# Efficient and resource-constrained dynamic neural networks

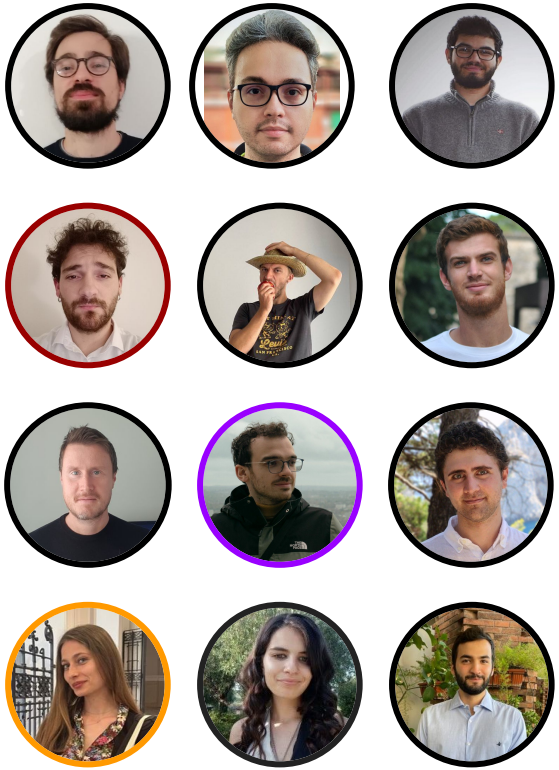
**Simone Scardapane**

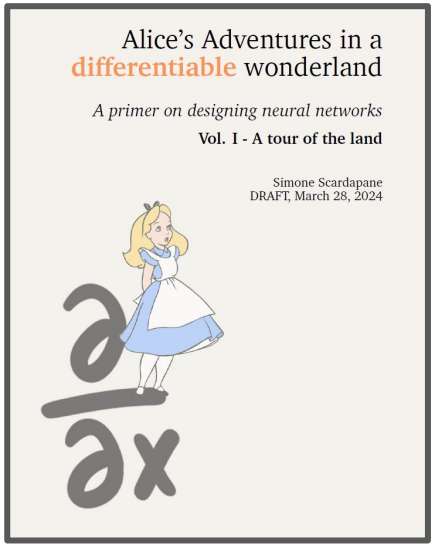
SAPIENZA
UNIVERSITÀ DI ROMA

intelligent signal processing
and multimedia lab

# About “us”


Simone Scardapane
Associate Professor, **Sapienza**
Affiliate Researcher, **INFN**
*Member*, **CNIT** / **ELLIS**
*Junior Fellow*, **Sapienza School of Advanced Study**

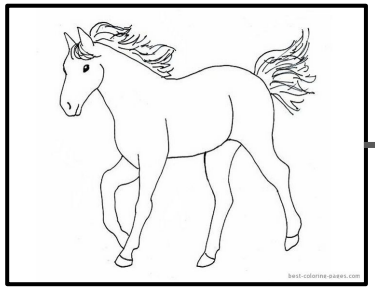





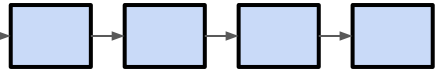

Book: Alice’s Adventures in a differentiable wonderland - Simone Scardapane

https://www.sscardapane.it/

https://twitter.com/s_scardapane



Repository



“*Horse*”

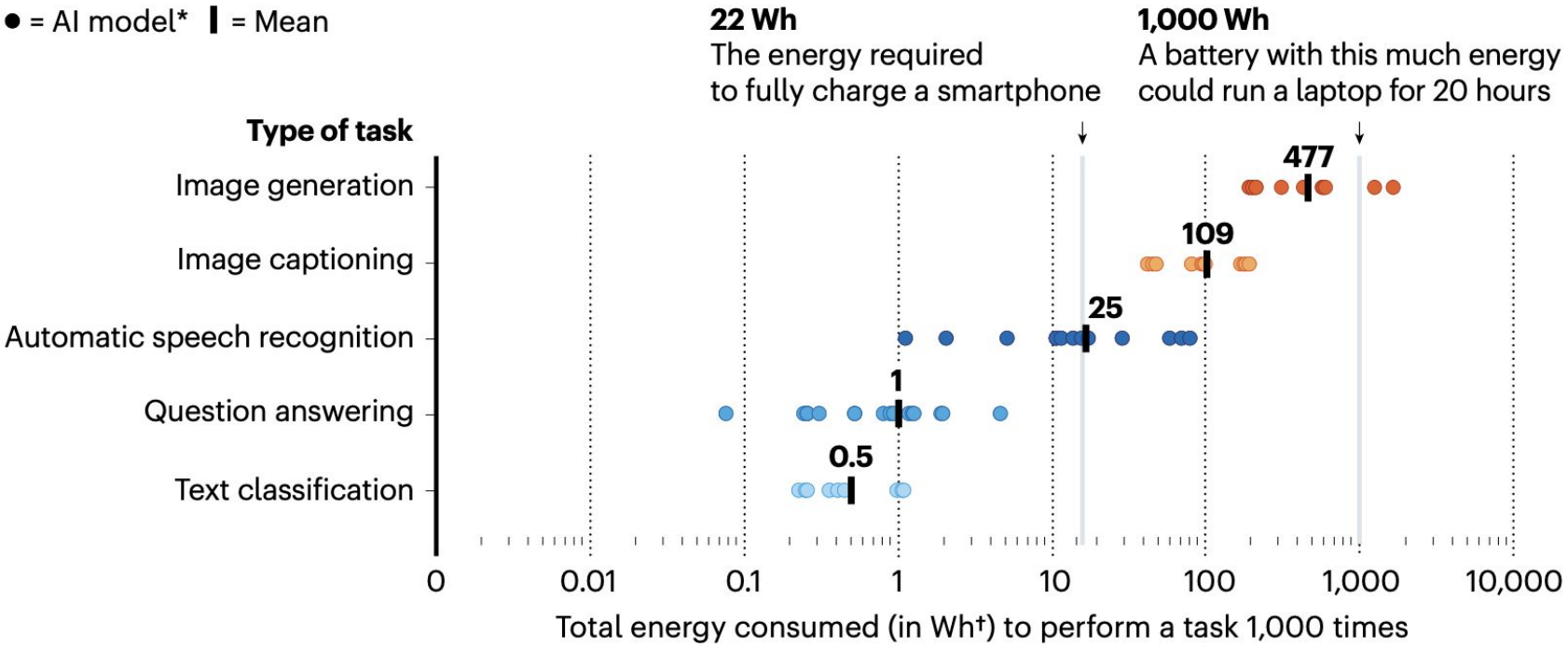
**Fixed**: depends on the **model** and the **hardware**

Computational cost

Light bulbs have energy ratings — so why can't AI chatbots?

# AI'S ENERGY FOOTPRINT

The power consumed by artificial intelligence (AI) tools varies greatly depending on the task. An AI model that provides answers to queries is much less energy-intensive than one that generates images from text prompts, for example. And the data show that even AI models of the same type can vary widely in energy consumption.




*Tests conducted on 20 popular open-source models. Each dot represents one model; †1 Watt-hour represents power consumption of 1 W extended over 1 hour.

**Elon Musk** ✓ 𝕏
@elonmusk

Abbonati ···

230k GPUs, including 30k GB200s, are operational for training Grok @xAI in a single supercluster called Colossus 1 (inference is done by our cloud providers).

At Colossus 2, the first batch of 550k GB200s & GB300s, also for training, start going online in a few weeks.

**Sam Altman** ✓
@sama

···

we have signed a deal for an additional 4.5 gigawatts of capacity with oracle as part of stargate. easy to throw around numbers, but this is a _gigantic_ infrastructure project.

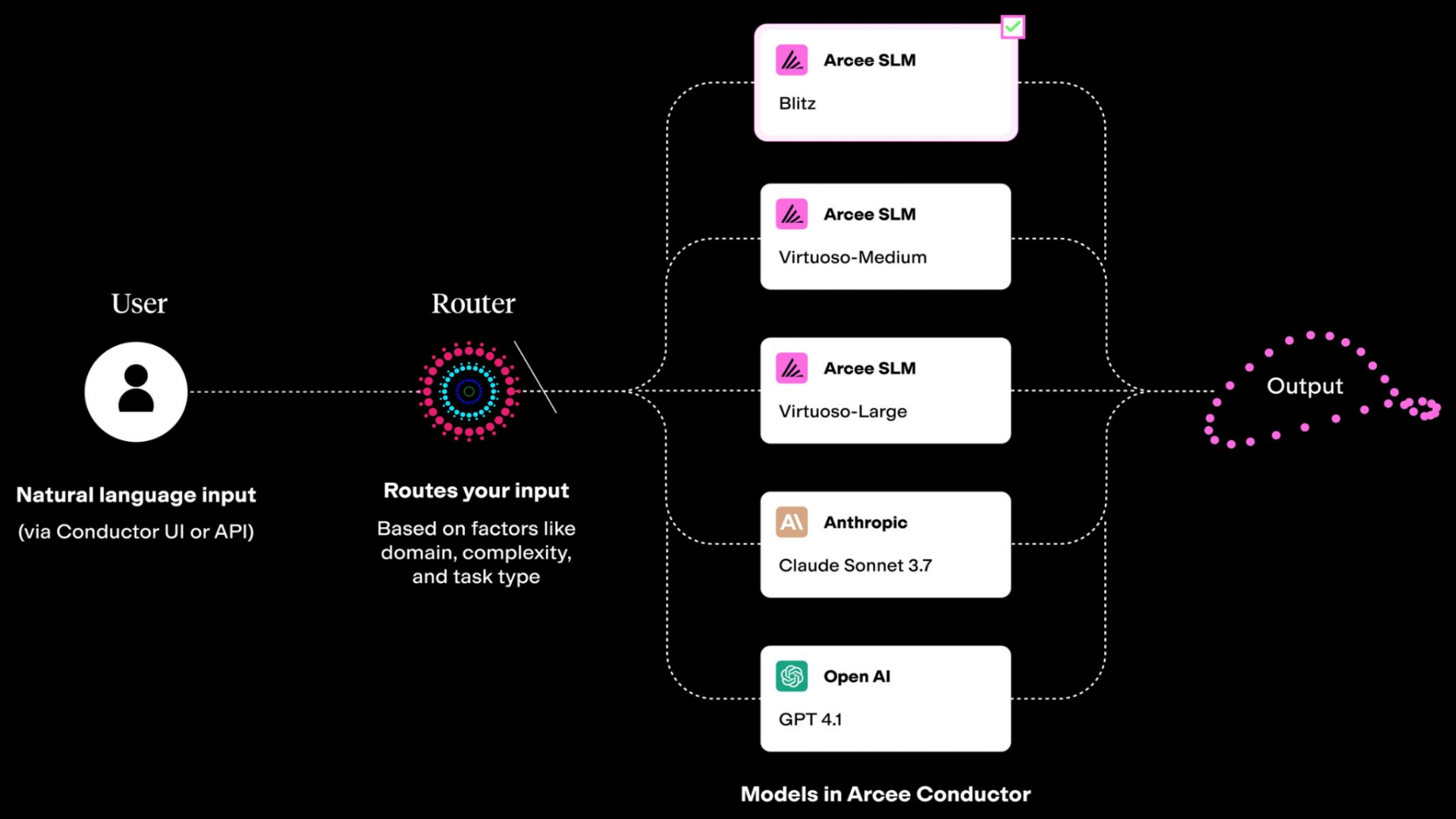
some progress photos from abilene:

