TÉLÉCOM PARIS · SETI MASTER · LECTURE 04

Knowledge Distillation

A large network teaches a small one without changing deployment cost

04

Soft targets, temperature, response and attention transfer, fair comparisons

Wednesday 27 January 2027 · Session 4 of 8 · lecture + Lab 4

Where we are

S1 9 Dec	S2 6 Jan	S3 13 Jan	S4 27 Jan	S5 3 Feb	S6 10 Feb	S7 24 Feb	S8 3 Mar
L01 + Lab 1	L02 + Lab 2	L03 + Lab 3	**L04 + Lab 4**	Lab 5	Lab 6	Lab 7 (I)	Lab 7 (II)

What we keep from S3

Structure is what the DPU rewards
a smaller dense network on the critical path; channel alignment affects efficiency.

Removing capacity costs accuracy
fine-tuning recovers part of it – at extreme ratios, not all of it.

Pruning finds a shape
trained from scratch, the pruned *architecture* often does as well: the shape was the discovery.

Today's question

A compact network trained on labels alone may underperform its capacity.

Can a large teacher improve the same compact student's accuracy without increasing its deployment cost?

S2 and S3 changed numbers and network shape. Today we change how the compact student is trained.

THE THREAD – TODAY'S ANGLE

How much useful intelligence can we obtain from a limited budget of computation, memory and energy?

Spend compute once, at training time, so that you can save it forever, at inference time.

The teacher stays off the deployed board; the student runs repeatedly. A related transfer signal trains the width-sharing supernet in TP03 (L03). Here we compare separate teacher/student models under controlled budgets.

By the end of this session you can

Compute

by hand

a softmax at temperature T, a CE and a KL term, and why the KD term is multiplied by T^2.

Explain

with the right words

dark knowledge, response vs feature transfer, attention maps, converged vs fixed-budget, same-data vs transfer-data.

Implement

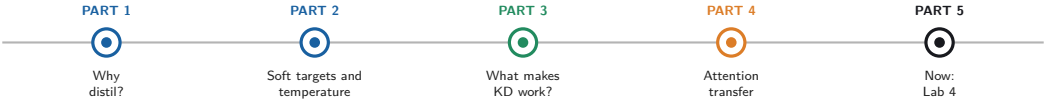
in PyTorch

response KD with `log_softmax`, `softmax`, `kl_div`, `batchmean`, a detached teacher – and check it.

Decide

with controls

which comparison isolates the KD objective, the effect of extra data and the chosen student capacity.



PART 1

Why distil?

Small networks are cheap to run and hard to train. A teacher changes the second half of that sentence.

1

Small models are easy to run – and hard to train

Capacity is not the only limit

a small network *could* represent a good function; gradient descent on one-hot labels rarely finds it.

Labels are a poor teacher

"this is a 9" and nothing else: at most $\log_2 10 \approx 3.3$ bits per image.

A trained network is a better teacher

a full distribution – for every image, labelled or not.

The classic result

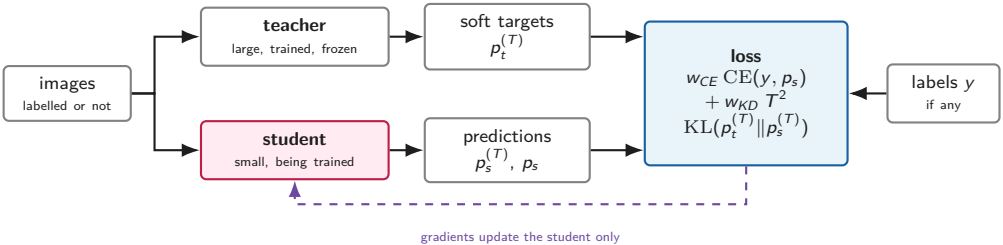
Hinton, Vinyals, Dean, 2015

speech recognition	frame acc.	WER
baseline (1 model)	58.9%	10.9%
ensemble of 10 models	61.1%	10.7%
1 model distilled from the ensemble	**60.8%**	**10.7%**

one model at inference cost captures almost all of what ten models learned. The idea is older: Buciluǎ et al., *Model Compression*, KDD 2006.

Distillation attacks the optimisation problem of small networks, not their capacity.

The idea: a teacher during training, a student in deployment



Offline KD

teacher trained first, then frozen; its outputs can be cached (Lab 4 script 05).

