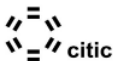


MICROCREDENCIAL ALGORITMOS VERDES PARA LA IA DISEÑO E IMPLEMENTACIÓN

Tema: Natural language processing

Sesión 1: Green NLP Today

Ponente: Olga Zamaraeva



UNIVERSIDADE DA CORUÑA



Financiado por la
Unión Europea
NextGenerationEU



GOBIERNO
DE ESPAÑA

MINISTERIO
PARA LA TRANSFORMACIÓN DIGITAL
Y DE LA FUNCIÓN PÚBLICA



2026



LET'S START WITH A SENTENCE

*Unha **tía miña**, que todo o solucionaba con ditos e refráns, dicía algo
así como*

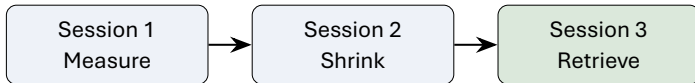
“un é escravo do que di e dono do que cala”

from “Speechwriter” by Paulo Naseiro

MODULE OVERVIEW

Three sessions, one running project

- Session 1: **Measure** the baseline
- Session 2: **Shrink** it — quantization
- Session 3: **Retrieve** on demand

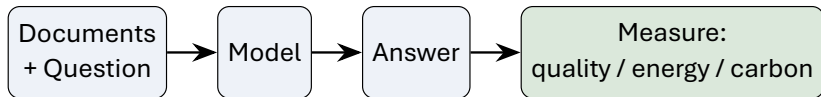


olga.zamaraeva@udc.es • UDC Teams • CITIC, 2nd floor

THE RUNNING PROJECT: GREEN NLP BENCHMARK LAB

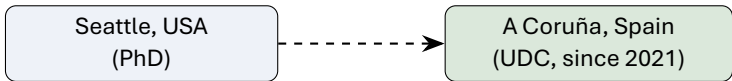
One pipeline, one change at a time

- Same pipeline, all three sessions — change **one** thing, re-measure
- Quality vs. energy vs. carbon — your own leaderboard



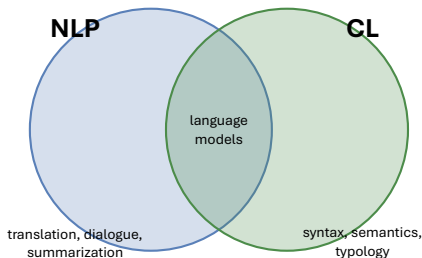
INSTRUCTOR'S PROFILE

- Computational linguist — syntax and machine translation

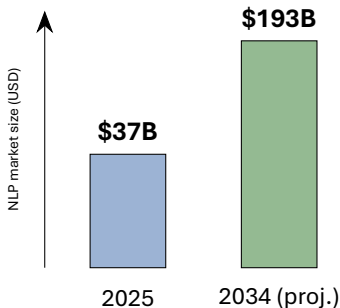


NATURAL LANGUAGE PROCESSING & COMPUTATIONAL LINGUISTICS

- NLP: building systems that *do* things with language
- CL: using computation to *study* language



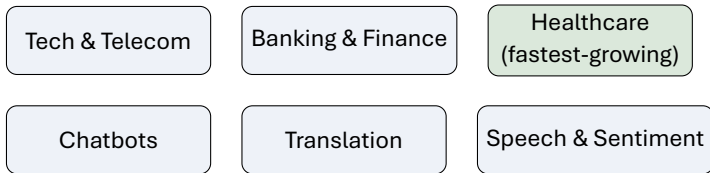
HOW BIG IS NLP, REALLY?