Traduci post

# Mistral AI Secures $1 Billion Funding for Europe's Largest AI Data Center

Coin World • Wednesday, Jul 9, 2025 3:10 am ET

4min read

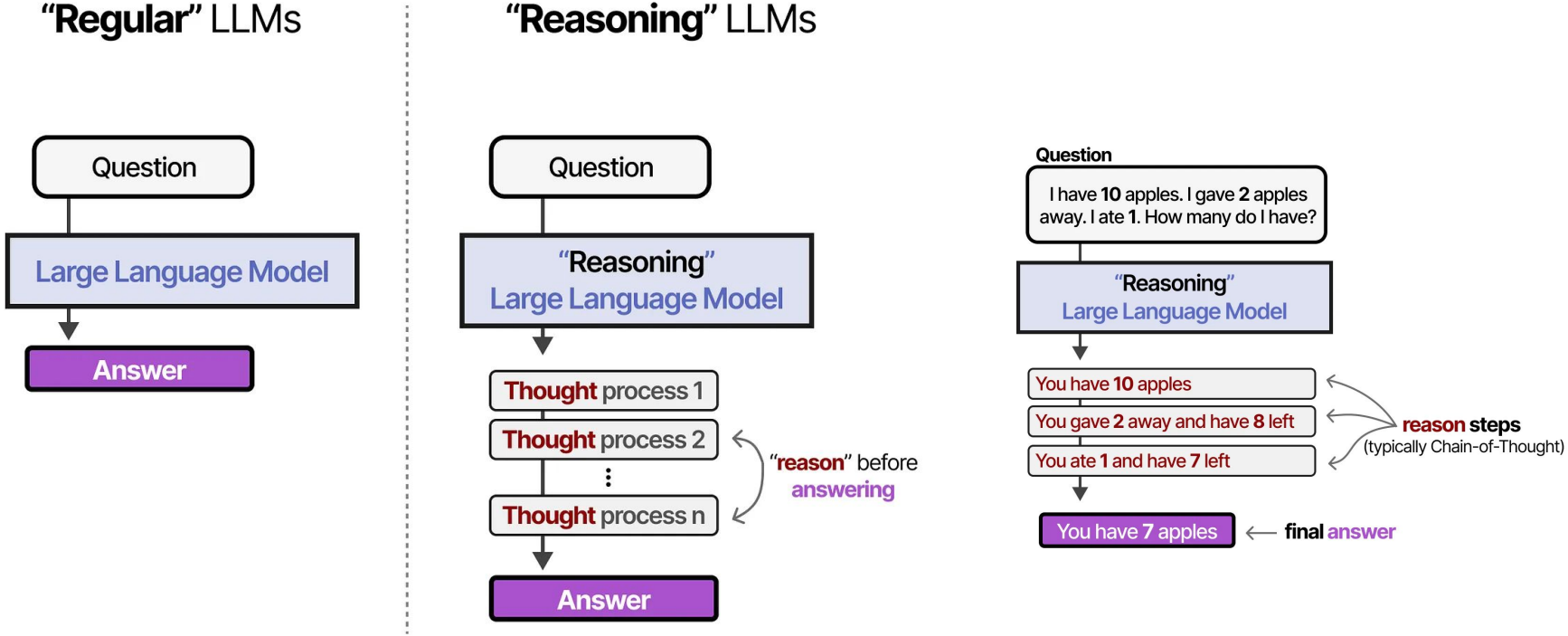

User
Router
Arcee SLM
Blitz
Arcee SLM
Virtuoso-Medium
Arcee SLM
Virtuoso-Large
Anthropic
Claude Sonnet 3.7
Open AI
GPT 4.1
Output
Natural language input
(via Conductor UI or API)
Routes your input
Based on factors like domain, complexity, and task type
Models in Arcee Conductor

# Test-time scaling

Can we go **higher** with the same model?

# “Reasoning”

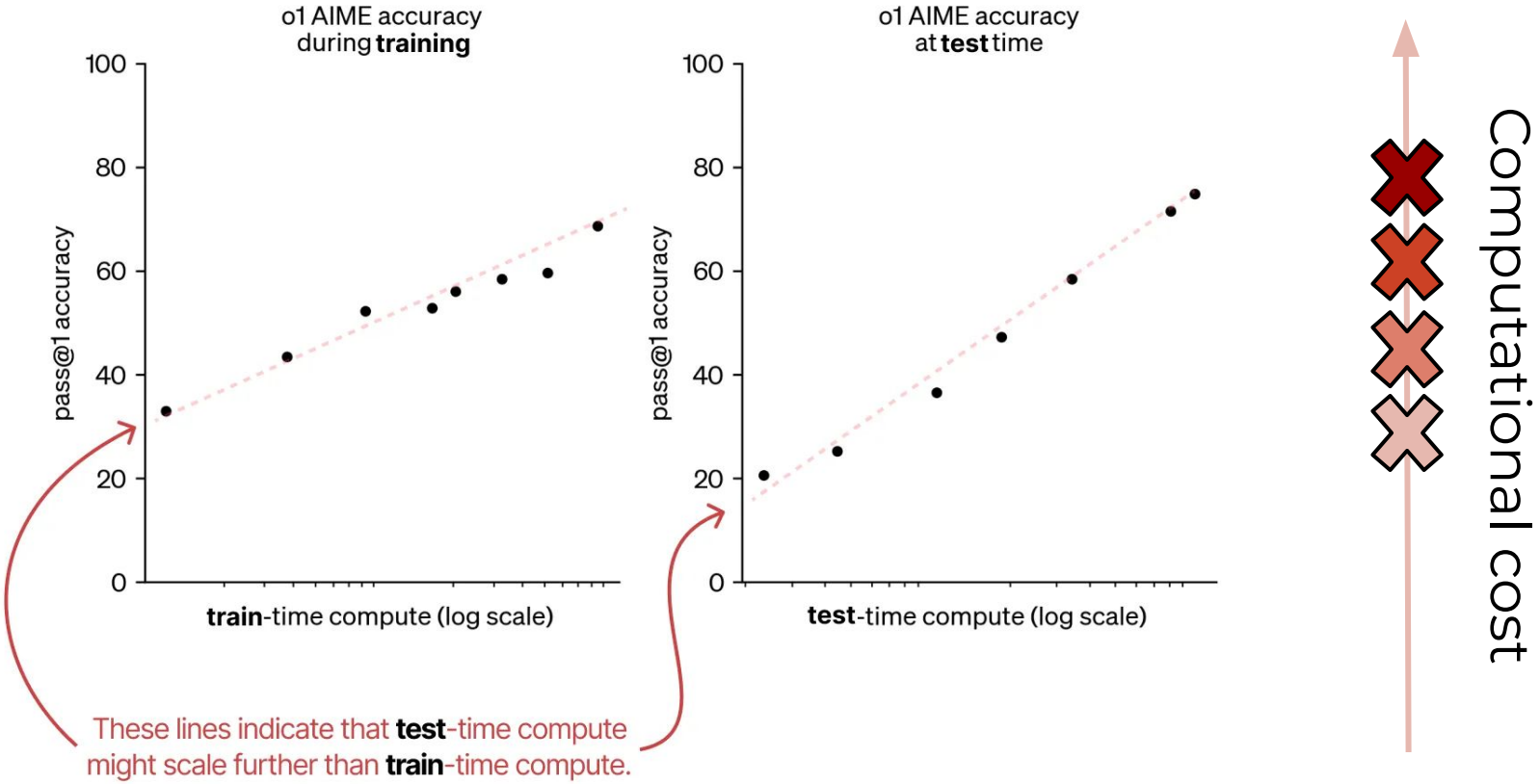
[A Visual Guide to Reasoning LLMs - by Maarten Grootendorst]




Cost depends (more or less linearly) on how many *reasoning* tokens are needed.

# Scaling laws for inference

[A Visual Guide to Reasoning LLMs - by Maarten Grootendorst]



# Looped models

[2311.12424] Looped Transformers are Better at Learning Learning Algorithms

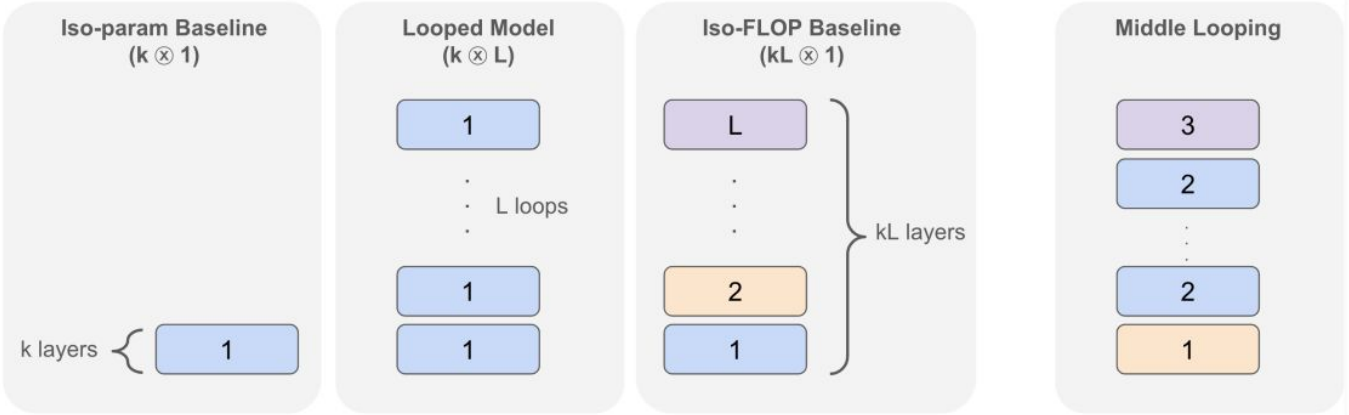



Figure 1: Illustration of the simple and architecture agnostic looping mechanism that we consider. A $k$-layer block looped $L$ times (**middle**) is denoted by $(k \otimes L)$, which can essentially be viewed as a weighted shared model. The iso-param baseline, $(k \otimes 1)$, is a $k$-layer model with the same number of *distinct* parameters. The iso-FLOP baseline, $(kL \otimes 1)$, is a $kL$-layer model with the same depth but $L$ times more parameters. Middle looping is a strategy that is inspired from prior works on model stacking (e.g. (Saunshi et al., 2024)).

