

3. Using encoder-decoder architectures

3.1. Tasks, training and inference

Training a transformer

This is painful to train from scratch, though it is possible.

Requires some form a curriculum learning in general (easy → hard examples).

Simpler to train encoder and decoder separately, then train cross attention.

See the “VisionEncoderDecoderModel” factory in HF Transformers.

→ So we'll talk about encoder and decoders separately to show how they are trained.

Tasks vanilla transformers are good at

Summarization — $|in| >> |out|$

Question-Answering (based on input + question) = conditioned summarization

Translation — $|in| \approx |out|$ (but \neq) = special case of QA (importance of 1st token)

~~Content generation~~

Better on **abstractive** tasks than **extractive** ones, due to **decoder**.

Somehow “slowed down” by the **unnecessary encoder for generative tasks**.

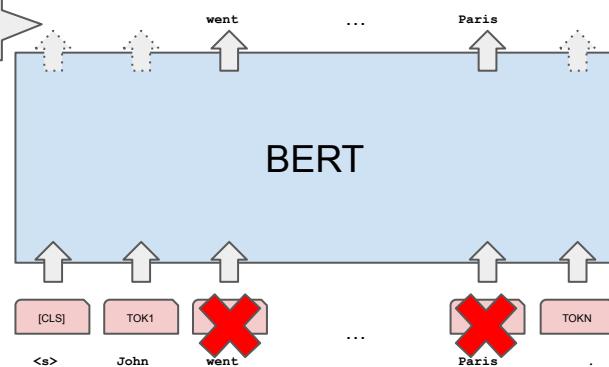
Now, encoders and decoders are often used separately.

And most important training them separately is more efficient.

3.2. Encoder-only

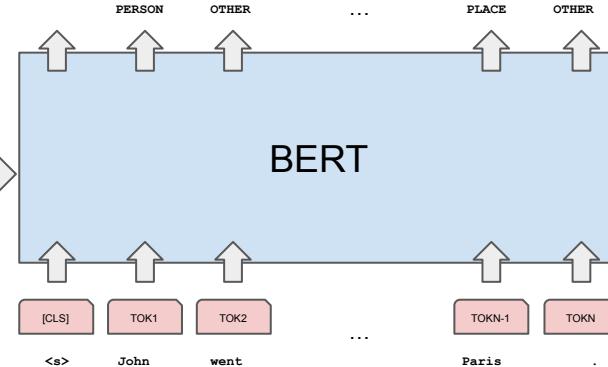
Encodeurs : entraînement auto-supervisé facilité

Pré-entraînement en valorisant les masses de données brutes, non annotées.



Tâche prétexte de masquage de termes

Entraînement supervisé (spécialisation) sur une tâche précise, en affinant le modèle pré-entraîné.



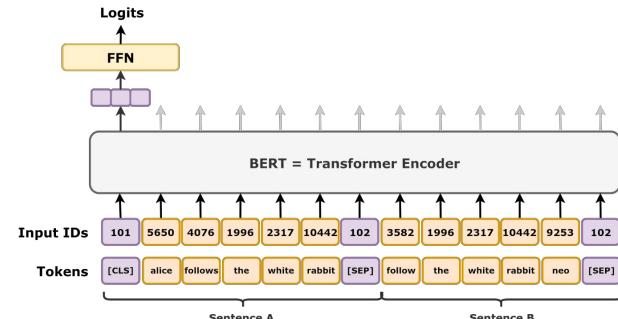
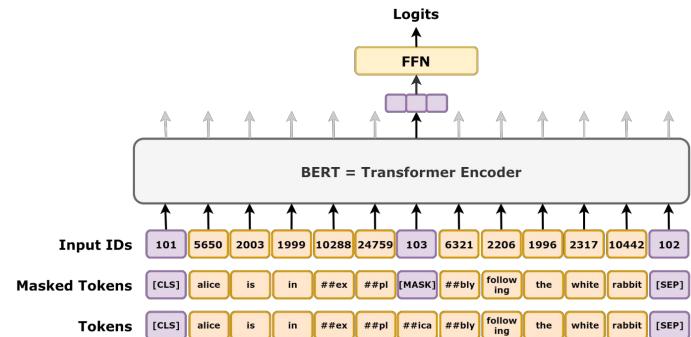
Tâche finale de reconnaissance d'entités nommées

Training details

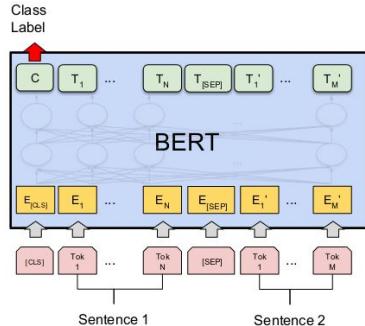
Masked Language Modeling (MLM):

- Task: **recover masked tokens** (denoising problem)
- Keep 85% of the tokens unchanged, and don't care about the predicted value
- For the 15% remaining tokens, check whether the output is correct while
 - 80% of the time, replace the input token with [MASK] (learn to reconstruct)
 - 10% of the time, replace the input token with a random word (noise tolerance)
 - 10% of the time, keep the original token unchanged (bias toward the correct value)

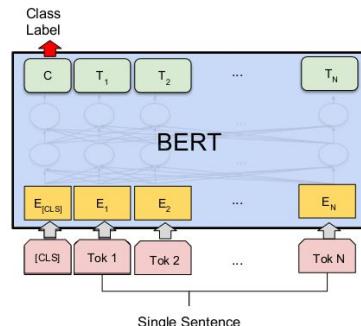
Next Sequence Predictions is not used anymore.
Check the ModernBERT paper for more information.



