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| ≈ |out| (but ≠) = 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**.

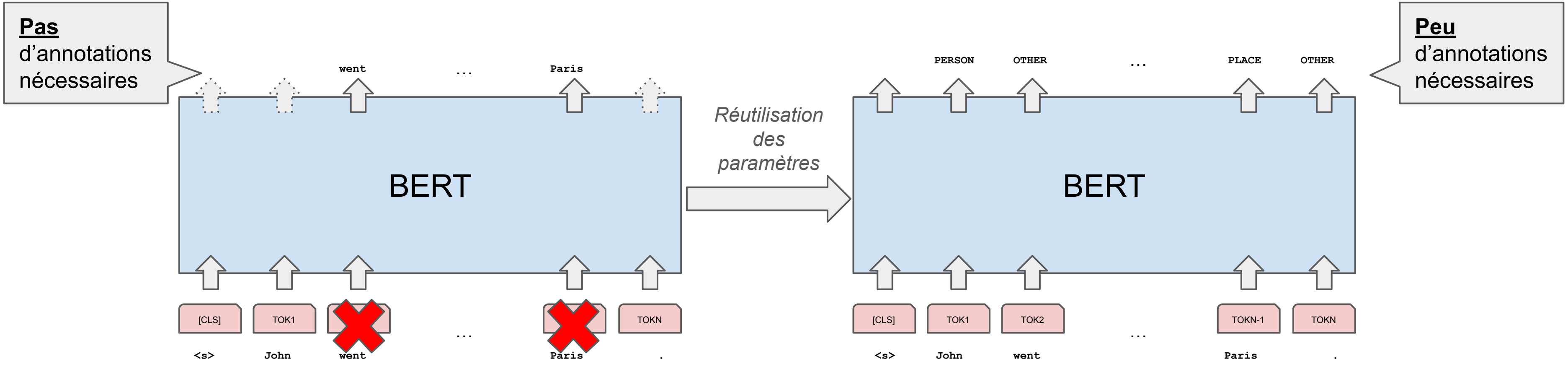
Now, <u>encoders and decoders</u> are often <u>used separately</u>.
And most important <u>training them separately</u> is more efficient.

3.2. Encoder-only

Encodeurs : entraînement auto-supervisé facilité

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

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

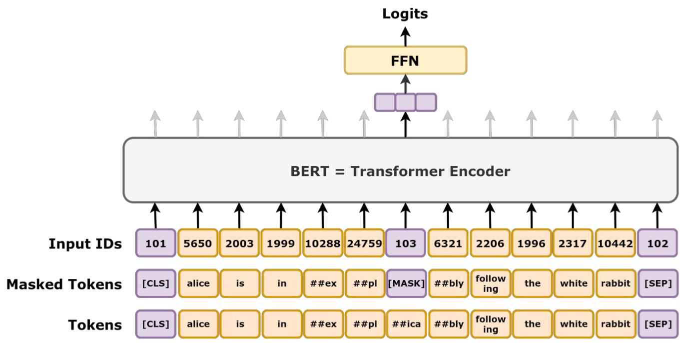


Tâche prétexte de masquage de termes

Tâche finale de reconnaissance d'entités nommées

Architecture BERT [Devlin et al. 2019]

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.