# “Tiny” recursive models

[2510.04871] Less is More: Recursive Reasoning with Tiny Networks

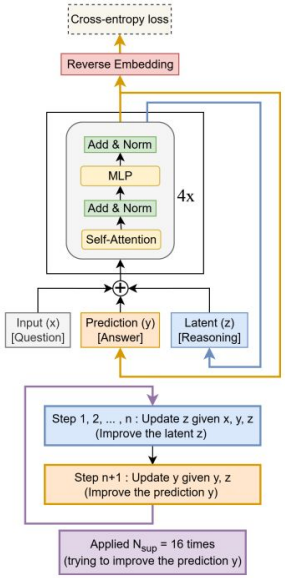



*Figure 1.* Tiny Recursion Model (TRM) recursively improves its predicted answer $y$ with a tiny network. It starts with the embedded input question $x$ and initial embedded answer $y$, and latent $z$. For up to $N_{sup} = 16$ improvements steps, it tries to improve its answer $y$. It does so by i) recursively updating $n$ times its latent $z$ given the question $x$, current answer $y$, and current latent $z$ (recursive reasoning), and then ii) updating its answer $y$ given the current answer $y$ and current latent $z$. This recursive process allows the model to progressively improve its answer (potentially addressing any errors from its previous answer) in an extremely parameter-efficient manner while minimizing overfitting.

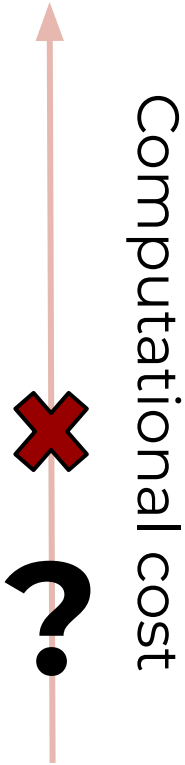
```python
def latent_recursion(x, y, z, n=6):
    for i in range(n): # latent reasoning
        z = net(x, y, z)
    y = net(y, z) # refine output answer
    return y, z

def deep_recursion(x, y, z, n=6, T=3):
    # recursing T-1 times to improve y and z (no gradients needed)
    with torch.no_grad():
        for j in range(T-1):
            y, z = latent_recursion(x, y, z, n)
    # recursing once to improve y and z
    y, z = latent_recursion(x, y, z, n)
    return (y.detach(), z.detach()), output_head(y), Q_head(y)

# Deep Supervision
for x_input, y_true in train_dataloader:
    y, z = y_init, z_init
    for step in range(N_supervision):
        x = input_embedding(x_input)
        (y, z), y_hat, q_hat = deep_recursion(x, y, z)
        loss = softmax_cross_entropy(y_hat, y_true)
        loss += binary_cross_entropy(q_hat, (y_hat == y_true))
        loss.backward()
        opt.step()
        opt.zero_grad()
        if q_hat > 0: # early-stopping
            break
```

*Figure 3.* Pseudocode of Tiny Recursion Models (TRMs).

# Test-time “downscaling”

A single model that **dynamically reduces** its computational footprint depending on, e.g., user requests, the environment, or input complexity.

“*Do not use more than 3% battery.*”

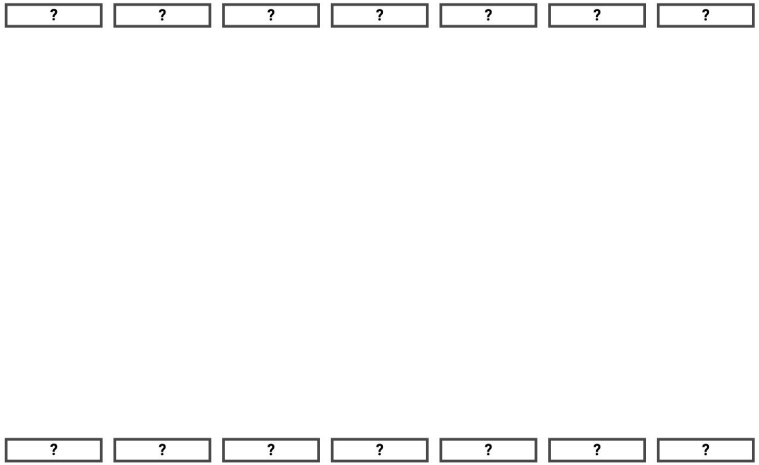


# Dynamic neural networks

Motivating **examples**

# Not all layers are needed

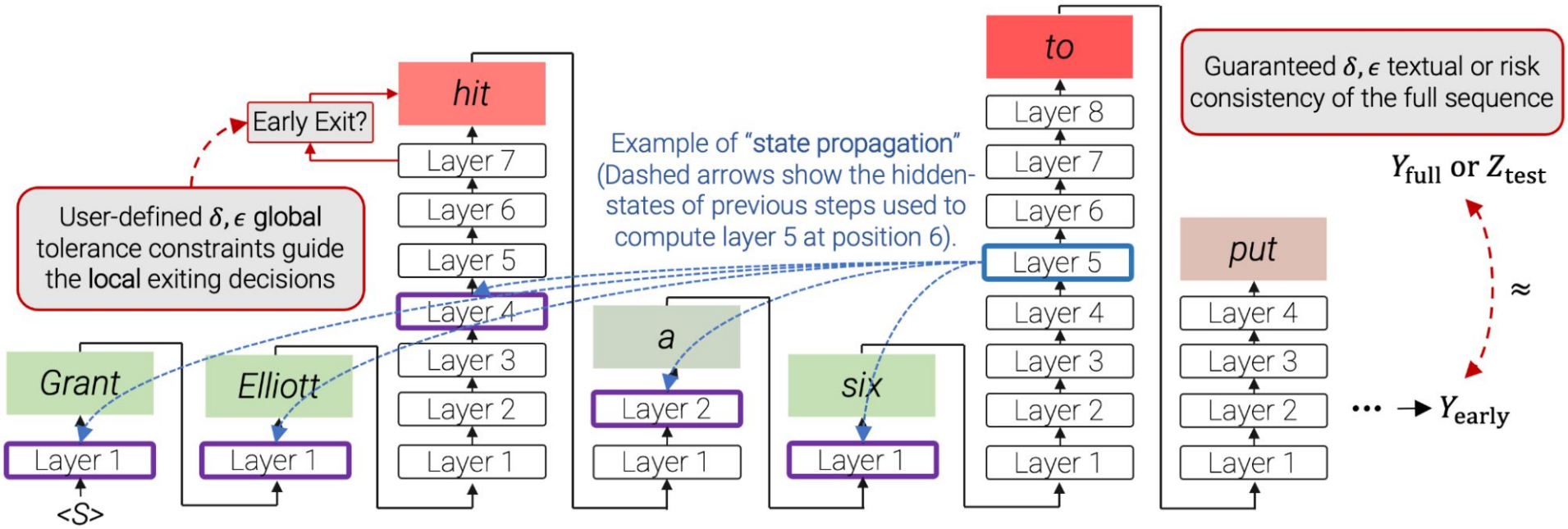
(Schuster et al., 2022)



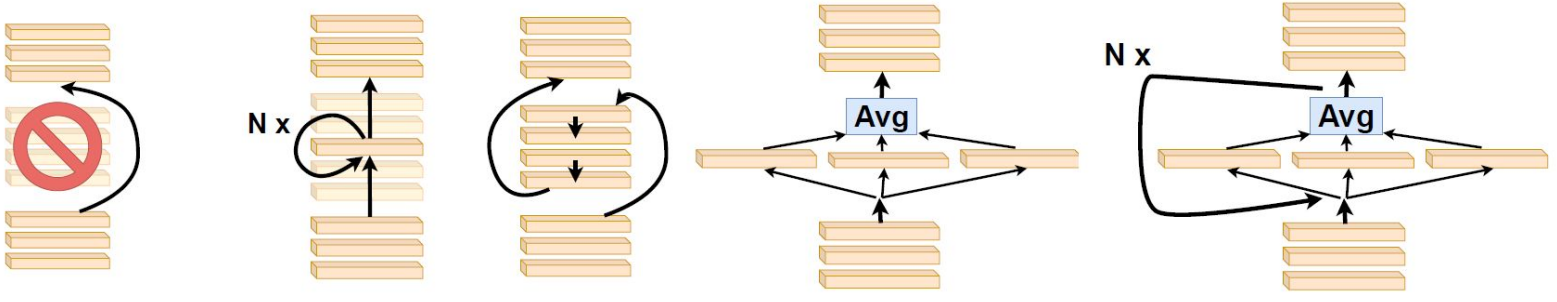

Known as **early exiting** in the literature: end the computation at an earlier layer for “simple” inputs or tokens.

