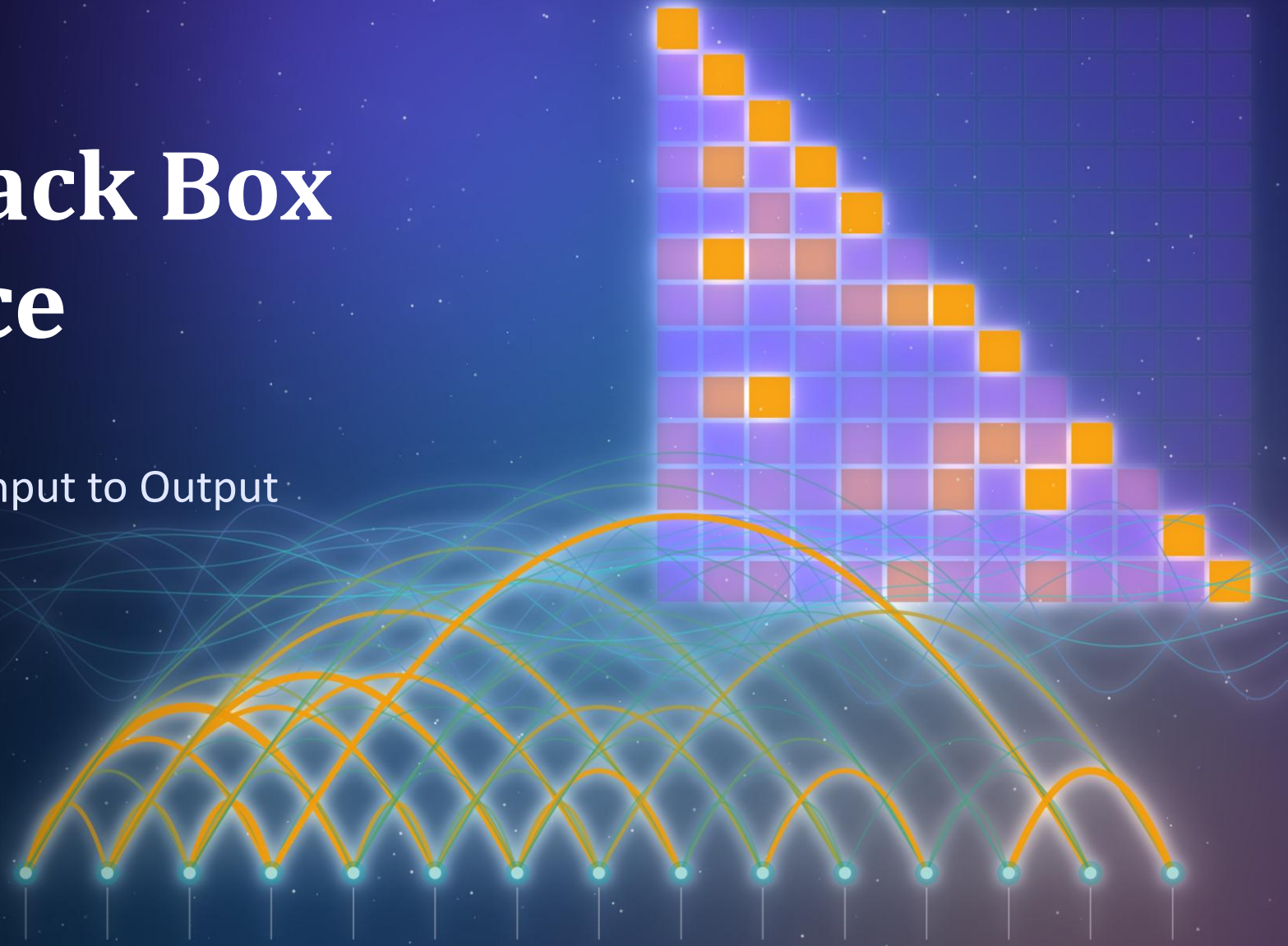
© Aridio Silva · @aridiosilva · October 2026 · ORCID 0009-0008-2411-6995

PAPER · INFOGRAPHIC

Opening the Black Box of LLM Inference

Anatomy of the Transformer, from Input to Output Token



Aridio Silva
Independent Researcher · @aridiosilva
ORCID 0009-0008-2411-6995

© Aridio Silva · @aridiosilva · October 2026

Roadmap

From text to token: 9 steps

1 **Tokenization**

Text becomes numbers (IDs)

2 **Embeddings + position**

IDs become vectors with order

3 **Self-attention**

Each token consults the others

4 **Multi-head + masks**

Several relations in parallel

5 **Add & Norm**

Stability with depth

6 **FFN**

Non-linear per-token transform

7 **Stacking**

Layer-by-layer refinement

8 **Weights and training**

What the model has learned

9 **Output**

Logits → softmax → word

© Aridio Silva · @aridiosilva · October 2026

What this paper answers

> *“O banco estava lotado”*
>
> How does the model decide, without explicit rules, whether “banco” is a bank branch or a bench?

Goal

Open the “black box”: concept, mechanism, technique, numerical example and references for every inference step.

Conventions

n = sequence length; d_model = model dimension; d_k = per-head dimension. References as (Author, year).

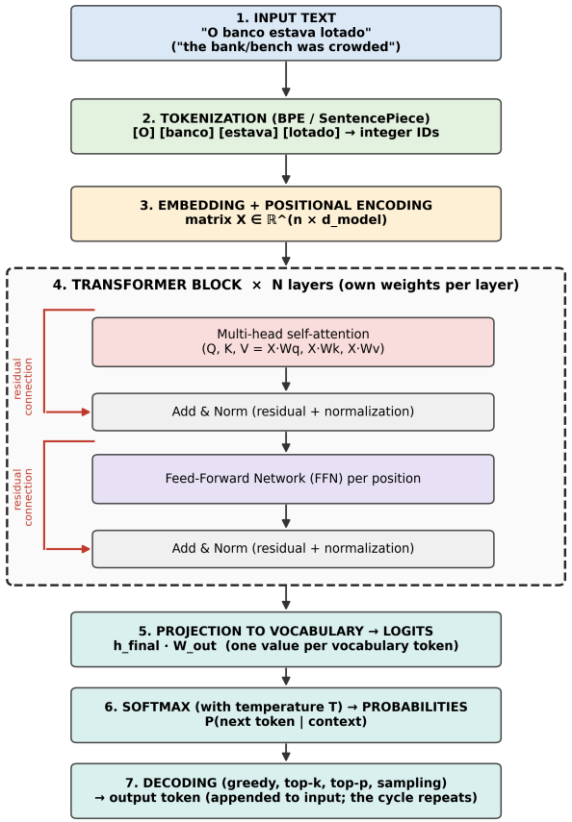
Numerical notice

Numbers are didactic: tiny dimensions, hand-set or random weights, computed by code. They are not measurements of a real model.