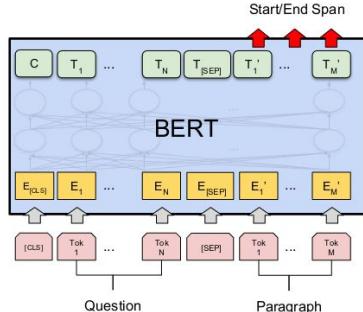
Une architecture, des tâches multiples



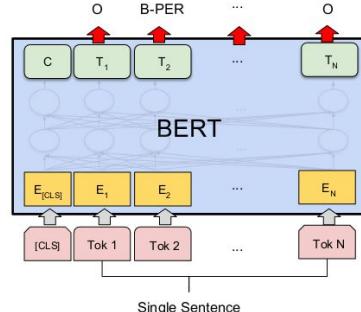
(a) Sentence Pair Classification Tasks:
MNLI, QQP, QNLI, STS-B, MRPC,
RTE, SWAG



(b) Single Sentence Classification Tasks:
SST-2, CoLA



(c) Question Answering Tasks:
SQuAD v1.1

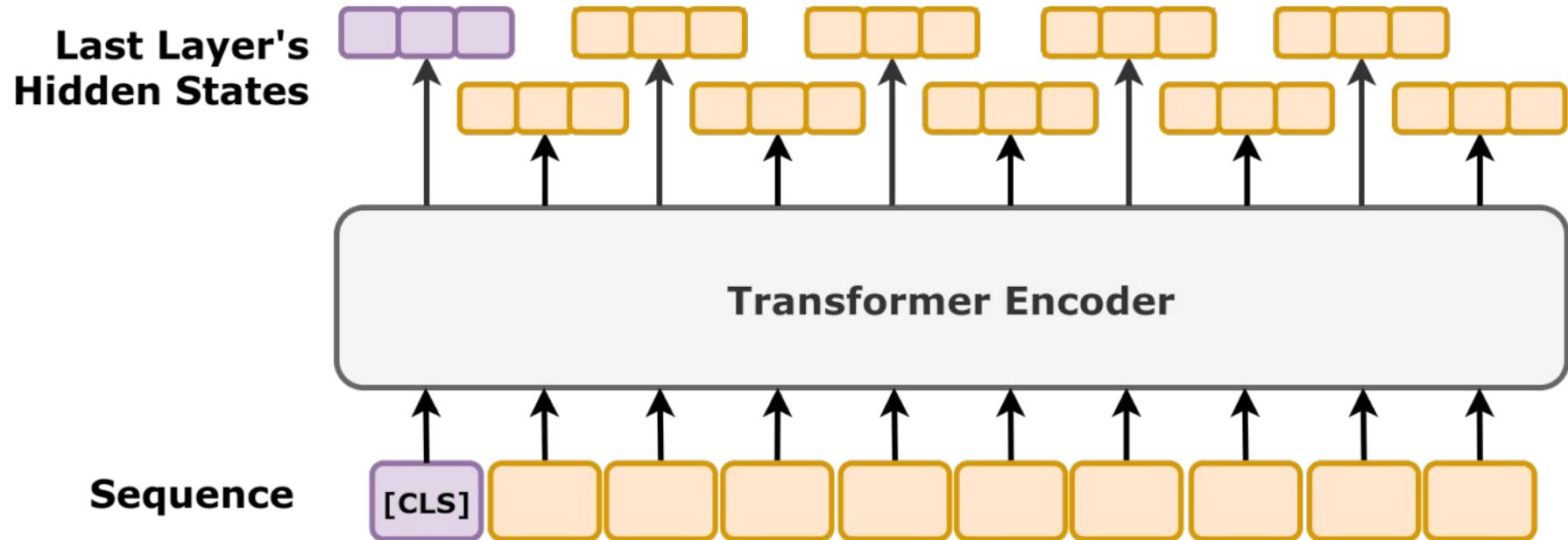


(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER

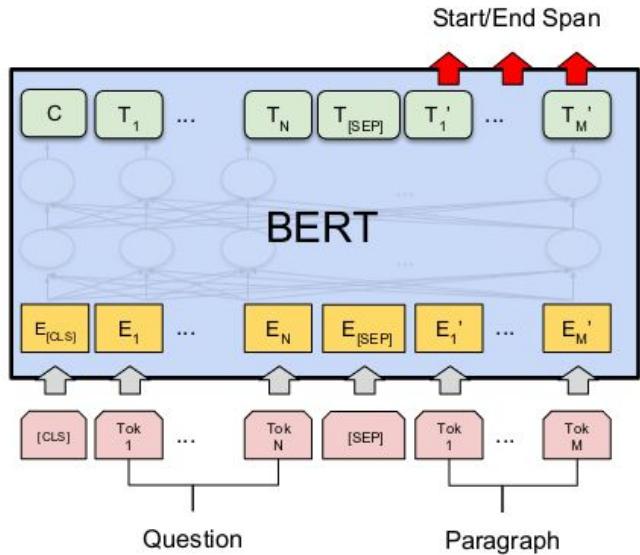
En utilisant des séparateurs spéciaux intégrés à la séquence de mots/tokens à analyser, il est possible de spécialiser un encodeur sur de **multiples tâches** :

- a) Indiquer si 2 phrases sont dans un **ordre logique ou cohérentes**.
- b) Affecter une **catégorie** à une **phrase**.
- c) Poser des **questions**.
- d) Affecter des **catégories** à chaque mot.

