

The Path to AI

François Fleuret

AlphaFold Changed Science. After 5 Years, It's Still Evolving

WIRED spoke with DeepMind's Pushmeet Kohli about the recent past—and promising future—of the Nobel Prize-winning research project that changed biology and chemistry forever.

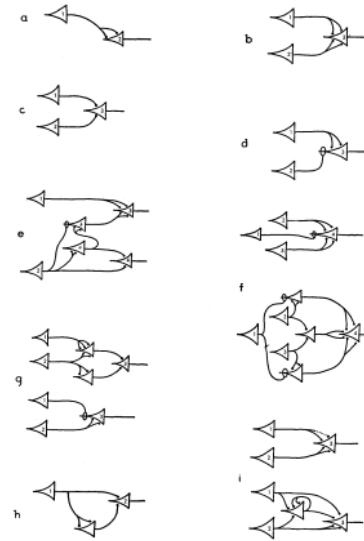
NewScientist

Mathematics

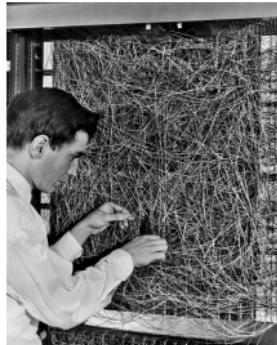
DeepMind and OpenAI claim gold in International Mathematical Olympiad

Two AI models have achieved gold medal standard for the first time in a prestigious competition for young mathematicians – and their developers claim these AIs could soon crack tough scientific problems

Cognition and learning can be engineered



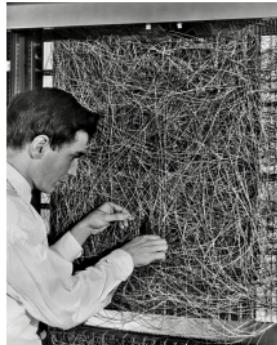
(McCulloch and Pitts, 1943)



Frank Rosenblatt working on the Mark I perceptron (1956)

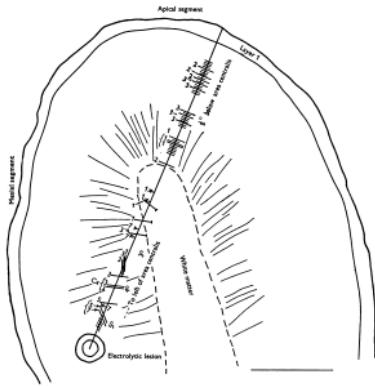
1949 – Donald Hebb proposes the Hebbian Learning principle (Hebb, 1949).

1951 – Marvin Minsky creates the first ANN (Hebbian learning, 40 neurons).



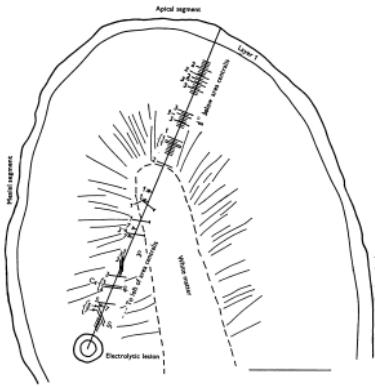
Frank Rosenblatt working on the Mark I perceptron (1956)

- 1949 – Donald Hebb proposes the Hebbian Learning principle (Hebb, 1949).
- 1951 – Marvin Minsky creates the first ANN (Hebbian learning, 40 neurons).
- 1958 – Frank Rosenblatt creates a perceptron to classify 20×20 images.



(Hubel and Wiesel, 1962)

1959 – David H. Hubel and Torsten Wiesel demonstrate orientation selectivity and columnar organization in the cat's visual cortex (Hubel and Wiesel, 1962).

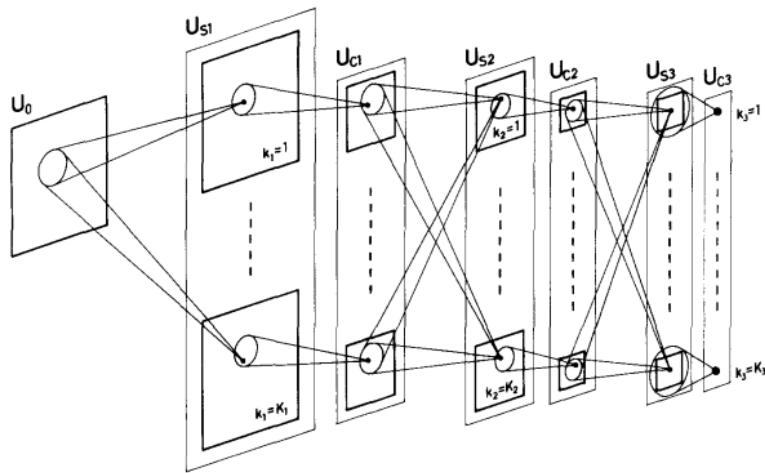


(Hubel and Wiesel, 1962)

1959 – David H. Hubel and Torsten Wiesel demonstrate orientation selectivity and columnar organization in the cat's visual cortex (Hubel and Wiesel, 1962).

1982 – Paul Werbos proposes back-propagation for ANNs (Werbos, 1981).

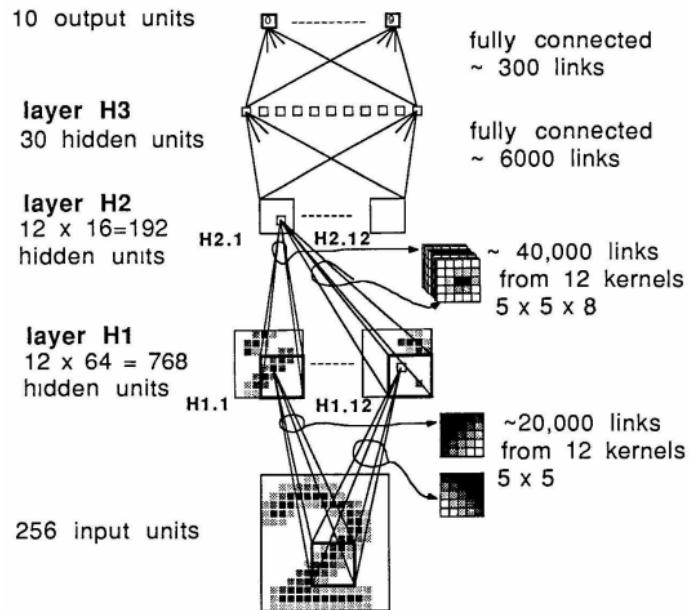
Neocognitron



(Fukushima, 1980)

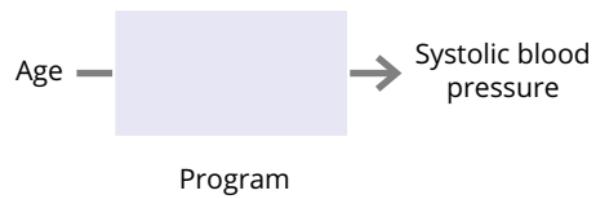
This model follows Hubel and Wiesel's results.