Any input will do

KD needs no label: unlabelled images become a *transfer set*.

Nothing of the teacher is deployed

the XMODEL is the student graph only.

Distillation changes the weights a student ends up with – never the cost of running it.

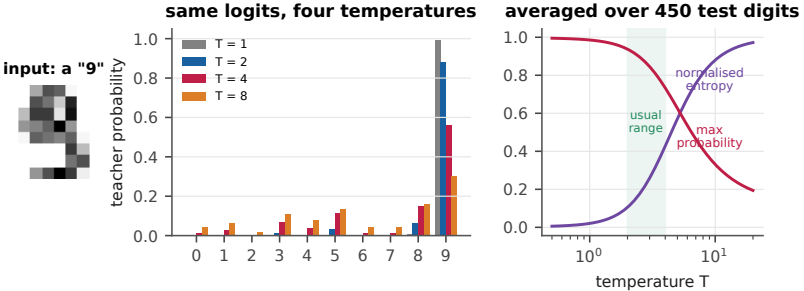
PART 2

Soft targets and temperature

2

What a teacher's distribution says that a label cannot, how temperature exposes it, and the loss that transfers it.

A label says “9”. The teacher says much more.



Left, middle: a trained teacher is almost certain it sees a 9 (grey). Divide its logits by T before the softmax and the **dark knowledge** appears: this 9 looks a bit like an 8, a 3 or a 5 – and nothing like a 6. That is the similarity structure of the classes.

Right: higher T spreads probability; at very high T probabilities approach uniform. With T^2 scaling, the loss still matches centered logits; it does not vanish. Moderate temperatures are a useful empirical starting point.

Computed for this course: CNN teacher (16→32 channels, 128 hidden) on scikit-learn `digits`. G. Hinton et al., *Distilling the Knowledge in a Neural Network*, 2015.

The label is one bit of certainty. The teacher’s full distribution tells the student how the classes relate.

Temperature: softer confidence, same ranking

$$p^{(T)}(i) = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)}, \qquad T > 0$$

T	A	B	C	D	norm. entropy
1	0.831	0.112	0.041	0.015	0.429
2	0.579	0.213	0.129	0.078	0.801
4	0.409	0.248	0.193	0.150	0.948
8	0.325	0.253	0.224	0.197	0.987

logits $z = [4, 2, 1, 0]$ – Lab 4 script 02a prints this table with `LOGITS=4,2,1,0`

For any two classes:

$$\frac{p^{(T)}(i)}{p^{(T)}(j)} = \exp\left(\frac{z_i - z_j}{T}\right)$$

The ranking never changes
$z_i > z_j \Rightarrow p_i > p_j$ for every $T > 0$: same argmax.

The ratios shrink
T compresses the logit gaps; secondary classes get visible probability.

High-T limit
$p \to$ uniform, yet $T^2 D_{KL} \to \frac{1}{2K} \|(z_s - \bar{z}_s) - (z_t - \bar{z}_t)\|^2$ (K classes): centered-logit matching, not zero.

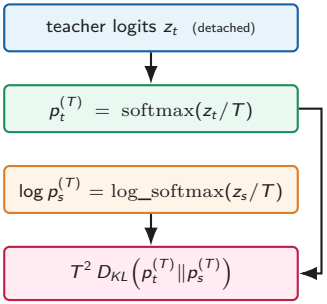
Temperature is not more knowledge. It controls how loudly the teacher's secondary opinions are heard.

The response-KD loss

$$\mathcal{L}_{KD} = T^2\, D_{KL}\left(p_t^{(T)} \,\|\, p_s^{(T)}\right)$$

$$= T^2 \sum_i p_t^{(T)}(i)\, \log \frac{p_t^{(T)}(i)}{p_s^{(T)}(i)}$$

$$\mathcal{L} = w_{CE}\, \mathcal{L}_{CE}(y, p_s) + w_{KD}\, \mathcal{L}_{KD}$$



Teacher = target

p_t is the distribution we want; p_s is the one we move. KL is not symmetric: the order matters.

Teacher fixed, both softened

its logits are detached (no gradient reaches the teacher); the *same* T divides both logits; CE uses the student at $T = 1$.

Response KD: reproduce the teacher's whole answer, not just its argmax.

