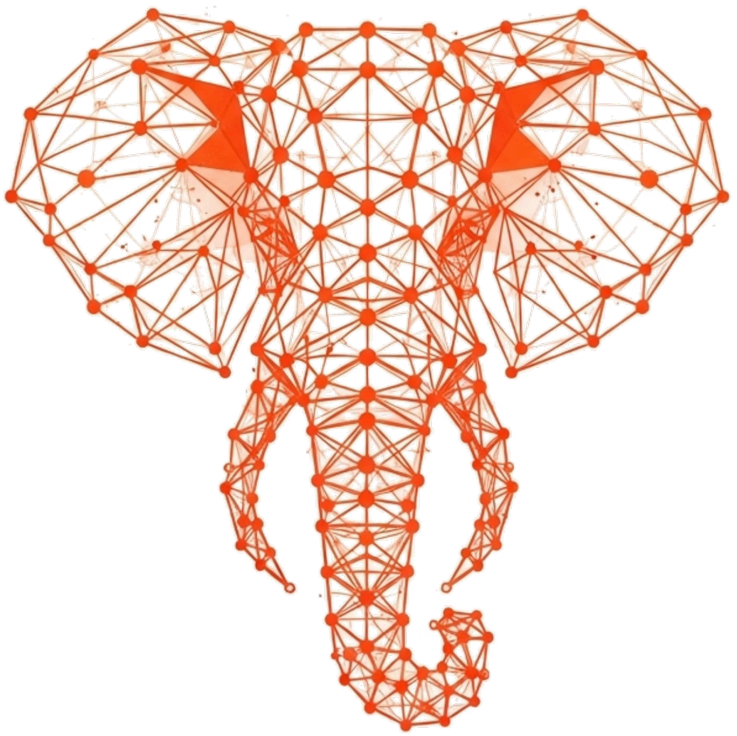
Key Advantages & Impact

- **Real-time Feasibility:** Lightweight, distilled student models enable practical, low-latency action anticipation on edge devices.
- **Improved Performance:** Outperforms more complex architectures, especially at lower resolutions, for a fraction of the computational cost.
- **Fairer Benchmarking:** The new streaming protocol provides a more realistic assessment of model utility in dynamic, egocentric scenarios.



Furnari, A., & Farinella, G. M. (2023). Streaming egocentric action anticipation: An evaluation scheme and approach. Computer Vision and Image Understanding, 234, 103763.

Conclusions and Next Steps



Uni ct DEEP LEARNING
ADVANCED MODELS AND METHODS

We Have Explored:

- **Foundational Theory**: LUPI framework, dark knowledge, and the role of temperature in softening predictions
- **Response-Based Methods**: Vanilla KD and Decoupled KD separating target class and non-target class knowledge
- **Feature-Based Distillation**: FitNets, Attention Transfer, and Neuron Selectivity Transfer for intermediate layer matching
- **Relation-Based Approaches**: FSP matrices and RKD for capturing structural relationships between examples
- **Contrastive Methods**: CRD maximizing mutual information between teacher-student representations
- **Alternative Paradigms**: Deep Mutual Learning, Data-Free KD, Born-Again Networks, and Teacher Assistant
- **Modern Applications**: Transformers, reasoning distillation (DeepSeek), diffusion models, and the industrial compression trinity

References

1. Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. arXiv:1503.02531
2. Romero, A., et al. (2014). FitNets: Hints for thin deep nets. arXiv:1412.6550
3. Zagoruyko, S., & Komodakis, N. (2017). Paying more attention to attention. ICLR
4. Yim, J., et al. (2017). A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. CVPR
5. Park, W., et al. (2019). Relational knowledge distillation. CVPR
6. Tian, Y., et al. (2020). Contrastive representation distillation. ICLR
7. Zhang, L., et al. (2018). Deep mutual learning. CVPR
8. Chen, H., et al. (2019). Data-free learning of student networks. ICCV
9. Furlanello, T., et al. (2018). Born again neural networks. ICML
10. Zhao, B., et al. (2022). Decoupled knowledge distillation. CVPR