LINGUA CUSTODIA

Training Large Scale Decoders for Multilingual Machine Translation

EUROPEAN LANGUAGE DATA SPACE (LDS) WORKSHOP

Raheel Qader

19 Juin 2024

Lingua Custodia

- Lingua Custodia is a French company that leverages **AI models** to offer various solutions for the **financial sector**.
- We provide two main solutions
  - A translation solution
  - A document analysis solution
- We also provide tools for automatic speech-to-text and a PDF table extractor solution.

Clients

AXA Investment Managers · BNP PARIBAS ASSET MANAGEMENT · caceis INVESTOR SERVICES · NATIXIS INVESTMENT MANAGERS

Rothschild & Co · Crédit Mutuel Investment Managers · HSBC · FRANKLIN TEMPLETON

HAUCK & AUFHÄUSER · LSEG · SYCOMORE ASSET MANAGEMENT · INTESA SANPAOLO

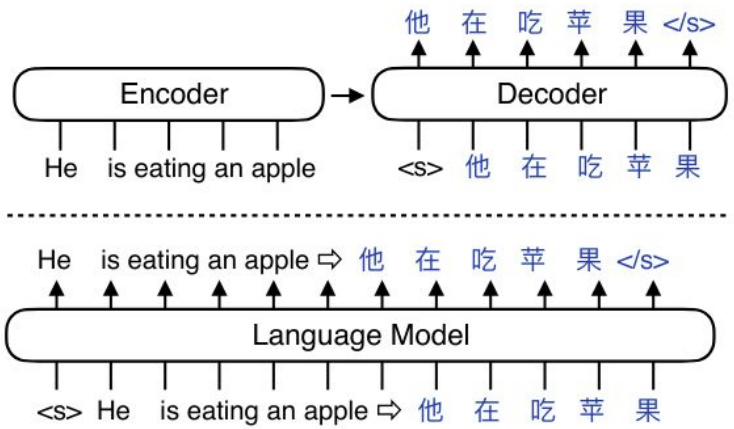
Fidelity INVESTMENTS · Kepler Cheuvreux · EURAZEO

R&D collaborations

DGA DIRECTION GÉNÉRALE DE L'ARMEMENT · Limsi · UGA Université Grenoble Alpes · cnrs · AGENCE INNOVATION DÉFENSE

Introduction

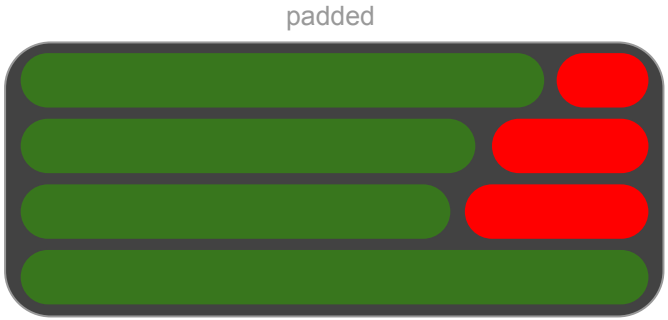
- Most modern machine translation systems are based on ***Transformer*** with an encoder-decoder architecture (introduced in 2017)
- Since then: decoder-only models have taken off thanks to LLMs



https://arxiv.org/pdf/2304.04052

Why Decoder-only Models?

- Decoder-only models on par with encoder-decoder models with a much ***simpler*** architecture
- They are ***easier*** to train on massive amounts of data



Why Decoder-only Models?

- Decoders treat all tokens similarly, while encoder-decoders make a distinction between input **source** tokens and **output** tokens.

Why Decoder-only Models?