Sequence classification: prediction for the [CLS] token



Dépasser la tâche de détection ?



Dans le cas des **questions / réponses** :

- La **question** est intégrée comme un **contexte supplémentaire**.
- Répondre consiste à **déetecter la portion pertinente** dans le contenu à analyser.

Les encodeurs restent **limités** à une forme de **détection**, et la performance est assez décevante en pratique.

Bilan des encodeurs

Forces

- **Pré-entraînement auto-supervisé**
⇒ *valorisation des données non-annotées*
- **Spécialisation facile avec peu de données**
- **Bonne performance pour**
 - Embedding (intermediate MHA feat.)
 - Sequence classification ([CLS] token)
 - Extractive tasks (token classification)

Faiblesses

- Limité à l'affectation d'une **catégorie à chaque mot ou à la phase complète**.
⇒ *Problématique si l'information n'est pas explicitement présente.*
- Pas de **relation causale**
prédictions (sorties) = # entrées
⇒ *Comment traduire entre deux langages avec un nombre de tokens très différents ?*
(ex. : *FR ↔ CN*)
⇒ *Comment construire progressivement une réponse (agent de dialogue) ?*
- *Besoin de tokens spéciaux pour gérer des nouveaux types de tâches.*

3.3. Decoder-only

What is a language model?

$$P(S = \{X_t, X_{t-1}, X_{t-2}, \dots, X_0\}) \text{ or } P(X_t \in S)$$

A system which can compute the probability of a sequence, or of any part (word) of it, given the assumption this sequence (word) belong to a particular language.

Can be autoregressive: *The capital of France is [??]*

But not necessarily: *Humans have [??] hands (in general).*

Plus it can be conditioned externally: *[IMAGE] The color of this table is [??]
(or is it another form of content in the input stream?)*

What is a autoregressive (or “semi-causal”) model?

$$P(X_t | X_{t-1}, \dots, X_{t-k})$$

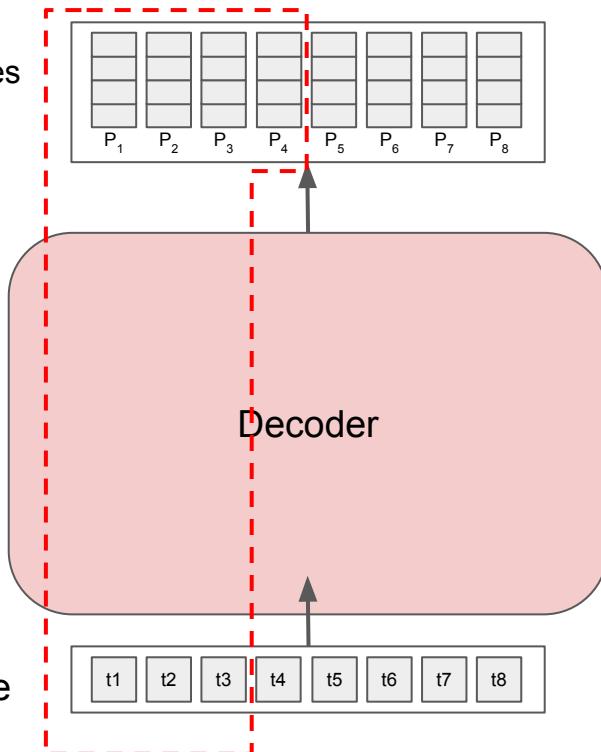
A system which predicts the probability of the next observation, given past observation(s).

Many **families**: HMMs, RNNs, but also Kalman & Particle Filters...

Many **training** strategies.

Decoder use 1: as a causal language model

Token probabilities
for each position



Enables to assess the likelihood of some sequence according to a learned causal language model: *traditional use in OCR, speech recognition as seq. rescorer...*

Causal, because with **masked attention** the decoder can only **look at the past tokens** to judge the current token.

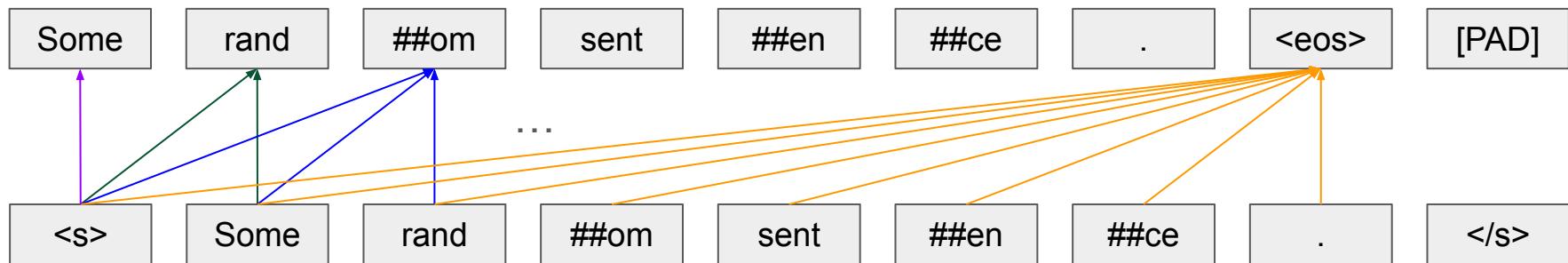
$$P(t_i | t_{i-1}, t_{i-2}, \dots, t_{i-n})$$

Given an **input sequence**, probabilities for each token can be **computed in parallel**.