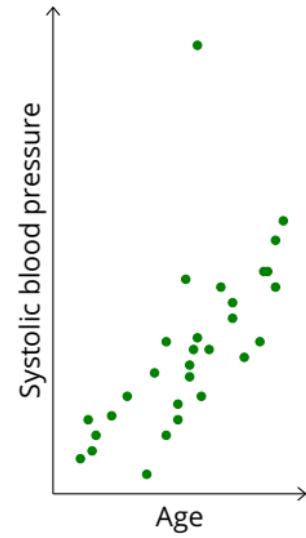
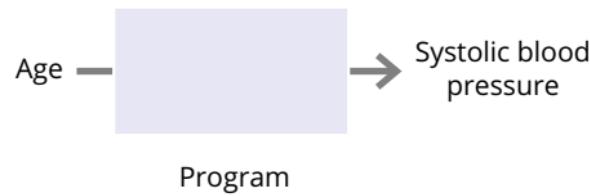
LeNet family

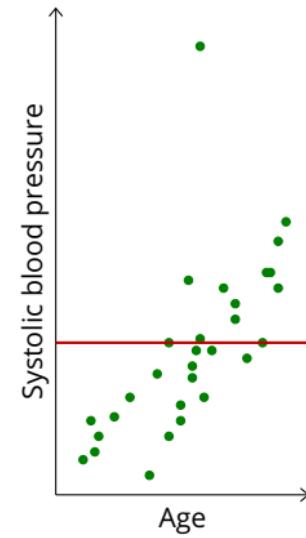
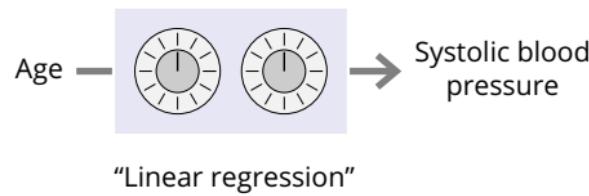


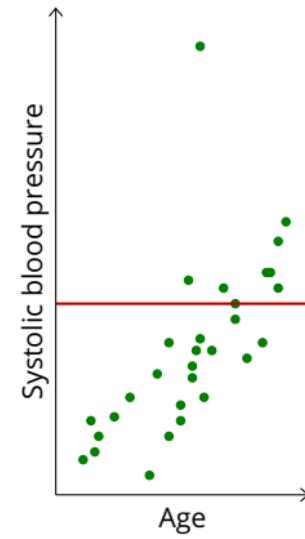
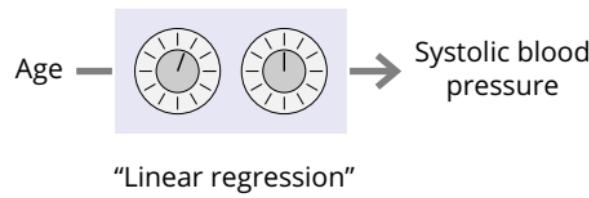
(LeCun et al., 1989)

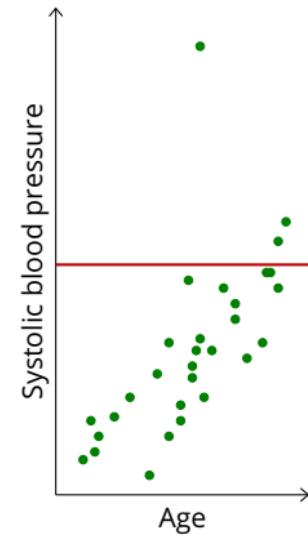
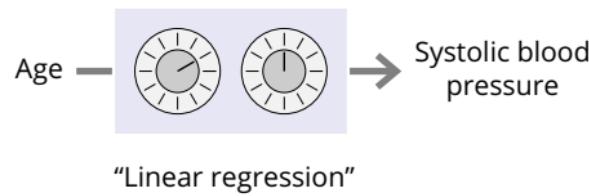
Stupid learning leads to smart models

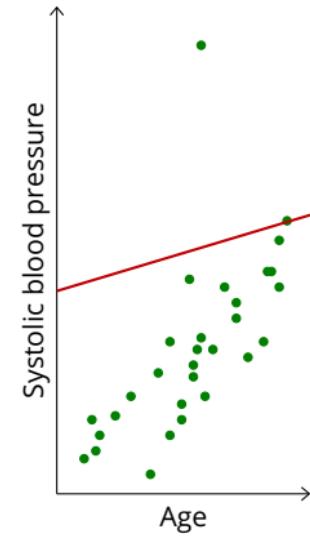
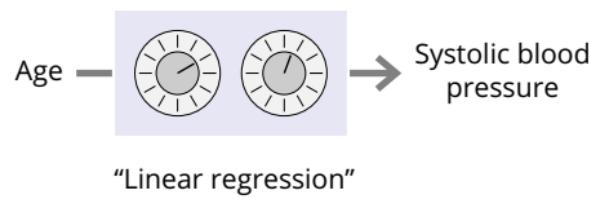


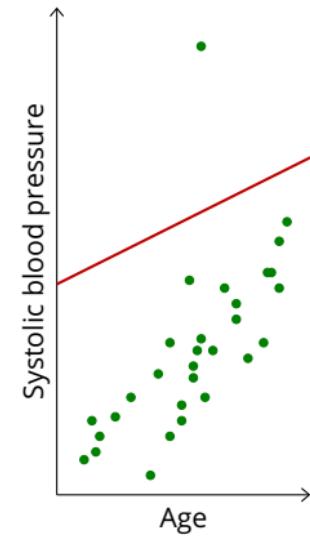
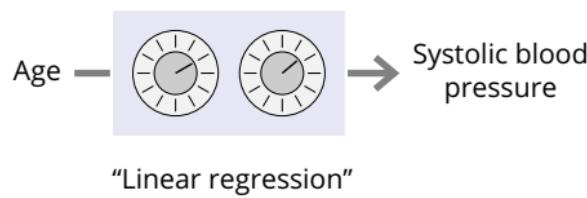


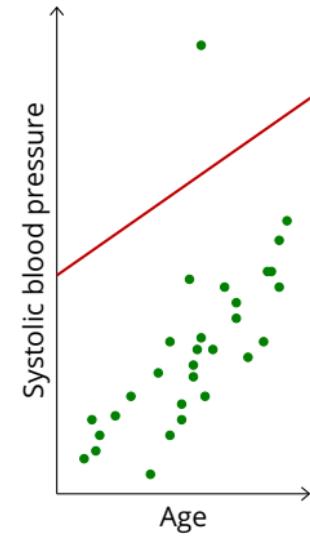
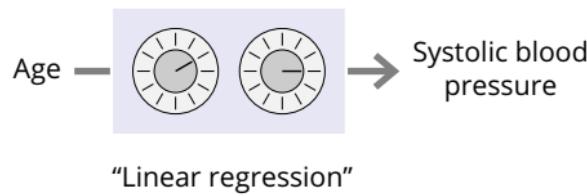


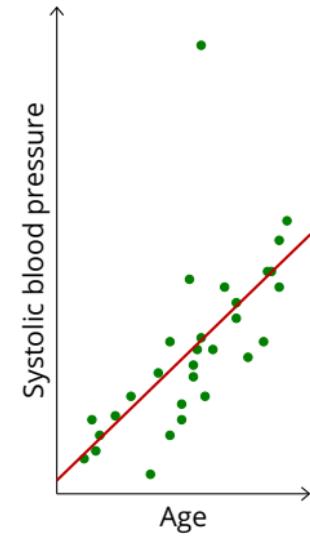
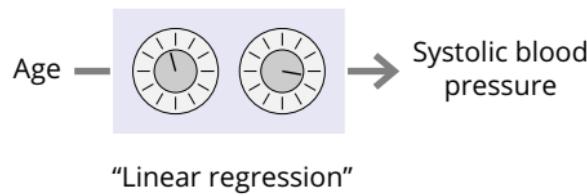