Same cross-entropy, different distillation loss

Correct class A; the teacher says $p_t = [0.70,\ 0.20,\ 0.08,\ 0.02]$

	$p_s(A)$	CE $= -\log p_s(A)$	$D_{KL}(p_t \| p_s)$
student 1: $[0.60,\ 0.25,\ 0.10,\ 0.05]$	0.60	0.511	**0.027**
student 2: $[0.60,\ 0.05,\ 0.30,\ 0.05]$	0.60	0.511	**0.261**

What CE sees

only $p_s(A)$. Both students are equally good – CE cannot tell them apart.

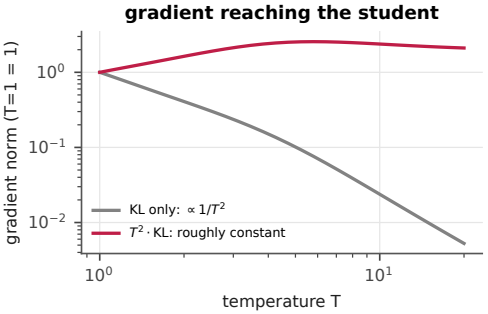
What KD sees

how the remaining 40% is spread. Student 2 thinks C is likely; the teacher thinks B is: 10× larger KL.

Same construction as Lab 4 script `02b_concept_response_kd.sh`, which uses its own students and also prints the gradients.

CE supervises the correct class. KD also shapes the mistakes – which is where a small network learns the geometry of the problem.

Why multiply the KD term by T^2?



Differentiating through z_s/T:

$$\frac{\partial D_{KL}}{\partial z_s} = \frac{1}{T}\left(p_s^{(T)} - p_t^{(T)}\right)$$

and at high T the difference itself shrinks roughly as $1/T$. So the KD gradient decays like $1/T^2$ (grey, measured).

The correction
multiplying by T^2 keeps the KD signal at a stable scale (red) – so changing T does not silently re-weight KD against CE.

Only the KD term
not CE, not attention, not the whole objective.

Measured for this course: gradient norm at the student logits, trained CNN teacher vs untrained student, 256 digits.

T^2 restores the gradient scale; without it, a high T switches KD off.

Why keep the labels – and why coefficients are not contributions

CE is the anchor

- the ground truth, where the teacher is wrong;
- a direct classification objective;
- protection against a poorly calibrated teacher.

KD-only removes the anchor: useful as an ablation, rarely the final choice.

$w_{KD} = 0.7$ does not mean "70% KD"

the optimiser sees $w_{CE}\mathcal{L}_{CE}$ and $w_{KD}\mathcal{L}_{KD}$ – raw values and gradients of different scales. A small coefficient on a large term can still dominate.

Lab 4 prints both: `RAW KD` and `WT KD` $= w_{KD}\mathcal{L}_{KD}$, epoch by epoch.

$w_{CE} = 0.3$ $w_{KD} = 0.7$ – Lab 4 default, at $T = 2$

coefficients: a starting point to investigate, one factor at a time

Read the weighted contributions, not the coefficients – and change one of them at a time.

Response KD: training contract and implementation checks

The training contract of Lab 4

1. Teacher in evaluation mode, student in training mode: the teacher is fixed context.
2. Teacher targets without gradient, at temperature T: $p_t^{(T)} = \mathrm{softmax}(z_t / T)$, computed under `no_grad` or read from the cache.
3. Student at the same T: $\log p_s^{(T)} = \log_\mathrm{softmax}(z_s / T)$.
4. $\mathcal{L}_{KD} = T^2 \, \frac{1}{B} \sum_{b=1}^{B} D_{KL}\left(p_{t,b}^{(T)} \,\|\, p_{s,b}^{(T)}\right)$: one KL per sample, averaged over the batch.
5. $\mathcal{L}_{CE}$ on the labelled samples only, and skipped when a batch has none.
6. $\mathcal{L} = w_{CE}\,\mathcal{L}_{CE} + w_{KD}\,\mathcal{L}_{KD}$ with the recorded coefficients; only the student is updated; validation runs in evaluation mode.

1. Argument order

`kl_div(input, target)`: student *log*-probabilities first, teacher probabilities second.

2. `batchmean`

the true KL per sample; `mean` also divides by the number of classes.

