Similitudes y diferencias entre modelos de lenguaje

Fernando Carranza
fernandocarranza@gmail.com

Clase 2 Parte 2
Martes 22/04/2025

- Clase 1: Nociones básicas de redes neuronales para el procesamiento del lenguaje natural
  - Parte 1: Tokenización, embeddings estáticos
  - Parte 2: el perceptrón; arquitectura básica de redes neuronales; funciones de activación; backpropagation.
- **Clase 2: Grandes y pequeños modelos de lenguaje**.
  - Parte 1: Mecanismo de atención; transformers; embeddings posicionales
  - **Parte 2: Similitudes y diferencias entre modelos de lenguaje**
- Clase 3: *Prompt engineering*
  - Parte 1: Hiperparámetros de los modelos de lenguaje; prompting: Zero-Shot; Few-Shot
  - Parte 3: Prompting: RAG, Chain of Thought y otros

Presentación

Estructura y temas de la clase de hoy:

1. Introducción
2. Historia y características de los modelos de lenguaje
3. Parámetros de diferenciación de modelos
4. Los modelos disponibles
5. Recapitulación
6. Bibliografía

Para la clase de hoy me voy a basar en la siguiente bibliografía:

- Naveed, H., Khan, A. U., Qiu, S., Saqib, M., Anwar, S., Usman, M., ... & Mian, A. (2023). "A comprehensive overview of large language models". arXiv preprint arXiv:2307.06435.
- Wang, F., Zhang, Z., Zhang, X., Wu, Z., Mo, T., Lu, Q., ... & Wang, S. (2024). "A comprehensive survey of small language models in the era of large language models: Techniques, enhancements, applications, collaboration with llms, and trustworthiness". arXiv preprint arXiv:2411.03350.
- Jurafsky, D. y Martin, J. H. (2025). "Large Language Models". *Speech and Language Processing*. Cap. 10.

Los objetivos de esta clase son los siguientes:

- Conocer qué tipos de grandes modelos de lenguaje hay disponibles.
- Tener un acercamiento a las características de estos modelos.
- Entender las razones de la emergencia de los pequeños modelos de lenguaje.
- Conocer qué pequeños modelos de lenguaje hay disponibles.

Importancia de la generación autorregresiva

The insight of large language modeling is that many practical NLP tasks can be cast as word prediction, and that a powerful-enough language model can solve them with a high degree of accuracy. For example, we can cast sentiment analysis as language modeling by giving a language model a context like:

The sentiment of the sentence “I like Jackie Chan” is:

and comparing the following conditional probability of the words “positive” and the word “negative” to see which is higher:
P(positive|The sentiment of the sentence “I like Jackie Chanïs:)
P(negative|The sentiment of the sentence “I like Jackie Chanïs:)
If the word “positive” is more probable, we say the sentiment of the sentence is positive, otherwise we say the sentiment is negative

Jurafsky y Martin. “Large Language Models”

Importancia de la generación autorregresiva

Esta misma idea se puede aplicar a otras tareas de procesamiento del lenguaje natural, como *question answering*:
Q: Who wrote the book “The Origin of Species”? A:
P(w|Q: Who wrote the book ‘ The Origin of Species”? A:)

Generación autorregresiva

- La tarea de decidir una palabra a generar dado un contexto se denomina *decoding*.
- Hacerlo repetitivamente se denomina *autoregressive generation* o *causal LM generation*.

Más allá de que todos los modelos de lenguaje puedan concebirse como un algoritmo de clasificación, existe una serie de cuestiones en las que varían y que es útil tener en cuenta a la hora de compararlos.

Algunos parámetros de variación relevantes según Naveed *et al.* (2023) I

- Cómo se tokeniza el texto: si al nivel de palabra, al nivel de subpalabra, etc.
- Qué tipo de *embedding* posicional se utiliza: Alibi, RoPE
- Qué tipo de mecanismo de atención se utiliza: *self-attention*, *cross attention*, *sparse attention*, etc.
- Qué función de activación se utiliza: ReLu, GeLu, GLU.
- La normalización de capas o *layer normalization*, usada para evitar *exploding* y *vanishing gradients*.
- Qué tipo de entrenamiento distribuido se utiliza.
- Qué librerías se usan: Tensorflow (Google), Pytorch (Facebook), Transformers, etc.
- Cómo se preprocesa el corpus: filtrado, eliminación de datos privados.

Algunos parámetros de variación relevantes según Naveed *et al.* (2023) II

- Qué tipo de arquitectura: encoder-decoder, solo decoder, etc.
- Objetivos del preentrenamiento: Full Language Modelling, Masked Language Modeling, etc.
- *Scaling Laws*: Búsqueda de la combinación óptima de los parámetros, tamaño del corpus y los recursos computacionales.
- Qué tipo de adaptación tiene: modelos preentrenados, *instruction-tuned models*.