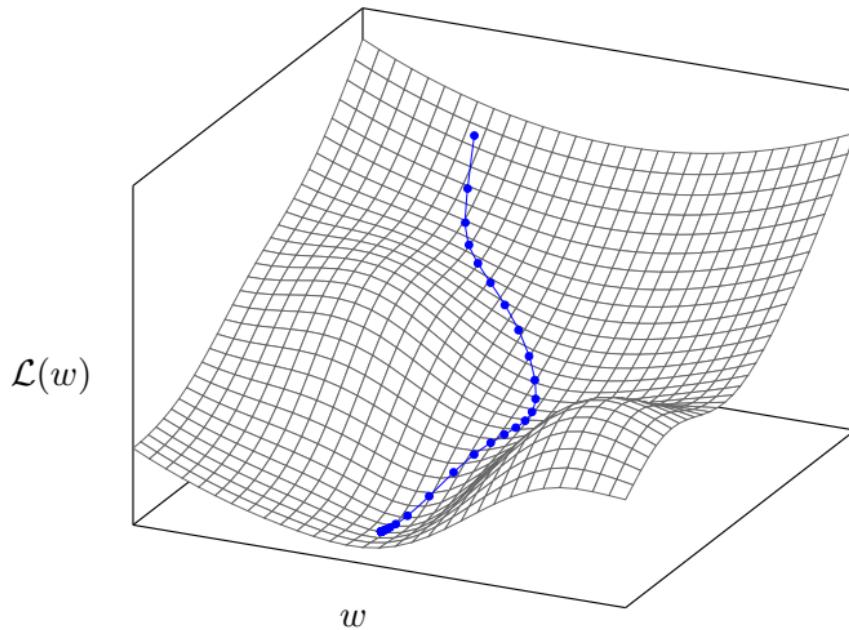
A trainable model is a composition of mappings

$$\phi(\cdot; w) = h(\cdot; w) \circ g(\cdot; w) \circ f(\cdot; w)$$

and training aims at minimizing an error

$$w^* = \underset{w}{\operatorname{argmin}} \mathcal{L}(\phi(\cdot; w); \mathcal{D}).$$

In the general case, computing the proper parameters has to be done numerically through gradient descent.



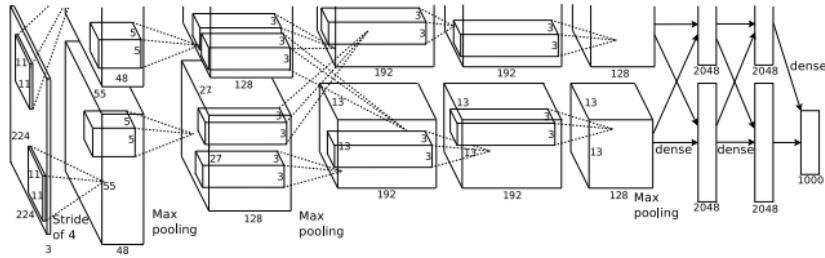
We need to compute

$$\nabla_{|w} \mathcal{L}(\phi(\cdot; w); \mathcal{D})$$

The chain rule leads to a training procedure that goes through the components backward, this is the “backpropagation” of the gradient.

```
Model {  
    model = nn.Sequential(  
        nn.Conv2d( 1, 32, 5), nn.MaxPool2d(3), nn.ReLU(),  
        nn.Conv2d(32, 64, 5), nn.MaxPool2d(2), nn.ReLU(),  
        nn.Flatten(),  
        nn.Linear(256, 200), nn.ReLU(),  
        nn.Linear(200, 10)  
    )  
  
    criterion = nn.CrossEntropyLoss()  
  
    optimizer = torch.optim.SGD(model.parameters(), lr = 1e-2)  
  
    for e in range(nb_epochs):  
        for input, target in data_loader_iterator(train_loader):  
            output = model(input)  
            loss = criterion(output, target)  
            optimizer.zero_grad()  
            loss.backward()  
            optimizer.step()  
}
```

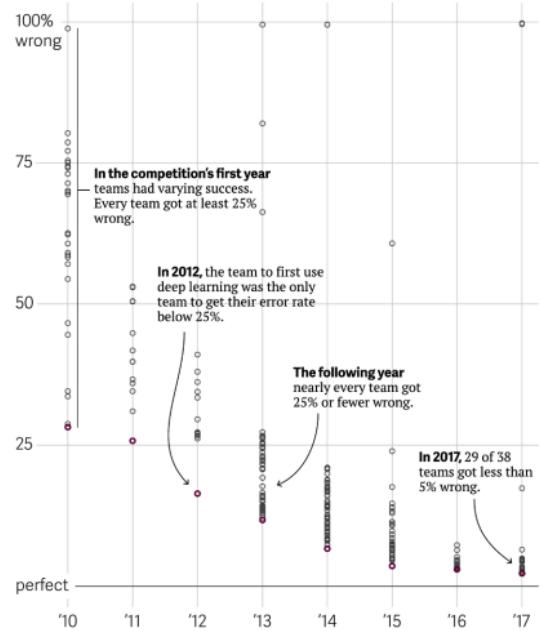
Quantity has a quality of its own



Very large models + GPUs (AlexNet, 2012)



ImageNet



(Gershgorin, 2017)

The last decade of progress in AI corresponds to a vast increase of the “training sets” sizes. The most successful deployed methods rely on human-labeled data.

Data-set	Year	Nb. images	Size
MNIST	1998	60K	12Mb
Caltech 256	2007	30K	1.2Gb
ImageNet	2012	1.2M	150Gb
LAION-5B	2022	5.85B	240Tb

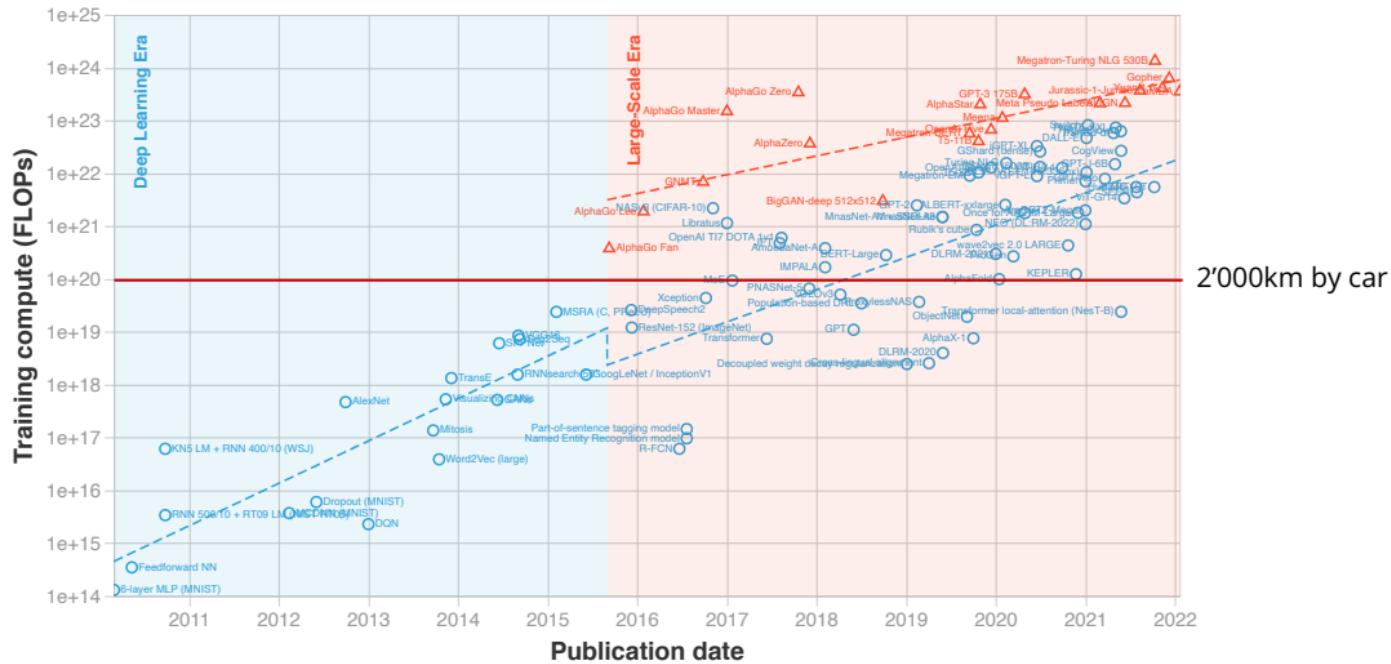
Data-set	Year	Nb. books (250p)	Size*
SST2	2013	40K	20Mb
WMT-18	2018	14M	7Gb
The Pile	2020	1.6B	825Gb
OSCAR	2020	12B	6Tb