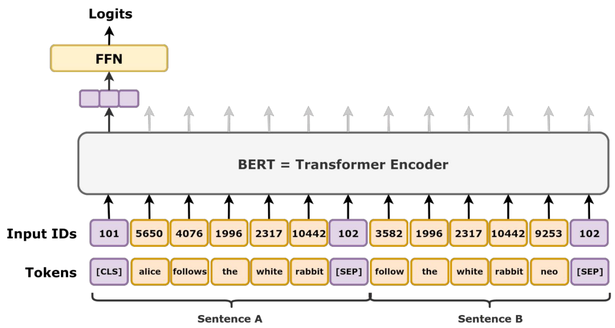
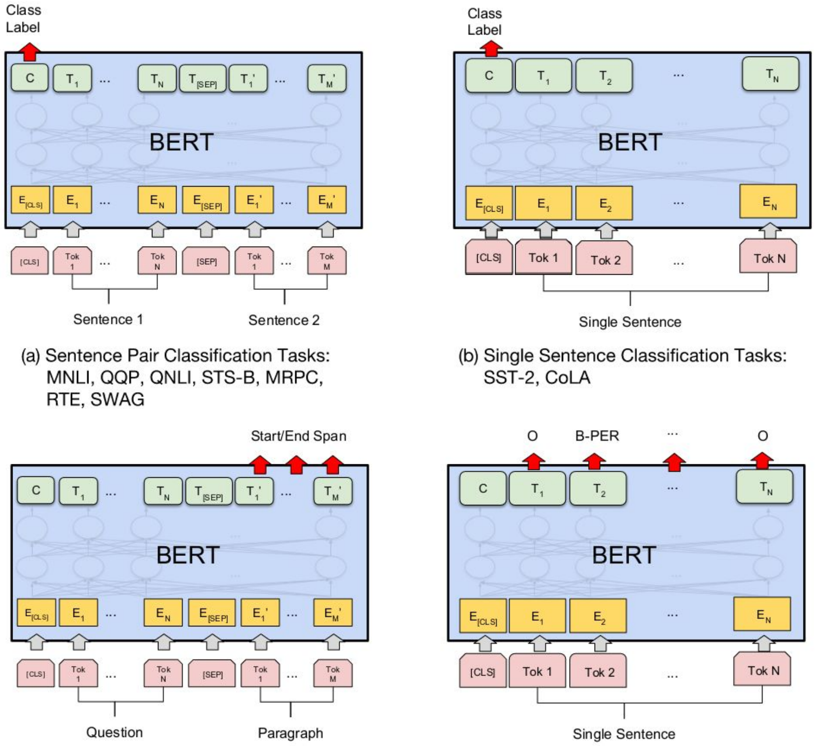


Image source: Voigt Godoy, D. [dvgodoy](https://github.com/dvgodoy) / [CC BY](https://creativecommons.org/licenses/by/4.0/) / https://github.com/dvgodoy/dl-visuals/

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

Extrait de [Devlin et al. 2019]

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

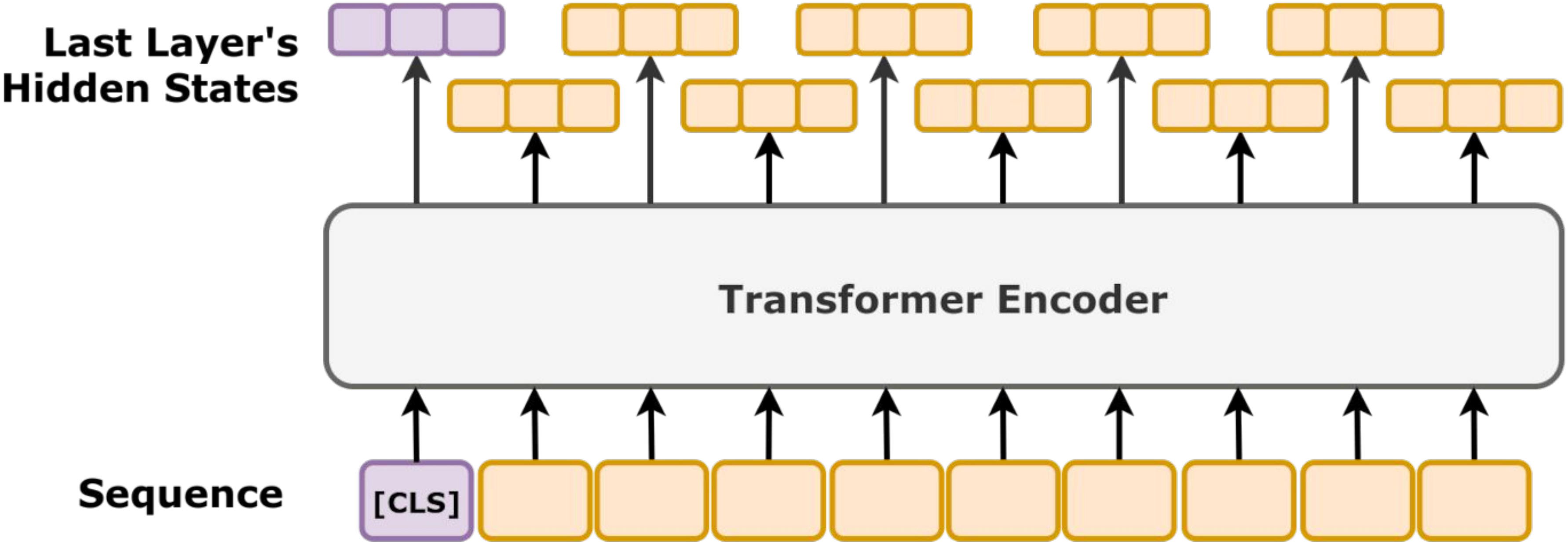
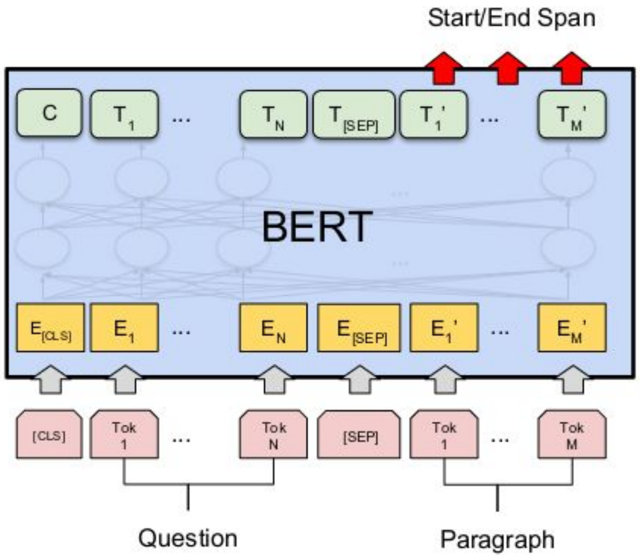