- ~20% annual growth — but estimates vary widely across firms (as low as \$25B for the same year, elsewhere)

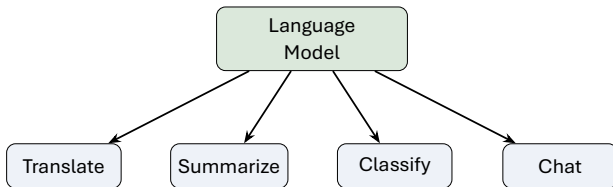
Source: Fortune Business Insights, 2025

WHERE NLP SHOWS UP



ONE ENGINE NOW RUNS MANY TASKS

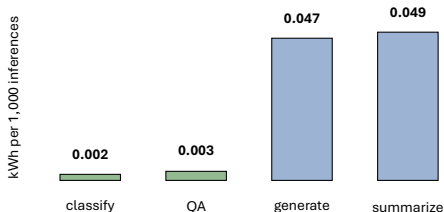
separate models, one per task



one model, prompted — this is where “NLP” and “AI” blur

Source: Radford et al., “Language Models are Unsupervised Multitask Learners,” 2019; Brown et al., “Language Models are Few-Shot Learners,” NeurIPS 2020

NLP TASKS: WHAT THEY ACTUALLY COST

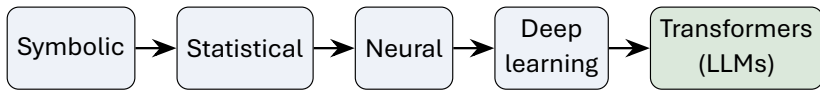


- Discriminative tasks (classify, answer) are cheap; generative tasks (write, summarize) cost 15–20 $\times$ more
- Image generation: 2.9 kWh per 1,000 — about 1,450 $\times$ the cheapest text task

Source: Luccioni, Jernite & Strubell, FAccT 2024 (arXiv:2311.16863), Table 2

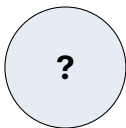
HOW WE GOT HERE

- Symbolic: rules written by hand (1950s–80s) → Statistical: counting from real text (1990s–2000s) → Neural: learned representations, more data and compute needed (2010s+) → Transformers: attention-based (2017+)
- Each shift trades hand-engineering for more data and compute — directly relevant to a course about the cost of that compute

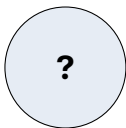


BEFORE WE START: GUESS THE FOOTPRINT

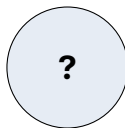
1. How many metric tons of CO₂eq did training GPT-3 emit?
2. What share of global electricity do data centers consume today?
3. How many times more carbon does a general-purpose LLM emit per query than a chatbot trained and built just for that task?



training
emissions



global
electricity



task-specific vs.
general-purpose

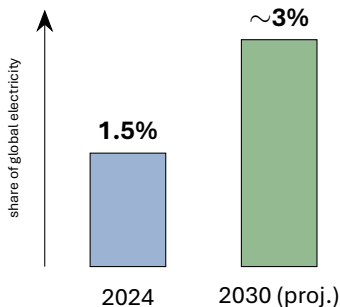
REVEAL 1: TRAINING A LARGE MODEL

552 t CO₂eq $\approx$ 
3 $\times$ SF $\leftrightarrow$ NYC round trips

- A *one-time* cost — paid once, by whoever trains the model

Source: Patterson et al., “Carbon Emissions and Large Neural Network Training,” 2021 (arXiv:2104.10350)

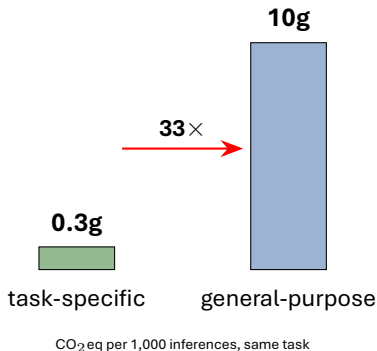
REVEAL 2: DATA CENTERS, GLOBALLY



- Growing 17% in just 2025 — driven mainly by AI

Source: International Energy Agency, “Energy and AI,” 2024–2025

REVEAL 3: GENERAL-PURPOSE LLMS ARE COSTLY

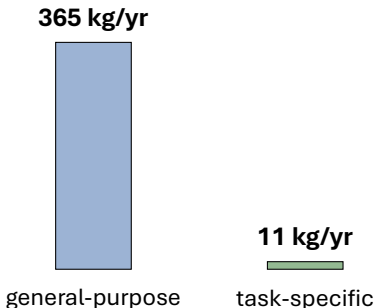


- Same task, 33 $\times$ apart

Source: Luccioni, Jernite & Strubell, FAccT 2024 (arXiv:2311.16863)