(* all the text of Wikipedia is 45Gb)

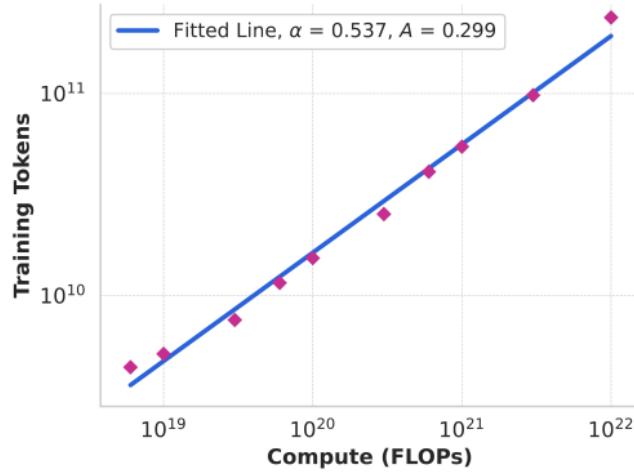
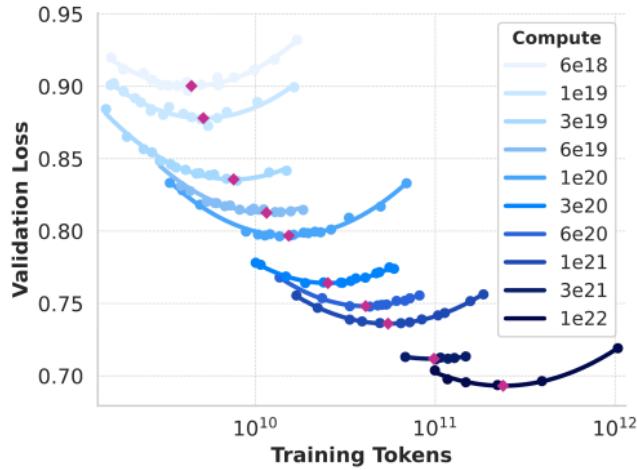
A \$1'500 GPU executes $\simeq 35'000$ billions operations per second.

Training compute (FLOPs) of milestone Machine Learning systems over time

$\theta = 102^\circ$

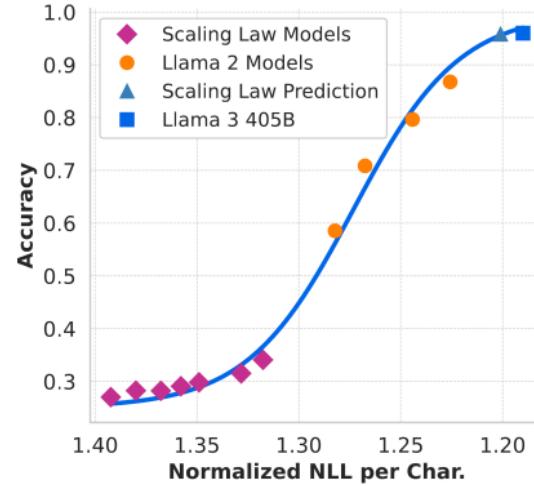
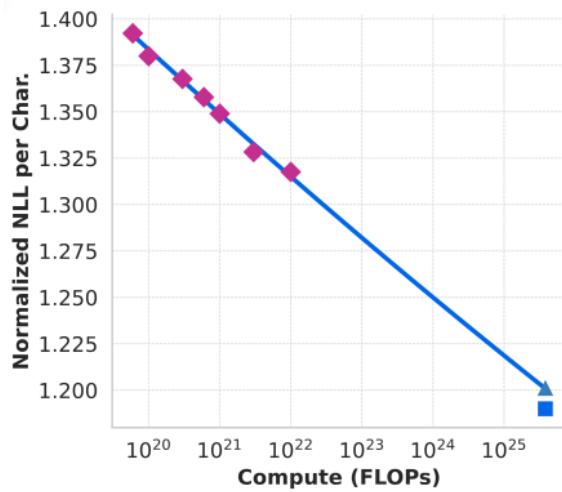


The performance of large models when the amount of compute and data set size increase follows very regular scaling laws.



(Dubey et al., 2024)

The performance of large models when the amount of compute and data set size increase follows very regular scaling laws.



(Dubey et al., 2024)

Optimizability trumps optimality

Using deeper architectures has been key in improving performance in many applications.

model	top-1 err.	top-5 err.
VGG-16 [41]	28.07	9.33
GoogLeNet [44]	-	9.15
PReLU-net [13]	24.27	7.38
plain-34	28.54	10.02
ResNet-34 A	25.03	7.76
ResNet-34 B	24.52	7.46
ResNet-34 C	24.19	7.40
ResNet-50	22.85	6.71
ResNet-101	21.75	6.05
ResNet-152	21.43	5.71

Table 3. Error rates (%), **10-crop** testing) on ImageNet validation.
VGG-16 is based on our test. ResNet-50/101/152 are of option B
that only uses projections for increasing dimensions.

(He et al., 2015)

Using deeper architectures has been key in improving performance in many applications.