Image source: Voigt Godoy, D. dvgodoy / CC BY / https://github.com/dvgodoy/dl-visuals/?tab=readme-ov-file

Dépasser la tâche de détection ?



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

- La **question** est intégrée comme un **contexte supplémentaire**.
- Répondre consiste à **détecter 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.

Extrait de [Devlin et al. 2019]

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?

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

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?

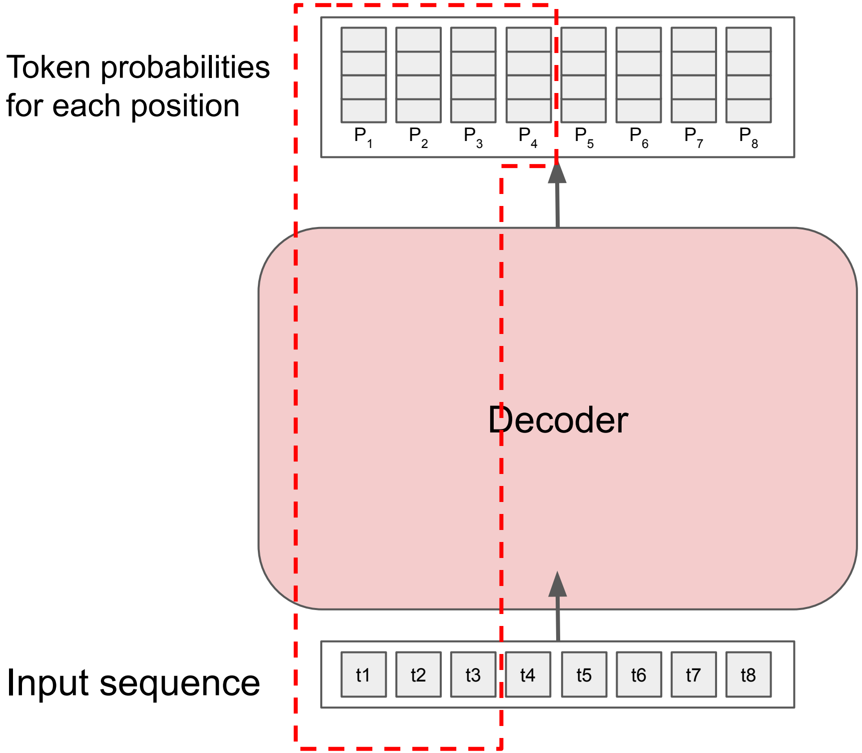
$$P(X_t \mid X_{t-1}, \ldots, X_{t-k})$$

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



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.