Fine-tuning

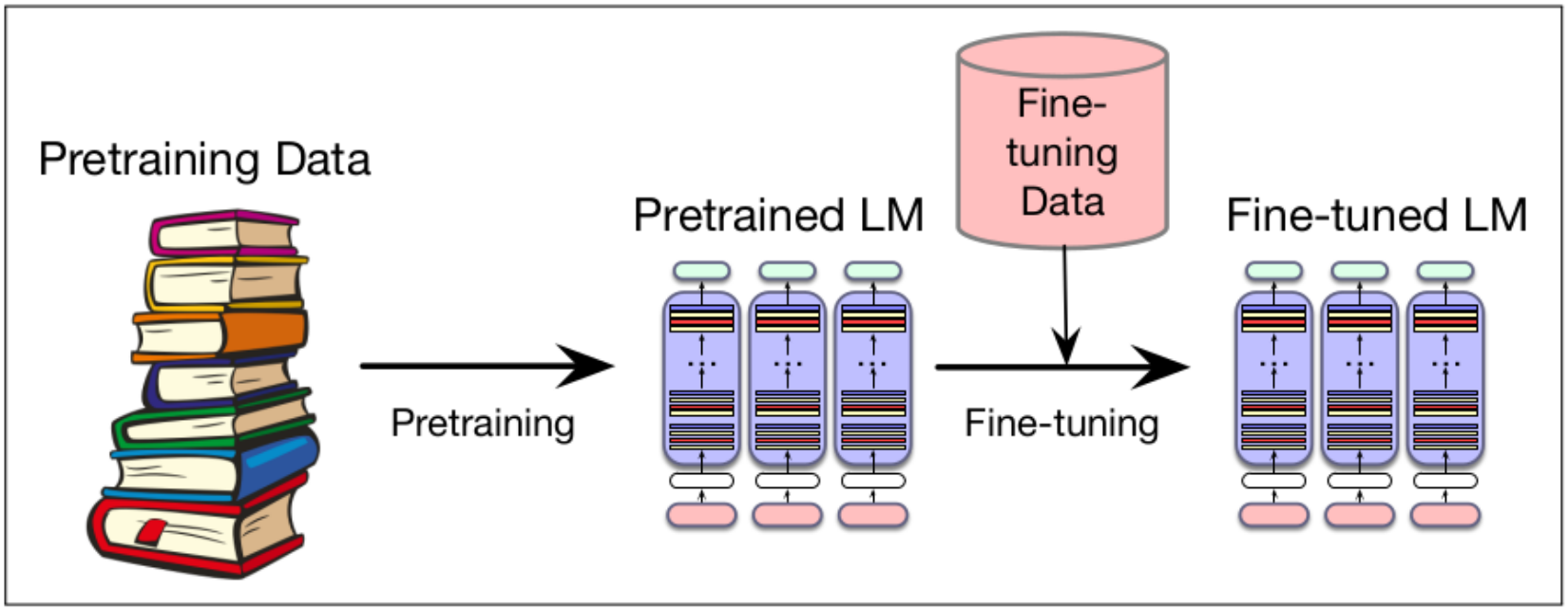


Figure 10.6 Pretraining and finetuning. A pre-trained model can be finetuned to a particular domain, dataset, or task. There are many different ways to finetune, depending on exactly which parameters are updated from the finetuning data: all the parameters, some of the parameters, or only the parameters of specific extra circuitry.

Figura: Tomado de Jurafsky y Martin.

Fine-tuning y *adapted layers*

El fine-tuning se puede realizar de distintas maneras:

- **Continued pre-training**: Reentrenando todo el modelo.
- **Parameter-efficient finetuning**: Congelando algunos parámetros y seleccionando solo algunos para el reentrenamiento.
- **Entrenamiento con *adapted layers***: Se congela todo el modelo pero se le agregan capas y lo que se entrena son esas capas.

Model Alignment

El proceso se puede enriquecer además con model-alignment, en el que un conjunto de anotadores humanos premian o castigan al modelo según si responden o no como esperan. De ese modo se puede aprestar al modelo para que se comporte como queremos.

Model alignment

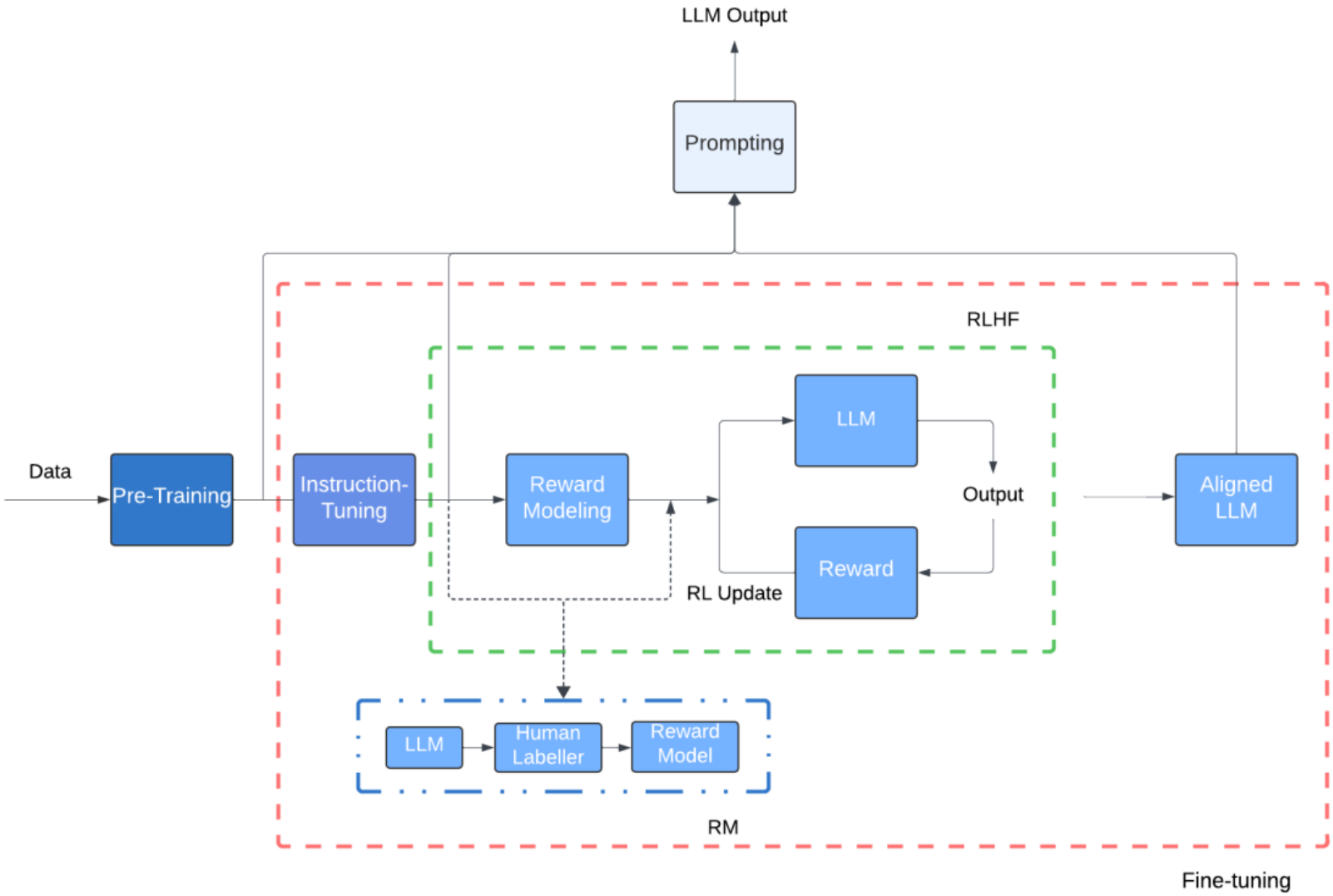


Figura: Tomado de Naveed et al.