Training a GPT-like RNN using “Teacher Forcing”

Very important self-supervised training technique.

Use the full expected sequence both as input and shifted output.



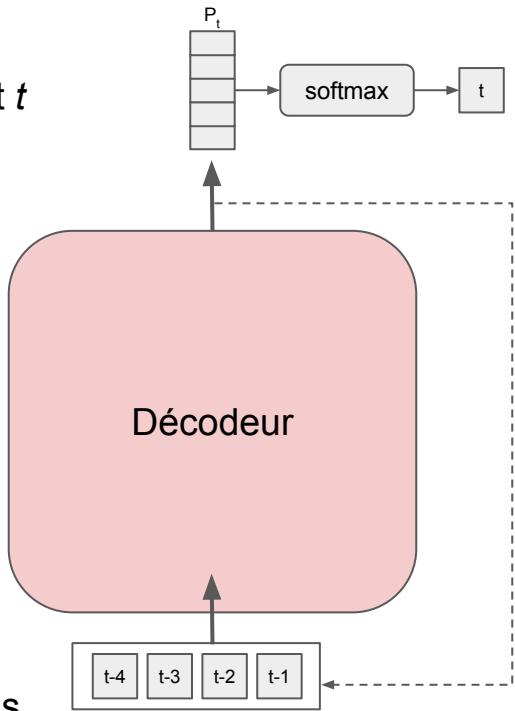
Outputs are computed in parallel thanks to masking.

Advantage over MLM: **all tokens are used for training** (masking for each token).

Can **add noise** to make the training more robust.

Decoder use 1: as a generative model

Predictions at t



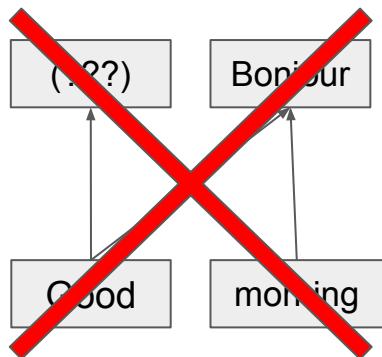
The decoder emits predictions **one by one**, using a **for-loop**.

It can therefore generate a **variable number of tokens!**

For each prediction, it only considers **past tokens**.

Teacher forcing is the critical element to enable this behavior.

Translation in practice



Training Dataset

I	am	a	student	<to-fr>	je	suis	étudiant
let	them	eat	cake	<to-fr>	Qu'ils	mangent	de
good	morning	<to-fr>	Bonjour				

- ✓ Input-then-output
- ✓ use special tokens to switch between sub-tasks

Summarization in practice

The image shows two versions of a Wikipedia article page for "Positronic brain". The left version is the original page, and the right version is a summarized version. Both pages have the same URL: <https://en.wikipedia.org/w/index.php?title=Positronic%20brain&oldid=10200000>.

Original Page (Left):

This page is about a fictional technological device. For the manufacturing company based in Springfield, Missouri, see [Positronic \(company\)](#).

Summary (Right):

This article needs additional citations for verification. Please help improve this article by adding citations to reliable sources. Unreliable sources may be challenged and removed. [\(Editor's note\)](#) [\(Author's note\)](#) [\(Learn how and when to remove this template message\)](#)

A positronic brain is a fictional technological device, originally conceived by science fiction writer Isaac Asimov.^[1] It functions as a central processing unit (CPU) for robots, and, in some unspecified way, provides them with a form of consciousness recognizable to humans. When Asimov wrote his first robot stories in 1939 and 1940, the concept of a positronic brain never occurred to him, and he had no idea what it was. In his 1950 novel *Runaround*, Asimov elaborates on the concept, in the context of the fictional Three Laws of Robotics.

Conceptual overview [edit]

Asimov remained vague about the technical details of positronic brains except to assert that their substrate was formed from an alloy of platinum and iridium. They were said to be vulnerable to radiation and apparently involve a type of volatile memory (since robots in storage required a power source keeping their brains "alive"). The focus of Asimov's stories on the software of robots—such as the Three Laws of Robotics—that are the hardware in which it was implemented, rather than the hardware itself, led to the creation of a positronic brain within the Three Laws that would have been necessary to spend years redesigning the fundamental approach towards the brain itself.

Within his stories of robotics on Earth and their development by U.S. Robots, Asimov's positronic brain is less of a physical device and more of a technological item worthy of study.

A positronic brain originally built without incorporating the Three Laws, any modifications thereof would actually modify robot behavior. Behavioral dilemmas resulting from conflicting potentials set by incompatible laws of the robot for the Three Laws made up the bulk of Asimov's stories concerning robots. They are resolved by applying the science of logic and psychology together with mathematics, the supreme solution finder being Dr. Susan Calvin, Chief Psychophysicist of U.S. Robots.