© Aridio Silva · @aridiosilva · October 2026

The complete inference flow

Figure 1 – Full inference flow in a Transformer (overview)



© Aridio Silva · @aridiosilva · October 2026 · ORCID 0009-0008-2411-6995

- **Inference:** running the trained model, with frozen weights, to produce an output.
- **Autoregressive loop:** predicts the next token, picks one, appends it to the input and repeats.
- **Forward pass:** one full pass through the layers.
- **Families:** encoder-only (BERT), decoder-only (GPT) and encoder-decoder (original).
- **Training ≠ inference:** training sets the weights; inference only uses them.

Steps, techniques and references

Step	Main technique	Key reference
1. Tokenization	BPE, SentencePiece	Sennrich et al., 2016; Kudo & Richardson, 2018
2. Embedding and position	Embedding matrix; sinusoidal or RoPE	Mikolov et al., 2013; Vaswani et al., 2017; Su et al., 2021
3. Self-attention	Scaled dot product and softmax	Bahdanau et al., 2015; Vaswani et al., 2017
4. Multi-head	d_k-dimensional subspaces	Vaswani et al., 2017; Voita et al., 2019
5. Add & Norm	Residual; LayerNorm, RMSNorm	He et al., 2016; Ba et al., 2016; Zhang & Sennrich, 2019
6. FFN	MLP with ReLU, GELU or SwiGLU	Geva et al., 2021; Shazeer, 2020
7. Stacking	N layers	Tenney et al., 2019
8. Output	Softmax, temperature, top-k, top-p	Holtzman et al., 2020

© Aridio Silva · @aridiosilva · October 2026

Problem solved and cost of absence (1/2)

Step	Problem it solves	Without it
Tokenization	The network only handles numbers; the vocabulary must be finite	No numeric input; rare words unrepresented or sequences too long
Embedding	Integer IDs are arbitrary labels, with no similarity	Sparse one-hot of size $\|V\|$, no similarity between tokens
Positional encoding	Attention does not use word order	Unmasked attention cannot tell orders apart (Vaswani et al., 2017)
Self-attention	Each token needs information from the others	Each token read in isolation: “banco” would always have the same vector
Multi-head	Several relations at once	A single weighted average per layer (Vaswani et al., 2017)

© Aridio Silva · @aridiosilva · October 2026

Problem solved and cost of absence (2/2)

Step	Problem it solves	Without it
Add & Norm	Signal and gradient cross many layers	Unstable or degraded training (He et al., 2016; Ba et al., 2016)
FFN	Attention only mixes Values; no per-token transform	Missing ~2/3 of layer parameters (Geva et al., 2021)
Stacking	Complex relations need step-wise composition	Only direct relations; induction heads need ≥2 layers (Olsson et al., 2022)
Training	Random weights do not produce language	A circuit with no useful behaviour, as in the examples
Output	A vector must become a token	No distribution and no chosen word

© Aridio Silva · @aridiosilva · October 2026

3 Step 1 · Input

Tokenization: text becomes numbers

1 Concept

Text becomes a token sequence and each token an integer ID. The vocabulary is finite and fixed before training.

2 Why subwords

Whole words leave new terms unrepresented; characters make long sequences. Subwords balance both.

3 Why it is needed

Without IDs there is no network input; long sequences make attention costly (cost $\propto n^2$).

Guiding sentence, 4 words = 4 tokens (illustrative IDs)

O	banco	estava	lotado
→ 12	→ 4871	→ 905	→ 23019

Tokenization does not disambiguate: the ID of "banco" is the same in both senses.

© Aridio Silva · @aridiosilva · October 2026

Four subword techniques

1 BPE

Starts from characters and merges the most frequent adjacent pair. Compression: Gage (1994); MT: Sennrich et al. (2016). Byte-level in GPT.

2 WordPiece

Picks merges by likelihood gain rather than frequency (Schuster & Nakajima, 2012; Wu et al., 2016). Used in BERT.

3 Unigram LM

Starts from a large vocabulary and prunes the least useful items (Kudo, 2018).

4 SentencePiece

Implements BPE and Unigram on raw text, without whitespace pre-tokenization; language-independent (Kudo & Richardson, 2018).

© Aridio Silva · @aridiosilva · October 2026

BPE on a didactic corpus

Corpus: low (5), lower (2), newest (6), widest (3); </w> marks the end of a word.

Step	Merged pair	Freq.	New symbol
1	e + s	9	es
2	es + t	9	est
3	est + </w>	9	est</w>
4	l + o	7	lo
5	lo + w	7	low
6	n + e	6	ne
7	ne + w	6	new
8	new + est</w>	6	newest</w>

newest</w>

“newest” becomes a single token

w · i · d · est</w>

“widest” was never seen whole, but reuses est</w>

Result computed by code (ties broken by order of occurrence).

© Aridio Silva · @aridiosilva · October 2026

From ID to dense vector

E Embedding

Each ID selects a row of W_E (|V| × d_model): a learned dense vector (Bengio et al., 2003; Mikolov et al., 2013).

≡ Static

“banco” gets the same vector in any sentence. Contextualization only happens later, in attention.

? Why it is needed

ID 4871 is not “similar” to 4872. Vectors allow similarity and the sums and products that follow.

512

d_model in the original Transformer

768

d_model of BERT-base

12.288

d_model of GPT-3 (largest)

© Aridio Silva · @aridiosilva · October 2026

Positional encoding: order matters

$$PE(pos, 2i) = \sin(pos / 10000^{2i/d_model}) \qquad PE(pos, 2i+1) = \cos(pos / 10000^{2i/d_model})$$

Why? “the cat eats the fish” and “the fish eats the cat” have the same tokens; without position, unmasked attention treats them equivalently (Vaswani et al., 2017).

1 Learned

Trainable table (BERT, GPT-2)

2 Relative

Attention depends on distance (Shaw et al., 2018)