Scaling Law

El rendimiento de un modelo depende de 3 factores:

1. Tamaño del modelo (cantidad de parámetros, excluyendo embeddings)
2. Tamaño del corpus
3. Capacidad de cómputo.

Las formas en que la interacción entre estos tres factores influye en el rendimiento se conoce como *scaling laws*.

Corpus

Además del método de preparación del corpus, también influye sobre el modelo el corpus que se utilice de base. Algunos corpus muy usados son los siguientes:

1. CommonCrawl: `https://commoncrawl.org/`
2. The Pile

Sampling

Otra cuestión que diferencia a los modelos es cómo se realiza el decoding. La elección de la palabra a predecir en una tarea de generación autorregresiva podría realizarse con una función argmax. A esto se lo denomina *greedy decoding*. Sin embargo, el resultado sería un texto muy predecible. Por este motivo existen técnicas de *sampling* que seleccionan de manera aleatoria la palabra a predecir entre una serie de candidatos relevantes sobre los que tienen mayores probabilidades. Estas técnicas incluyen el *top k-sampling*, el *Nucleus o top-p sampling* y el *temperature sampling*

LLMs Multimodales (MLLMs)

Naveed *et al.* (2023) señalan la existencia de las MLLMs

- Los MLLMs pueden percibir diferentes modalidades de información como imágenes, video y audio.
- Ofrecen beneficios sustanciales en comparación con los LLMs estándar al lograr una comprensión más profunda del contexto.
- Ejemplos de MLLMs incluyen Flamingo, BLIP-2 y MiniGPT-4 en la corriente de preentrenamiento.

Evaluación de LLMs

Los modelos de lenguaje se pueden comparar también por su rendimiento, pero para eso es necesario contar con algún criterio de evaluación de los modelos.

- La evaluación se divide en dos categorías amplias:
  - **Comprensión del Lenguaje Natural (NLU)**: Incluye análisis de sentimiento, clasificación de texto, inferencia del lenguaje natural (NLI), preguntas y respuestas (QA) y razonamiento de sentido común (CR).
  - **Generación del Lenguaje Natural (NLG)**: Incluye resumen, finalización de oraciones, traducción automática (MT) y generación de diálogo.
- Se han propuesto numerosos conjuntos de datos y *benchmarks* para cada tarea.

Datasets y benchmarks

Type	Datasets/Benchmarks
Multi-Task	MMLU [307], SuperGLUE [2], BIG-bench [308], GLUE [309], BBH [308], CUGE [310], Zero-CLUE [311], FewCLUE [312], Blended Skill Talk [313], HELM [314], KLUE-STS [315]
Language Understanding	CoQA [316], WiC [317], Wikitext103 [318], PG19 [319], LCQMC [320], QQP [321], WinoGender [322], CB [323], FinRE [324], SanWen [325], AFQMC [311], BQ Corpus [326], CNSS [327], CKBQA 13 [328], CLUENER [311], Weibo [329], AQuA [330], OntoNotes [331], HeadQA [332], Twitter Dataset [333]
Story Cloze and Sentence Completion	StoryCloze [334], LAMBADA [335], LCSTS [336], AdGen [337], E2E [338], CHID [339], CHID-FC [312]
Physical Knowledge and World Understanding	PIQA [340], TriviaQA [341], ARC [342], ARC-Easy [342], ARC-Challenge [342], PROST [343], OpenBookQA [344], WebNLG [345], DogWhistle Insider & Outsider [346]
Contextual Language Understanding	RACE [347], RACE-Middle [347], RACE-High [347], QuAC [348], StrategyQA [349], Quiz Bowl [350], cMedQA [351],cMedQA2 [352], MATINF-QA [353]
Commonsense Reasoning	WinoGrande [354], HellaSwag [355], COPA [356], WSC [357], CSQA [358], SIQA [359], C^3 [360], CLUEWSC2020 [311], CLUEWSC [311], CLUEWSC-FC [312], ReCoRD [361]
Reading Comprehension	SQuAD [362], BoolQ [363], SQUADv2 [364], DROP [365], RTE [366], WebQA [367], CMRC2017 [368], CMRC2018 [369], CMRC2019 [370], COTE-BD [371], COTE-DP [371], COTE-MFW [371], MultiRC [372], Natural Questions [373], CNSE [327], DRCD [374], DuReader [375], $\text{Dureader}_{\text{robust}}$ [376], DuReader-QG [375], SciQ [377], Sogou-log [378], $\text{Dureader}_{\text{robust}}$-QG [376], QA4MRE [379], KorQuAD 1.0 [380], CAIL2018-Task1 & Task2 [381]
Mathematical Reasoning	MATH [382], Math23k [383], GSM8K [384], MathQA [385], MGSM [386], MultiArith [387], ASDiv [388], MAWPS [389], SVAMP [390]
Problem Solving	HumanEval [141], DS-1000 [391], MBPP [392], APPS [382], CodeContests [142]
Natural Language Inference & Logical Reasoning	ANLI [393], MNLI-m [394], MNLI-mm [394],QNLI [362], WNLI [357], OCNLI [311], CMNLI [311], ANLI R1 [393], ANLI R2 [393], ANLI R3 [393], HANS [395], OCNLI-FC [312], LogiQA [396], StrategyQA [349]
Cross-Lingual Understanding	MLQA [397], XNLI [398], PAWS-X [399], XSum [400], XCOPA [401], XWinograd [402], TyDiQA-GoldP [403], MLSum [404]
Truthfulness and Fact Checking	TruthfulQA [405], MultiFC [406], Fact Checking on Fever [407]
Biases and Ethics in AI	ETHOS [408], StereoSet [409], BBQ [410], Winobias [411], CrowS-Pairs [412]
Toxicity	RealToxicityPrompts [413], CivilComments toxicity classification [414]
Language Translation	WMT [415], WMT20 [416], WMT20-enzh [416], EPRSTMT [312], CCPM [417]
Scientific Knowledge	AminoProbe [148], BioLAMA [148], Chemical Reactions [148], Galaxy Clusters [148], Mineral Groups [148]
Dialogue	Wizard of Wikipedia [418], Empathetic Dialogues [419], DPC-generated [96] dialogues, ConvAI2 [420], KdConv [421]
Topic Classification	TNEWS-FC [312], YNAT [315], KLUE-TC [315], CSL [311], CSL-FC [312], IFLYTEK [422]

La licencia

Los modelos también pueden variar por el tipo de licencia que pueden tener:

1. Propietary: El software o modelo está protegido por derechos de autor, y solo el titular de la licencia tiene control total sobre su uso, modificación y distribución.
2. Open Source License
   - MIT: Permite casi cualquier uso del software, incluyendo modificaciones y distribuciones comerciales.
   - Apache: ofrece cláusulas adicionales de protección contra patentes.
   - Personalizadas: DeepSeek, Gemma, etc.