# An example: Confident Adaptive LM

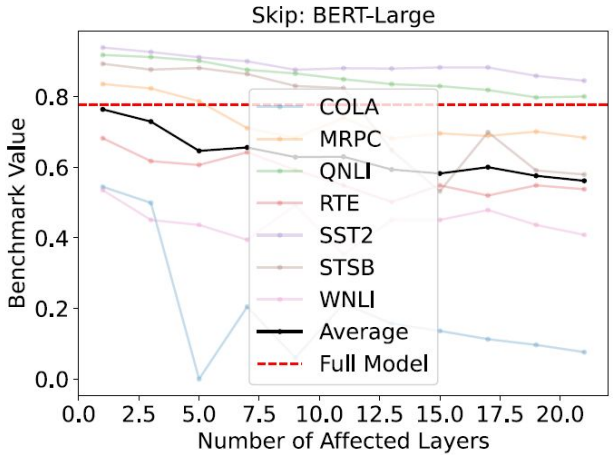
(Schuster et al., 2022)

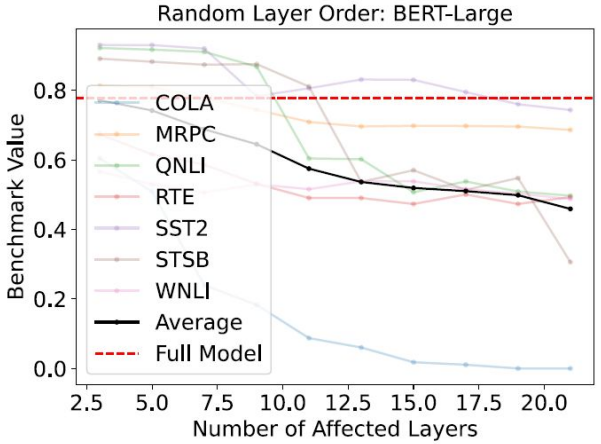


[2407.09298] Transformer Layers as Painters



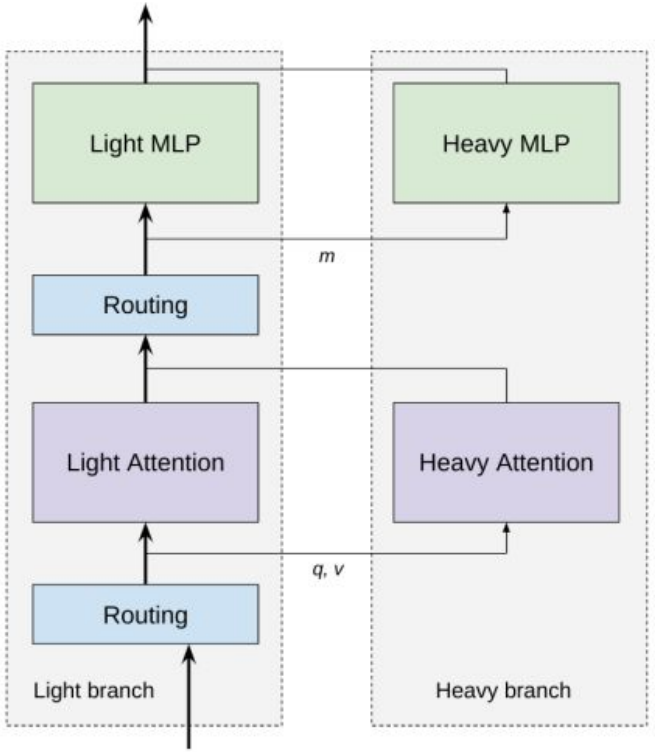

(a) Skip (b) Middle Repeat (c) Reverse (d) Parallel (e) Looped Parallel



# The *entire layer* is not needed

[2303.09752] CoLT5: Faster Long-Range Transformers with Conditional Computation



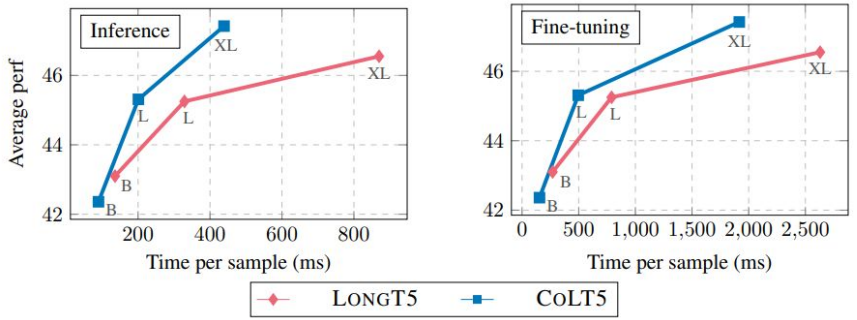




Figure 2: **CoLT5 achieves stronger performance than LongT5 at any speed.** Average performance on all datasets as a function of inference and fine-tuning time per sample (ms) for LongT5 and CoLT5 Base, Large, and XL models. LongT5 does not use MQA, but we report speed as though it had for a conservative baseline.

# Mixture-of-depths

[2404.02258] Mixture-of-Depths: Dynamically allocating compute in transformer-based language models

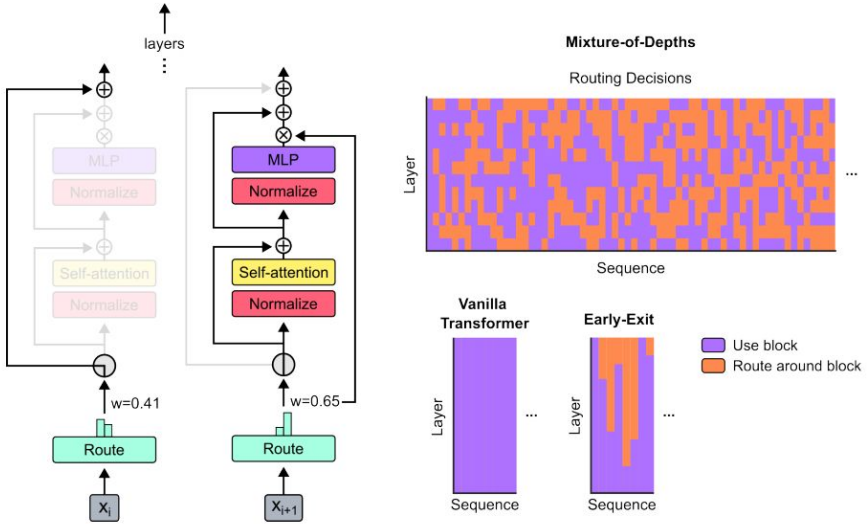



Figure 1 | **Mixture-of-Depths Transformer.** As in mixture-of-experts (MoE) transformers we use a router to choose among potential computational paths. But unlike in MoE transformers the possible choices are a standard block's computation (i.e., self-attention and MLP) or a residual connection. Since some tokens take this second route, Mixture-of-Depths (MoD) transformers have a smaller total FLOP footprint compared to vanilla or MoE transformers. On the top right is depicted a trained model's routing decisions for a short sequence truncated to 64 tokens for visualization purposes. When examining the choices one can find tokens processed by later blocks' layers, despite passing through relatively few total blocks throughout the model's depth. This is a unique feature of MoD compared to conventional halting-based, or "early-exit" conditional computation, which instead engage blocks serially, or vanilla transformers, which engage every block.

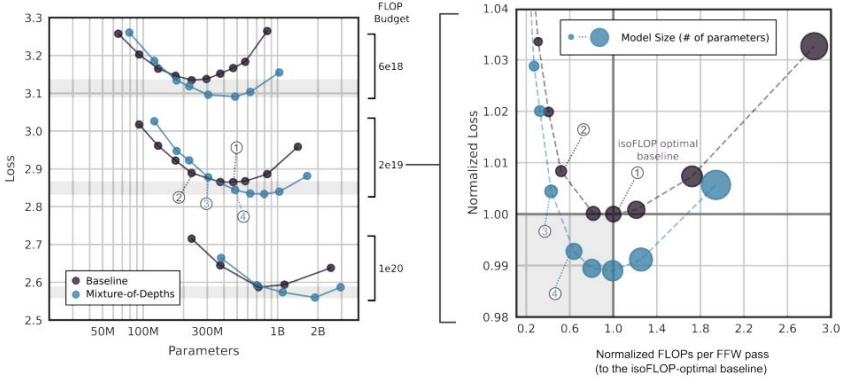



Figure 4 | **isoFLOP analysis.** We used the 12.5% capacity MoD variant to perform an isoFLOP analysis for 6e18, 2e19, and 1e20 FLOPs, training models varying in size from 60M to 3B parameters. Depicted on the right are the relative FLOPs per forward pass (normalized to the isoFLOP optimal baseline). There exist MoD variants that are both faster to step (by virtue of requiring fewer FLOPs per forward pass) and better performing than the isoFLOP optimal baseline.

# MatFormer

[2310.07707] MatFormer: Nested Transformer for Elastic Inference

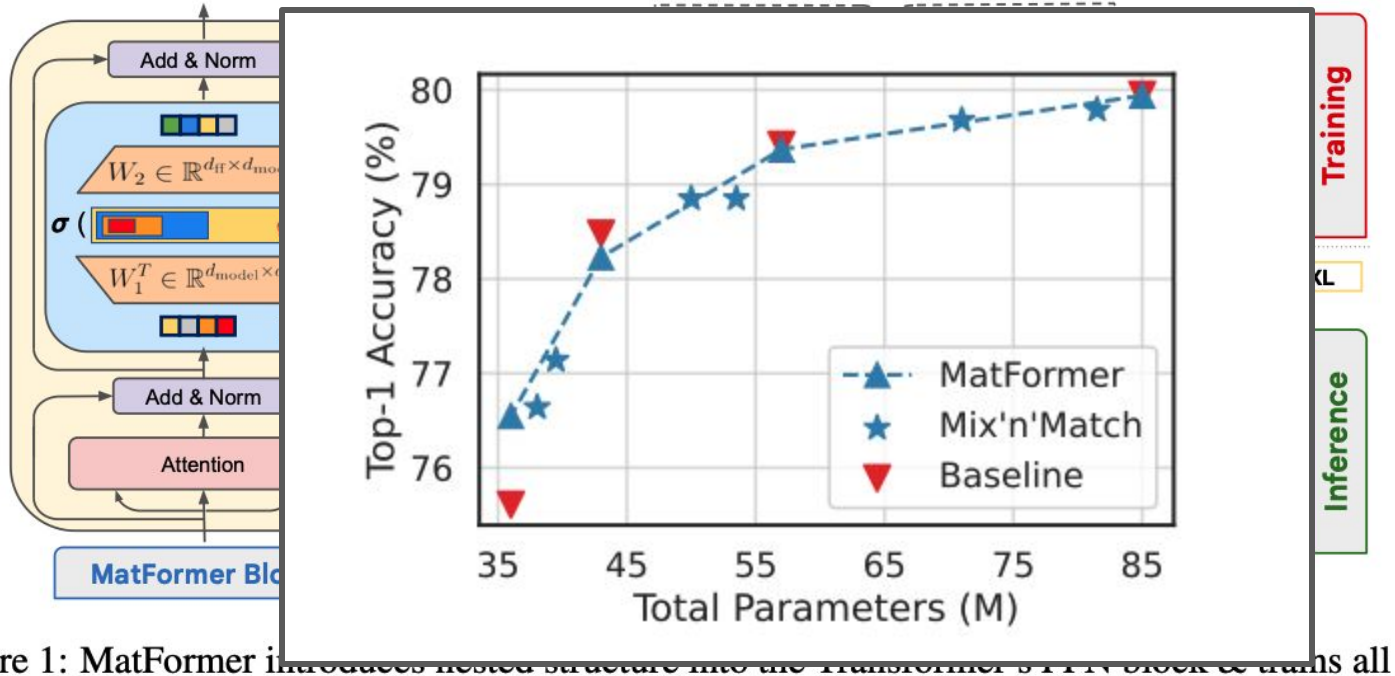



Figure 1: MatFormer introduces nested structure into the Transformer's FFN block & trains all the submodels, enabling free extraction of hundreds of accurate submodels for elastic inference.