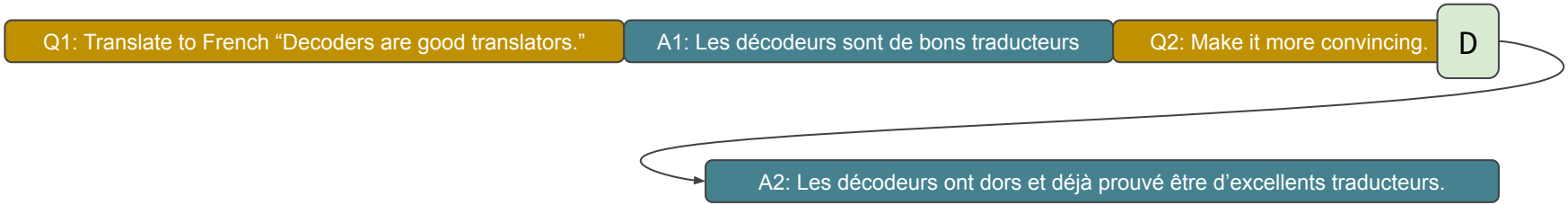
- Decoders treat all tokens similarly, while encoder-decoders make a distinction between input **source** tokens and **output** tokens.

Why Decoder-only Models?

- Decoders treat all tokens similarly, while encoder-decoders make a distinction between input **source** tokens and **output** tokens.

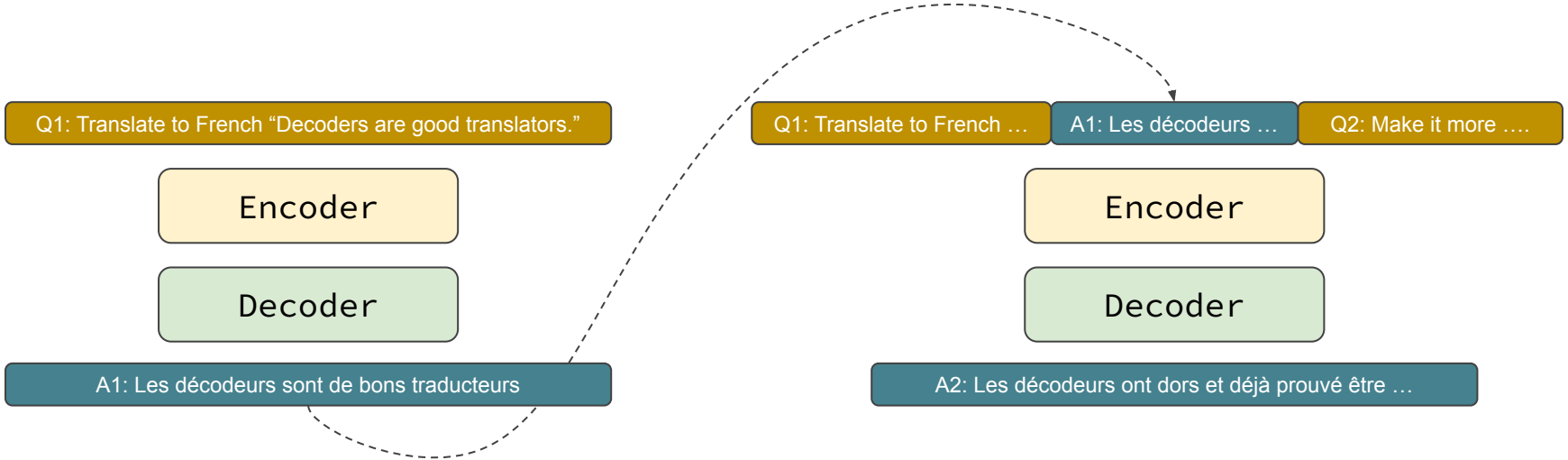
Why Decoder-only Models?

- Decoders treat all tokens similarly, while encoder-decoders make a distinction between input **source** tokens and **output** tokens.



Why Decoder-only Models?

- Decoders treat all tokens similarly, while encoder-decoders make a distinction between input **source** tokens and **output** tokens.