EXAMPLE: THE 33 $\times$, AT DEPLOYMENT SCALE

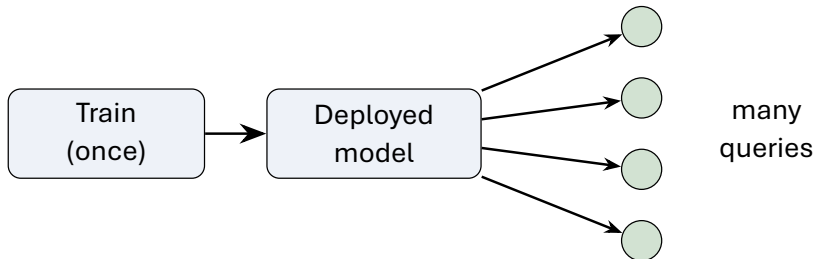
- A chatbot answering **100,000 questions/day**, general-purpose LLM API
- Anchor: 10g CO₂eq / 1,000 inferences (Reveal 3)



- $100k/day \times 10g/1,000 \approx 365 \text{ kg/year}$ — about a month of driving a car
- Swap to task-specific (Reveal 3's 33 $\times$) $\rightarrow \approx 11 \text{ kg/year}$

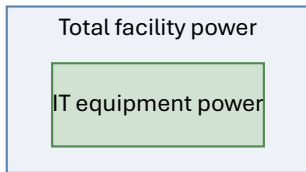
TRAINING VS. INFERENCE

- Train **once**. Infer **millions of times**
- Cumulative inference cost dominates a model's lifetime footprint



VOCABULARY FOR THIS COURSE

- **FLOPs** (floating-point operations): how much computation
- **Carbon intensity**: CO₂eq per unit of electricity — depends on *where/when*
- **PUE** (Power Usage Effectiveness): facility power $\div$ IT equipment power
- **Operational** vs. **embodied** carbon: running it vs. building it



$$\text{PUE} = \text{facility power} \div \text{IT equipment power}$$

CASE STUDY: A FAMOUS NUMBER, REVISITED



“= 5 cars, lifetime emissions”

- One 2019 paper, number repeated everywhere for years — later shown $\sim 10\times$ too high (actual number 10x smaller)

Source: Strubell et al., ACL 2019

Activity: Class discussion (10 min)

- Why is estimating carbon footprint from a paper so hard?
- What would you need to *trust* a number like this?
- Catchy comparisons: help, or hurt?



DISCUSSION: SOME POSSIBLE ANSWERS

- **Hard because:** hardware, PUE, run time, grid — rarely all reported
- **To trust it:** the full method, ideally reproducible code
- **Catchy comparisons:** memorable — risky if the number is wrong

BACK TO THAT SENTENCE

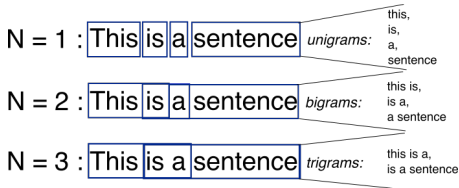
*Unha **tía miña**, que todo o solucionaba con ditos e refráns, dicía algo así como
“un é escravo do que di e dono do que cala”*

an aunt's saying, from the novel *Speechwriter* by Paulo Naseiro

- Galician: a **low-resource** language
- Does a huge general model actually handle a case like this, or does it rely on what it has plenty of data for — Spanish, Portuguese, English?
- But... what's a “model”?

WHAT IS A LANGUAGE MODEL?