3–5. Detach, T^2, `dim=1`

teacher under `no_grad()`; T^2 on the KD term only; softmax over classes, not the batch.

Lab 4's challenge checks exactly these behaviours – reversed KL, missing T^2, an updated teacher all fail.

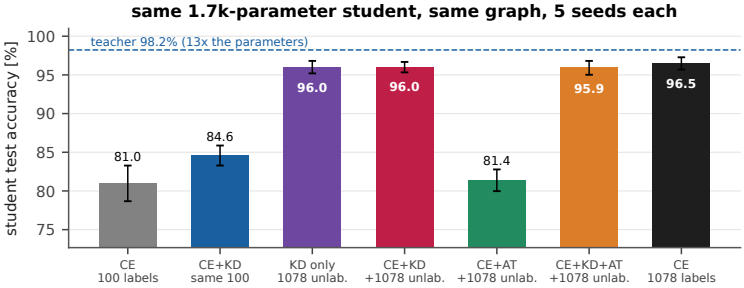
PART 3

What makes KD work?

Controlled experiments: where the gain comes from, how temperature and balance matter, and what a fair comparison needs.

3

Where does the gain come from? Seven students, one graph



The loss alone (same 100 labelled images): CE+KD gains **+3.6 points** over CE. Soft targets carry more per image than labels.

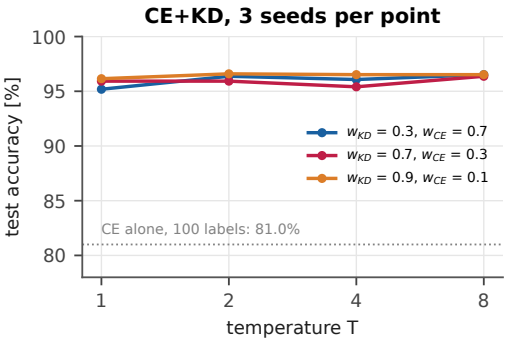
The data (+1 078 unlabelled images the teacher labels softly): another **+11 points**, to 96.0%. The teacher turns unlabelled data into supervision.

The ceiling: 96.0% with 100 labels is close to the 96.5% of the same student trained on all 1 078 labels – and below the teacher: capacity still matters.

Computed for this course: CNN teacher 16→32/128 (22.6 k parameters); student 4→8/32 (1.7 k); `digits`; best-validation checkpoint; mean ± std over 5 seeds. Lab 4 uses the same widths on MNIST, 1 000 labels + 10 000 transfer images.

Before crediting the KD loss, separate what the loss did from what the extra data did.

Temperature and balance: a small sweep, a flat optimum



$T = 1$ is already useful

without any softening, KD on the transfer set reaches 95–96%: most of the gain is the teacher-labelled data.

A plateau

every setting from $T = 1$ to 8 and w_{KD} from 0.3 to 0.9 lands between 95.2% and 96.6% – far above CE alone (81.0%).

Read it as a study, not a recipe

3 seeds per point, one student, one dataset: differences below one point are within the noise. Lab 4 lets you change `T`, `CE_WEIGHT`, `KD_WEIGHT` one at a time.

Computed for this course: CE on 100 labels + KD on 1 078 transfer images, same student and teacher as the previous slide.

KD is robust to its hyper-parameters in a reasonable range. The data regime matters far more than the third decimal of T.

Teacher quality and student capacity

The teacher must be better

and trained before the student

a teacher that is only marginally better – or wrong in a systematic way – teaches its mistakes. Check the teacher's validation accuracy before distilling from it.

Bigger is not always better

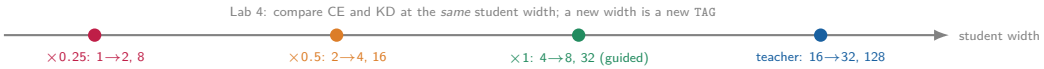
Cho & Hariharan, ICCV 2019; Mirzadeh et al., AAAI 2020

a very large teacher can be too hard to imitate: an intermediate "teacher assistant" or an early-stopped teacher can transfer more.

Students do not copy perfectly

Stanton et al., NeurIPS 2021

even when it helps accuracy, a student often agrees with its teacher far less than expected: KD is a regulariser as much as a copy.



KD helps most when the student has room to learn from a clearly better teacher. It cannot give a student capacity it does not have.

What makes a KD comparison fair?