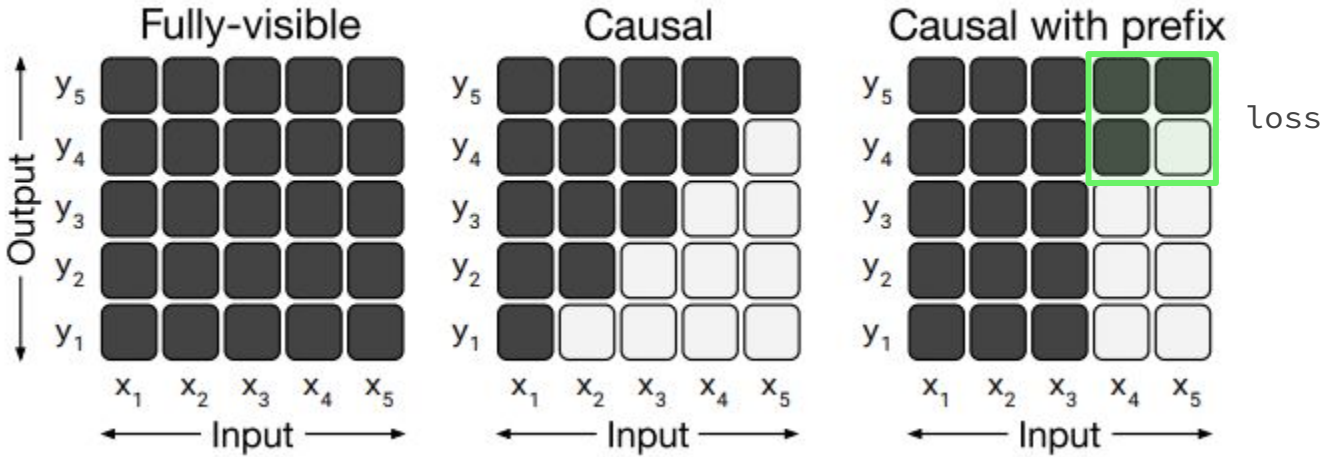


Objectives

- Train **large scale** decoder-only models for multilingual translation from scratch
- Study **scaling laws** of decoder-only models for translation

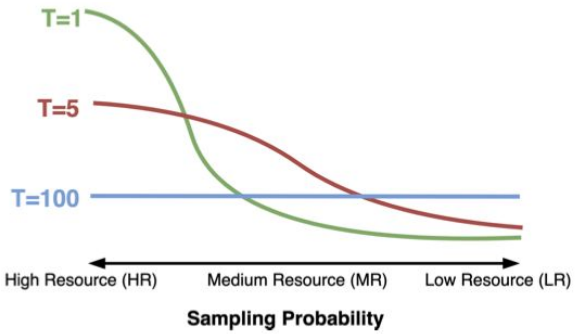
Adapting Decoder-only Models to Machine Translation

- Change causal mask to causal with prefix mask
- Calculate the loss on the target tokens only



Data

- Public datasets: CCMatrix, WikiMatrix, UN Parallel Corpus, Paracrawl and Europarl
- In-house: Lingua Financial dataset
- We applied temperature sampling, t=5



Direction	Sentences	Tokens	%
ende	47.82 M	2760.09 M	9.63
enes	53.21 M	3596.69 M	12.55
enfr	89.68 M	5925.24 M	20.67
enit	26.95 M	1693.75 M	5.91
ennl	44.10 M	2121.77 M	7.40
enpt	42.63 M	2109.17 M	7.36
ensv	46.60 M	2190.31 M	7.64
frde	25.06 M	1543.60 M	5.38
fres	33.38 M	2755.18 M	9.61
frit	29.12 M	1907.48 M	6.65
frnl	32.56 M	2063.92 M	7.20
Total:	471.11 M	28 667.20 M	100

Table 1: Distribution of the training dataset.