The Three Laws are also a bottleneck in brain sophistication. Very complex brains designed to handle world economy interpret the First Law in expanded sense to include the Second Law. For example, in Asimov's later novel *Robot and Empire* the law is referred to as the "Second Law". A less complex brain designed as a calculating machine, as opposed to being a robot control circuit, was designed to have a flexible, childlike personality so that it was able to pursue difficult problems without the Three Laws limiting its complexity. Specialized brains created for overseeing world economies were stated to have no personality at all.

Under specific conditions, the Three Laws are violated, with the modification of the actual robotic design:

- Robots that are of low enough value can have the Third Law deleted. They do not have to protect themselves from harm, and the brain size can be reduced by half.
- Robots that are not required to obey the First Law can have the Second Law deleted, and therefore require smaller brain again, though they do not require the Third Law.
- Robots that are disposable can have the First Law deleted, so that they do not have to not harm a human, nor not require even the First Law. The sophistication of particular robots determines a brain's size and how it could potentially rewire itself through self-repair.

Robots of the latter type directly parallels contemporary industrial robotics practice, though not the robots do contain safety sensors and systems, in a concern for human safety or weak form of the First Law, the robot is a safe tool to use, but has no "subjective", which is implicit in Asimov's own stories.

In Asimov's trilogy [edit]

Several robot stories have been written by other authors following Asimov's death. For example, in Roger MacBride Allen's *Cathleen Ni Ogair*, a Sparrow robot called Gubier Asimov invents the gravitronic brain. It offers speed and capacity improvements over traditional positronic designs, but the strong influence of tradition make robotics labs reject Asimov's work. Only one robotics lab, Fredie Loring, chooses to adopt gravitronics, because it offers her a bank stake on which she could explore alternatives to the Three Laws. Because they are not dependent upon centuries of earlier research, gravitonic brains can be programmed with the standard Laws, variations of the Laws, or even empty pathways which specify no Laws at all.

Main page **Contents** **Featured content** **Current events** **Random article** **Donate** **Help** **Special pages** **Wikisource store**

Interaction **Help** **About Wikipedia** **Community portal** **Recent changes** **Contact page**

Tools **What links here** **Related edits** **Upload file** **Special pages** **Printable version** **Page information** **Permanent link** **Cite this page**

Print/export **Create a book** **Download PDF** **Printable version**

Languages **English** **Français** **日本語** **Deutsch** **Português** **Svenska** **Türkçe**

Links **What links here** **Related edits** **Upload file** **Special pages** **Printable version** **Page information** **Permanent link** **Cite this page**

Print/export **Create a book** **Download PDF** **Printable version**

Conceptual overview [edit]

Asimov remained vague about the technical details of positronic brains enough to assert that their substrate was formed from an alloy of platinum and iridium. They were said to be vulnerable to radiation and apparently involve a type of volatile memory (since robots in storage required a power source keeping their brains "alive"). The focus of Asimov's stories was directed more towards the software of robots—such as the Three Laws of Robotics—that is the hardware in which it was implemented, rather than the hardware itself, leading to the creation of a positronic brain within the Three Laws that would have been necessary to spend years redesigning the fundamental approach towards the brain itself.

Within his stories of robotics on Earth and their development by U.S. Robots, Asimov's positronic brain is less of a physical device and more of a technological item worthy of study.

A positronic brain cannot outlast the body without incorporating the Three Laws, any modification thereof actually modifies robot behavior. Behavioral dilemmas resulting from conflicting potentials set by incompatible laws of the robot for the Three Laws make up the bulk of Asimov's stories concerning robots. They are resolved by applying the science of logic and psychology together with mathematics, the supreme solution finder being Dr. Susan Calvin, Chief Psychophysicist of U.S. Robots.

The Three Laws are also a bottleneck in brain sophistication. Very complex brains designed to handle world economy interpret the First Law in expanded sense to include the Second Law. For example, in Asimov's later novel *Robot and Empire* the law is referred to as the "Second Law". A less complex brain designed as a calculating machine, as opposed to being a robot control circuit, was designed to have a flexible, childlike personality so that it was able to pursue difficult problems without the Three Laws limiting its complexity. Specialized brains created for overseeing world economies were stated to have no personality at all.

Under specific conditions, the Three Laws can be violated, with the exception of the Second Law, as follows:

- Robots that are of low enough value can have the Third Law deleted. They do not have to protect themselves from harm, and the brain size can be reduced by half.
- Robots that are not required to obey the First Law can have the Second Law deleted, and therefore require smaller brain again, though they do not require the Third Law.
- Robots that are disposable can have the First Law deleted, so that they do not have to not harm a human, nor not require even the First Law. The sophistication of particular robots determines a brain's size and how it could potentially rewire itself through self-repair.

Robots of the latter type directly parallels contemporary industrial robotics practice, though not the robots do contain safety sensors and systems, in a concern for human safety or weak form of the First Law, the robot is a safe tool to use, but has no "subjective", which is implicit in Asimov's own stories.