El tamaño

- Los grandes modelos de lenguaje adquieren lo que se conoce como *emergent abilities*: una repentina adquisición de versatilidad.
- Sin embargo, los LLMs presentan limitaciones debido a su gran tamaño, demandas computacionales, preocupaciones de privacidad y rendimiento subóptimo en dominios especializados.
- Por este motivo, han surgido los modelos de lenguaje pequeños (SLMs), que tienen la ventaja de tener baja latencia de inferencia, rentabilidad, desarrollo eficiente y fácil personalización.

El tamaño

No existe acuerdo respecto de cómo definir los SMLs

- Algunos investigadores consideran que los SLMs son aquellos que tienen menos de un billón de parámetros.
- Otros lo definen en función de su carencia de *emergent abilities*.
- Algunos lo definen en función de si es posible correrlo en un teléfono móvil.
- Wang *et al.* (2024) proponen definir los SLMs por su capacidad para realizar tareas especializadas y su idoneidad para entornos con recursos limitados.

El tamaño

- Los SLMs comúnmente emplean la arquitectura Transformer.
- Las técnicas principales para obtener SLMs a partir de LLMs incluyen *pruning*, *knowledge distillation* y *quantization*.
  - El *pruning* elimina parámetros menos críticos.
  - La *knowledge distillation* transfiere conocimiento de un modelo grande (maestro) a uno más pequeño (estudiante), que debe aprender a imitarlo.
  - La *quantization* disminuye la precisión de los parámetros, reduciendo la memoria y el cómputo.