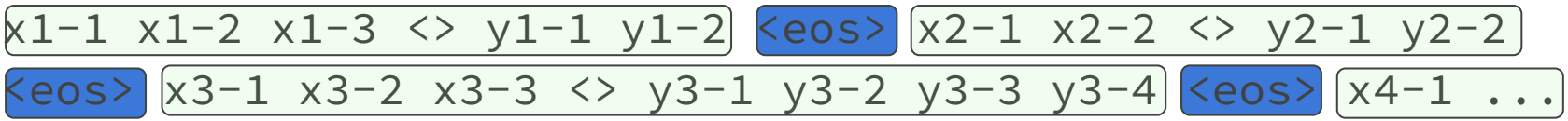
Tokenizer

- Byte-Level BPE tokenizer trained from scratch on the multilingual dataset
- 100K tokens
- Reserve a small set of special tokens to represent the supported languages

The buyer pays at an ATM. <lang_fr> <lang_en> L'acheteur effectue le paiement sur les bornes automatiques. <eos>

The <eos> Token Issue

- <eos> token is at the end of the source or beginning of target sequence?



The <eos> Token Issue

- Adding <eos> before every source sequence systematically improves the scores

Model	without <eos>	with <eos>
70M	30.80	41.11
160M	39.12	45.13
410M	40.85	46.82

Table 2: BLEU scores of the same models when sources are prefixed with and without the <eos> token.

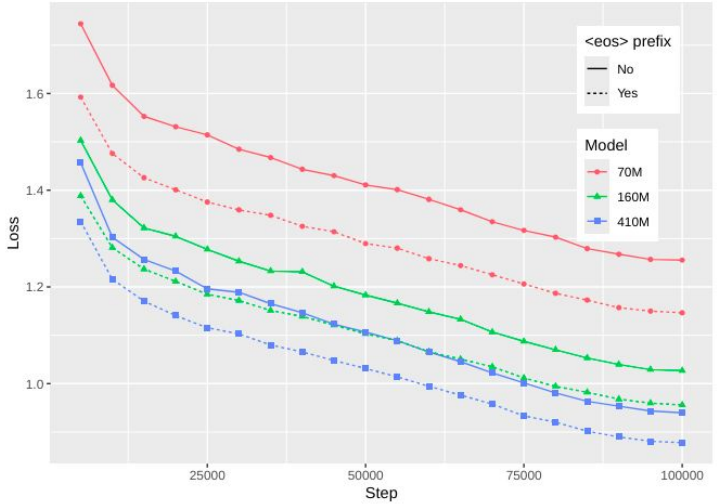


Figure 1: Test loss of three smallest models (70M, 160M and 410M) with and without <eos> prefix.

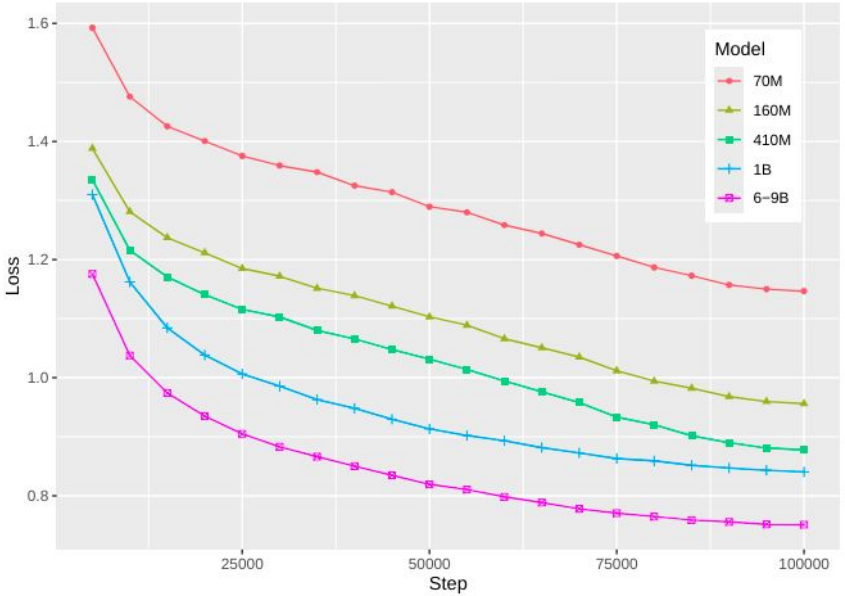
Model Architectures

- We use architectures used in the Pythia suite
- We trained the models using the GPTNeoX library
- Changed data processing scripts to ignore source tokens during the loss computation