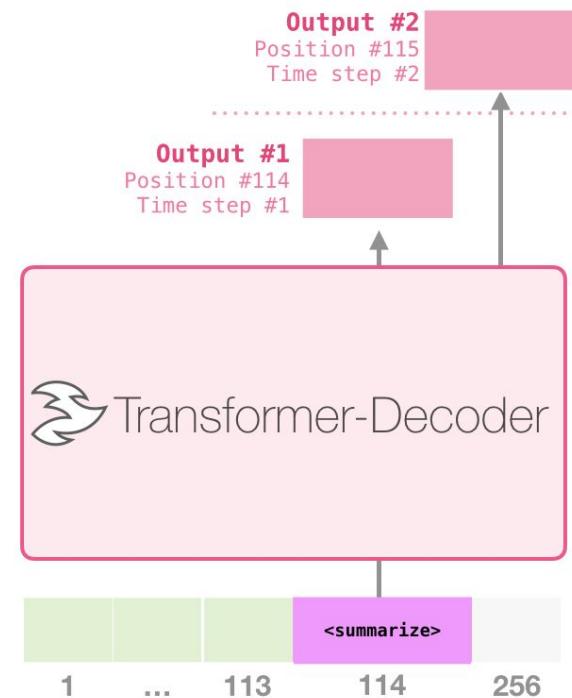
ARTICLE

Besides Isaac Asimov's original stories, many others have been written by other authors following Asimov's death. For example, in Roger MacBride Allen's *Cathleen Ni Ogair*, a Sparrow robot called Gubier Asimov invents the gravitronic brain. It offers speed and capacity improvements over traditional positronic designs, but the strong influence of tradition make robotics labs reject Asimov's work. Only one robotics lab, Fredie Loring, chooses to adopt gravitronics, because it offers her a bank stake on which she could explore alternatives to the Three Laws. Because they are not dependent upon centuries of earlier research, gravitonic brains can be programmed with the standard Laws, variations of the Laws, or even empty pathways which specify no Laws at all.

Summarization in practice

Training Dataset

Article #1 tokens		<summarize>	Article #1 Summary
Article #2 tokens	<summarize>	Article #2 Summary	padding
Article #3 tokens	<summarize>	Article #3 Summary	

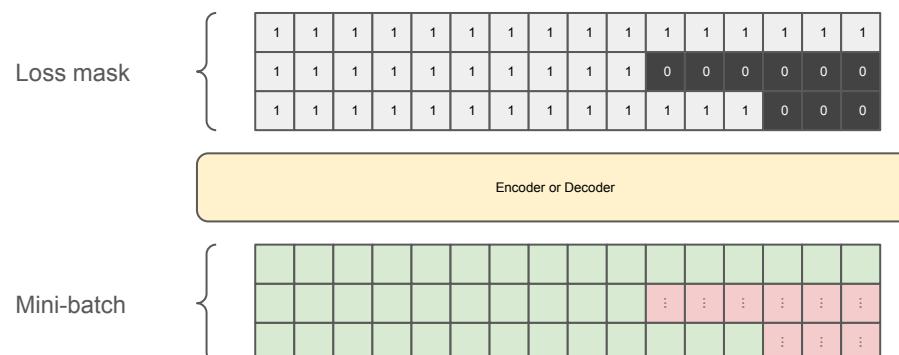


Training in batch and padding (*training detail*)

We want to be able to train on **multiple sequences** at the same time, however they may **not have the same length**.

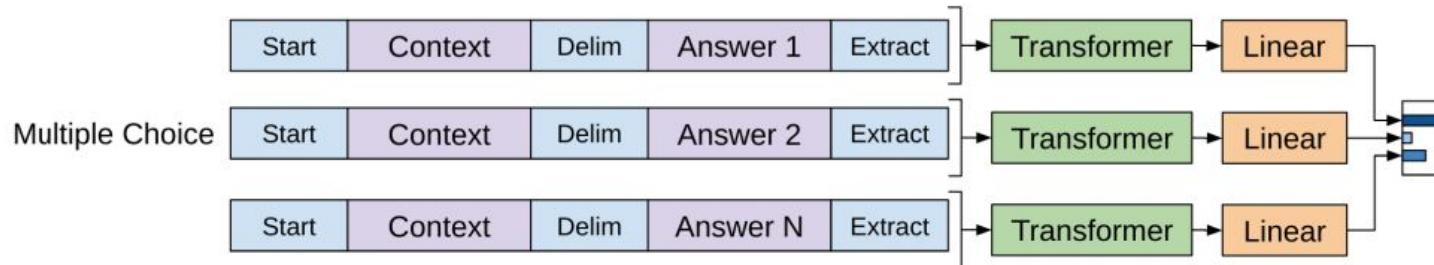
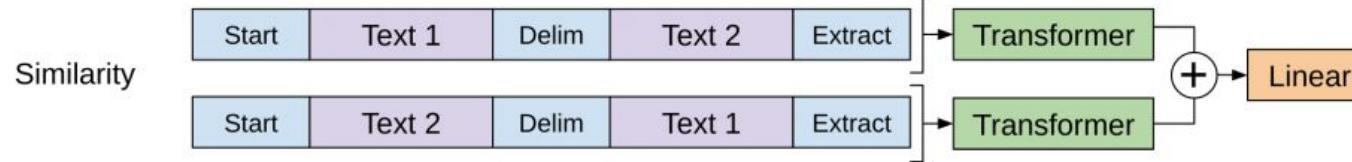
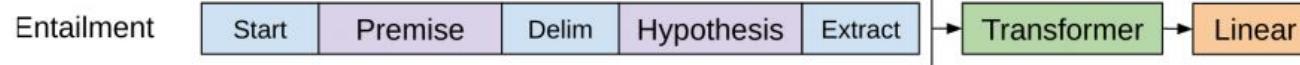
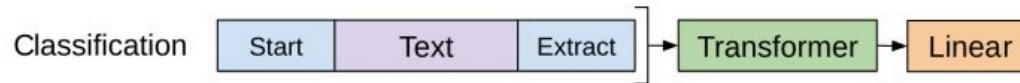
Option 1 (bad): crop sequences