- Predicts the next word — a full probability distribution over the vocabulary
- N-grams: the simplest version



Credit: Jurafsky & Martin, *Speech and Language Processing*

Interactive: [N-grams, step by step](#)

N-GRAMS: A WORKED EXAMPLE

my aunt stayed quiet • my aunt spoke wisely • my
uncle stayed quiet

$$P(w_i | w_{i-1}) = \frac{\text{Count}(w_{i-1} w_i)}{\text{Count}(w_{i-1})}$$

- $P(\text{aunt} | \text{my}) = 2/3 \approx 0.67$ $P(\text{uncle} | \text{my}) = 1/3 \approx 0.33$
- Counting, then dividing — that's the whole model
- A Galician model has far fewer sentences to count from than a Spanish or English one — noisier estimates

Activity: N-grams Playground (~15 min, individual)

- Open `01b_ngrams_playground.ipynb` — CPU only, no GPU needed
- Part A: rebuild this exact bigram model yourself, then generate text with it
- Part B: use it to fix the word order of a broken word-for-word translation — a preview of Notebook 1's “statistical” approach



FROM COUNTS TO VECTORS

- Count co-occurrences instead of sequences → a vector per word

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	1	8	15
soldier	2	2	12	36
fool	37	58	1	5
clown	6	117	0	0

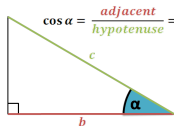
COMPARING VECTORS: COSINE SIMILARITY

$$\cos(u, v) = \frac{u \cdot v}{\|u\| \|v\|}$$

- Toy example: aunt = (4, 2), uncle = (3, 3), car = (1, 4)
- $\cos(\text{aunt}, \text{uncle}) \approx 0.95$ $\cos(\text{aunt}, \text{car}) \approx 0.65$
- Dividing by length makes this about *direction* alone

$$\text{dot-product}(\vec{v}, \vec{w}) = \vec{v} \cdot \vec{w} = \sum_{i=1}^N v_i w_i = v_1 w_1 + v_2 w_2 + \dots + v_N w_N$$

$$\begin{aligned}\vec{a} \cdot \vec{b} &= |\vec{a}| |\vec{b}| \cos \theta \\ \frac{\vec{a} \cdot \vec{b}}{|\vec{a}| |\vec{b}|} &= \cos \theta\end{aligned}$$



FROM WORDS TO VECTORS: EMBEDDINGS

- **Learned, dense** vectors — every dimension has a value
- Similar contexts → similar vectors — same cosine idea, learned this time



same word, two representations

x_1 

Je

x_2 

suis

x_3 

étudiant

Credit: jalamar.github.io/illustrated-transformer/

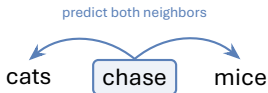
HOW EMBEDDINGS ARE ACTUALLY LEARNED

Skip-gram (Mikolov et al., 2013): predict the neighbors

A tiny corpus

cats chase mice
dogs chase balls
cats eat fish
dogs eat meat

Window = 1



16 training pairs

input → word to predict

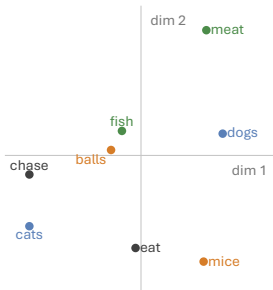
cats	→ chase	cats	→ eat
chase	→ cats	eat	→ cats
chase	→ mice	eat	→ fish
mice	→ chase	fish	→ eat
dogs	→ chase	dogs	→ eat
chase	→ dogs	eat	→ dogs
chase	→ balls	eat	→ meat
balls	→ chase	meat	→ eat

- The task: given *cats*, give high probability to *chase* and *eat*
- The task never mentions word vectors. The vectors appear because the model needs them to make predictions.

THE MODEL'S PARAMETERS: A TABLE OF NUMBERS

One row per word. It starts random.

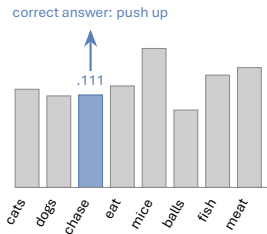
word	dim 1	dim 2
cats	-0.41	-0.26
dogs	0.30	0.08
chase	-0.41	-0.07
eat	-0.02	-0.34
mice	0.23	-0.39
balls	-0.11	0.02
fish	-0.07	0.09
meat	0.24	0.46



- Two dimensions so we can draw it. word2vec used 300.
- Color = same neighbors in the corpus. The model never sees the colors.