Model	Non-embedding	Embedding	Layers	Dim	Heads	Max lr
70M	70 295 552	51 380 224	6	512	8	$1e^{-3}$
160M	162 126 336	77 070 336	12	768	16	$1e^{-3}$
410M	405 071 872	102 760 448	24	1024	16	$1e^{-3}$
1B	1 011 257 344	205 520 896	16	2048	8	$1e^{-4}$
6.9B	6 855 204 864	411 041 792	32	4096	32	$1e^{-4}$

Experiments

- Larger decoder models always converge faster and require less training data



Test loss of all model checkpoints

Experiments

- Lower loss correlate with performance increase

Model	BLEU	COMET	CometKiwi
70M	29.62	0.813	0.807
160M	32.43	0.840	0.834
410M	33.60	0.848	0.841
1B	34.42	0.851	0.843
6.9B	36.07	0.859	0.848

Table 4: Translation performances of the five models trained during this study.

Scaling Laws

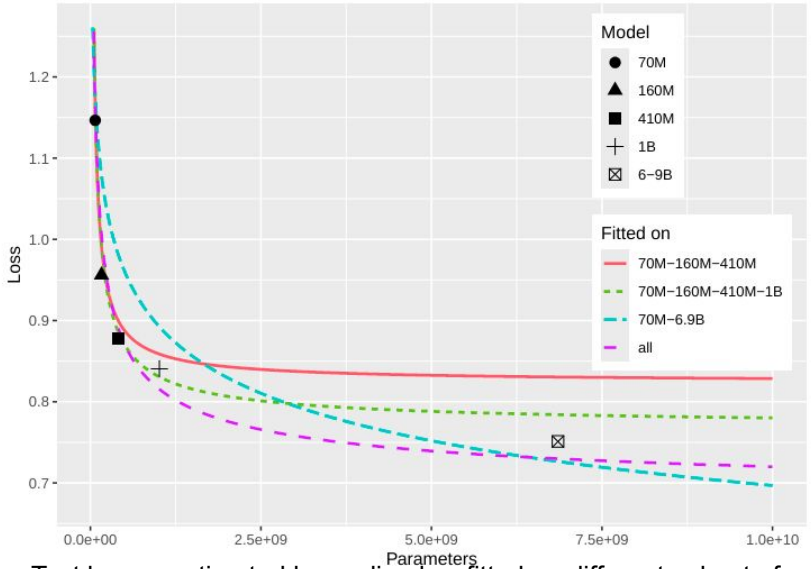
- Scaling-law studies show that larger models exhibit better generalization capabilities
- We follow the **Chinchilla** scaling law:

$$L(N, D) = E + \frac{a}{N^{\alpha}} + \frac{b}{D^{\beta}}$$

E, **a**, **α**, **b** and **β** are variable fitted using mean squared error ; N and D are: the number of non-embedding parameters and number of training samples.