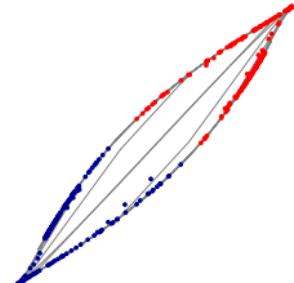
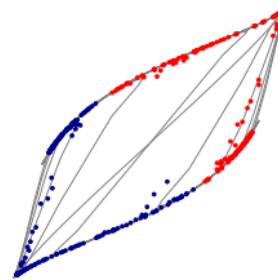
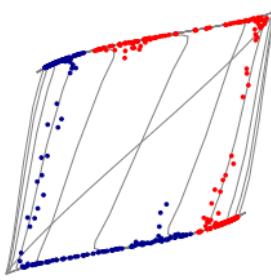
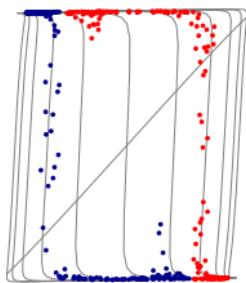
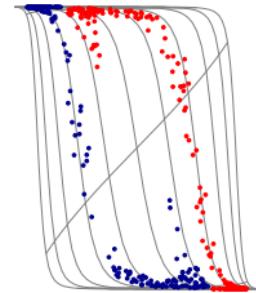
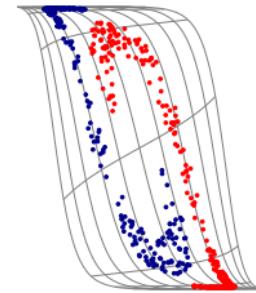
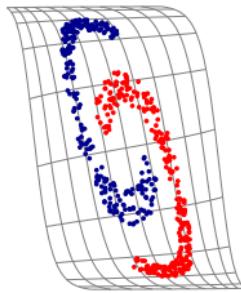
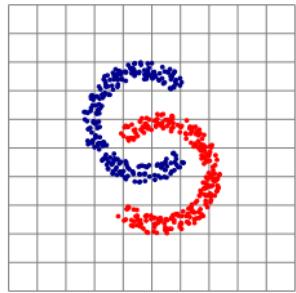
model	top-1 err.	top-5 err.
VGG-16 [41]	28.07	9.33
GoogLeNet [44]	-	9.15
PRelu-net [13]	24.27	7.38
plain-34	28.54	10.02
ResNet-34 A	25.03	7.76
ResNet-34 B	24.52	7.46
ResNet-34 C	24.19	7.40
ResNet-50	22.85	6.71
ResNet-101	21.75	6.05
ResNet-152	21.43	5.71

Table 3. Error rates (%), **10-crop** testing) on ImageNet validation.
VGG-16 is based on our test. ResNet-50/101/152 are of option B
that only uses projections for increasing dimensions.

(He et al., 2015)

“We did not depart from the classical ConvNet architecture of LeCun et al. (1989), but improved it by substantially increasing the depth. ”

(Simonyan and Zisserman, 2014)



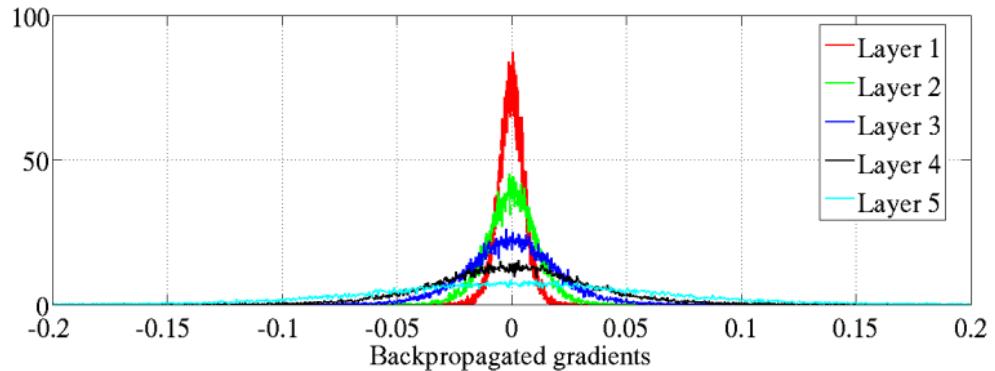
Considering ReLU MLPs with a single input/output, there exists a network f with D^* layers, and $2D^*$ internal units, such that, with g a single hidden layer network

$$\|f - g\|_1 \geq 1 - 2 \frac{W^{(1)}}{2^{D^*}}.$$

To approximate f properly, the width $W^{(1)}$ of g 's hidden layer has to increase exponentially with f 's depth D^* .

(Telgarsky, 2015, 2016).

The gradient “vanishes” exponentially with the depth if the ws are ill-conditioned or the activations are in the saturating domain of σ .



(Glorot and Bengio, 2010)

Four key elements allow the design of extremely deep architectures:



Four key elements allow the design of extremely deep architectures:



- + Smart initialization to normalize all activations.

Four key elements allow the design of extremely deep architectures:



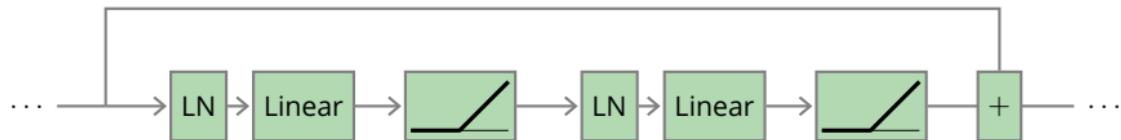
- + Smart initialization to normalize all activations.
- + Better non-linearities to prevent gradient vanishing.

Four key elements allow the design of extremely deep architectures:



- + Smart initialization to normalize all activations.
- + Better non-linearities to prevent gradient vanishing.
- + Normalization layers to keep activation moments.

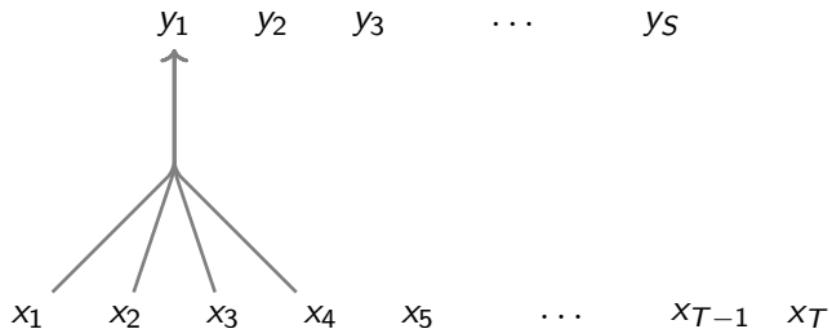
Four key elements allow the design of extremely deep architectures:



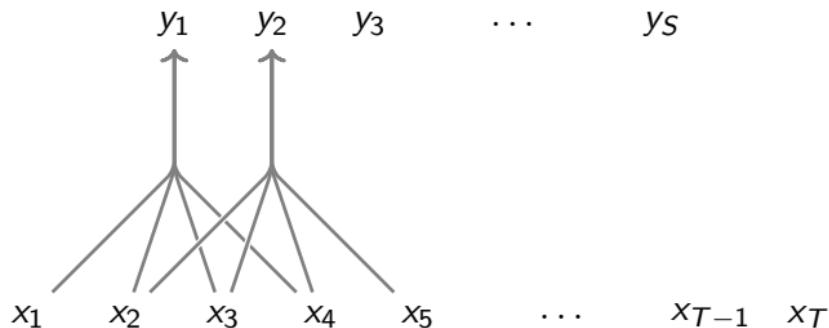
- + Smart initialization to normalize all activations.
- + Better non-linearities to prevent gradient vanishing.
- + Normalization layers to keep activation moments.
- + Skip connections, to add the computed component to the current activations.

Attention is all you need

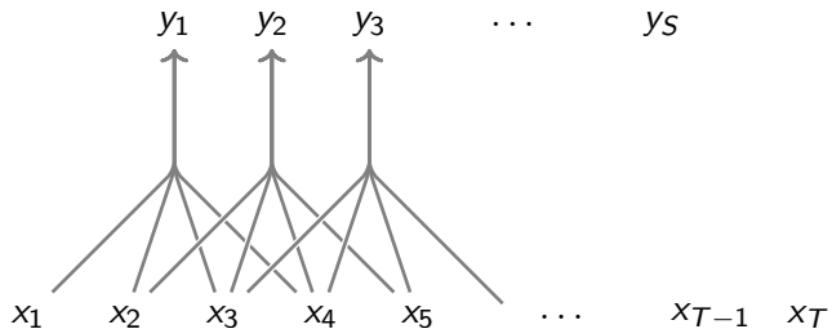
A convolution applies the **same operation** everywhere in the signal.