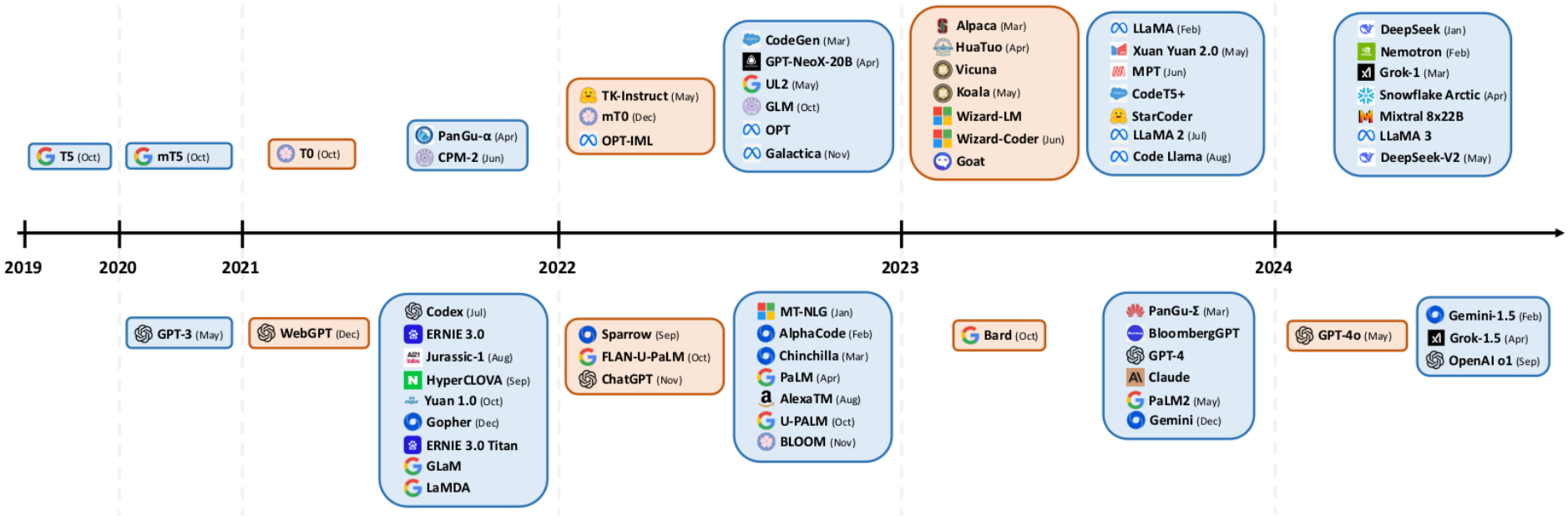
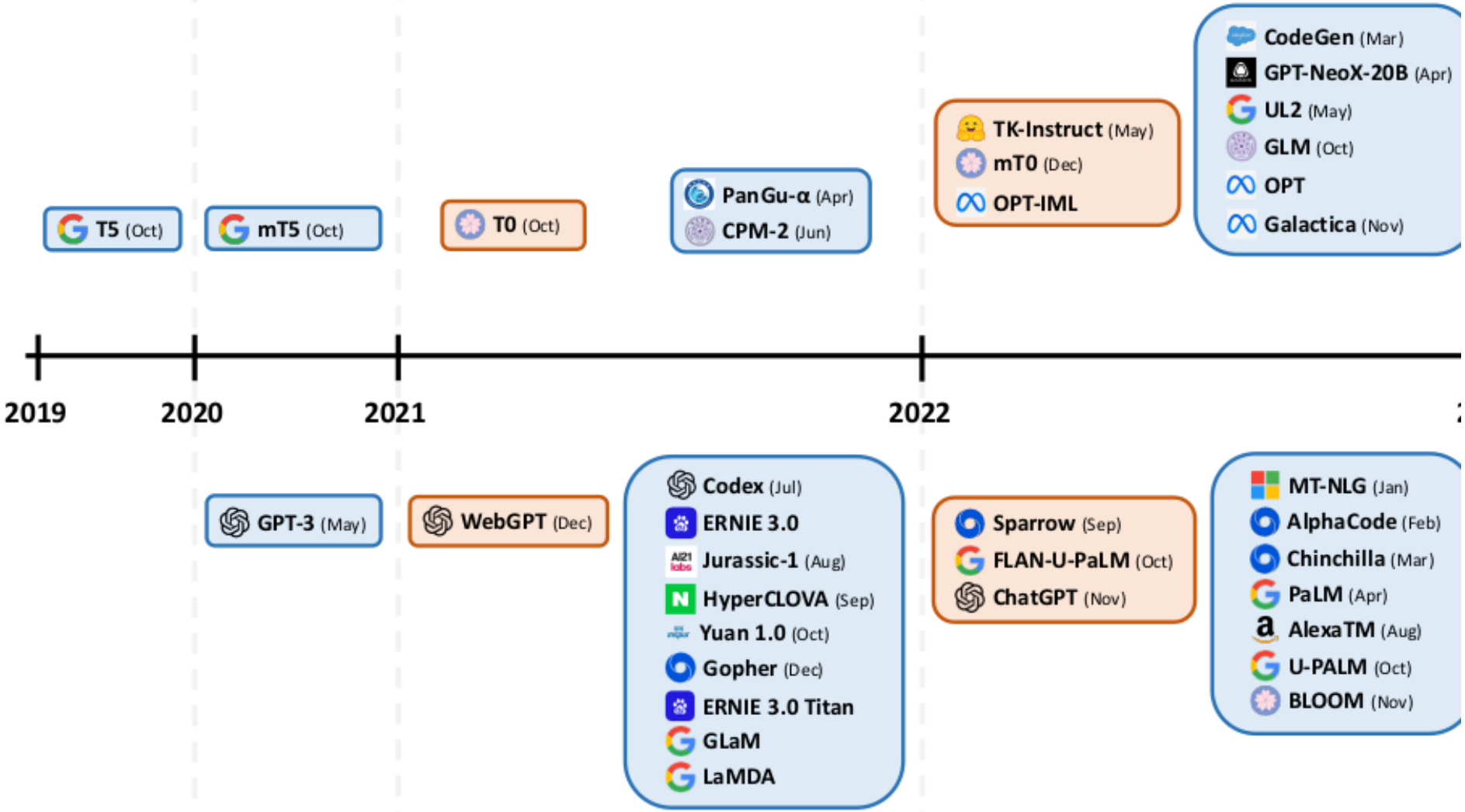
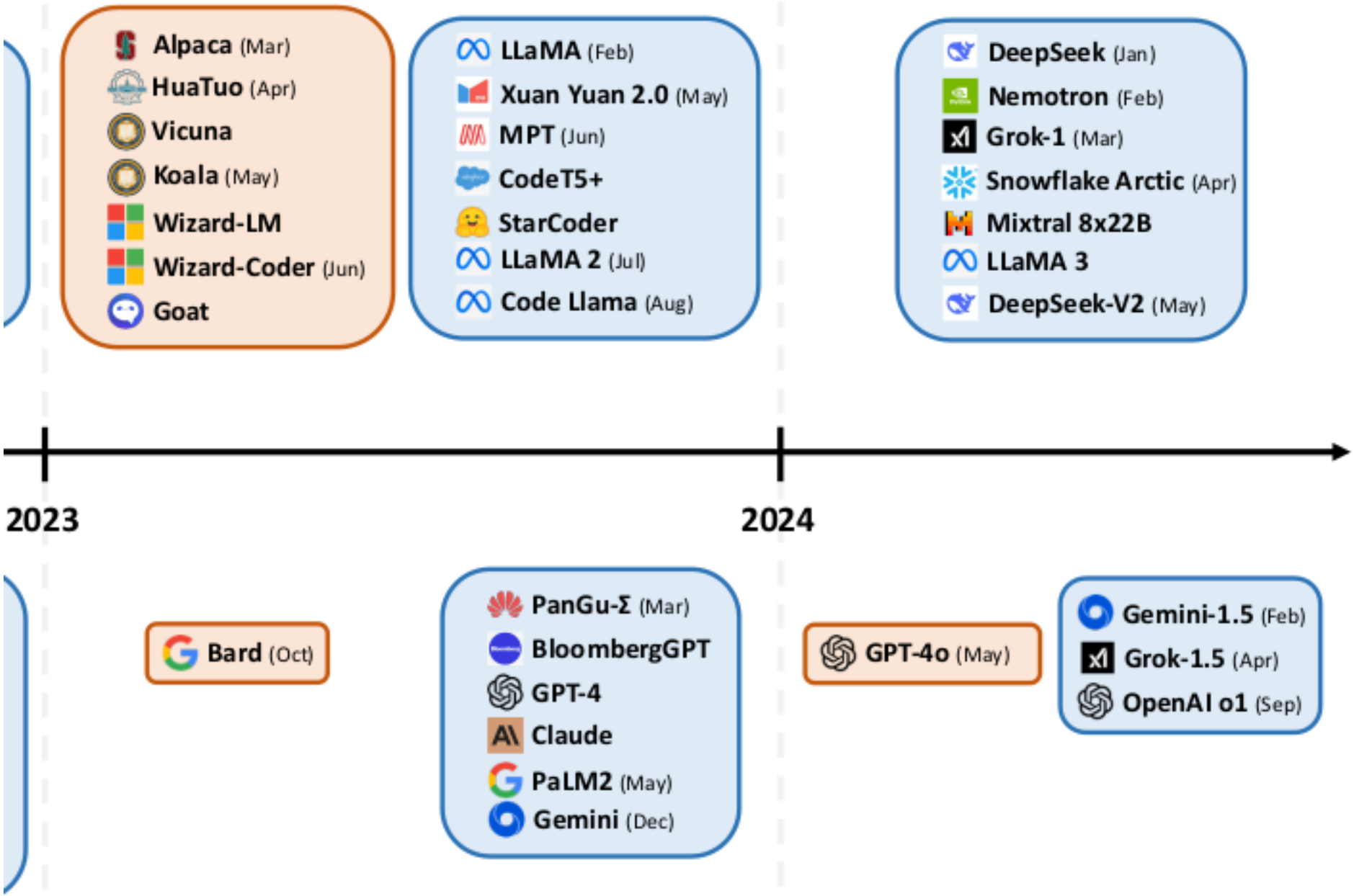


Figura: Cronología de grandes modelos de lenguaje. Las tarjetas azules son modelos preentrenados; las naranjas son 'instruction-tuned' models. Los de arriba son open-source; los de abajo son propietarios. Tomado de Naveed *et al.* (2023)