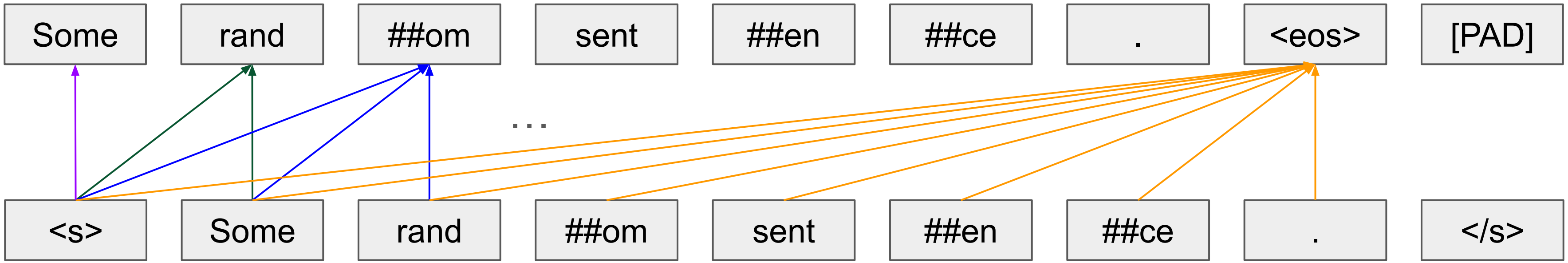
$$P(ti \mid t_{i-1}, t_{i-2}, \ldots, t_{i-n})$$

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**.

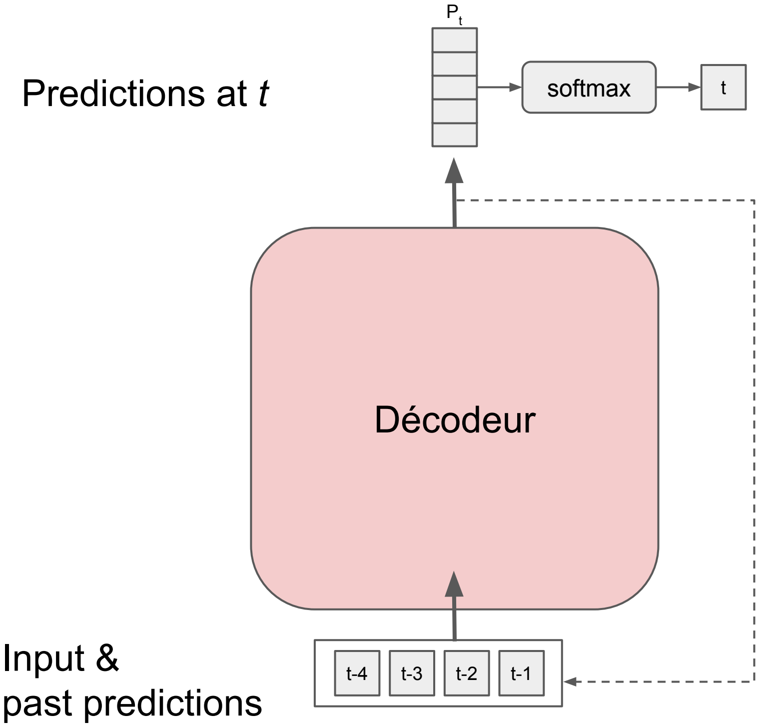


Outputs are computed in parallel thanks to masking.

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



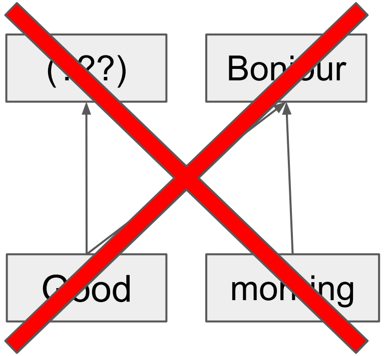
The decoder emits predictions **one by one**, using a **for-loop**.