# Not all *tokens* are needed

[2112.07658] AdaViT: Adaptive Tokens for Efficient Vision Transformer

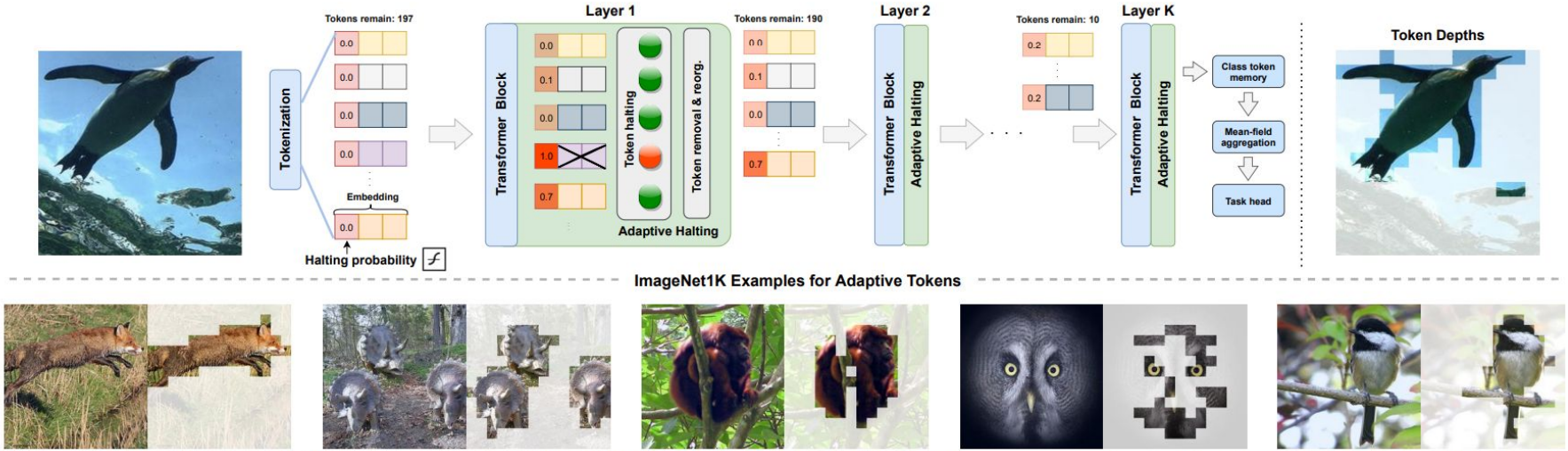


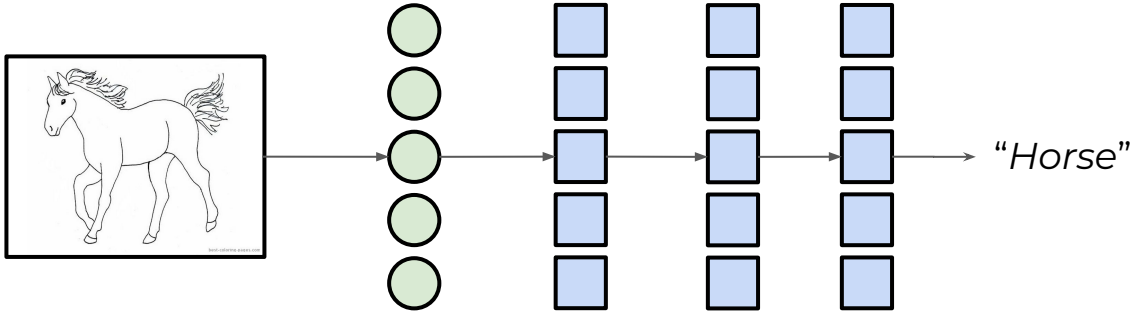

Figure 1. We introduce A-ViT, a method to enable ***adaptive token*** computation for vision transformers. We augment the vision transformer block with adaptive halting module that computes a halting probability per token. The module reuses the parameters of existing blocks and it borrows a single neuron from the last dense layer in each block to compute the halting probability, imposing no extra parameters or computations. A token is discarded once reaching the halting condition. Via adaptively halting tokens, we perform dense compute only on the active tokens deemed informative for the task. As a result, successive blocks in vision transformers gradually receive less tokens, leading to faster inference. Learnt token halting vary across images, yet align ***surprisingly well*** with image semantics (see examples above and more in Fig. 3). This results in immediate, out-of-the-box inference speedup on off-the-shelf computational platform.

# Dynamic neural networks

A **categorization**

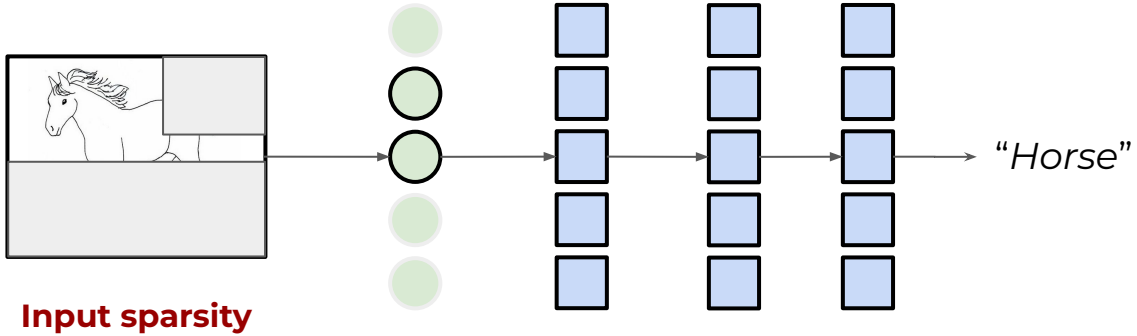
# Dynamic neural networks

Conditional computation in neural networks: Principles and research trends



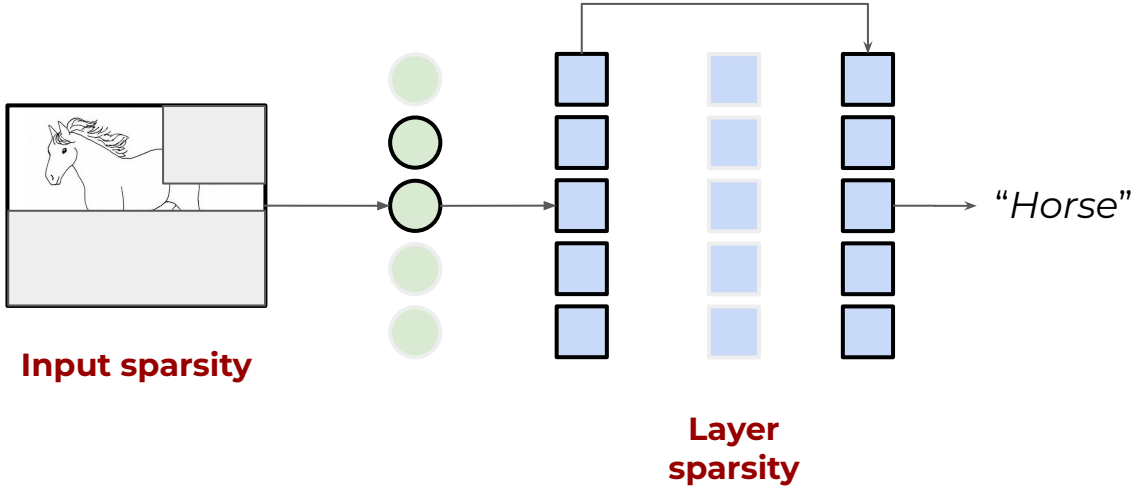
# Dynamic neural networks

[Conditional computation in neural networks: Principles and research trends]



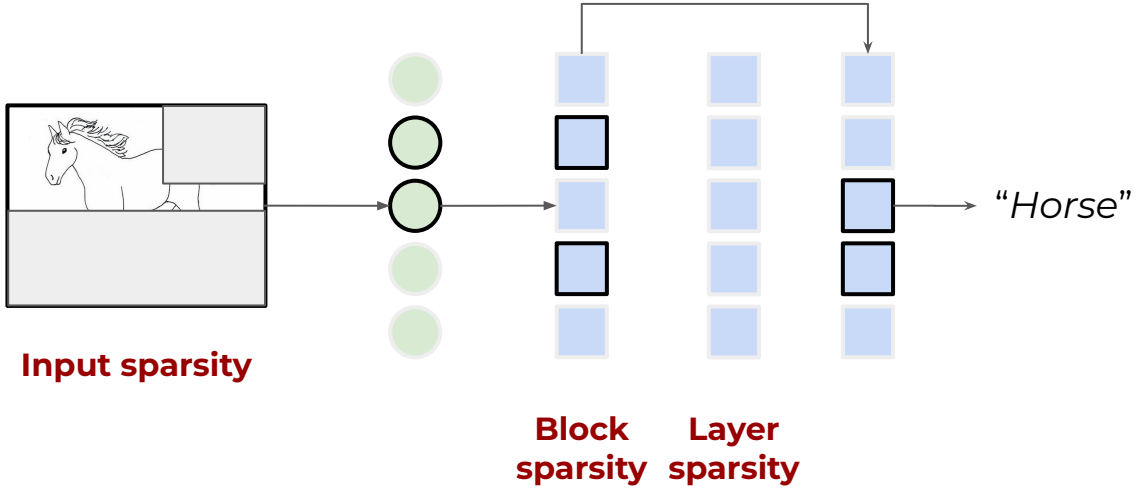
# Dynamic neural networks

[Conditional computation in neural networks: Principles and research trends]