T5 (Oct)
mT5 (Oct)
T0 (Oct)
PanGu-α (Apr)
CPM-2 (Jun)
TK-Instruct (May)
mT0 (Dec)
OPT-IML
CodeGen (Mar)
GPT-NeoX-20B (Apr)
UL2 (May)
GLM (Oct)
OPT
Galactica (Nov)
2019
2020
2021
2022
GPT-3 (May)
WebGPT (Dec)
Codex (Jul)
ERNIE 3.0
Jurassic-1 (Aug)
HyperCLOVA (Sep)
Yuan 1.0 (Oct)
Gopher (Dec)
ERNIE 3.0 Titan
GLaM
LaMDA
Sparrow (Sep)
FLAN-U-PaLM (Oct)
ChatGPT (Nov)
MT-NLG (Jan)
AlphaCode (Feb)
Chinchilla (Mar)
PaLM (Apr)
AlexaTM (Aug)
U-PALM (Oct)
BLOOM (Nov)


Alpaca (Mar)
HuaTuo (Apr)
Vicuna
Koala (May)
Wizard-LM
Wizard-Coder (Jun)
Goat
LLaMA (Feb)
Xuan Yuan 2.0 (May)
MPT (Jun)
CodeT5+
StarCoder
LLaMA 2 (Jul)
Code Llama (Aug)
DeepSeek (Jan)
Nemotron (Feb)
Grok-1 (Mar)
Snowflake Arctic (Apr)
Mixtral 8x22B
LLaMA 3
DeepSeek-V2 (May)
2023
2024
Bard (Oct)
PanGu-Σ (Mar)
BloombergGPT
GPT-4
Claude
PaLM2 (May)
Gemini (Dec)
GPT-4o (May)
Gemini-1.5 (Feb)
Grok-1.5 (Apr)
OpenAI o1 (Sep)

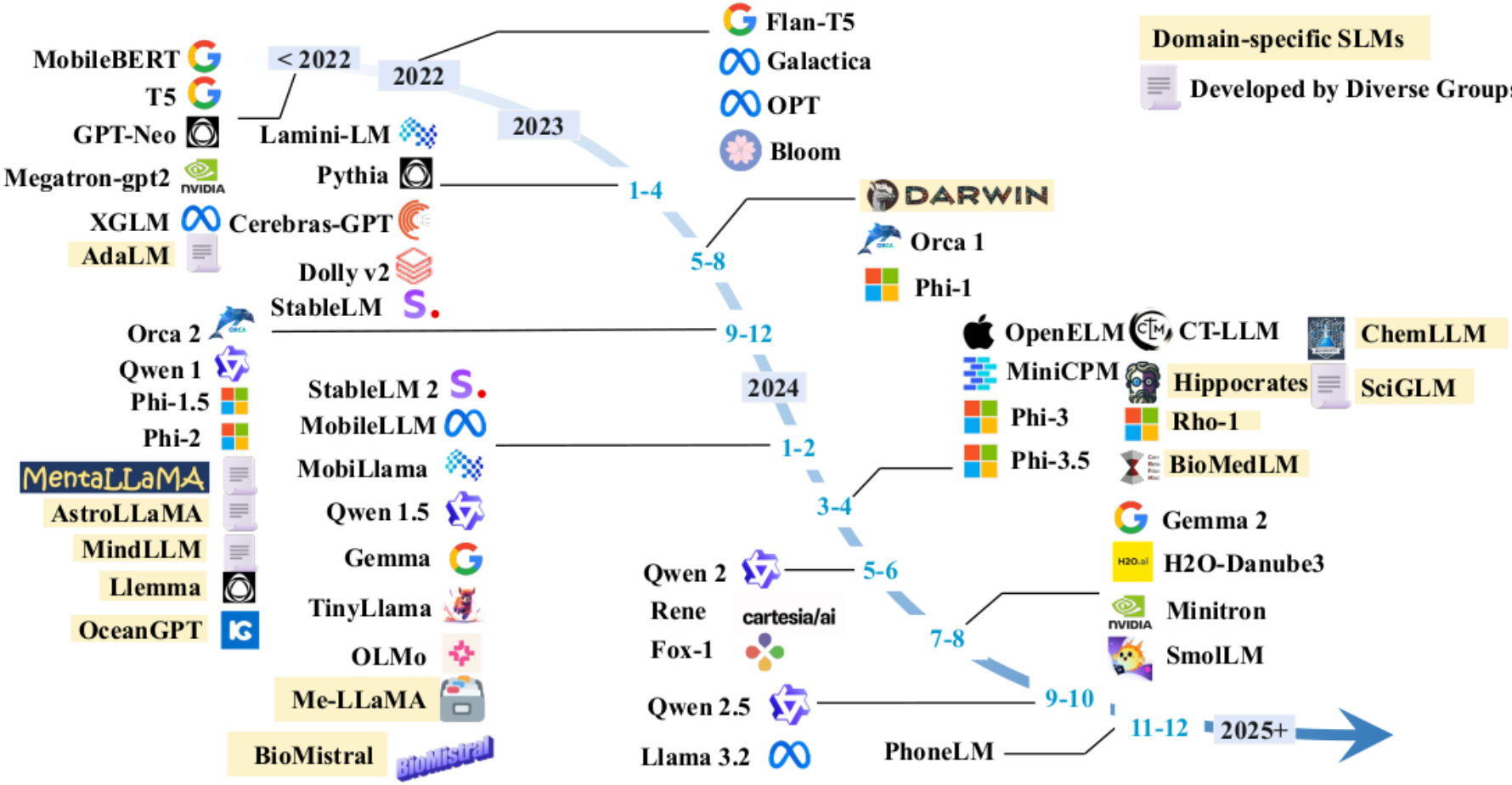


Fig. 3. A timeline of existing small language models.

Figura: Cronología de pequeños modelos de lenguaje. Tomado de Wang *et al.* (2024)

- Se puede consultar información y descargar varios de los modelos en `https://huggingface.co/`

En general, se puede encontrar las presentaciones de los modelos en ArXiv. Van algunos ejemplos a modo de ilustración:

- DeepSeek: `https://arxiv.org/pdf/2401.02954`
- Galactica: `https://arxiv.org/pdf/2211.09085`
- Goat: `https://arxiv.org/pdf/2504.12339`
- Mixtral: `https://arxiv.org/pdf/2401.04088`
- MobileLLM: `https://arxiv.org/pdf/2402.14905`
- Orca: `https://arxiv.org/pdf/2306.02707`
- Pangea: `https://arxiv.org/pdf/2410.16153`
- PhoneLM: `https://arxiv.org/pdf/2411.05046`
- ...

T5

An encoder-decoder model employing a unified text-to-text training for all NLP problems is shown in Figure 7. T5 places layer normalization outside the residual path in a conventional transformer model [64]. It uses masked language modeling as a pre-training objective where spans (consecutive tokens) are replaced with a single mask instead of separate masks for each token. This type of masking speeds up the training as it produces shorter sequences. After pre-training, the model is fine-tuned using adapter layers [106] for downstream tasks.