Same student
same graph, same initial weights (Lab 4 prints an initialisation hash), same data split.

Same stopping rule
patience on *validation* accuracy, best checkpoint restored, test set evaluated once.

Same data – or say so
CE on 1 000 labels vs KD on 10 000 transfer images compares data regimes, not only losses. `TRANSFER_SAMPLES=1000` isolates the loss.

Converged or fixed budget?
"KD reaches 95% two epochs earlier" (speed) is a different claim from "KD ends higher" (final quality). A fixed-compute claim needs a budget-matched study.

Several seeds
a one-point gain on one seed is not a result; the Lab 4 reference runs are a starting point.

Selected epoch $\neq$ last epoch
different methods may stop at different epochs under the same rule – that is not unfair.

A KD gain is only a KD gain once graph, initialisation, data, optimiser and stopping rule are shown to be the same.

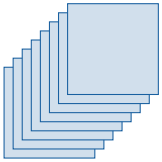
PART 4

Attention transfer

4

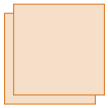
Distilling intermediate representations when teacher and student do not even have the same number of channels.

Matching features: the channel problem



teacher: 32 channels

$\neq$



student: 8 channels

Direct matching fails

different channel counts and scales; channel k of one network need not mean channel k of the other.

FitNets (2015)

a trainable adapter maps student to teacher features: more parameters, more hyper-parameters.

Attention transfer: match *where*, not *what*

collapse the channels into one spatial energy map, then normalise:

$$Q(F)_{h,w} = \frac{1}{C}\sum_{c} F_{c,h,w}^2, \qquad A(F) = \frac{\mathrm{vec}(Q(F))}{\|\mathrm{vec}(Q(F))\|_2}$$

$$\mathcal{L}_{AT} = \frac{1}{|\mathcal{S}|}\sum_{\ell\in\mathcal{S}} \left\| A(F_s^\ell) - A(F_t^\ell) \right\|_2^2$$

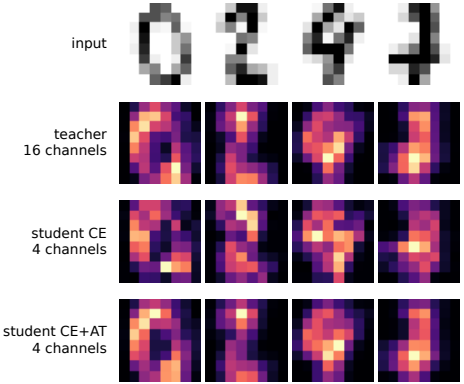
Channel counts may differ; the **spatial resolutions** of the chosen stages must match.

A. Romero et al., *FitNets*, ICLR 2015; S. Zagoruyko, N. Komodakis, *Paying More Attention to Attention*, ICLR 2017.

Attention transfer keeps the spatial pattern and drops channel identity.

What the student learns to look at

First-stage attention maps (8×8), four test digits.



Teacher
16 channels: energy on the discriminative strokes.

Student, CE only
its own pattern: a mean squared distance of 0.18 to the teacher's maps (test set).

Student, CE+AT
distance 0.009 – 20× closer: it now looks where the teacher looks.

But accuracy barely moves
81.0% → 81.4% (5 seeds, 100 labels): looking in the right place is not knowing the class.

Computed for this course: attention after the first convolution + ReLU; AT on both stages; same teacher and student.

A smaller attention distance shows imitation, not accuracy: measure both.

Attention transfer: feature contract and Lab 4 objectives

```
# Contract: features have shape [N, C, H, W].
# Aggregate squared responses over channels.
# Flatten spatial positions for each sample.
# Normalize each attention vector with an L2 norm.
# Compare matching spatial stages, not channel counts.
# Freeze teacher targets and keep teacher in eval mode.
# Average the stage losses as specified in the handout.
# Implement the numerical details in the student TODOs.
```

student	labels	soft logits	attention	script
CE	✓			04
KD-only		✓		06a
CE+KD	✓	✓		06b
CE+KD+AT	✓	✓	✓	06c
CE+AT	✓		✓	06d

CE+KD adds response transfer; CE+AT adds feature transfer. To isolate the objective, match data access as well as initialization and stopping. AT reads the teacher's attention maps, which Lab 4 caches together with its logits; the teacher runs online only with random augmentation. The deployed student graph stays unchanged.