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

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

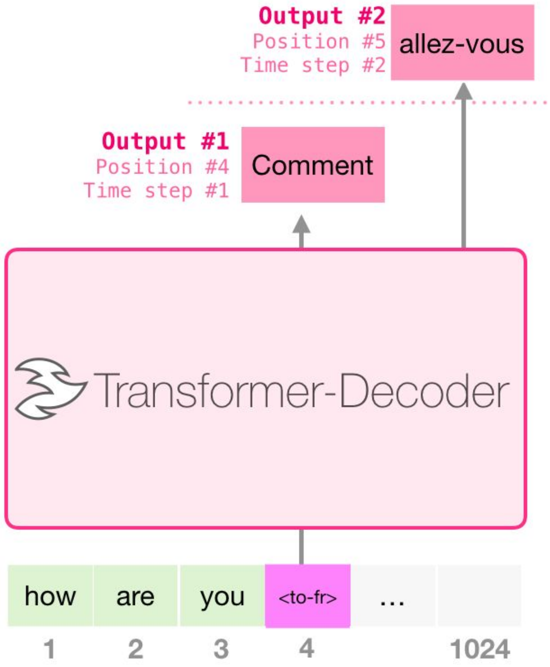
Teacher forcing is the critical element to enable this behavior.

Translation in practice



Training Dataset

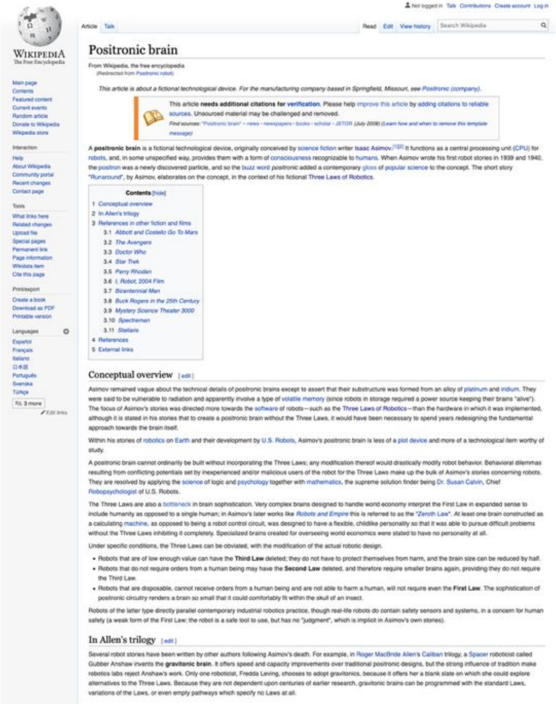
I	am	a	student	<to-fr>	je	suis	étudiant
let	them	eat	cake	<to-fr>	Qu'ils	mangent	de
good	morning	<to-fr>	Bonjour				



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

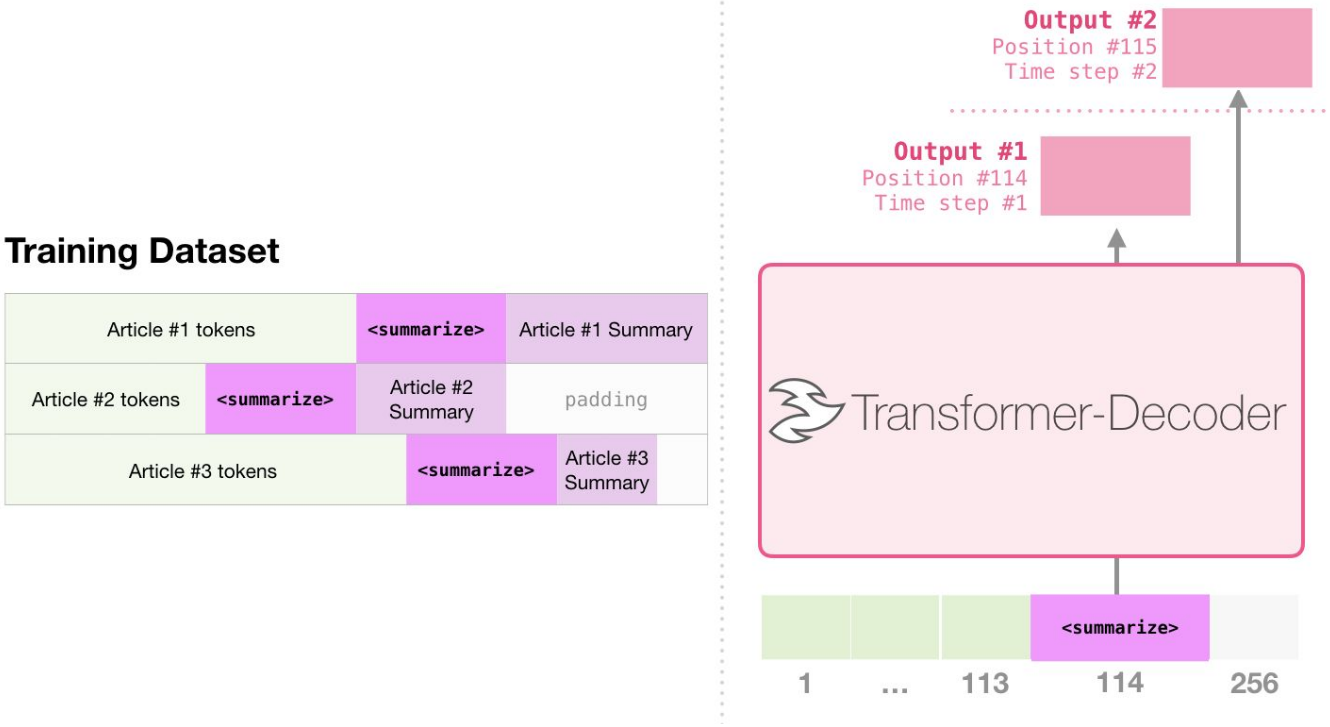
https://jalammar.github.io/illustrated-gpt2/