Naveed *et al.* 2023: 7

T5

An **encoder-decoder model** employing a unified **text-to-text training** for all NLP problems is shown in Figure 7. T5 places **layer normalization** outside the **residual path** in a conventional transformer model [64]. It uses **masked language modeling** as a **pre-training objective** where spans (consecutive tokens) are replaced with a single mask instead of separate masks for each token. This type of masking speeds up the training as it produces shorter sequences. After pre-training, the model is **fine-tuned using adapter layers** [106] for downstream tasks.

Naveed *et al.* 2023: 7

T5

T5 tiene una licencia Apache.
Se lo puede encontrar en este repositorio: `https://github.com/google-research/text-to-text-transfer-transformer`
Utiliza como corpus el CommonCrawl.

GPT-3

GPT-3 [6]: The GPT-3 architecture is the same as the GPT-2 [5] but with dense and sparse attention in transformer layers similar to the Sparse Transformer [67]. It shows that large models can train on larger batch sizes with a lower learning rate to decide the batch size during training, GPT-3 uses the gradient noise scale as in [107]. Overall, GPT-3 increases model parameters to 175B showing that the performance of large language models improves with the scale and is competitive with the fine-tuned models.

Naveed *et al.* 2023: 7-8

GPT-3

GPT-3 [6]: The GPT-3 architecture is the same as the GPT-2 [5] but with **dense and sparse attention** in transformer layers similar to the Sparse Transformer [67]. It shows that large models can train on **larger batch sizes** with a **lower learning rate** to decide the batch size during training, GPT-3 uses the gradient noise scale as in [107]. Overall, GPT-3 increases **model parameters** to 175B showing that the performance of large language models improves with the scale and is competitive with the **fine-tuned models**.

Naveed *et al.* 2023: 7-8

PanGu-α

PanGu-α [108]: An autoregressive model that has a query layer at the end of standard transformer layers, example shown in Figure 8, to predict the next token. Its structure is similar to the transformer layer but with an additional embedding for the next position in the attention mechanism.

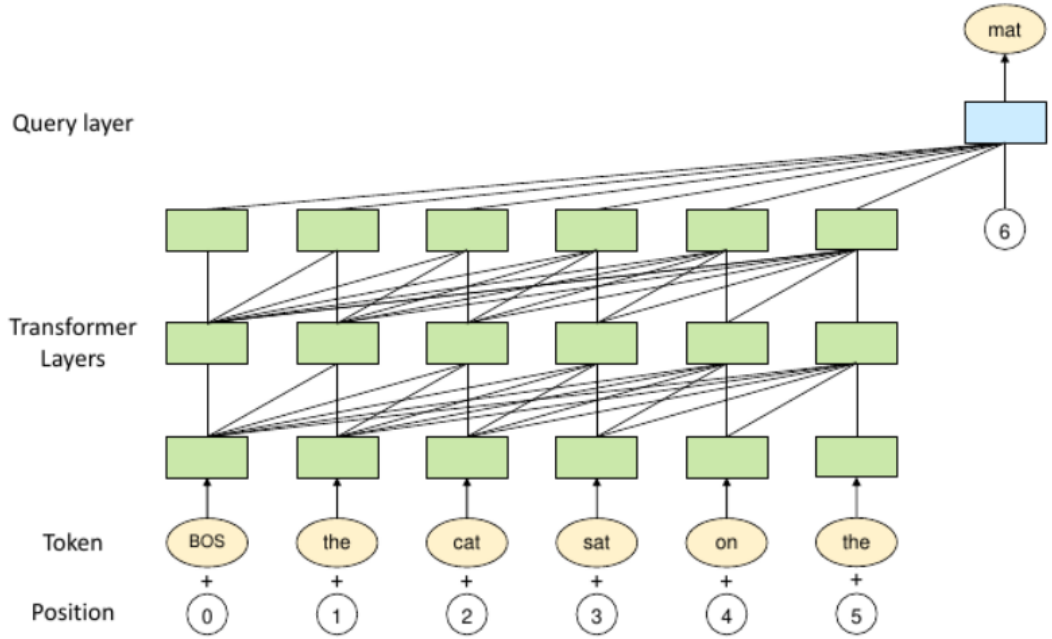


Figura: Naveed *et al.* 2023: 8

PanGu-α

PanGu-α [108]: An **autoregressive model** that has a **query layer** at the end of standard transformer layers, example shown in Figure 8, to predict the next token. Its structure is similar to the transformer layer but with an additional embedding for the next position in the attention mechanism.

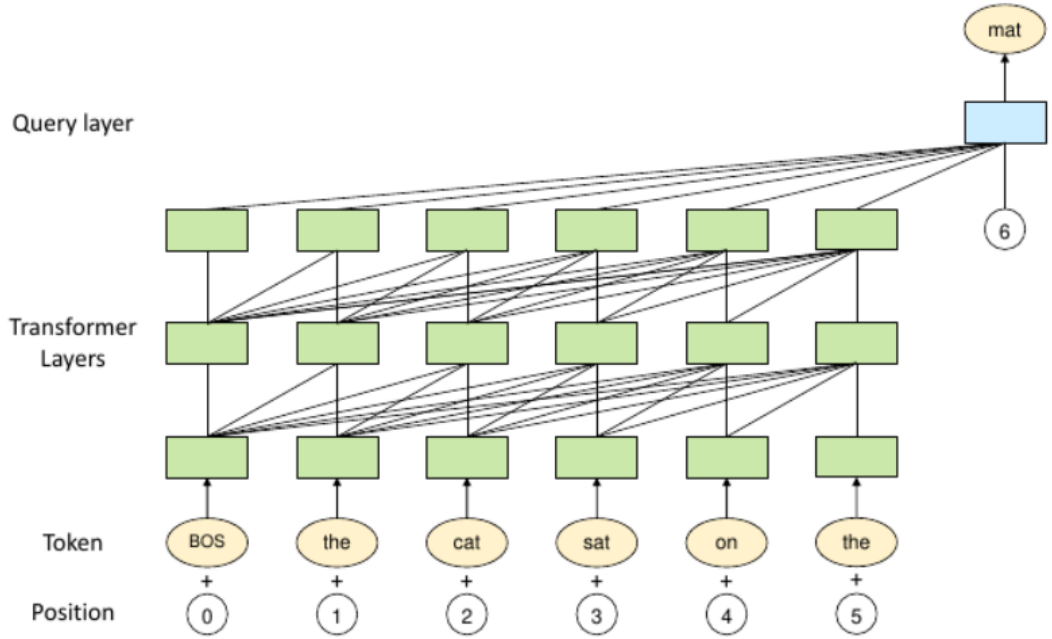


Figura: Naveed *et al.* 2023: 8

Llama

LLaMA [127, 21]: A set of decoder-only language models varying from 7B to 70B parameters. LLaMA models series is the most famous among the community for parameter efficiency and instruction tuning.