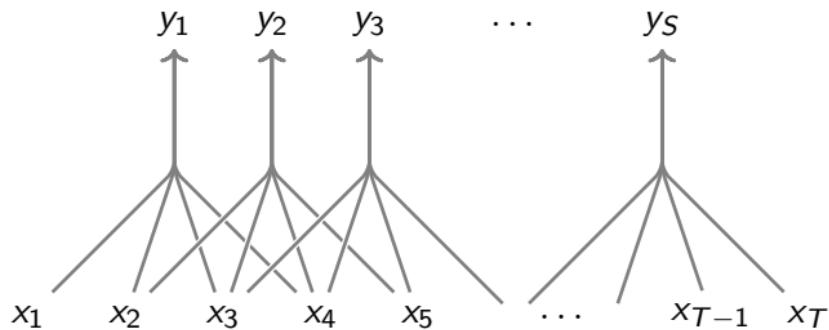
A convolution applies the **same operation** everywhere in the signal.



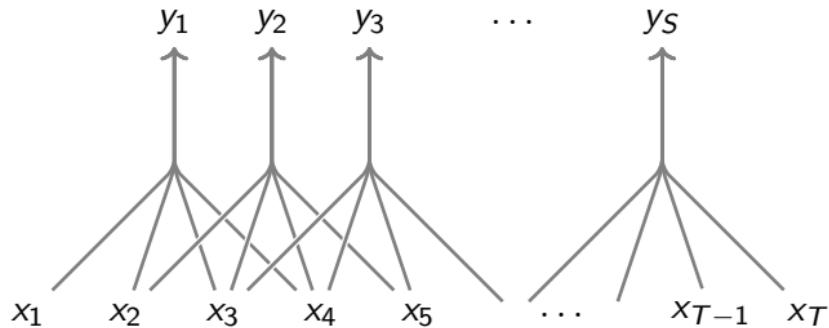
A convolution applies the **same operation** everywhere in the signal.



A convolution applies the **same operation** everywhere in the signal.



A convolution applies the **same operation** everywhere in the signal.



Such mechanisms are very efficient when

- + the signal is stationary, and
- + local structures are very informative.

Some tasks involve more than local structures, e.g. translation:

"An apple that had been on the tree in the garden for weeks had finally been **picked up.**"

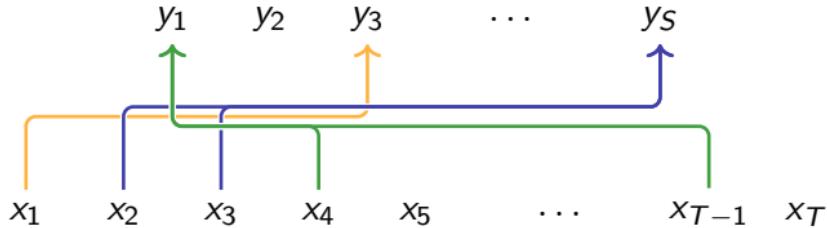
"Une pomme qui était sur l'arbre du jardin depuis des semaines avait finalement été **ramassée.**"

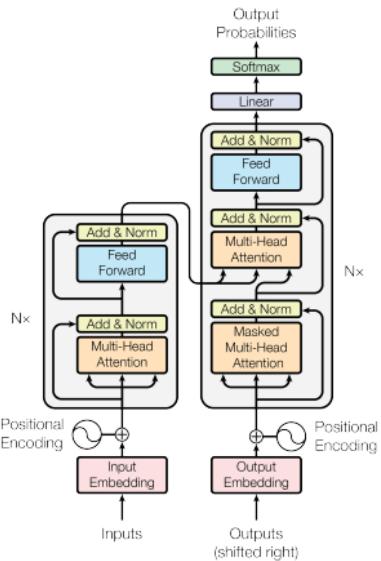
Some tasks involve more than local structures, e.g. translation:

"An apple that had been on the tree in the garden for weeks had finally been **picked up.**"

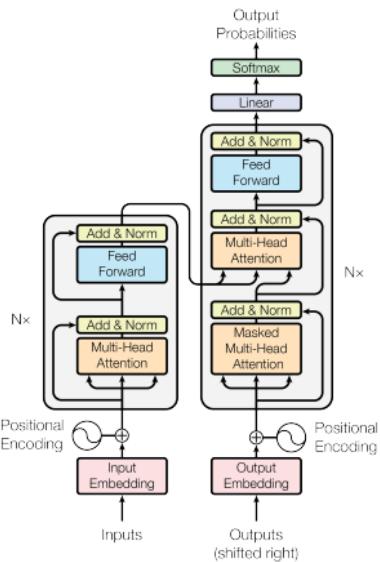
"Une pomme qui était sur l'arbre du jardin depuis des semaines avait finalement été **ramassée.**"

It has motivated **attention-based processing** to transport information from parts of the signal to other parts dynamically identified.

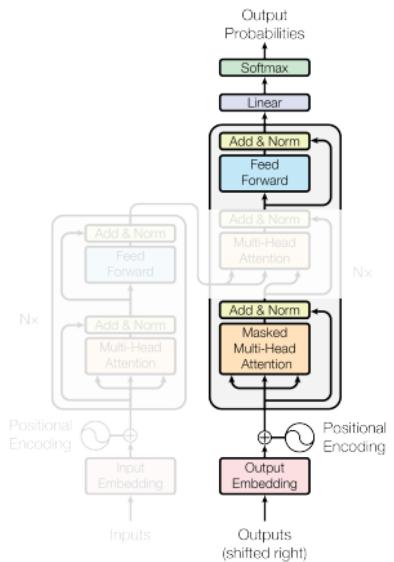




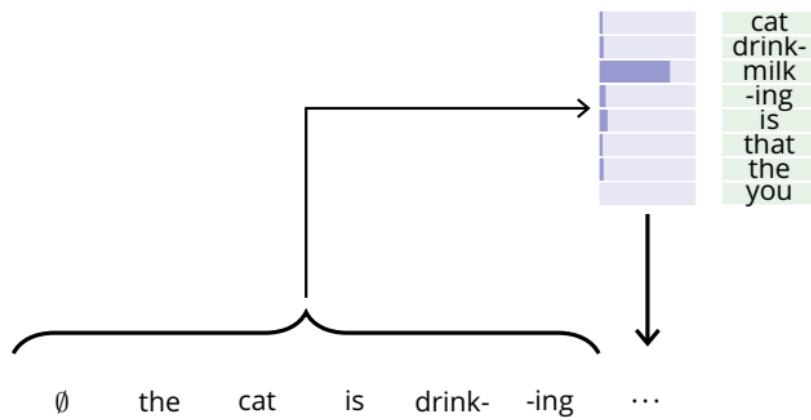
Transformer
(Vaswani et al., 2017)