Five objectives, one student graph: every difference in the table is a difference in what the student was told.

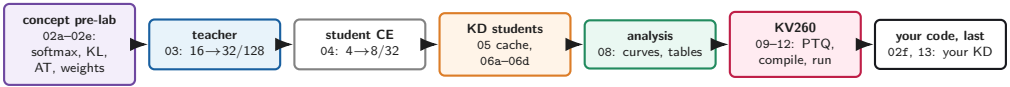
PART 5

Now: Lab 4

A concept pre-lab on small tensors, a teacher and five controlled students on MNIST, a deployment on the KV260 – and, last, your own KD implementation.

5

Lab 4 in one picture



Setting	Guided value (fixed by the `TAG` at setup)
data regime	MNIST, 1 000 labelled images + 10 000-image transfer set (`TRANSFER_SAMPLES`)
objective	$0.3\,\mathcal{L}_{CE} + 0.7\,\mathcal{L}_{KD}$ at $T = 2$; `AT_WEIGHT` for the attention variants
stopping	no epoch ceiling (`MAX_EPOCHS=0`): patience on validation accuracy, best checkpoint restored, test once
controls	one student graph and one initialisation hash for all variants of a `TAG`

Change structural choices (width, data budget) with a new `TAG`; change training hyper-parameters on the command line – and they are recorded.

Deployment: same graph, same cost – different weights

```
TAG=your_mnist_kd bash scripts/09_prepare_host_comparison.sh # host
TAG=your_mnist_kd bash scripts/10_quantize_compile_all.sh # PTQ + vai_c_xir for every student
TAG=your_mnist_kd bash scripts/11_copy_all_to_board.sh
cd /home/root/c5/TP04/your_mnist_kd # on the KV260
bash 12_run_all_fpga_variants_on_board.sh .
bash 12b_profile_energy_all_variants_on_board.sh .
bash 12c_dpu_profile_on_board.sh . # xdputil, vaitrace
TAG=your_mnist_kd bash scripts/12_fetch_and_summarize.sh # back on the host
```

Expect the same

parameters and dense MACs: the student graph is unchanged. Compiled size, latency and energy require measurements; fixed costs and compiler choices also matter.

Expect differences in

float, PTQ and KV260 accuracy – the weights differ.

If the cost differs

check graph, compiler metadata, timing variance and measurement scope before attributing it to architecture.

Distillation changes training while preserving the student architecture. Structural costs stay fixed; measure deployment behavior separately.

Final task: implement response KD yourself

```
# challenge_kd/student/kd_experiment.py
def response_kd_loss(student_logits, teacher_logits, T):
    ... # T^2 * KL(p_t^T || p_s^T), teacher detached

def training_step(student, teacher, images, labels, optimizer,
                  method, temperature, ce_weight, kd_weight,
                  label_mask=None, **context):
    ... # forward, CE + KD, backward, update the student
```

```
TAG=your_challenge TRANSFER_SAMPLES=1000 \
TEACHER_SOURCE=copy SOURCE_TAG=your_mnist_kd \
  bash scripts/00_setup_experiment.sh
C=your_challenge # shorthand used below
TAG=$C bash scripts/03d_copy_teacher_from_tag.sh
TAG=$C bash scripts/13_check_challenge.sh
TAG=$C METHOD=ce bash scripts/13_run_challenge.sh
TAG=$C METHOD=ce_kd bash scripts/13_run_challenge.sh
TAG=$C bash scripts/13_validate_challenge.sh
```

Same data on purpose

`TRANSFER_SAMPLES=1000`: CE and CE+KD see the same images, so the comparison isolates the loss.

Behavioural checks

the value of $T^2 D_{KL}$, detachment, temperature scaling, teacher immutability, the limits $w_{CE} = 0$ and $w_{KD} = 0$.

Your conclusion

better validation accuracy, better test accuracy, faster convergence – or none? Then one extension with a new `TAG` and a falsifiable prediction.

The validator checks that your comparison is controlled. It does not decide whether your conclusion is right – your table does.

Before you start the lab

Checklist

1. Concept pre-lab first
predict each printed number before reading it (C1–C4); your own primitives (C5) come at the end, with the final task.