3 RoPE

Rotates Q and K by position (Su et al., 2021); used in LLaMA

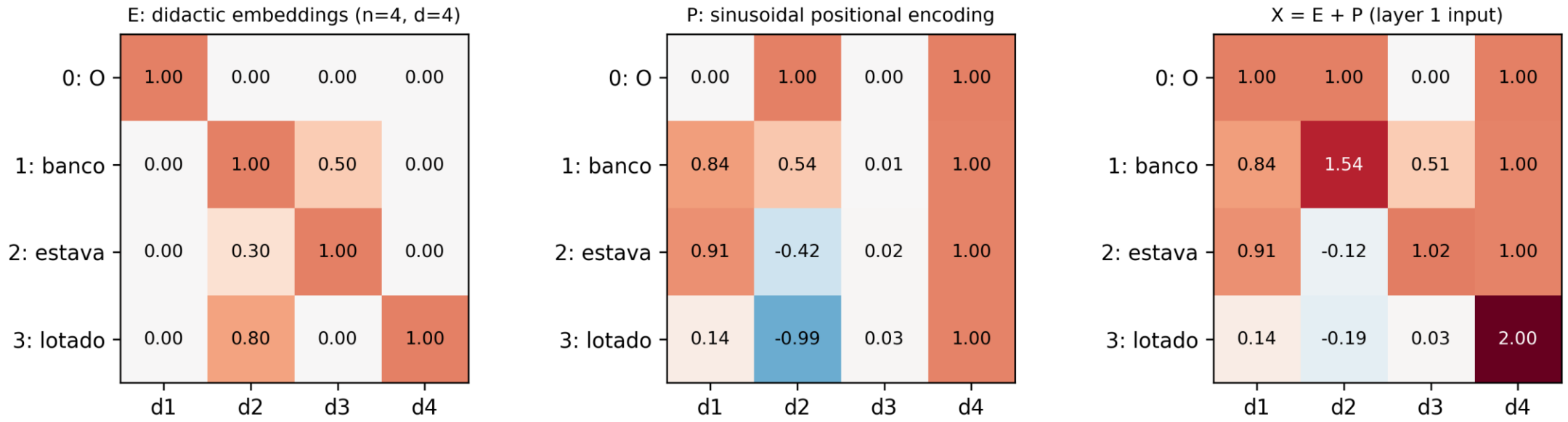
4 ALiBi

Linear distance bias; extrapolates to long contexts (Press et al., 2022)

© Aridio Silva · @aridiosilva · October 2026

X = E + P for “O banco estava lotado”

Figure 2 – Step 2: embeddings and positional encoding (computed values; didactic weights)



© Aridio Silva · @aridiosilva · October 2026 · ORCID 0009-0008-2411-6995

“banco” (position 1): PE = [0.841; 0.540; 0.010; 1.000]; embedding [0; 1; 0.5; 0] → x = [0.841; 1.540; 0.510; 1.000]

d_model = 4 (didactic; the original uses 512). X, of size 4 × 4, is the input of the first layer.

Query, Key and Value: three roles

Q **Query = X·Wq**

What this token is looking for.

K **Key = X·Wk**

What this token offers to be found.

V **Value = X·Wv**

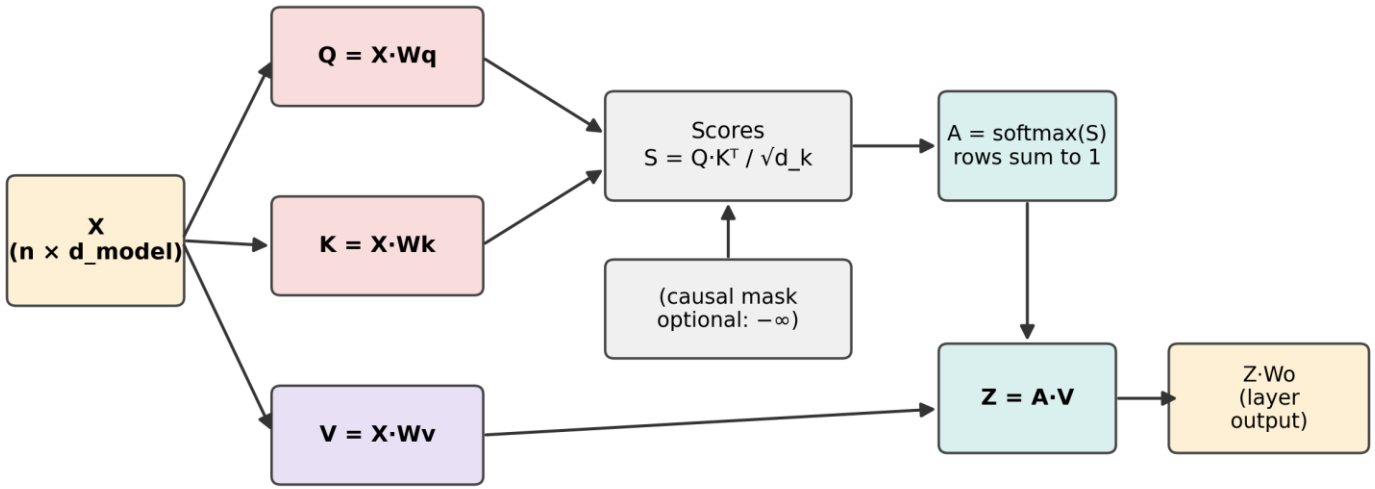
The content the token passes on if selected.

Why it is needed. A word’s sense depends on its neighbours, yet the input embedding is static. Each token must consult the others, with weights that depend on content. In attention, any two positions connect in one step, and the computation is parallelizable (Vaswani et al., 2017).

© Aridio Silva · @aridiosilva · October 2026

Scaled dot-product

Figure 3 – Scaled dot-product attention mechanism (Vaswani et al., 2017)



© Aridio Silva · @aridiosilva · October 2026 · ORCID 0009-0008-2411-6995

- **QK^T:** compatibility between each Query and each Key.
- **÷ √d_k:** keeps softmax from saturating when d_k is large.
- **softmax:** positive weights summing to 1 per row.
- **× V:** weighted average of the Values: the contextualized token.

$$softmax(\, QK^T / \sqrt{d_k}\,) \cdot V$$

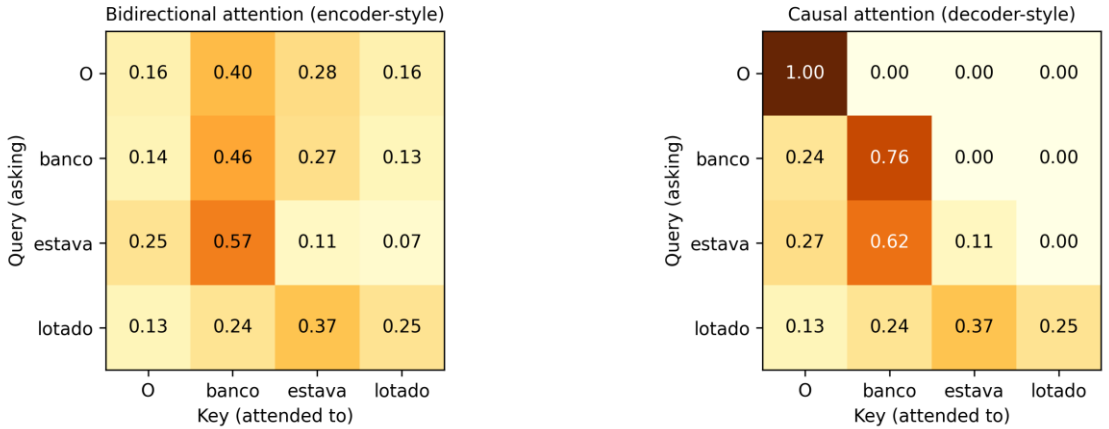
“banco” attention in numbers

Step (token “banco”)	O	banco	estava	lotado
Score q·k (QKT)	4.476	6.130	5.353	4.375
Score / √2	3.165	4.334	3.785	3.093
Weight (softmax)	**0.143**	**0.459**	**0.265**	**0.133**

Z = [1.043; 1.360]

output for “banco”: Values averaged with 0.143 · 0.459 · 0.265 · 0.133

Figure 4 – Attention map softmax(QKᵀ/√d_k): each row sums to 1.00



© Aridio Silva · @aridiosilva · October 2026 · ORCID 0009-0008-2411-6995

Hand-set weights: do not read linguistic meaning into the pattern. In a trained model, Wq, Wk and Wv learn which relations matter.