Summarization in practice



https://jalammar.github.io/illustrated-gpt2/

Summarization in practice



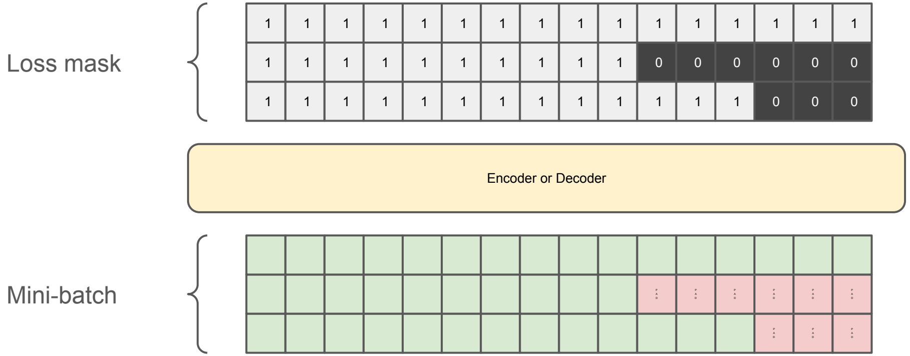
https://jalammar.github.io/illustrated-gpt2/

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**.

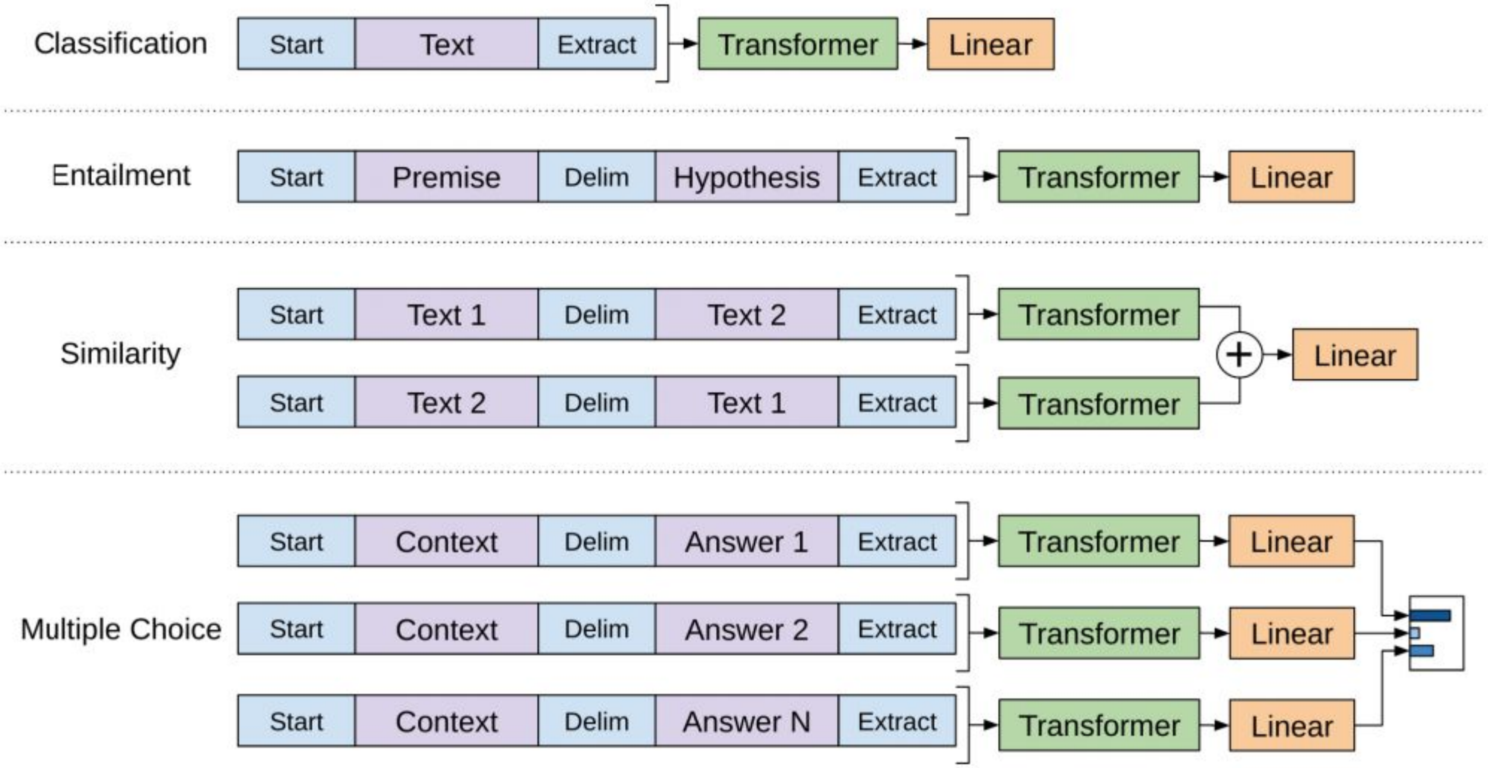
Option 1 (bad): crop sequences

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”)

Interesting: Khandelwal et al. (2019), *Sample Efficient Text Summarization Using a Single Pre-Trained Transformer*. http://arxiv.org/abs/1905.08836

(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.

T. B. Brown *et al.*, "Language Models are Few-Shot Learners," Jul. 2020, http://arxiv.org/abs/2005.14165 ← OpenAI, GPT-2