ONE TRAINING STEP: CATS $\rightarrow$ CHASE

Model's prediction: $P(\cdot \mid \text{cats})$



Training pair: cats $\rightarrow$ chase

Training nudges the parameters so that $P(\text{chase} \mid \text{cats})$ grows.

One of the nudged rows is *cats*:

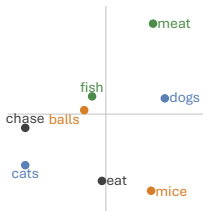
before: $(-0.414, -0.263)$

after: $(-0.384, -0.261)$

$P(\text{chase} \mid \text{cats}): 0.111 \rightarrow 0.115$

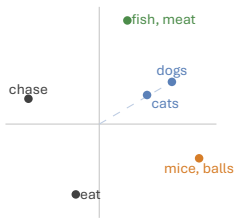
- Then: cats $\rightarrow$ eat, dogs $\rightarrow$ chase, dogs $\rightarrow$ eat, ... all 16 pairs, thousands of times

THE SAME TABLE, REDRAWN DURING TRAINING



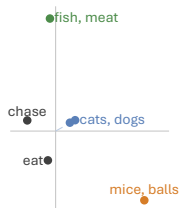
start

axes: -0.5 to 0.5



100 passes

axes: -2 to 2.5



3000 passes

axes: -2 to 5

- One pass = all 16 training pairs once
- Same color → same direction from the origin. Nobody told the model which words belong together.

SIMILAR COMPANY, SIMILAR VECTORS

The company each word keeps

cats	chase, eat
dogs	chase, eat
mice	chase
balls	chase
fish	eat
meat	eat

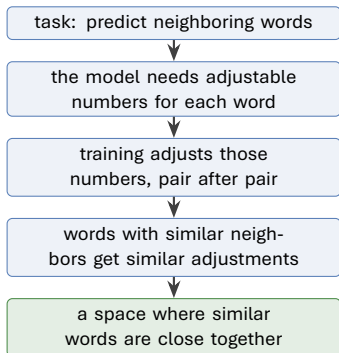
The table after training

cats	0.63	0.37
dogs	0.86	0.48
mice	3.89	-3.15
balls	3.93	-2.94
fish	-0.34	5.00
meat	-0.16	4.91

- Same neighbors $\rightarrow$ same nudges, pair after pair $\rightarrow$ same direction
- $\cos(\text{cats}, \text{dogs}) = 1.00$ $\cos(\text{cats}, \text{fish}) = 0.44$
- Task solved: $P(\text{chase} \mid \text{cats}) 0.11 \rightarrow 0.47$, $P(\text{eat} \mid \text{cats}) 0.12 \rightarrow 0.52$

“You shall know a word by the company it keeps.” — J. R. Firth (1957)

WHAT WE KEEP: THE EMBEDDING TABLE



- After training, discard the prediction part and **keep the table**. Its rows are the **word embeddings**.
- Embeddings = the parameters the model uses to solve its prediction task
- Rare word → few pairs → few nudges → poorly placed vector
- Less Galician text online → weaker embeddings for many Galician words

QUICK REFRESHER: MULTIPLYING MATRICES

- To multiply $A (m \times n)$ by $B (n \times p)$: the **inner** dimensions must match; the result is $m \times p$
- Each entry of the result is a **dot product**: multiply matching entries, add them up

$$\begin{pmatrix} 1 & 2 & 0 \\ 3 & 1 & 4 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 1 & 3 \\ 0 & 2 \end{pmatrix} = \begin{pmatrix} 4 & 6 \\ 7 & 11 \end{pmatrix}$$

top-left entry: $1 \times 2 + 2 \times 1 + 0 \times 0 = 4$

- A vector is just a matrix with one row — and a **one-hot** vector (all zeros but a single 1) does something special:

$$\begin{pmatrix} 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0.1 & 0.9 \\ 0.3 & 0.2 \\ 0.7 & 0.5 \\ 0.4 & 0.8 \end{pmatrix} = \begin{pmatrix} 0.7 & 0.5 \end{pmatrix}$$

the 1 in position 3 copies out row 3 — every other row gets multiplied by 0

INSIDE SKIP-GRAM: WHAT ACTUALLY GETS MULTIPLIED

- p : one row per vocabulary word — the input's one-hot vector just **selects a row**; that row is h , the word's current embedding
- p' : turns h into a predicted probability distribution over the vocabulary, for each nearby position
- Training adjusts p and p' together until those predictions match real text