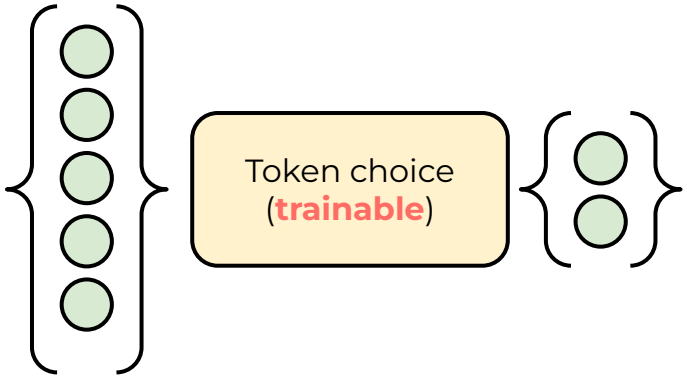


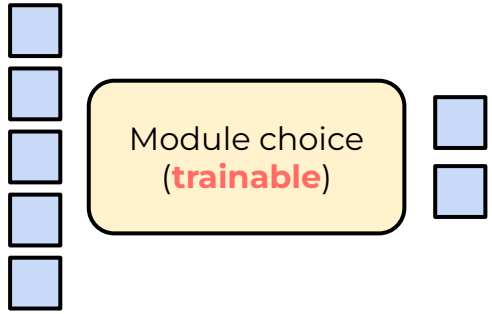
# Dynamic neural networks

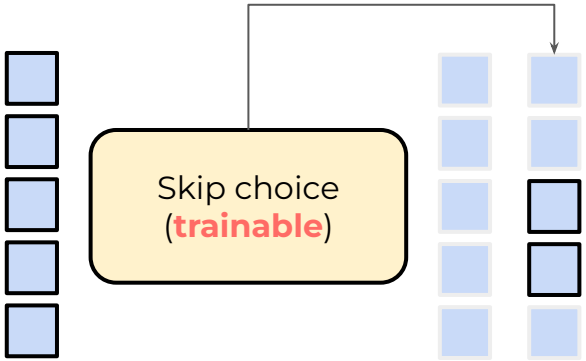
[Conditional computation in neural networks: Principles and research trends]



# Dynamic choice

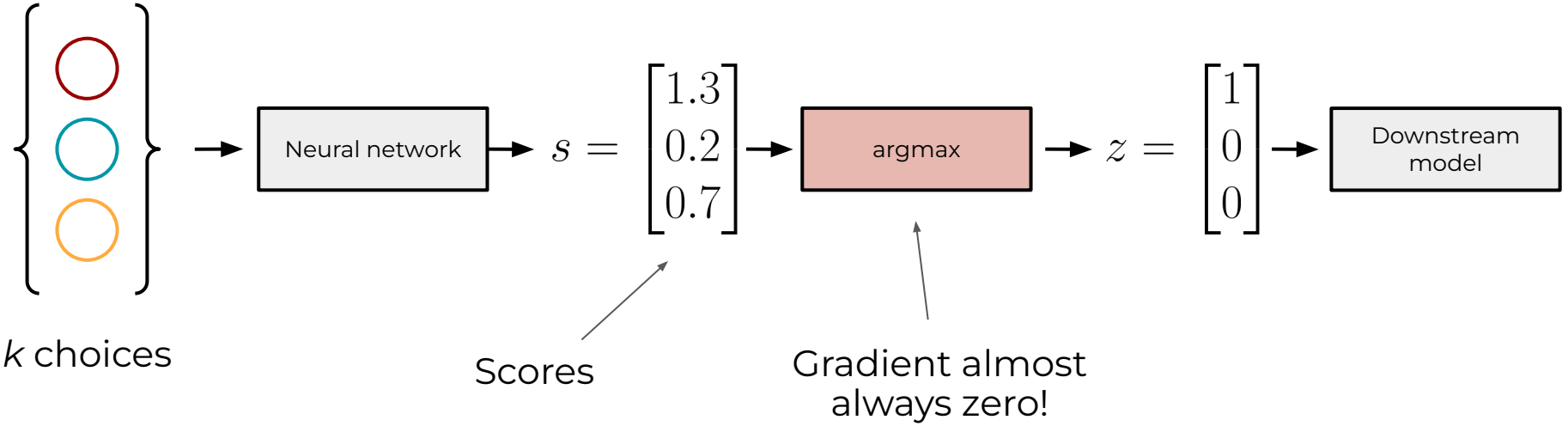


# Formalizing the problem

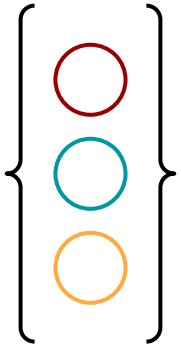
[2301.07473] Discrete Latent Structure in Neural Networks




**Note**: for *k=2*, we can take binary decisions (e.g., keep or discard a patch).

# Softmax approximation

[1611.01144] Categorical Reparameterization with Gumbel-Softmax



*k* choices

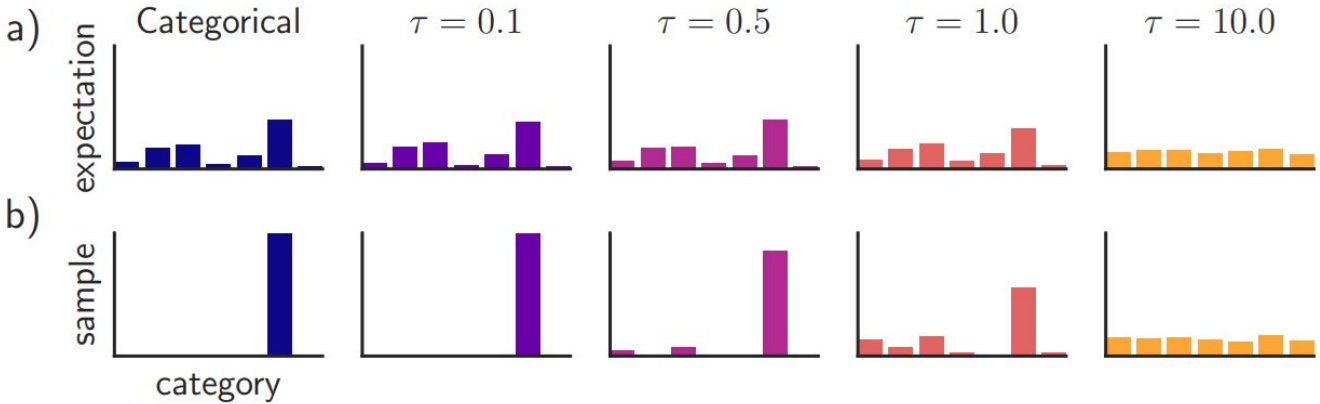
Neural network

$s = \begin{bmatrix} 1.3 \\ 0.2 \\ 0.7 \end{bmatrix}$

softmax

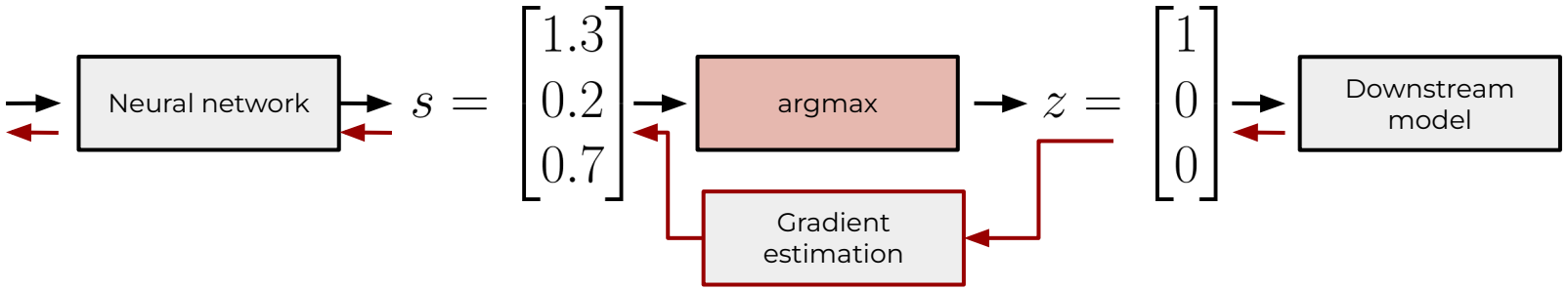
$z = \begin{bmatrix} 0.9 \\ 0.03 \\ 0.07 \end{bmatrix}$

Downstream model



# Gradient estimation

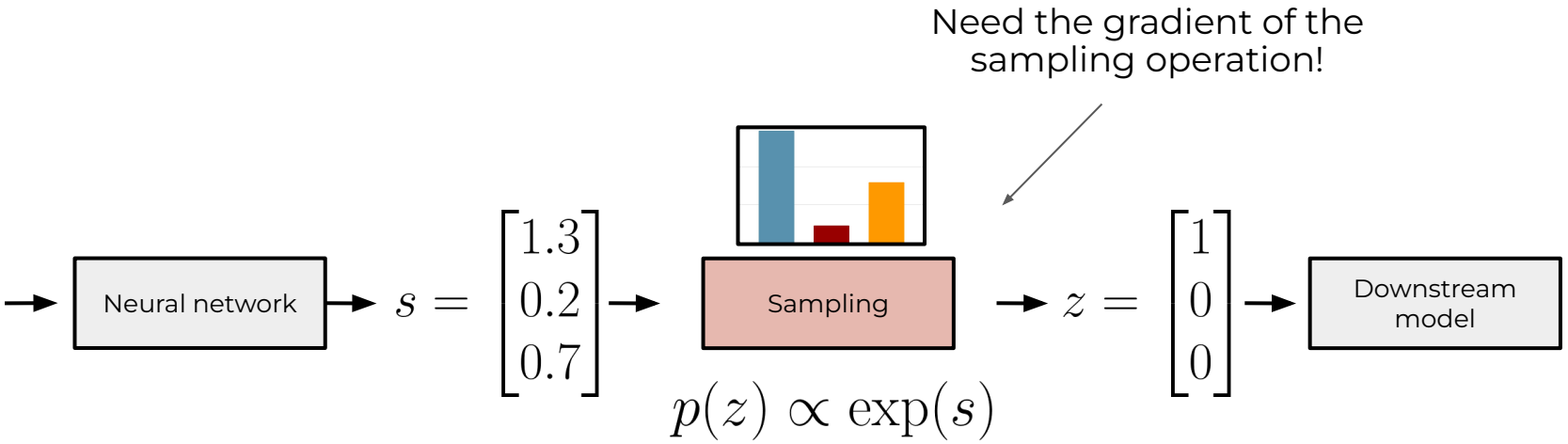
[2301.07473] Discrete Latent Structure in Neural Networks