Scaling Laws for Translation vs Language Modeling

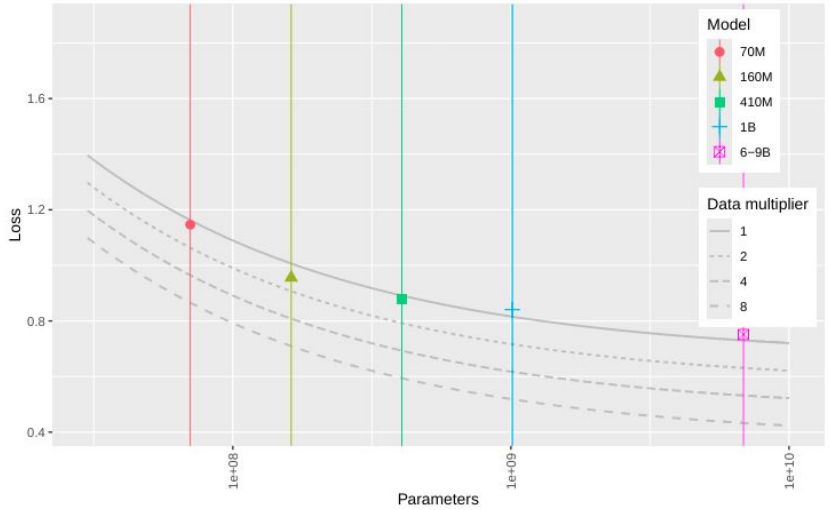
- Loss of our models can be estimated by the scaling law fitted on all observations



Test losses estimated by scaling law fitted on different subset of models.

Scaling Laws for Translation vs Language Modeling

- It is better to just train our models on more data, instead of training a larger model



Estimation of models' test losses if we trained them on more data.

Scaling Strategies

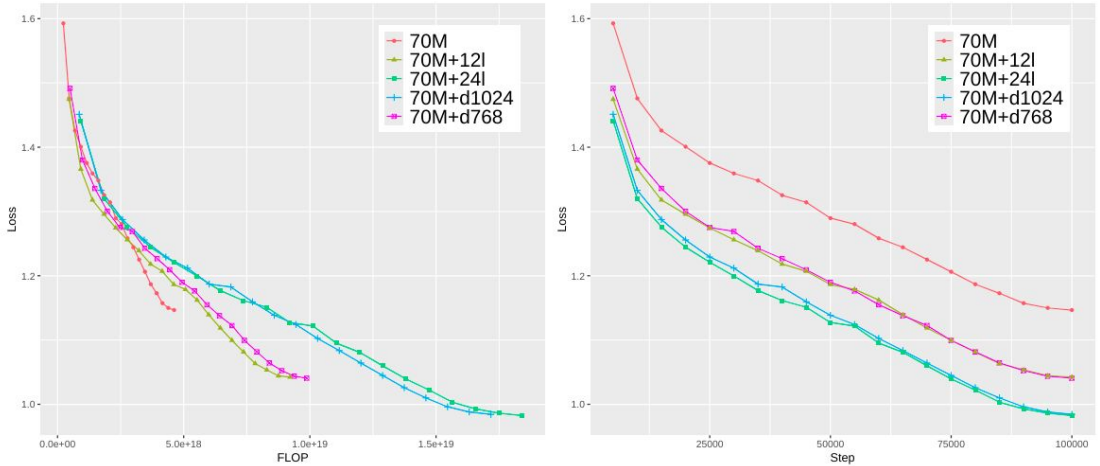
- Should we favor scaling the depth (increasing the number of layers) or the width (increasing the hidden size) of a decoder model

Model	Layers	Dim	Non-embedding	Embedding	FLOP per s.	Samples per s.
70M	6	512	70 295 552	51 380 224	1.06×10^{14}	1170
70M+d768	6	768	119 599 104	77 070 336	1.74×10^{14}	900
70M+12l	12	512	178 339 840	51 380 224	1.37×10^{14}	760
70M+d1024	6	1024	178 339 840	102 760 448	2.43×10^{14}	725
70M+24l	24	512	127 038 464	51 380 224	1.6×10^{14}	445

Sizes and architecture of models scaled in depth (70M+12l and 70M+24l) and models scaled in width (70M+d768 and 70M+d1024) compared to the base 70M model.

Scaling Strategies

- Should we favor scaling the depth (increasing the number of layers) or the width (increasing the hidden size) of a decoder model



We increased the width and the depth of the 70M model so the additional cost in terms of FLOP is similar (left). Scaling the depth or the width can lead to similar performance gains (right).

Next!

- Train larger models
- Fine-tune large scale pre-trained models on MT and related tasks
- Use RLHF to guide the model better follow instructions

Questions!