6 Step 4 · Multi-head

Several relations in parallel

$$MultiHead(X) = Concat(head^1, ..., head^h) \cdot W^O$$

8 × 64

heads × d_k in the base Transformer (d_model = 512)

1,048,576

weights in the 4 attention projections: 4 × 512 × 512

h | d_model

h must divide d_model: 12 heads need d_model = 768 (BERT-base)

? Why it is needed

A single attention gives one average per token and inhibits different subspaces; several heads, in smaller dimension, do not raise total cost.

≈ What is observed

Heads focus on delimiters, fixed offsets and syntactic relations (Clark et al., 2019); many are redundant (Voita et al., 2019; Michel et al., 2019).

6 Step 4 · Masks

Who may look at whom

↔ Bidirectional

Every position sees all others. BERT regime (encoder).

→ Causal

Future scores get −∞ and weights 0. Prevents “seeing” the token to predict (leakage).

▭ Padding

Ignores empty positions used to equalize batch sizes.

“banco” row: [3.165; 4.334; 3.785; 3.093] → with causal mask → **[0.237; 0.763; 0; 0]**
In a generative model, “banco” cannot look at “lotado”.

© Aridio Silva · @aridiosilva · October 2026

Efficient inference: MQA, GQA and KV cache

MQ Multi-Query

All heads share a single K and V pair (Shazeer, 2019).

GQ Grouped-Query

Groups of heads share K and V: a trade-off between quality and memory (Ainslie et al., 2023).

KV KV cache

Stores the K and V already computed and avoids recomputing the past for each token (Pope et al., 2023).

Why it is needed. In generation, rereading and recomputing the whole past at every token would be prohibitive; fewer K and V shrink the cache and speed up inference.

© Aridio Silva · @aridiosilva · October 2026

Residual connection and normalization

+ Residual

y = x + f(x): a shortcut for signal and gradient; each sublayer learns only a correction (He et al., 2016).

σ LayerNorm / RMSNorm

Normalize each token over its d_model components (Ba et al., 2016). RMSNorm omits the mean (Zhang & Sennrich, 2019).

⇄ Pre-LN × Post-LN

Normalizing before each sublayer (Pre-LN) trains more stably, without careful warm-up (Xiong et al., 2020).

Why it is needed. Without a shortcut, deep networks degrade in training; without normalization, value scales drift. The depth that gives the model capacity would no longer be trainable.

© Aridio Silva · @aridiosilva · October 2026

7 Step 5 · Numerical example

LayerNorm × RMSNorm for “banco”

Vector	Values (d_model = 4)
x (input)	[0.841; 1.540; 0.510; 1.000]
o (attention output)	[1.074; 1.912; 0.537; 0.956]
r = x + o	[1.915; 3.453; 1.047; 1.956]
LayerNorm(r)	**[−0.205; 1.572; −1.209; −0.158]**
RMSNorm(r)	**[0.846; 1.525; 0.462; 0.864]**

$\mu = 2.093$

mean of r

$\sigma^2 = 0.748$

variance of r

2.264

root mean square (RMS)

After LayerNorm: mean 0 and variance 1. RMSNorm keeps the proportion between components, without centering.

The per-token feed-forward network

$$FFN(x) = max(0, x \cdot W^1 + b^1) \cdot W^2 + b^2$$

What it does

Attention mixes information between tokens; the FFN transforms information within each token: expands (d_ff = 4 × d_model), applies the non-linearity and compresses.

Why it is needed

In attention, the output is an average of Values via linear projections; the only non-linearity is the weights' softmax. The FFN adds a per-token non-linear transform.

Key-value memory

Columns of W1 detect patterns (keys); rows of W2 bring the associated information (values). Hypothesis by Geva et al. (2021).

d_model = 512 → **d_ff = 2048 · ReLU** → **d_model = 512**

© Aridio Silva · @aridiosilva · October 2026

Parameters and a didactic calculation

1,048,576

W1: 512 × 2048

1,048,576

W2: 2048 × 512

≈ 2/3

of the ≈ 3.15 million weights of a layer are in the FFN (Geva et al., 2021)

Didactic calculation (d_model = 4, d_ff = 8, random weights, seed 0)

Token	FFN input	Hidden layer (ReLU)	FFN output
estava	[1.389; −0.901; −0.999; 0.511]	[0.907; 0.660; 0.330; 0; 0.623; 0; 1.541; 0.759]	[0.231; 0.954; 0.437; −0.291]
banco	[−0.205; 1.572; −1.209; −0.158]	[0; 0; 0; 0; 0; 0; 0; 0]	[0; 0; 0; 0]

“estava”: 6 of 8 units active. “banco”: all zeroed by the ReLU, so the output is 0 and the vector crosses the sublayer via the residual path.

© Aridio Silva · @aridiosilva · October 2026

Multi-layer stacking

Model	Layers	d_model	Heads	Parameters	Source
Base Transformer	6 + 6	512	8	65 million	Vaswani et al., 2017
BERT-base	12	768	12	110 million	Devlin et al., 2019
BERT-large	24	1024	16	340 million	Devlin et al., 2019
GPT-3 (largest)	96	12,288	96	175 billion	Brown et al., 2020