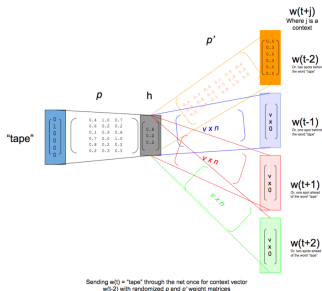


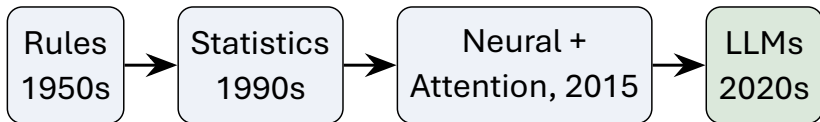
Illustration: district-data-labs, "Forward Propagation: Building a Skip-Gram Net from the Ground Up" (Medium)

Activity: Embeddings Playground (~15–20 min, individual)

- Open `01c_embeddings_playground.ipynb` — CPU only, no GPU needed
- Part A: real word embeddings, real cosine similarity, a 2-D map of related words
- Part B: do Galician, Spanish, and Portuguese words for “aunt” really look distinct to a multilingual model?



MACHINE TRANSLATION: A VERY SHORT HISTORY



- One of AI's oldest goals — first public demo: 1954 (Georgetown–IBM)
- 1966: the ALPAC report cuts U.S. funding for over a decade
- 2006: Google Translate, built on statistical MT
- Galician is living through this same arc right now

Source: Hutchins, "Machine Translation: A Concise History," 2007

ATTENTION WAS BORN IN MACHINE TRANSLATION

- Early neural MT: whole sentence $\rightarrow$ **one vector** — a bottleneck for long sentences
- Bahdanau, Cho & Bengio (2015): let the decoder look back, word by word — **attention**
- Invented for translation, years before today's LLMs

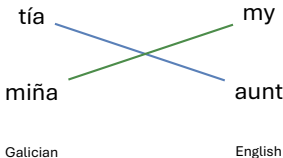
*Unha **tía miña**, que todo o solucionaba con ditos e refráns, dicía algo así como “un é escravo do que di e dono do que cala”.*

Source: Bahdanau, Cho & Bengio, ICLR 2015 (arXiv:1409.0473)

Interactive: [Attention, step by step](#)

THE ALIGNMENT PROBLEM IN TRANSLATION

Unha *tía miña*, que todo o solucionaba con ditos e refráns, dicía algo así como “un é escravo do que di e dono do que cala”.



- The aunt is introduced as “*tía miña*” — noun before possessive, the reverse of English
- Attention learns this reordering automatically, no separate step

ATTENTION SCORES: HOW MUCH SHOULD I LOOK HERE?

- **Query:** what am I looking for? **Key:** what do I offer?
- $\text{Score} = q \cdot k$, then **softmax:** weights that sum to 1
- **Value:** the actual content — weighted average, using those weights

WORKED EXAMPLE: ALIGNING “TÍA MIÑA” → “MY AUNT”

Unha **tía miña**, que todo o solucionaba con ditos e refráns, dicía algo así como “un é escravo do que di e dono do que cala”.

$$k_{\text{tía}} = (1, 0) \quad k_{\text{miña}} = (0, 1)$$

To generate aunt, query $q_{\text{aunt}} = (3, 0)$:

$$\text{scores: } 3, 0 \quad \Rightarrow \quad \text{softmax: } 0.95, 0.05$$

To generate my, query $q_{\text{my}} = (0, 3)$:

$$\text{scores: } 0, 3 \quad \Rightarrow \quad \text{softmax: } 0.05, 0.95$$

- 95% attention on tía for aunt; 95% on miña for my — a clean word-level correspondence, learned from data

Activity: Attention Playground (~15 min, individual)

- Open `01d_attention_playground.ipynb` — CPU only, no GPU needed
- Part A: rebuild this exact worked example yourself, verify the 95/5 split
- Part B: pull real attention weights out of a real translation model, on our own sentence, and see how they compare



WHAT ALIGNMENT ACTUALLY GIVES YOU

Unha tía miña, que todo o solucionaba con ditos e refráns, dicía algo así como “un é escravo do que di e dono do que cala”.