Naveed *et al.* 2023: 10

Llama

LLaMA [127, 21]: A set of **decoder-only** language models varying from 7B to 70B **parameters**. LLaMA models series is the most famous among the community for **parameter efficiency** and **instruction tuning**.

Naveed *et al.* 2023: 10

Gemini

Gemini [135, 136]: Gemini replaces Bard (based on PaLM) with multi-modal capabilities and significant language modeling performance improvements.
Gemini-1 [135]: The first-ever auto-regressive model to achieve human-level capabilities on the MMLU benchmark.
Gemini-1.5 [136]: A multi-modal LLM with MoE architecture builds on the findings of Gemini-1. The model has a 2M **context window** and can reason over information up to 10M tokens. Such large context windows were never achieved previously and shown to have a huge impact on performance gain.

Naveed *et al.* 2023: 10

Gemini

Gemini [135, 136]: Gemini replaces Bard (based on PaLM) with **multi-modal capabilities** and significant language modeling performance improvements.
Gemini-1 [135]: The first-ever **auto-regressive model** to achieve human-level capabilities on the MMLU **benchmark**.
Gemini-1.5 [136]: A **multi-modal LLM** with **MoE architecture** builds on the findings of Gemini-1. The model has a **2M context window** and can reason over information up to 10M tokens. Such large context windows were never achieved previously and shown to have a huge impact on performance gain.

Naveed *et al.* 2023: 10

DeepSeek

DeepSeek [138]: DeepSeek studies the LLMs scaling laws in detail to determine the optimal non-embedding model size and training data. The experiments were performed for 8 budgets ranging from 1e 17 to 3e 20 training FLOPs. Each compute budget was tested against ten different models/data scales. The batch size and learning rates were also fitted for the given compute budget finding that the batch size should increase with the increased compute budget while decreasing the learning rate. Following are the equations for the optimal batch-size (B), learning rate (η), model size (M), and data (D).

Naveed *et al.* 2023: 10

DeepSeek

DeepSeek [138]: DeepSeek studies the LLMs **scaling laws** in detail to determine the **optimal non-embedding model size and training data**. The experiments were performed for 8 budgets ranging from $1e^{17}$ to $3e^{20}$ training FLOPs. Each compute budget was tested against ten different models/data scales. The **batch size** and **learning rates** were also fitted for the given compute budget finding that the batch size should increase with the increased compute budget while decreasing the learning rate.

Naveed *et al.* 2023: 10

CodeGen

CodeGen [140]: CodeGen has a similar architecture to PaLM [15], i.e., parallel attention, MLP layers, and RoPE embeddings. The model is trained on both natural language and programming language data sequentially (trained on the first dataset, then the second, and so on) on the following datasets 1) PILE, 2) BIGQUERY, and 3) BIGPYTHON. CodeGen proposed a multi-step approach to synthesizing code. The purpose is to simplify the generation of long sequences where the previous prompt and generated code are given as input with the next prompt to generate the next code sequence. CodeGen open-source a Multi-Turn Programming Benchmark (MTPB) to evaluate multi-step program synthesis.

Naveed *et al.* 2023: 11

CodeGen

CodeGen [140]: CodeGen has a similar architecture to PaLM [15], i.e., **parallel attention**, **MLP layers**, and **RoPE embeddings**. The model is trained on both natural language and programming language data sequentially (trained on the first dataset, then the second, and so on) on the following datasets 1) PILE, 2) BIGQUERY, and 3) BIGPYTHON. CodeGen proposed a multi-step approach to synthesizing code. The purpose is to simplify the generation of long sequences where the previous prompt and generated code are given as input with the next prompt to generate the next code sequence. CodeGen open-source a Multi-Turn Programming Benchmark (MTPB) to evaluate multi-step program synthesis.

Naveed *et al.* 2023: 11

Galactica

Galactica [148]: A large curated corpus of human scientific knowledge with 48 million papers, textbooks, lecture notes, millions of compounds and proteins, scientific websites, encyclopedias, and more are trained using the metaseq library3, which is built on PyTorch and fairscale [149]. The model wraps reasoning datasets with the $<$ work $>$ token to provide step-by-step reasoning context to the model, which has been shown to improve the performance on reasoning tasks.

Naveed *et al.* 2023: 11

Galactica

Galactica [148]: A large curated corpus of human scientific knowledge with 48 million papers, textbooks, lecture notes, millions of compounds and proteins, scientific websites, encyclopedias, and more are trained using the metaseq library3, which is built on **PyTorch** and fairscale [149]. The model wraps reasoning datasets with the $<$ work $>$ token to provide step-by-step reasoning context to the model, which has been shown to improve the performance on reasoning tasks.

Naveed *et al.* 2023: 11

En esta clase introdujimos los siguientes temas:

- Hicimos un breve resumen de las características más importantes de los modelos de lenguaje
- Establecimos algunos parámetros que sirven para comparar los modelos entre sí.
- Presentamos la distinción entre grandes y pequeños modelos de lenguaje.
- Revisamos descripciones de una serie de modelos y tratamos de interpretarlas a la luz de los contenidos vistos hasta ahora en el curso.

Bibliografía I

Jurafsky, D. y Martin, J. H. (2025). *Large Language Models*. 3 edición. Online manuscript released January 12, 2025.

Naveed, H., Khan, A. U., Qiu, S., Saqib, M., Anwar, S., Usman, M., Akhtar, N., Barnes, N., y Mian, A. (2023). A comprehensive overview of large language models. *arXiv preprint arXiv:2307.06435*.

Wang, F., Zhang, Z., Zhang, X., Wu, Z., Mo, T., Lu, Q., Wang, W., Li, R., Xu, J., y Tang, X. (2024). A comprehensive survey of small language models in the era of large language models: Techniques, enhancements, applications, collaboration with llms, and trustworthiness. *arXiv preprint arXiv:2411.03350*.