Straight-through estimation (STE): Gradient estimation =

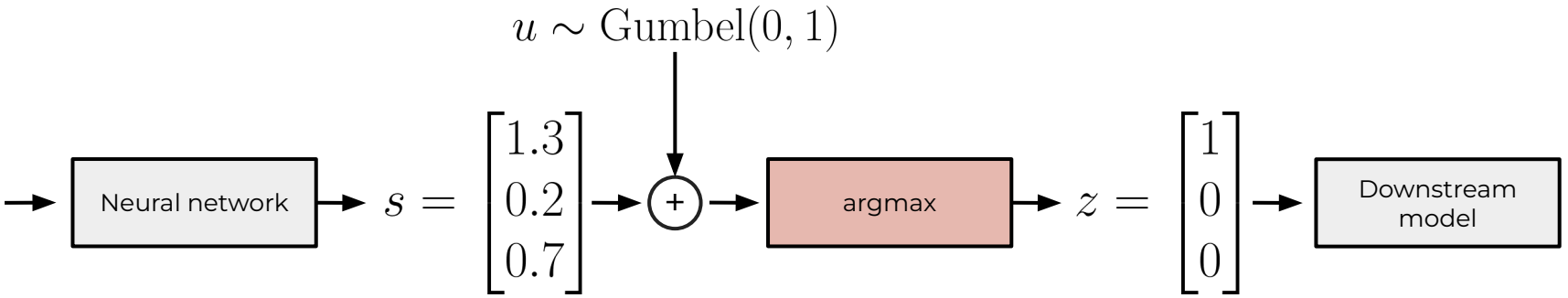
Softmax estimation: Gradient estimation = Softmax (with temperature)

# Improving exploration



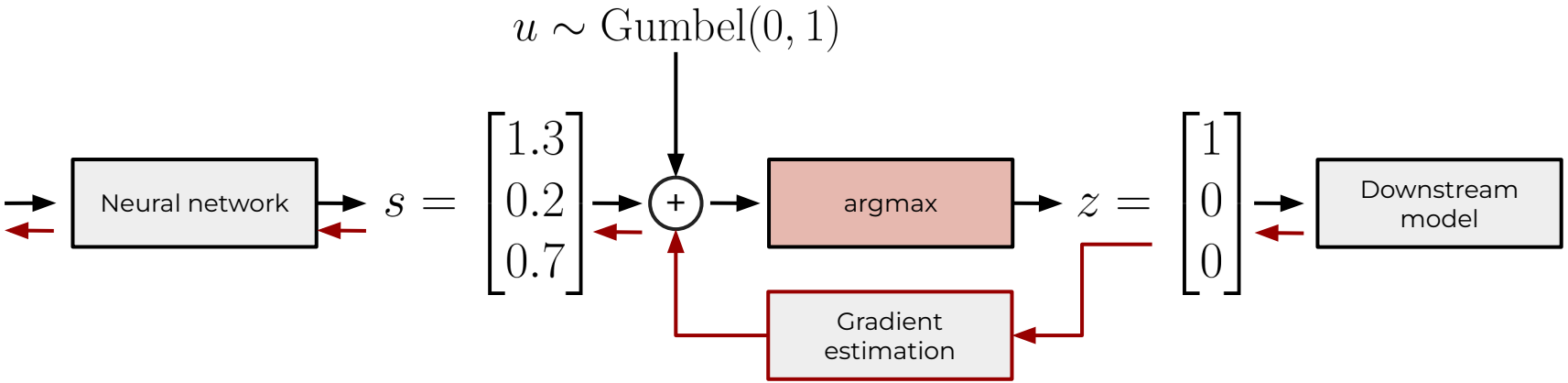

If one score is much higher than the others, sampling or maximizing has similar behaviour, but if the scores are mixed, this can promote exploration.

# Reparameterization (Gumbel-max)




It can be shown that the sample *z* is sampled according to the correct distribution.

# Gumbel-softmax (full diagram)




Luckily, this is all already implemented in PyTorch. :-)

# **Gumbel-softmax** (the equations)

$$z = \arg\max_{z} \; (s+u)^\top z \approx \mathrm{softmax}((s+u)/\tau)$$

This is an example of **perturbed optimization** or **perturb-and-MAP**, which is a very general principle!

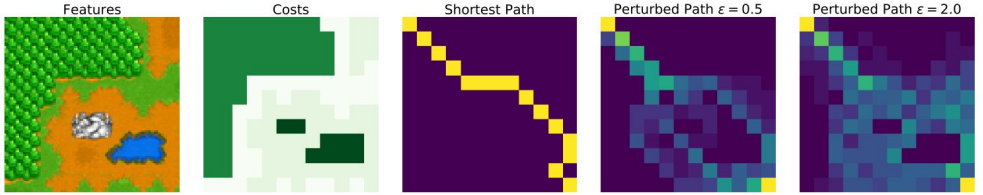



Figure 5: In the shortest path experiment, training features are images. Shortest paths are computed based on terrain costs, hidden to the network. Training responses are shortest paths based on this cost.

[2002.08676] Learning with Differentiable Perturbed Optimizers

# Beyond Gumbel-Softmax

[2301.07473] Discrete Latent Structure in Neural Networks

- Other regularized operators (e.g., entmax).
- Different types of approximations (e.g., score function estimators).
- Algorithms for differentiable sampling beyond categorical variables (e.g., subgraphs).

... This is just an introduction!

# Dynamic neural networks

Making models truly **dynamic**

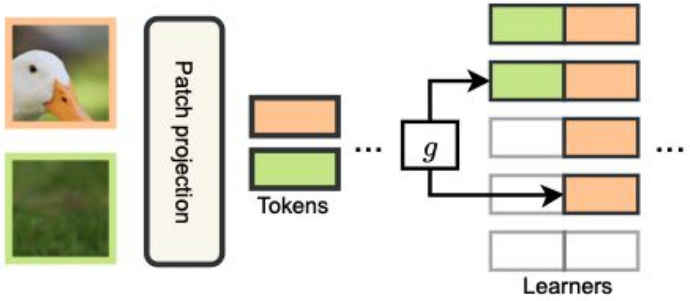
# Overview

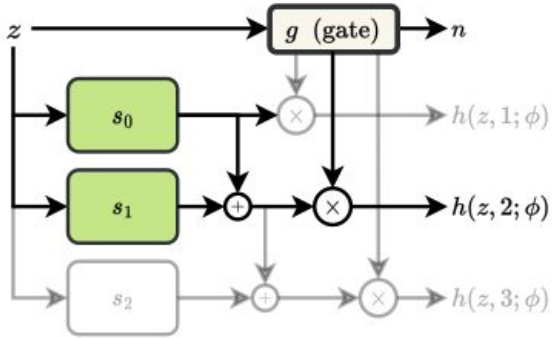
- Fine-tuning existing models is preferable (to leverage the ecosystem).
- Computational cost should be an external constraint if possible (**computation-accuracy** trade-off).
- Hardware latency is important (e.g., customized kernels).
- Lastly, can we also make training cost-efficient?

# Adaptive computation modules (AAAI, 2025)

Adaptive Computation Modules: Granular Conditional Computation for Efficient Inference | AAAI

Fine-tune a pre-trained models to transform some components into *adaptive* blocks:



# Adaptive computation modules (AAAI, 2025)

[Adaptive Computation Modules: Granular Conditional Computation for Efficient Inference | AAAI]

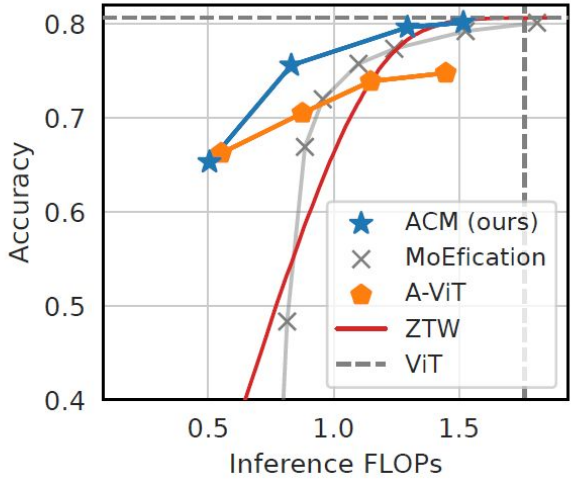



Figure 3: Performance-efficiency trade-offs of different conditional computation methods as measured on the ImageNet-1k dataset. ACM-based ViT-B achieves the Pareto frontier for a wide range of computational budgets.

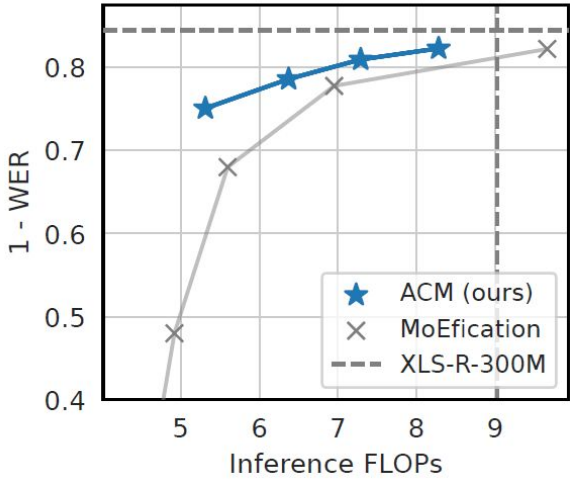



Figure 4: Performance-efficiency trade-offs of different conditional computation methods as measured on the CommonVoice-es dataset. The model's performance is reported in terms of Word Error Rate (WER). ACMs achieve lower WER for every evaluated computational budget.