- Word-for-word, attention gives you something perfectly fluent:
“**my aunt**”
- “Unha tía miña,” with that indefinite article, is introducing a *brand-new* character — “my aunt” implies one you already know
- A translator actually writes: “**I had an aunt who...**”

SELF-ATTENTION: ALIGNING A SENTENCE TO ITSELF

- Vaswani et al. (2017): the same mechanism, *within* one sentence — also at work inside the English translation of our story itself
- “It” → “the animal” — no hand-written rule needed

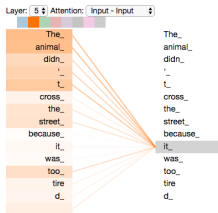
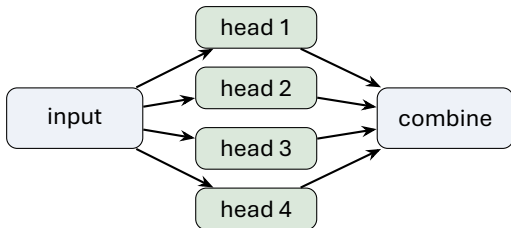


Illustration: [jalammar.github.io/illustrated-transformer/](https://github.com/jalammar/illustrated-transformer/) Source: Vaswani et al., NeurIPS 2017 (arXiv:1706.03762)

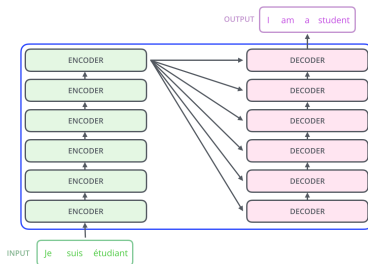
MULTIPLE ALIGNMENTS AT ONCE: MULTI-HEAD ATTENTION

- Several attention patterns in parallel, in different “heads”
- In our sentence, one head might track *tía-miña*; another, word order elsewhere in the passage
- Combined into one output per word



INSIDE A TRANSFORMER: STACKED BLOCKS

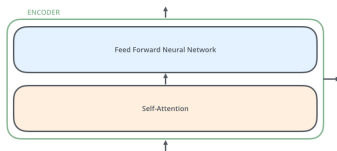
- Same kind of block (attention + feed-forward) repeated many times
- This picture is the classic translation shape — **encoders** on the left reading the source sentence, **decoders** on the right generating the output
- GPT-3 and most general-purpose LLMs use only the decoder side, alone — 96 blocks deep, no separate encoder at all



Credit: jalammar.github.io/illustrated-transformer/

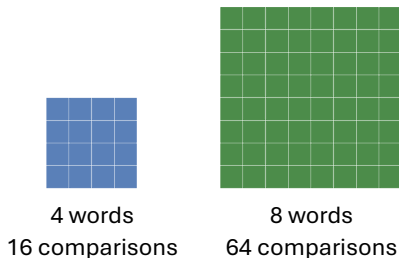
INSIDE ONE BLOCK

- Two sublayers, always in this order: **self-attention**, then a **feed-forward network**
- Self-attention mixes information across positions; the feed-forward layer does not — it runs on each word's vector separately
- This specific shape is an **encoder** block — **decoder** blocks add one more piece: **cross-attention** into the encoder's output



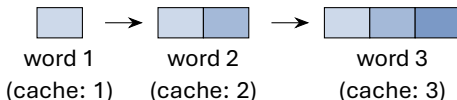
Credit: jalammar.github.io/illustrated-transformer/

WHY ATTENTION COST GROWS WITH CONTEXT LENGTH



- Double the words $\rightarrow$ four times the comparisons — growth is quadratic
- “tía miña” was two words. The full story is thousands

GENERATING TEXT: WHY INFERENCE IS SEQUENTIAL



- One word at a time — the **KV cache** grows with every word generated
- Translating the rest of our story this way: cache growing the whole time

WHY DOES MODEL SIZE COST SO MUCH?