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 d**o 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:

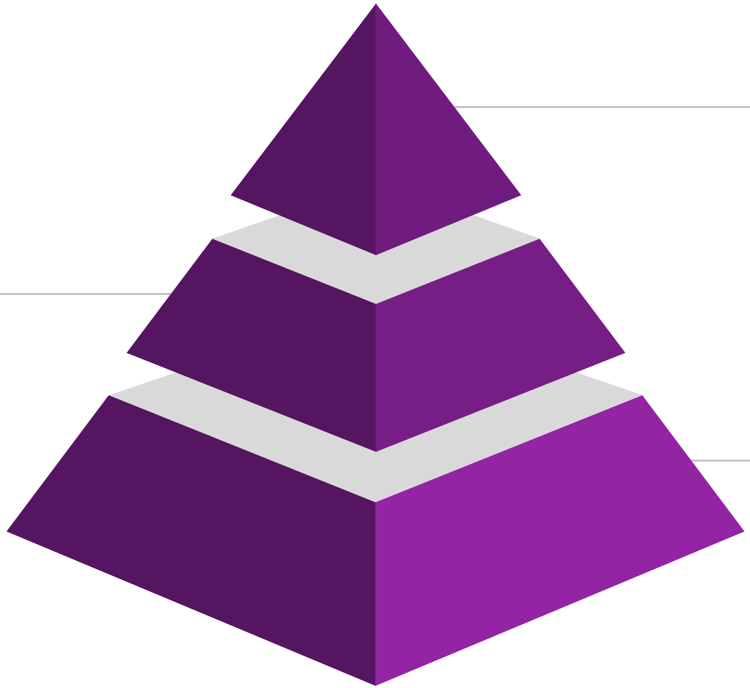
{sample prompt P, preferred answer A_W, less preferred answer A_L}

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



Alignment using RL

3

Given sample prompts and answer pairs ($A_W > A_L$), optimise the model toward predicting better answers.

Supervised Fine-Tuning (SFT)

2

Using labeled datasets on specific tasks to boost performance for particular problems.
Some of this data can be generated synthetically.

Self-supervised pre-training

1

Using massive amounts of data available on the Internet.

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.