# Adaptive computation modules

[Adaptive Computation Modules: Granular Conditional Computation for Efficient Inference | AAAI]

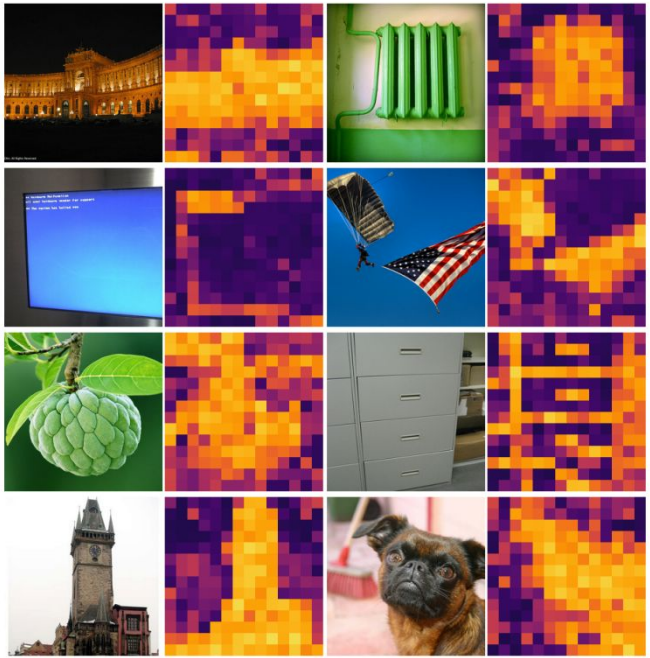


Figure 6: Computational load heatmaps for the model trained with $\beta_{\text{target}} = 0.6$. The model allocates its computational budget to semantically meaningful patches.

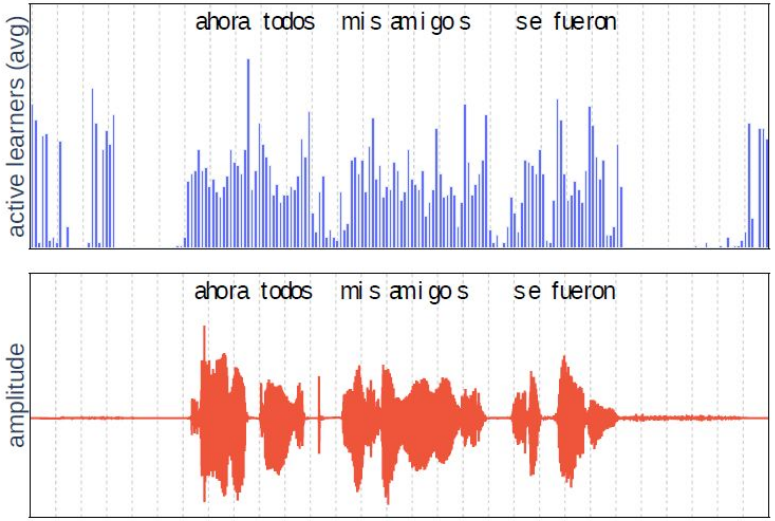
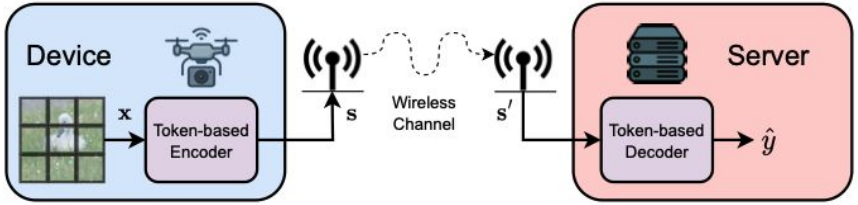
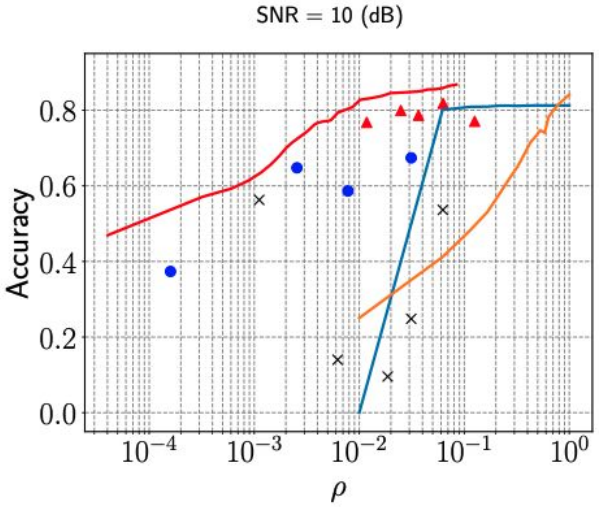


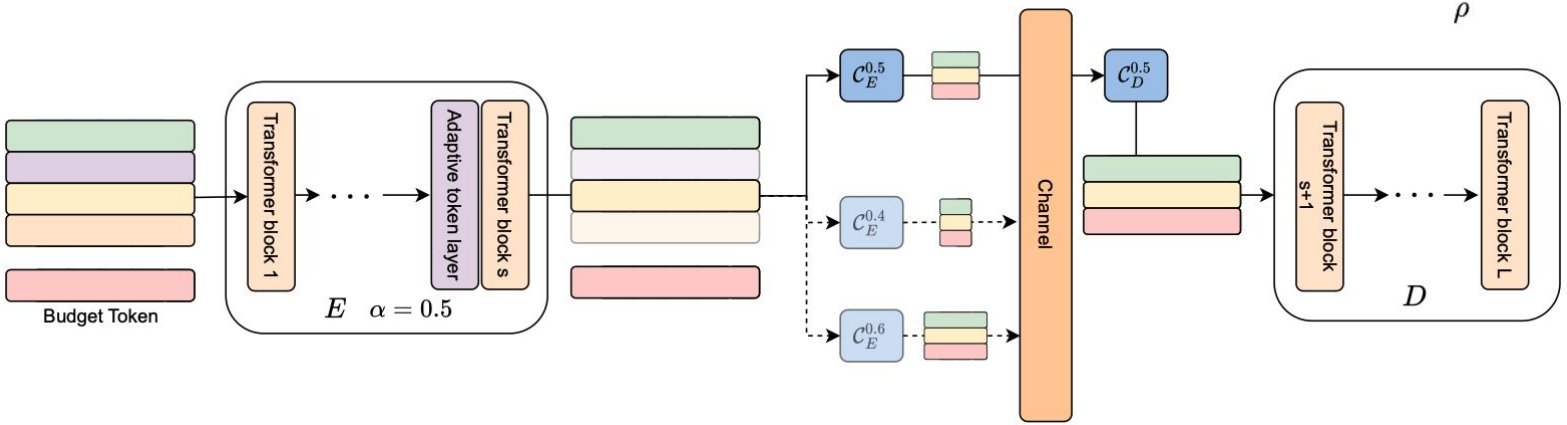

Figure 7: For each input audio token (red waveforms), we show the average number of learners that were activated in the ACMized model for $\beta_{\text{target}} = 0.25$ (blue bars). We can see that this model also learned to allocate its computational budget to important regions of the input.

[2505.17604] Adaptive Semantic Token Communication for Transformer-based Edge Inference



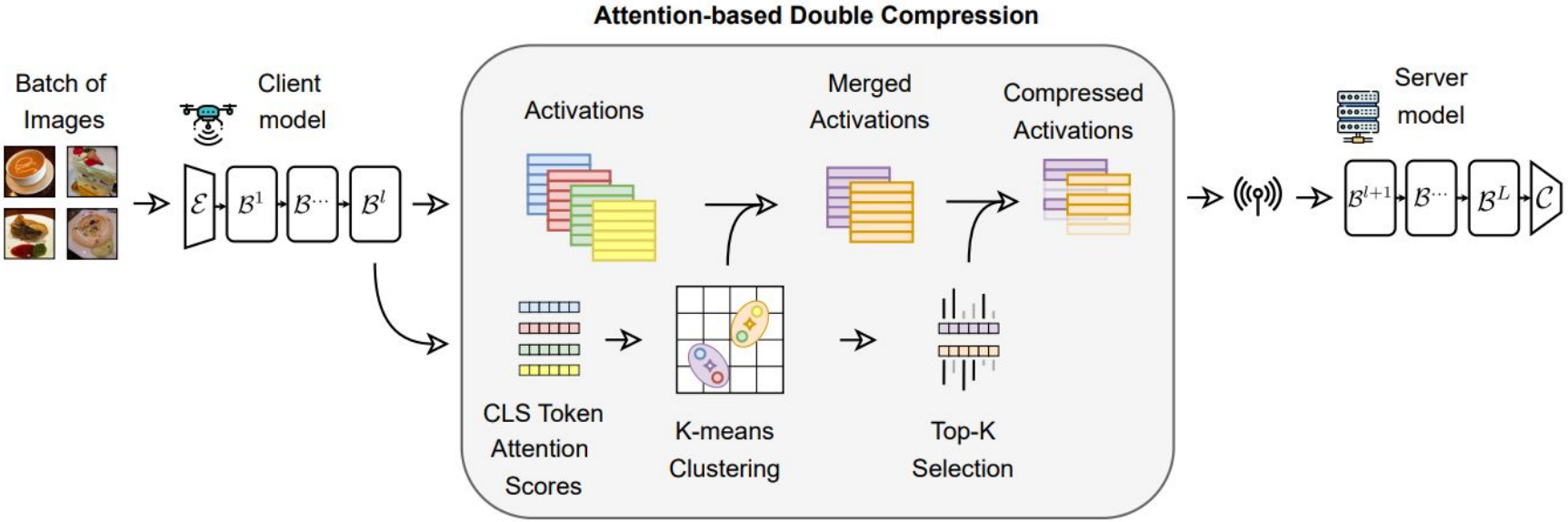

Can flexibly choose *how much data* (tokens) to send on the network (**token-based communication**).



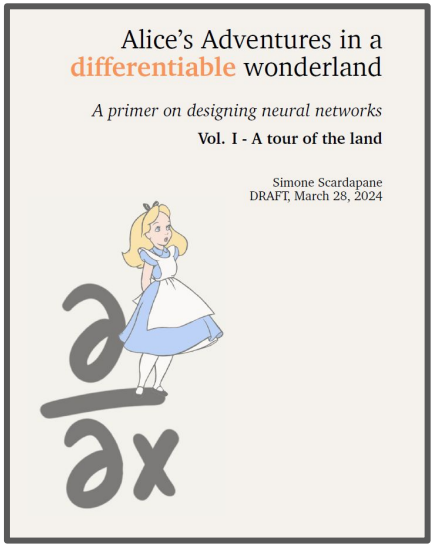



# Training efficiency

[2509.15058] Communication Efficient Split Learning of ViTs with Attention-based Double Compression




Simone Scardapane
Associate Professor, **Sapienza**
Affiliate researcher, **INFN**
*Member*, **CNIT** / **ELLIS**
*Junior fellow*, **Sapienza School of Advanced Study**


Book: Alice's Adventures in a differentiable wonderland - Simone Scardapane

https://www.sscardapane.it/

https://twitter.com/s_scardapane