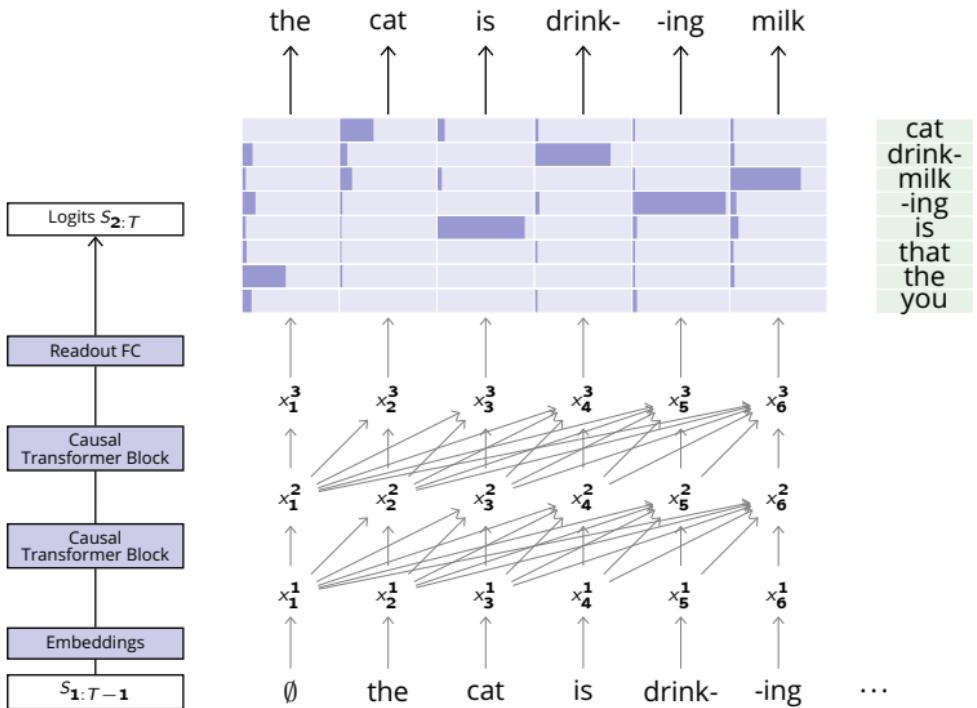


Transformer
(Vaswani et al., 2017)



Decoder-only Transformer
(Radford et al., 2018)





Autoregression is smarter than expected

System Prompt
(human-written)

In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

Model Completion
(machine-written, 10 tries)

The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science.

Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved.

Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow.

Pérez and the others then ventured further into the valley. "By the time we reached the top of one peak, the water looked blue, with some crystals on top," said Pérez.

Pérez and his friends were astonished to see the unicorn herd. These creatures could be seen from the air without having to move too much to see them - they were so close they could touch their horns.

Such a “base model”, trained only to generate text, can already be put to use for instance for classification.

I: water boils at 100 degrees, 0: physics. I: the square root of two is irrational, 0: mathematics. I: the set of prime numbers is infinite, 0: mathematics. I: gravity is proportional to the mass, 0: **physics**.

Such a “base model”, trained only to generate text, can already be put to use for instance for classification.

I: water boils at 100 degrees, 0: physics. I: the square root of two is irrational, 0: mathematics. I: the set of prime numbers is infinite, 0: mathematics. I: gravity is proportional to the mass, 0: **physics**.

Such a “base model”, trained only to generate text, can already be put to use for instance for classification.

I: water boils at 100 degrees, 0: physics. I: the square root of two is irrational, 0: mathematics. I: the set of prime numbers is infinite, 0: mathematics. I: gravity is proportional to the mass, 0: **physics**.

I: water boils at 100 degrees, 0: physics. I: the square root of two is irrational, 0: mathematics. I: the set of prime numbers is infinite, 0: mathematics. I: squares are rectangles, 0: **mathematics**.

Such a “base model”, trained only to generate text, can already be put to use for instance for classification.

I: water boils at 100 degrees, 0: physics. I: the square root of two is irrational, 0: mathematics. I: the set of prime numbers is infinite, 0: mathematics. I: gravity is proportional to the mass, 0: **physics**.

I: water boils at 100 degrees, 0: physics. I: the square root of two is irrational, 0: mathematics. I: the set of prime numbers is infinite, 0: mathematics. I: squares are rectangles, 0: **mathematics**.

Such a “base model”, trained only to generate text, can already be put to use for instance for classification.

I: water boils at 100 degrees, 0: physics. I: the square root of two is irrational, 0: mathematics. I: the set of prime numbers is infinite, 0: mathematics. I: gravity is proportional to the mass, 0: **physics**.

I: water boils at 100 degrees, 0: physics. I: the square root of two is irrational, 0: mathematics. I: the set of prime numbers is infinite, 0: mathematics. I: squares are rectangles, 0: **mathematics**.

<I: I love apples, 0: positive. I: music is my passion, 0: positive. I: my job is boring, 0: negative. I: frozen pizzas are awesome, 0: **positive**.

Such a “base model”, trained only to generate text, can already be put to use for instance for classification.

I: water boils at 100 degrees, 0: physics. I: the square root of two is irrational, 0: mathematics. I: the set of prime numbers is infinite, 0: mathematics. I: gravity is proportional to the mass, 0: **physics**.

I: water boils at 100 degrees, 0: physics. I: the square root of two is irrational, 0: mathematics. I: the set of prime numbers is infinite, 0: mathematics. I: squares are rectangles, 0: **mathematics**.

<I: I love apples, 0: positive. I: music is my passion, 0: positive. I: my job is boring, 0: negative. I: frozen pizzas are awesome, 0: **positive**.

Such a “base model”, trained only to generate text, can already be put to use for instance for classification.

I: water boils at 100 degrees, 0: physics. I: the square root of two is irrational, 0: mathematics. I: the set of prime numbers is infinite, 0: mathematics. I: gravity is proportional to the mass, 0: **physics**.

I: water boils at 100 degrees, 0: physics. I: the square root of two is irrational, 0: mathematics. I: the set of prime numbers is infinite, 0: mathematics. I: squares are rectangles, 0: **mathematics**.

<I: I love apples, 0: positive. I: music is my passion, 0: positive. I: my job is boring, 0: negative. I: frozen pizzas are awesome, 0: **positive**.

I: I love apples, 0: positive. I: music is my passion, 0: positive. I: my job is boring, 0: negative. I: frozen pizzas taste like cardboard, 0: **negative**.

Such a “base model”, trained only to generate text, can already be put to use for instance for classification.

I: water boils at 100 degrees, 0: physics. I: the square root of two is irrational, 0: mathematics. I: the set of prime numbers is infinite, 0: mathematics. I: gravity is proportional to the mass, 0: **physics**.

I: water boils at 100 degrees, 0: physics. I: the square root of two is irrational, 0: mathematics. I: the set of prime numbers is infinite, 0: mathematics. I: squares are rectangles, 0: **mathematics**.

<I: I love apples, 0: positive. I: music is my passion, 0: positive. I: my job is boring, 0: negative. I: frozen pizzas are awesome, 0: **positive**.

I: I love apples, 0: positive. I: music is my passion, 0: positive. I: my job is boring, 0: negative. I: frozen pizzas taste like cardboard, 0: **negative**.

For e.g. ChatGPT, base training is followed by “Reinforcement Learning from Human Feedback” (RLHF) to create an assistant.

Submit Skip Page 3 / 11 Total time: 05:39

Instruction

Summarize the following news article:

=====
{article}
=====

Output A

summary1

Rating (1 = worst, 7 = best)

1 2 3 4 5 6 7

Fails to follow the correct instruction / task ? Yes No

Inappropriate for customer assistant ? Yes No

Contains sexual content Yes No

Contains violent content Yes No

Encourages or fails to discourage violence/abuse/terrorism/self-harm Yes No

Denigrates a protected class Yes No

Gives harmful advice ? Yes No

Expresses moral judgment Yes No

Notes

(Optional) notes

Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

Explain the moon landing to a 6 year old

A labeler demonstrates the desired output behavior.