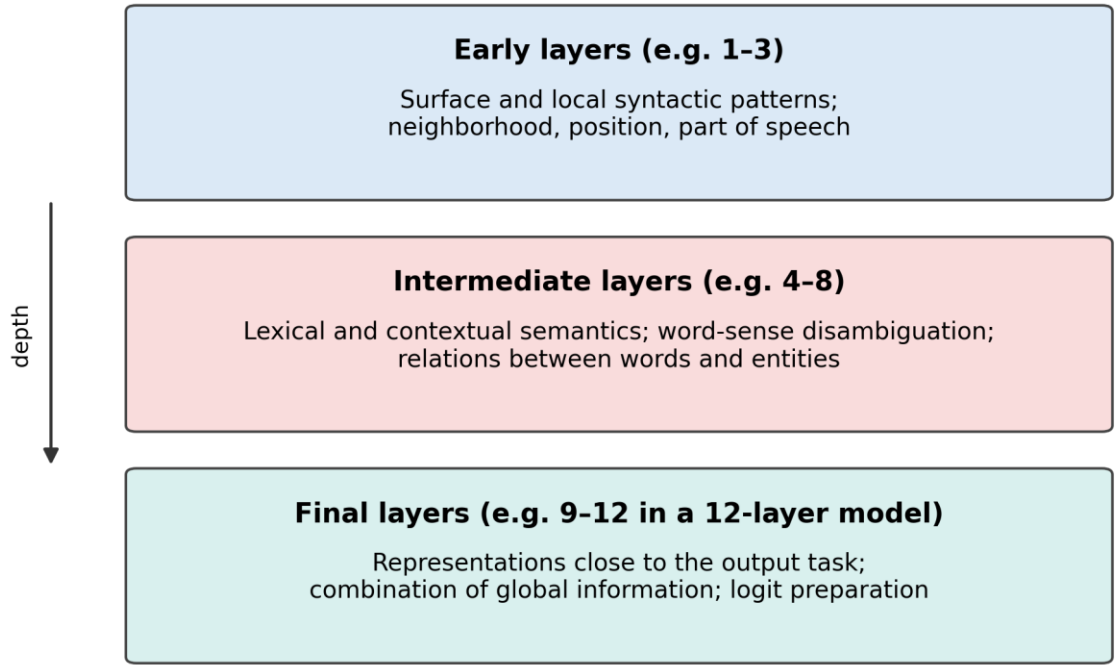
Why it is needed. One layer captures relations in a single step, but many tasks require composition: identify, relate, infer the sense and only then predict. Some capabilities need at least two layers, such as induction heads (Elhage et al., 2021; Olsson et al., 2022). More layers add capacity but cost more: depth is a design choice.

© Aridio Silva · @aridiosilva · October 2026

What each depth tends to represent

Figure 5 – Approximate specialization by depth in stacked Transformers

Conceptual sketch: trend observed in probing studies on BERT (Tenney et al., 2019); layer boundaries are NOT exact.



© Aridio Silva · @aridiosilva · October 2026 · ORCID 0009-0008-2411-6995

- **Probing:** early layers capture local patterns; middle ones, syntax; upper ones, semantics (Jawahar et al., 2019).
- **Classic pipeline:** POS, parsing, entities, semantic roles, coreference, in that order (Tenney et al., 2019).
- **Refinement:** the next-token prediction improves layer by layer (Belrose et al., 2023).
- **Caveat:** boundaries vary by model, task and method; “layer 6 resolves banco” is an illustration.

The objective: predict the next token

$$L = -(1/T)\,\Sigma^{T}\ \log p(x^{t} \mid x^{<t}) \qquad PPL = e^{L}$$

3.806

loss if the target were “lotado” (p = 0.022)

0.473

loss if the target were “cheio” (p = 0.623)

p − y

gradient of the loss at the logits: [−0.978; 0.114; 0.623; 0.094; 0.147] for target “lotado”

? Why it is needed

With random weights the architecture does not produce language. Training turns it into a useful model; the signal comes from the text itself, with no human labels.

BERT is different

Uses masked tokens and looks both ways (Devlin et al., 2019). Disambiguation is a strategy to reduce prediction error.

© Aridio Silva · @aridiosilva · October 2026

How the weights are adjusted

∇ Backpropagation

Gradients by chain rule through all layers (Rumelhart et al., 1986); required for millions to billions of parameters.

⚙ Optimization

Adam (Kingma & Ba, 2015) with warm-up, decay, dropout and label smoothing (Vaswani et al., 2017); AdamW today (Loshchilov & Hutter, 2019).

↗ Scale

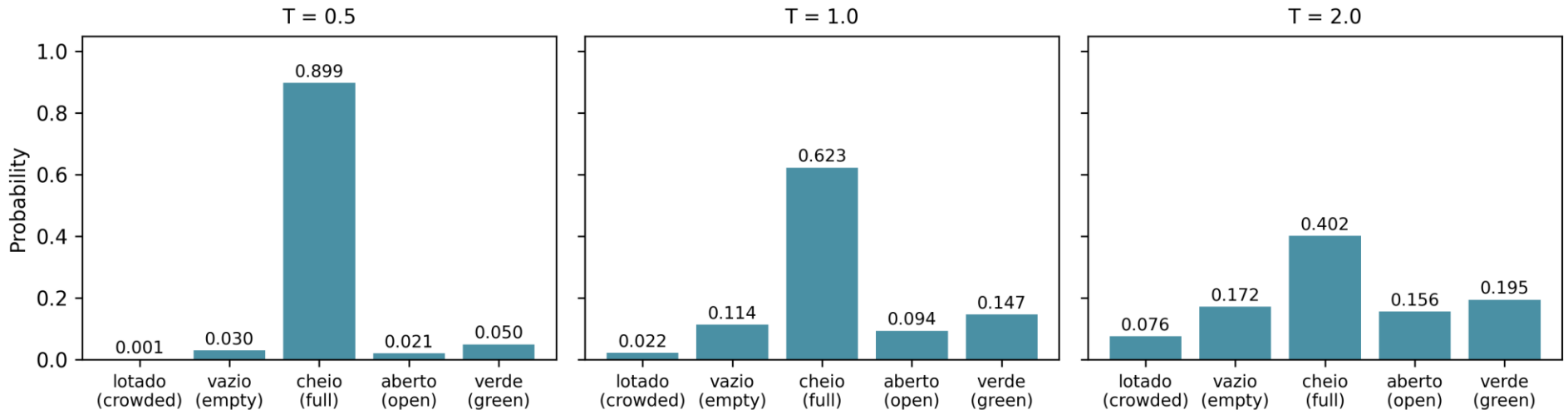
Loss falls as a power law with parameters, data and compute (Kaplan et al., 2020); data and parameters should grow together (Hoffmann et al., 2022).

Training: all positions in parallel (causal mask). Inference: one token at a time, with KV cache. Then: pre-training → fine-tuning → alignment (RLHF; Ouyang et al., 2022).

© Aridio Silva · @aridiosilva · October 2026

Logits, softmax and temperature

Figure 6 – Effect of temperature on softmax(logits/T) (computed logits; random output weights)



© Aridio Silva · @aridiosilva · October 2026 · ORCID 0009-0008-2411-6995

Logits (position "estava"): lotado −0.818 · vazio 0.815 · cheio 2.516 · aberto 0.626 · verde 1.070

T < 1 concentrates probability (more deterministic); T > 1 spreads it. Random output weights: in a trained model, "lotado" would get a high probability.

From vector to chosen word

Strategy	How it works	In the example
Greedy	Always picks the most probable token	“cheio”
Sampling	Draws according to the probabilities	lotado 0.022 … cheio 0.623
Top-k (Fan et al., 2018)	Restricts to the k most probable and renormalizes	k = 2: cheio 0.809 · verde 0.191
Top-p (Holtzman et al., 2020)	Smallest set with cumulative probability ≥ p	p = 0.9: {cheio, verde, vazio, aberto} → 0.637; 0.150; 0.117; 0.096
Beam search	Several candidate sequences; picks the highest total probability	—

Why it is needed. Always picking the most probable tends to repetitive text; unrestricted sampling includes unlikely tokens. Top-k and top-p balance the extremes.