Challenges with RNNs	Transformers
• Long range dependencies • Gradient vanishing and explosion • Large # of training steps • Sequential/recurrence → can't parallelize • Complexity per layer: $O(n \cdot d^2)$	• Can model long-range dependencies • No gradient vanishing and explosion • Fewer training steps • Can parallelize computation! • Complexity per layer: $O(n^2 \cdot d)$

- More parameters, more heads, more blocks → more compute per query
- Longer input → quadratically more attention cost
- **Size and context length are levers with a real, computable energy cost**

WHAT DID LLMS BRING TO MACHINE TRANSLATION?

- One model, 100+ languages — no separate model per pair
- Zero-shot, and **controllable** by instruction
- But: usually far more expensive per translation than a dedicated MT model

HOW WELL DO LLMS ACTUALLY SERVE GALICIAN?

- In mainstream LLMs, Galician's presence is “almost residual” — close enough to Spanish and Portuguese that models blur the three together
- Fix: a **smaller**, dedicated model (1.3B params), trained on real Galician text, that beat the giant general model at actually producing correct Galician
- Even bias research mostly skips languages like this: 2 papers out of dozens reviewed

Source: Gamallo et al., “A Galician-Portuguese Generative Model,” EPIA 2024; Proxecto Nós (USC)

IS THERE “GREEN MT” RESEARCH?

140× faster 98.7% less energy

for a small quality trade-off

- That’s an MT-specific result — Guta, Petrick & Polák, EMNLP Findings 2025
- The 2019 “5 cars” number itself came from a machine-translation architecture search
- Quantization for MT, specifically: Shterionov & Vanmassenhove, 2022 — Session 2’s topic

THE LEVERS WE'LL ACTUALLY USE

- **Precision** (Session 2) **Retrieval** (Session 3) **Task-specificity**
(today, twice over)



Precision



Retrieval

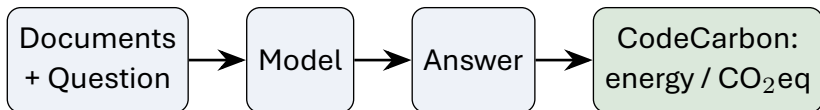


Task-specific

- Today, we build the **baseline** the other two get compared against

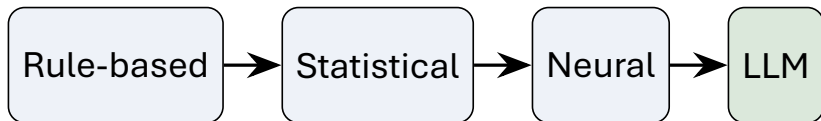
NOTEBOOK 1: BASELINE MEASUREMENT

- Task: answer questions about a small set of documents (about Green NLP itself)
- Instrumented with **CodeCarbon**: energy, CO₂eq, quality — one row on your leaderboard



NOTEBOOK 1, CONTINUED: FOUR WAYS TO TRANSLATE, THREE WAYS TO GRADE IT

- Part 2: the same sentence from this morning, through **rule-based**, **statistical**, **neural**, and **LLM** translation — the whole history, hands-on
- Part 3: **BLEU**, **BERTScore**, and an **LLM-as-judge** score the results — and we measure what grading itself costs



Activity: Notebook 1 (60–75 min baseline, +40–60 min Parts 2–3, individual)

- Open `01_baseline_measurement.ipynb` in Google Colab
- Runtime → Change runtime type → T4 GPU
- Part 1 first, then Parts 2–3 if time allows
- Stretch goal (optional): swap in a bigger model and compare



SHARE-OUT

- Part 1: how much energy/ CO_2 eq did your run measure? Same model, different hardware — did it matter?
- Parts 2–3: how many orders of magnitude separated your cheapest and priciest approach — and did the three MT metrics even agree?
- Next time: we quantize this exact model



you



neighbor



**GRACIAS POR SU
ATENCIÓN**

INDITEXUDC
Cátedra de IA en Algoritmos Verdes

<https://catedrainditexalgoritmosverdes.com>