Some people went to the moon...

This data is used to fine-tune GPT-3 with supervised learning.



Step 2

Step 2

Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

Explain the moon landing to a 6 year old

A B
Explain gravity... Explain way...
C D
Moon is natural satellite of... People went to the moon...

A labeler ranks the outputs from best to worst.

D > C > A = B

This data is used to train our reward model.

RM
D > C > A = B

Step 3

Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

Write a story about frogs

The policy generates an output.



Once upon a time...

The reward model calculates a reward for the output.



The reward is used to update the policy using PPO.

r_k

Learning to reason

It was then discovered that prompting the model with “step-by-step” reasoning examples improved dramatically its performance.

Standard Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. 

Chain-of-Thought Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. $5 + 6 = 11$. The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had $23 - 20 = 3$. They bought 6 more apples, so they have $3 + 6 = 9$. The answer is 9. 

This can be scaled up to complex reasoning, e.g. mathematics, with reinforcement learning to boost the generation of chains of thoughts.

This can be scaled up to complex reasoning, e.g. mathematics, with reinforcement learning to boost the generation of chains of thoughts.

The current standard method is “Group Relative Policy Optimization”:

1. Prompt the model to think step by step and to provide a well formatted answer.
2. Generate multiple response for a given prompt/question.
3. Compute the relative quality of each response.
4. Reinforce the network accordingly.

Meta's Computational World Model 32B (CWM, CodeGen team 2025):

Pre-training

- + **General pre-training:** 8T tokens of mostly english text with 30% coding data, 8k seqlen.

Get a strong language model with all the capabilities that go with it.

(*) 1T tokens \simeq 10M books \simeq 250 English Wikipedia

Meta's Computational World Model 32B (CWM, CodeGen team 2025):

Pre-training

- + **General pre-training:** 8T tokens of mostly english text with 30% coding data, 8k seqlen.

Get a strong language model with all the capabilities that go with it.

- + **Code world model mid-training:** 5T tokens of data providing examples of code execution effects, 131k seqlen.

The model now has stronger generative capabilities for code and long-context.

(*) 1T tokens \simeq 10M books \simeq 250 English Wikipedia

Meta's Computational World Model 32B (CWM, CodeGen team 2025):

Post-training

- + **Supervised fine-tuning:** 100B tokens of examples of instruction-following, tool use and reasoning traces, 32k seqlen.

The model becomes an assistant able to generate chains of thought.

(*) 1T tokens \simeq 10M books \simeq 250 English Wikipedia

Meta's Computational World Model 32B (CWM, CodeGen team 2025):

Post-training

- + **Supervised fine-tuning:** 100B tokens of examples of instruction-following, tool use and reasoning traces, 32k seqlen.

The model becomes an assistant able to generate chains of thought.

- + **Reasoning Reinforcement-Learning:** 172B tokens from its own reasoning and agentic behavior, 131k seqlen, up to 128 turns.

Boost the quality of the chains of thought and agentic capabilities.

(*) 1T tokens \simeq 10M books \simeq 250 English Wikipedia

Conclusion

- + Large language models are extraordinary general-purpose “cognitive engines” fitting to modern computational devices.

- + Large language models are extraordinary general-purpose “cognitive engines” fitting to modern computational devices.
- + The current trend focuses on improving training procedures to leverage text generation for reasoning and agentic behavior.

- + Large language models are extraordinary general-purpose “cognitive engines” fitting to modern computational devices.
- + The current trend focuses on improving training procedures to leverage text generation for reasoning and agentic behavior.
- + We have probably passed the “data peak” and need new ways to keep improving. Synthetic data and self play are probably key.

- + Large language models are extraordinary general-purpose “cognitive engines” fitting to modern computational devices.
- + The current trend focuses on improving training procedures to leverage text generation for reasoning and agentic behavior.
- + We have probably passed the “data peak” and need new ways to keep improving. Synthetic data and self play are probably key.
- + Architectures are lacking fundamental capabilities such as continuous and multi-modal reasoning.

F. CodeGen team. **Cwm: An open-weights ILM for research on code generation with world models**, 2025.

A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan, A. Goyal, A. Hartshorn, A. Yang, A. Mitra, A. Sravankumar, A. Korenev, A. Hinsvark, A. Rao, A. Zhang, A. Rodriguez, A. Gregerson, A. Spataru, B. Roziere, B. Biron, B. Tang, B. Chern, C. Caucheteux, C. Nayak, C. Bi, C. Marra, C. McConnell, C. Keller, C. Touret, C. Wu, C. Wong, C. C. Ferrer, C. Nikolaidis, D. Allonsius, D. Song, D. Pintz, D. Livshits, D. Esiobu, D. Choudhary, D. Mahajan, D. Garcia-Olano, D. Perino, D. Hupkes, E. Lakomkin, E. AlBadawy, E. Lobanova, E. Dinan, E. M. Smith, F. Radenovic, F. Zhang, G. Synnaeve, G. Lee, G. L. Anderson, G. Nail, G. Mialon, G. Pang, G. Cucurell, H. Nguyen, H. Korevaar, H. Xu, H. Touvron, I. Zarov, I. A. Ibarra, I. Kloumann, I. Misra, I. Evtimov, J. Copet, J. Lee, J. Geffert, J. Vranes, J. Park, J. Mahadeokar, J. Shah, J. van der Linde, J. Billock, J. Hong, J. Lee, J. Fu, J. Chi, J. Huang, J. Liu, J. Wang, J. Yu, J. Bitton, J. Spisak, J. Park, J. Rocca, J. Johnstun, J. Saxe, J. Jia, K. V. Alwala, K. Upasani, K. Plawiak, K. Li, K. Heafield, K. Stone, K. El-Arini, K. Iyer, K. Malik, K. Chiu, K. Bhalla, L. Rantala-Yeary, L. van der Maaten, L. Chen, L. Tan, L. Jenkins, L. Martin, L. Madaan, L. Malo, L. Blecher, L. Landzaat, L. de Oliveira, M. Muzzi, M. Pasupuleti, M. Singh, M. Paluri, M. Kardas, M. Oldham, M. Rita, M. Pavlova, M. Kambadur, M. Lewis, M. Si, M. K. Singh, M. Hassan, N. Goyal, N. Torabi, N. Bashlykov, N. Bogoychev, N. Chatterji, O. Duchenne, O. Çelebi, P. Alrassy, P. Zhang, P. Li, P. Vasic, P. Weng, P. Bhargava, P. Dubal, P. Krishnan, P. S. Koura, P. Xu, Q. He, Q. Dong, R. Srinivasan, R. Ganapathy, R. Calderer, R. S. Cabral, R. Stojnic, R. Raileanu, R. Girdhar, R. Patel, R. Sauvestre, R. Polidoro, R. Sumbaly, R. Taylor, R. Silva, R. Hou, R. Wang, S. Hosseini, S. Chennabasappa, S. Singh, S. Bell, S. S. Kim, S. Edunov, S. Nie, S. Narang, S. Raparthy, S. Shen, S. Wan, S. Bhosale, S. Zhang, S. Vandenhende, S. Batra, S. Whitman, S. Sootla, S. Collot, S. Gururangan, S. Borodinsky, T. Herman, T. Fowler, T. Sheasha, T. Georgiou, T. Scialom, T. Speckbacher, T. Mihaylov, T. Xiao, U. Karn, V. Goswami, V. Gupta, V. Ramanathan, V. Kerkez,