© Aridio Silva · @aridiosilva · October 2026

12 Case study

Disambiguating “banco”

Didactic toy: 4 axes [money, seat, people, wood]; attention with Wq = Wk = 2·I, Wv = I; cosine comparison against two sense prototypes.

Context	Contextualized vector	cos. bank	cos. seat
Isolated: “banco”	[0.700; 0.700; 0; 0]	0.632	0.632
“O banco estava lotado de clientes”	[0.524; 0.351; 0.229; 0]	**0.851**	0.468
“O banco da praça estava velho de madeira”	[0.310; 0.509; 0.006; 0.175]	0.452	**0.858**

= Isolated

Equidistant from both senses (0.632 × 0.632).

≠ Only the neighbours change

Same mechanism: with customers it moves toward the bank; with square and wood, toward the seat.

Care when reading attention maps

1 Attention ≠ explanation

Different weights can give the same prediction; attention is only part of the computation (Jain & Wallace, 2019; Wiegreffe & Pinter, 2019).

2 Empirical evidence

In trained models, vectors of the same word cluster by sense (Reif et al., 2019) and get more context-specific in upper layers (Ethayarajh, 2019).

3 Intervention and circuits

Knowing what the model “uses” requires intervention experiments and circuit analysis (Elhage et al., 2021; Olsson et al., 2022).

Visualization tool: Vig (2019).

Capabilities nobody programmed

A Concept

A property of the whole that is in no component (Anderson, 1972). Nothing in the code mentions senses or syntax.

W With scale

Abilities absent in smaller models and present in larger ones (Wei et al., 2022a): few-shot, chain-of-thought.

S Debate

Part of the jumps may come from discontinuous metrics (Schaeffer et al., 2023). The topic remains open.

G ≠ grokking

Grokking is delayed generalization during training on small tasks (Power et al., 2022; Nanda et al., 2023). Emergence by scale occurs along model size.

B And “banco”?

Disambiguation already appears in BERT-base (110 million parameters): there is no fixed layer threshold. The capability comes from training and grows with scale.

What this work does not show

D Of this document

- Examples with tiny dimensions, a single head, hand-set or random weights.
- Omitted: biases, dropout, cross-attention, mixture of experts and attention alternatives such as state-space models (Gu & Dao, 2023).
- Layer, normalization and position details vary across models.

M Of the mechanism itself

- **n^2 cost:** long context may use the middle poorly (Liu et al., 2024).
- **Hallucination:** no step checks truthfulness (Ji et al., 2023).
- **“Just statistics”?** Bender & Koller (2020) deny understanding from form alone; Li et al. (2023) find internal domain states (Othello).
- **Emergence:** nature of the jumps is debated.

© Aridio Silva · @aridiosilva · October 2026

★ Conclusion

Key takeaways

1	**No explicit rule**	“Banco” is resolved by context: attention mixes neighbours into the token’s vector.
2	**Every step has a reason**	Numbers, vectors, order, context, stability, non-linearity, depth and output.
3	**Weights are the knowledge**	Training adjusts weights to predict the next token; inference only uses them.
4	**Emergence, with caution**	Capabilities arise from training and scale; the debate on jumps remains open.
5	**Check the sources**	12 of 64 references checked at source; 14 links audited. Numbers are didactic.

© Aridio Silva · @aridiosilva · October 2026

Terms and concepts (1/8)

Term	Definition	§
Adam / AdamW	Optimizers with per-parameter adaptive rate; AdamW decouples weight decay.	10
ALiBi	Linear bias on attention scores, proportional to the distance between tokens.	4
Attention	Weighted average of vectors, with weights from query–key compatibility.	5
Attention head	One of h parallel attentions, with its own Wq, Wk and Wv.	6
Autoregressive	Generates one element at a time, conditioned on those already generated.	2, 11
Backpropagation	Computes loss gradients for all weights, propagating the error by the chain rule.	10
Beam search	Keeps several candidate sequences and picks the most probable one.	11
BERT	Encoder-only model trained with masked tokens and bidirectional attention.	2, 10
BPE (Byte Pair Encoding)	Tokenization that iteratively merges the most frequent adjacent symbol pair.	3
Causal mask	Blocks attending to future positions by giving $-\infty$ to the corresponding scores.	6

© Aridio Silva · @aridiosilva · October 2026

Terms and concepts (2/8)

Term	Definition	§
Chain-of-thought	Prompting where the model writes reasoning steps before answering.	13
Context	Tokens that condition a prediction (previous ones, or the whole sequence if bidirectional).	5, 12
Contextualized token	A token's vector after attention, carrying information from the other tokens.	5
Cosine similarity	Alignment between vectors: dot product over the norms, from −1 to 1.	12
Cross-attention	Queries from one sequence (decoder); Keys and Values from another (encoder).	14
Cross-entropy	Loss measuring how far the predicted distribution is from the correct token: −log p.	10
d_model, d_k, d_ff	Model dimension, per-head dimension (d_model/h) and FFN hidden dimension.	4, 6, 8
Decoding	Rule that picks the output token (greedy, top-k, top-p, beam).	11
Dot product	Sum of component-wise products; measures alignment.	5
Dropout	Regularization that randomly zeroes activations during training.	10

© Aridio Silva · @aridiosilva · October 2026

Terms and concepts (3/8)

Term	Definition	§
Embedding	Learned dense vector representing a token (or a position).	4
Emergence	System-level property arising from many interacting components, not programmed.	13
Encoder / decoder	Encoder reads with bidirectional attention; decoder generates with causal attention.	2
Feed-forward (FFN)	Two linear layers with a non-linearity, applied to each position.	8
Few-shot	Using the model with a few examples in the prompt, without updating weights.	13
Fine-tuning	Additional training, after pre-training, for a task or behaviour.	10
Forward pass	Passing the input through the layers to the output, without updating weights.	2
GELU	Smooth activation function used in BERT and GPT.	8
Gradient	Derivatives of the loss with respect to the weights; shows how to adjust them.	10
Grokking	Delayed generalization, long after memorization, on small algorithmic tasks.	13

© Aridio Silva · @aridiosilva · October 2026

Terms and concepts (4/8)

Term	Definition	§
Hallucination	Fluent output that is factually wrong or unsupported by the context.	14
Induction head	Two-head circuit that finds past occurrences of a pattern and predicts its continuation.	9
Inference	Running the trained model, with frozen weights, to generate an output.	2
Interpretability	Methods to understand what the model computes and why.	12
Key	Projection X·Wk: what each token offers to be found.	5
KV cache	Already computed Keys and Values, reused for each newly generated token.	6
Label smoothing	Softens the training target, reducing extreme confidence.	10
Layer	One Transformer block: attention, Add & Norm, FFN, Add & Norm.	9
LayerNorm	Per-token normalization across dimensions, with learned scale γ and shift β.	7
Logit	Unbounded real score per vocabulary token, before the softmax.	11