Option 2 (good): pad with special tokens (ignored in loss)



Option 3 (production): padding + clustering of samples per length in batches

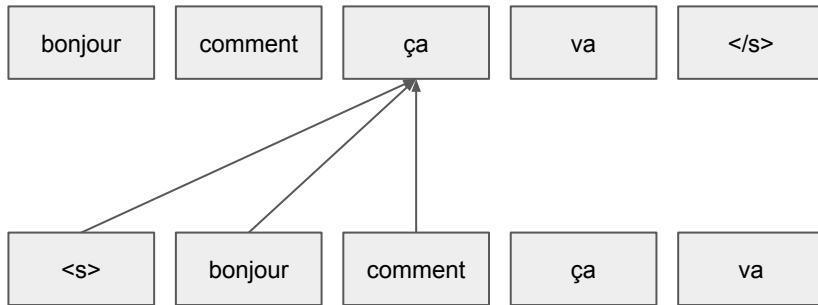
Sample tasks for decoders with special tokens



General training process

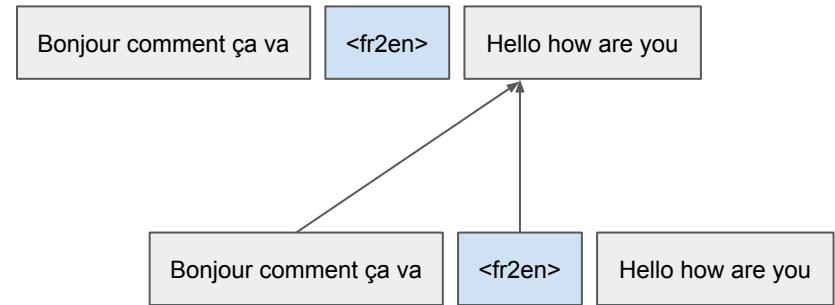
1

Self-supervised
pre-training



2

Supervised
training (“SFT”)



(Causal) LMs are few-shot learners

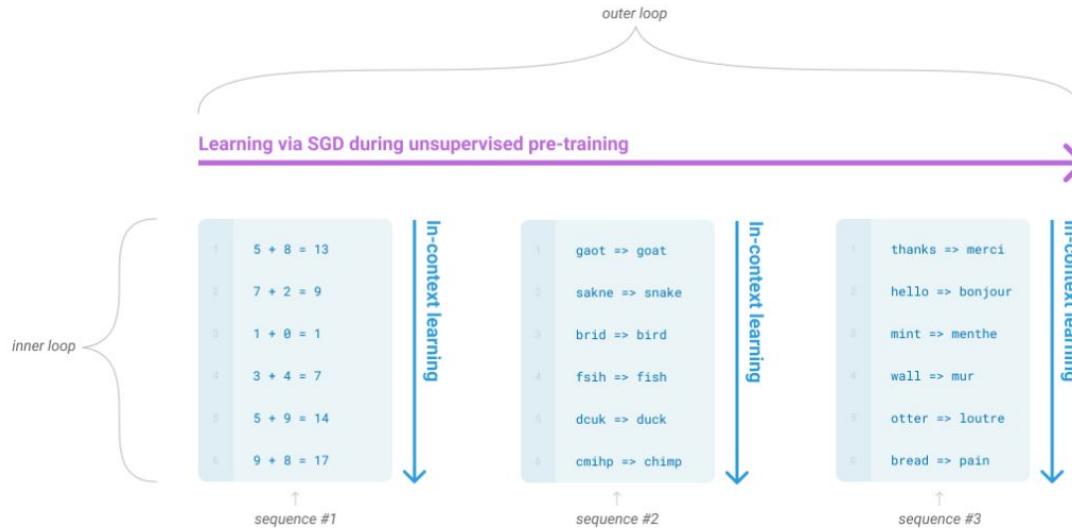


Figure 1.1: Language model meta-learning. During unsupervised pre-training, a language model develops a broad set of skills and pattern recognition abilities. It then uses these abilities at inference time to rapidly adapt to or recognize the desired task. We use the term “in-context learning” to describe the inner loop of this process, which occurs within the forward-pass upon each sequence. The sequences in this diagram are not intended to be representative of the data a model would see during pre-training, but are intended to show that there are sometimes repeated sub-tasks embedded within a single sequence.

Alignment and reinforcement learning (RL)

Natural language is more flexible than predefined tokens to express tasks, and can be used at inference time to define new tasks not seen during training.

What if we could tell the model what it must do instead of using hard-coded special tokens?