2. Check the teacher
is it clearly better than the student baseline? If not, stop and fix it.

3. Read raw and weighted losses
`RAW KD` vs `WT KD`, epoch by epoch.

4. Validation for choices, test once
never tune on the test set.

Predictions to write down now

CE vs CE+KD
how much of the gain comes from the 10 000 transfer images?

CE+AT
smaller attention distance – better accuracy?

On the board
which columns of the final table should be identical across students?

Keep your best student: S5 deploys a frugal model that combines pruning, distillation and INT8.

Distillation experiments are easy to run and easy to misread. Controls first, conclusions second.

THE THREAD – WHAT TODAY ADDED

How much useful intelligence can we obtain from a limited budget of computation, memory and energy?

A **teacher** transfers information from its responses or intermediate features to a compact student, without increasing the student's inference graph.

Pruning and architecture search chose the network shape in Lab 3; distillation improves how that shape is trained. Next session measures the resulting system on the SoC, including application FPS and energy.

Takeaways

Soft targets	The teacher's full distribution carries the class structure a one-hot label discards; temperature makes it audible.
The loss, correctly	$T^2\, D_{KL}(p_t^{(T)} \| p_s^{(T)})$ with a detached teacher, `batchmean`, plus CE as the anchor.
Data vs loss	Much of a KD gain can come from teacher-labelled extra data. Same-data controls isolate the loss.
Same deployed cost	Teacher, KL and attention never reach the XMODEL: KD composes with pruning and INT8 for free.
Fair model selection	Choose the best epoch on validation data, keep test results out of selection, and report training cost alongside accuracy.

Distillation spends compute during training to improve a student with the same inference cost. Compare methods with explicit controls.

Sources and further reading

Distillation

- C. Buciluǎ, R. Caruana, A. Niculescu-Mizil, *Model Compression*, KDD 2006.
- G. Hinton, O. Vinyals, J. Dean, *Distilling the Knowledge in a Neural Network*, arXiv:1503.02531, 2015.
- A. Romero et al., *FitNets*, ICLR 2015; S. Zagoruyko, N. Komodakis, *Attention Transfer*, ICLR 2017.
- J. H. Cho, B. Hariharan, *On the Efficacy of Knowledge Distillation*, ICCV 2019.
- S. I. Mirzadeh et al., *Teacher Assistant*, AAAI 2020.
- S. Stanton et al., *Does Knowledge Distillation Really Work?*, NeurIPS 2021.
- J. Gou et al., *Knowledge Distillation: A Survey*, IJCV 2021.

Further distillation reading

- G. Hinton et al., *Distilling the Knowledge in a Neural Network*, 2015.
- S. Zagoruyko, N. Komodakis, *Attention Transfer*, ICLR 2017.
- S. Stanton et al., *Does Knowledge Distillation Really Work?*, NeurIPS 2021.

Case study

- A. Mazouz et al., arXiv:2503.04832, 2025.

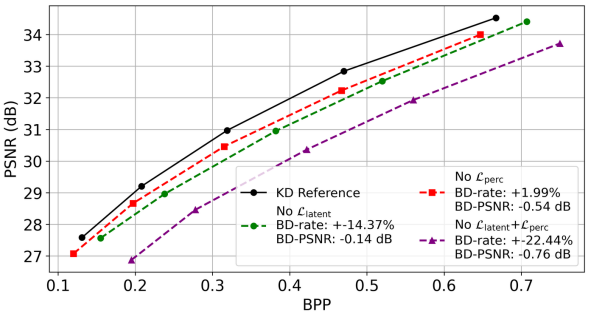
Figures marked "computed/measured for this course" come from the course's figure scripts, which are not part of the student release.

FOR READING

Supplementary material

A case study: distilling a learned image codec for the KV260.

Case: distilling a learned image codec for the KV260



A codec student must reproduce *images*, not a class: the teacher's intermediate representations carry what the pixels need.

Response + latent + perceptual
the student is trained to match the teacher's output, its latent representation, and perceptual features.

Remove the feature terms
-0.14 dB (no latent), -0.54 dB (no perceptual), -0.76 dB (neither), in BD-PSNR.

A. Mazouz et al., *Lightweight FPGA Deployment of Learned Image Compression with Knowledge Distillation and Hybrid Quantization*, arXiv:2503.04832, 2025.

For dense outputs, matching what the teacher computes inside matters as much as what it outputs – the idea of the next part.