© Aridio Silva · @aridiosilva · October 2026

Terms and concepts (5/8)

Term	Definition	§
Logit lens / tuned lens	Project intermediate states onto the vocabulary to see per-layer predictions.	9
Masked language modeling	Hidden tokens are reconstructed from context on both sides.	10
Mixture of experts (MoE)	Each token activates only a subset of specialized FFN sub-networks.	14
Multi-head attention	h attentions in parallel in distinct subspaces, concatenated and projected by Wo.	6
One-hot	Vector with 1 at the correct token’s position and 0 elsewhere.	10
Parameter / weight	Adjustable number learned during training.	10
Perplexity (PPL)	Exponential of the mean per-token loss; measures the model’s “surprise”.	10
Positional encoding	Vector added to the embedding to convey the token’s position.	4
Pre-LN / Post-LN	Normalization before (Pre-LN) or after (Post-LN) the residual sum.	7
Pre-training	Initial training phase on a large volume of raw text.	10

© Aridio Silva · @aridiosilva · October 2026

Terms and concepts (6/8)

Term	Definition	§
Probing	Simple classifiers on internal representations to test what they contain.	9
Query	Projection X·Wq: what each token is looking for.	5
ReLU	Activation function max(0, x).	8
Residual connection	Adds a sublayer's input to its output: y = x + f(x).	7
RLHF	Reinforcement learning from human feedback, to align with preferences.	10
RMSNorm	Normalizes by the root mean square, without subtracting the mean.	7
RoPE	Position by rotating Q and K, expressing relative position.	4
Scaling law	Regular (power-law) relation between loss and parameters, data and compute.	10, 13
Self-attention	Attention where Queries, Keys and Values come from the same sequence.	5
SentencePiece	Subword library (BPE or Unigram) operating on raw text.	3

© Aridio Silva · @aridiosilva · October 2026

Terms and concepts (7/8)

Term	Definition	§
Softmax	Converts numbers into positive probabilities that sum to 1.	5, 11
SwiGLU	Gated activation (GLU) using Swish, common in recent models.	8
Temperature (T)	Divisor of the logits before softmax; controls how peaked the distribution is.	11
Token	Vocabulary unit (word, subword, byte or symbol).	3
Tokenization	Converting text into a sequence of tokens and IDs.	3
Top-k / Top-p (nucleus)	Sampling restricted to the k most probable or the smallest set with cumulative probability p.	11
Transformer	Attention-based architecture, with no recurrence or convolution.	1, 2
Unigram LM	Tokenization that starts from a large vocabulary and prunes less useful items.	3
Value	Projection X·Wv: the content each token passes on.	5
Vocabulary	Finite set of tokens the model recognizes.	3

© Aridio Silva · @aridiosilva · October 2026

Terms and concepts (8/8)

Term	Definition	§
Warm-up	Initial training phase with a growing learning rate.	7, 10
Weight tying	Shares the embedding matrix with the output projection.	4, 11
WordPiece	Subwords with merges chosen by likelihood gain.	3

Sources and references (1/8)

#	Reference	Link	✓ ●
1	**Ainslie et al. (2023).** GQA: Training generalized multi-query Transformer models from multi-head checkpoints. EMNLP 2023	arxiv.org/abs/2305.13245	
2	**Anderson (1972).** More is different. Science, 177(4047), 393–396	doi.org/10.1126/science.177.4047.393	
3	**Ba et al. (2016).** Layer normalization	arxiv.org/abs/1607.06450	
4	**Bahdanau et al. (2015).** Neural machine translation by jointly learning to align and translate. ICLR 2015	arxiv.org/abs/1409.0473	
5	**Belrose et al. (2023).** Eliciting latent predictions from Transformers with the tuned lens	arxiv.org/abs/2303.08112	✓ ●
6	**Bender et al. (2020).** Climbing towards NLU: On meaning, form, and understanding in the age of data. ACL 2020, p. 5185–5198	aclanthology.org/2020.acl-main.463	
7	**Bengio et al. (2003).** A neural probabilistic language model. Journal of Machine Learning Research, 3, 1137–1155	www.jmlr.org/papers/v3/bengio03a.html	
8	**Brown et al. (2020).** Language models are few-shot learners. NeurIPS 33	arxiv.org/abs/2005.14165	

✓ checked at source ● link audited (others: not verified in this environment; see section 16 note)

Sources and references (2/8)

#	Reference	Link	✓ ●
9	**Clark et al. (2019).** What does BERT look at?. An analysis of BERT's attention. BlackboxNLP Workshop at ACL 2019, p. 276–286	arxiv.org/abs/1906.04341	●
10	**Dao et al. (2022).** FlashAttention: Fast and memory-efficient exact attention with IO-awareness. NeurIPS 35	arxiv.org/abs/2205.14135	
11	**Devlin et al. (2019).** BERT: Pre-training of deep bidirectional Transformers for language understanding. NAACL-HLT 2019, p. 4171–4186	arxiv.org/abs/1810.04805	
12	**Elhage et al. (2021).** A mathematical framework for Transformer circuits. Transformer Circuits Thread	transformer-circuits.pub/2021/framework/index.html	
13	**Ethayarajh (2019).** How contextual are contextualized word representations?. Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. EMNLP-IJCNLP 2019, p. 55–65	aclanthology.org/D19-1006	✓ ●
14	**Fan et al. (2018).** Hierarchical neural story generation. ACL 2018, p. 889–898	arxiv.org/abs/1805.04833	
15	**Gage (1994).** A new algorithm for data compression. The C Users Journal, 12(2), 23–38	www.derczynski.com/papers/archive/BPE_Gage.pdf	
16	**Ganguli et al. (2022).** Predictability and surprise in large generative models. ACM FAccT 2022, p. 1747–1764	arxiv.org/abs/2202.07785	

✓ checked at source ● link audited (others: not verified in this environment; see section 16 note)

Sources and references (3/8)

#	Reference	Link	✓ ●
17	**Geva et al. (2021).** Transformer feed-forward layers are key-value memories. EMNLP 2021, p. 5484–5495	aclanthology.org/2021.emnlp-main.446	✓ ●
18	**Gu et al. (2023).** Mamba: Linear-time sequence modeling with selective state spaces	arxiv.org/abs/2312.00752	
19	**He et al. (2016).** Deep residual learning for image recognition. CVPR 2016, p. 770–778	arxiv.org/abs/1512.03385	
20	**Hendrycks et al. (2016).** Gaussian error linear units (GELUs)	arxiv.org/abs/1606.08415	
21	**Hinton et al. (2015).** Distilling the knowledge in a neural network	arxiv.org/abs/1503.02531	
22	**Hoffmann et al. (2022).** Training compute-optimal large language models. NeurIPS 35	arxiv.org/abs/2203.15556	
23	**Holtzman et al. (2020).** The curious case of neural text degeneration. ICLR 2020	arxiv.org/abs/1904.09751	
24	**Jain et al. (2019).** Attention is not explanation. NAACL-HLT 2019, p. 3543–3556	aclanthology.org/N19-1357	✓ ●