Bonjour comment ça va

<fr2en>

Hello how are you

vs

<prompt>Bonjour comment ça va ===== **Traduis cette phrase en anglais.**</prompt><answer>...</answer>

Problem: because we do not know which problems will be posed, we cannot perform supervised learning (SFT).

Solution: use reinforcement learning to optimize the network policy.

Instruct models

The first trick requires to **remove the need for task-specific special tokens**.

Though modern LLMs still have special tokens for their internal use: reasoning, segmentation...

We now have **generic <prompt> / <answer> markers** in the “conversation”.

This bias the model toward behaving **like a chatbot**, and rather than predicting what would be the most probable sentence in the question, **try to answer instead**.

This is a form of “meta-language” learning...

Alignment and RL approaches

“Alignment” then refers to biasing the model toward a **preferable behavior**.
Answer question, avoid toxicity, provide explanation...

All approaches are based on human preference, expressed as triplets:

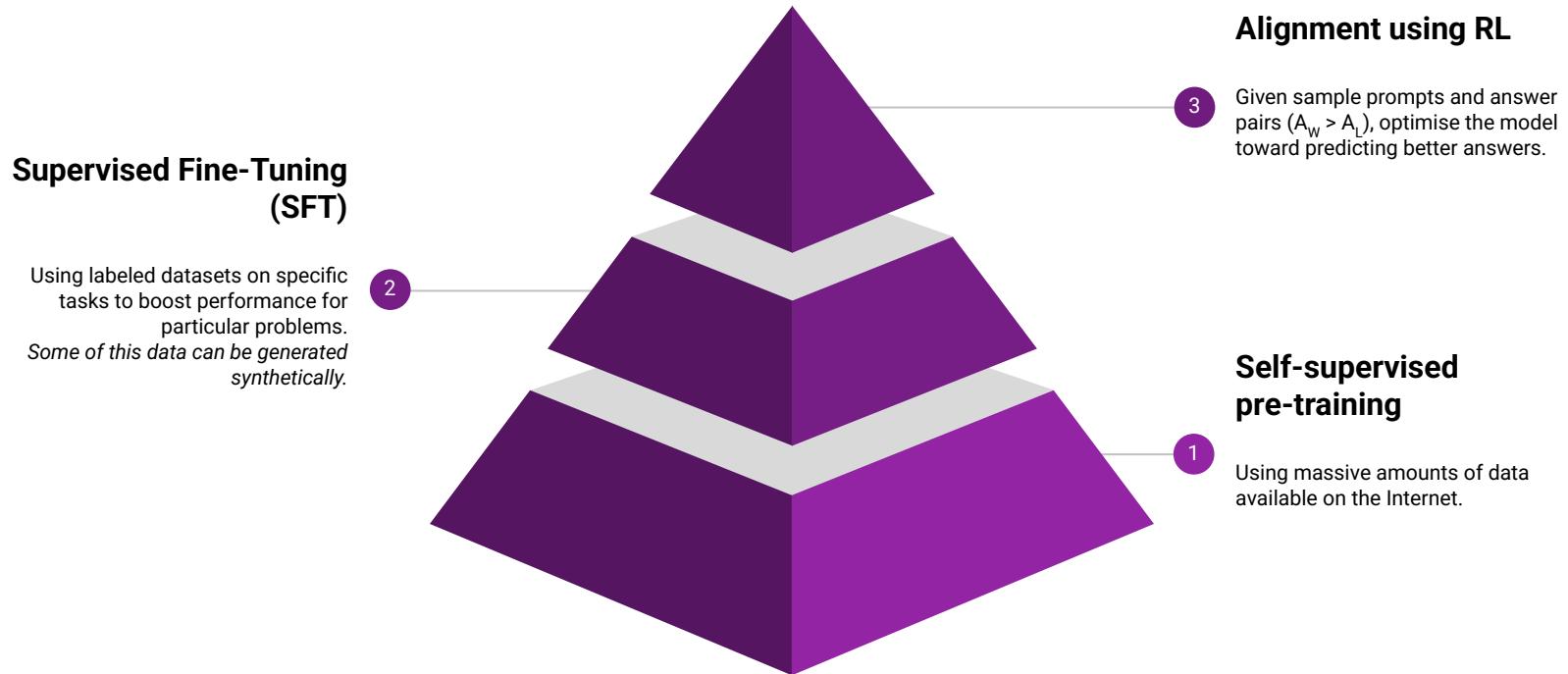
{sample prompt P, preferred answer A_w , less preferred answer A_L }

Current approaches:

- reinforcement learning from human feedback ([RLHF](#)) using Proximal Policy Optimization ([PPO](#)) → learn a reward model (score answers) from human examples
- Direct preference optimization ([DPO](#)) → optimize directly the policy in the network using a sort of contrastive loss (faster, better)

Very easy to use thanks to [HuggingFace TRL](#) library.

State-of-the art training pipeline for LLMs



Bilan des décodeurs

Forces

- Capacité à générer une **nouvelle séquence de longueur variable**
La traduction est possible.
- Possibilité d'**apporter progressivement de l'information**
Adapté aux agents de dialogue.
- **Pre-training massif** possible.
Utilisation de tous les tokens, teacher forcing...
- Capacité à s'adapter à des **tâches nouvelles**.

Faiblesses

- **Entraînement** délicat pour des données structurées.
- **Vitesse** d'inférence limitée.