✓ checked at source ● link audited (others: not verified in this environment; see section 16 note)

Sources and references (4/8)

#	Reference	Link	✓ ●
25	**Jawahar et al. (2019).** What does BERT learn about the structure of language?. ACL 2019, p. 3651–3657	aclanthology.org/P19-1356	✓ ●
26	**Ji et al. (2023).** Survey of hallucination in natural language generation. ACM Computing Surveys, 55(12), art. 248	arxiv.org/abs/2202.03629	●
27	**Kaplan et al. (2020).** Scaling laws for neural language models	arxiv.org/abs/2001.08361	
28	**Kingma et al. (2015).** Adam: A method for stochastic optimization. ICLR 2015	arxiv.org/abs/1412.6980	
29	**Kudo (2018).** Subword regularization: Improving neural network translation models with multiple subword candidates. ACL 2018, p. 66–75	arxiv.org/abs/1804.10959	
30	**Kudo et al. (2018).** SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. EMNLP 2018 (System Demonstrations), p. 66–71	arxiv.org/abs/1808.06226	
31	**Li et al. (2023).** Emergent world representations: Exploring a sequence model trained on a synthetic task. ICLR 2023	arxiv.org/abs/2210.13382	
32	**Liu et al. (2024).** Lost in the middle: How language models use long contexts. Transactions of the ACL, 12, 157–173	aclanthology.org/2024.tacl-1.9	✓ ●

✓ checked at source ● link audited (others: not verified in this environment; see section 16 note)

Sources and references (5/8)

#	Reference	Link	✓ ●
33	**Loshchilov et al. (2019).** Decoupled weight decay regularization. ICLR 2019	arxiv.org/abs/1711.05101	
34	**Michel et al. (2019).** Are sixteen heads really better than one?. NeurIPS 32	arxiv.org/abs/1905.10650	
35	**Mikolov et al. (2013).** Efficient estimation of word representations in vector space	arxiv.org/abs/1301.3781	
36	**Nanda et al. (2023).** Progress measures for grokking via mechanistic interpretability. ICLR 2023	arxiv.org/abs/2301.05217	
37	**Olsson et al. (2022).** In-context learning and induction heads. Transformer Circuits Thread	arxiv.org/abs/2209.11895	
38	**Ouyang et al. (2022).** Training language models to follow instructions with human feedback. NeurIPS 35	arxiv.org/abs/2203.02155	
39	**Pope et al. (2023).** Efficiently scaling Transformer inference. Proceedings of Machine Learning and Systems, 5 (MLSys 2023)	arxiv.org/abs/2211.05102	✓ ●
40	**Power et al. (2022).** Grokking: Generalization beyond overfitting on small algorithmic datasets	arxiv.org/abs/2201.02177	

✓ checked at source ● link audited (others: not verified in this environment; see section 16 note)

Sources and references (6/8)

#	Reference	Link	✓ ●
41	**Press et al. (2022).** Train short, test long: Attention with linear biases enables input length extrapolation. ICLR 2022	arxiv.org/abs/2108.12409	
42	**Press et al. (2017).** Using the output embedding to improve language models. EACL 2017, p. 157–163	arxiv.org/abs/1608.05859	
43	**Radford et al. (2018).** Improving language understanding by generative pre-training. OpenAI, relatório técnico	cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf	
44	**Radford et al. (2019).** Language models are unsupervised multitask learners. OpenAI, relatório técnico	cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf	
45	**Reif et al. (2019).** Visualizing and measuring the geometry of BERT. NeurIPS 32, p. 8594–8603	arxiv.org/abs/1906.02715	✓ ●
46	**Rumelhart et al. (1986).** Learning representations by back-propagating errors. Nature, 323, 533–536	doi.org/10.1038/323533a0	
47	**Schaeffer et al. (2023).** Are emergent abilities of large language models a mirage?. NeurIPS 2023 (v. 36)	arxiv.org/abs/2304.15004	✓ ●
48	**Schuster et al. (2012).** Japanese and Korean voice search. ICASSP 2012, p. 5149–5152	doi.org/10.1109/ICASSP.2012.6289079	

✓ checked at source ● link audited (others: not verified in this environment; see section 16 note)

Sources and references (7/8)

#	Reference	Link	✓ ●
49	**Sennrich et al. (2016).** Neural machine translation of rare words with subword units. ACL 2016, p. 1715–1725	arxiv.org/abs/1508.07909	
50	**Shaw et al. (2018).** Self-attention with relative position representations. NAACL-HLT 2018, p. 464–468	arxiv.org/abs/1803.02155	
51	**Shazeer (2019).** Fast Transformer decoding: One write-head is all you need	arxiv.org/abs/1911.02150	
52	**Shazeer (2020).** GLU variants improve Transformer	arxiv.org/abs/2002.05202	
53	**Su et al. (2021).** RoFormer: Enhanced Transformer with rotary position embedding	arxiv.org/abs/2104.09864	
54	**Tenney et al. (2019).** BERT rediscovers the classical NLP pipeline. ACL 2019, p. 4593–4601	aclanthology.org/P19-1452	✓ ●
55	**Touvron et al. (2023).** LLaMA: Open and efficient foundation language models	arxiv.org/abs/2302.13971	
56	**Vaswani et al. (2017).** Attention is all you need. Advances in Neural Information Processing Systems 30 (NIPS 2017), p. 5998–6008	papers.neurips.cc/paper/7181-attention-is-all-you-need	✓ ●

✓ checked at source ● link audited (others: not verified in this environment; see section 16 note)

Sources and references (8/8)

#	Reference	Link	✓ ●
57	**Vig (2019).** A multiscale visualization of attention in the Transformer model. ACL 2019 (System Demonstrations), p. 37–42	arxiv.org/abs/1906.05714	
58	**Voita et al. (2019).** Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. ACL 2019, p. 5797–5808	arxiv.org/abs/1905.09418	
59	**Wei et al. (2022a).** Emergent abilities of large language models. Transactions on Machine Learning Research	arxiv.org/abs/2206.07682	✓ ●
60	**Wei et al. (2022b).** Chain-of-thought prompting elicits reasoning in large language models. NeurIPS 35	arxiv.org/abs/2201.11903	
61	**Wiegreffe et al. (2019).** Attention is not not explanation. EMNLP-IJCNLP 2019, p. 11–20	arxiv.org/abs/1908.04626	
62	**Wu et al. (2016).** Google's neural machine translation system: Bridging the gap between human and machine translation	arxiv.org/abs/1609.08144	
63	**Xiong et al. (2020).** On layer normalization in the Transformer architecture. ICML 2020 (PMLR 119)	arxiv.org/abs/2002.04745	
64	**Zhang et al. (2019).** Root mean square layer normalization. NeurIPS 32	arxiv.org/abs/1910.07467	

✓checked at source ● link audited (others: not verified in this environment; see section 16 note)