

Inteligencia Artificial Aplicada para la Economía

Profesores

Profesor Magistral

Camilo Vega Barbosa

Asistente de Docencia

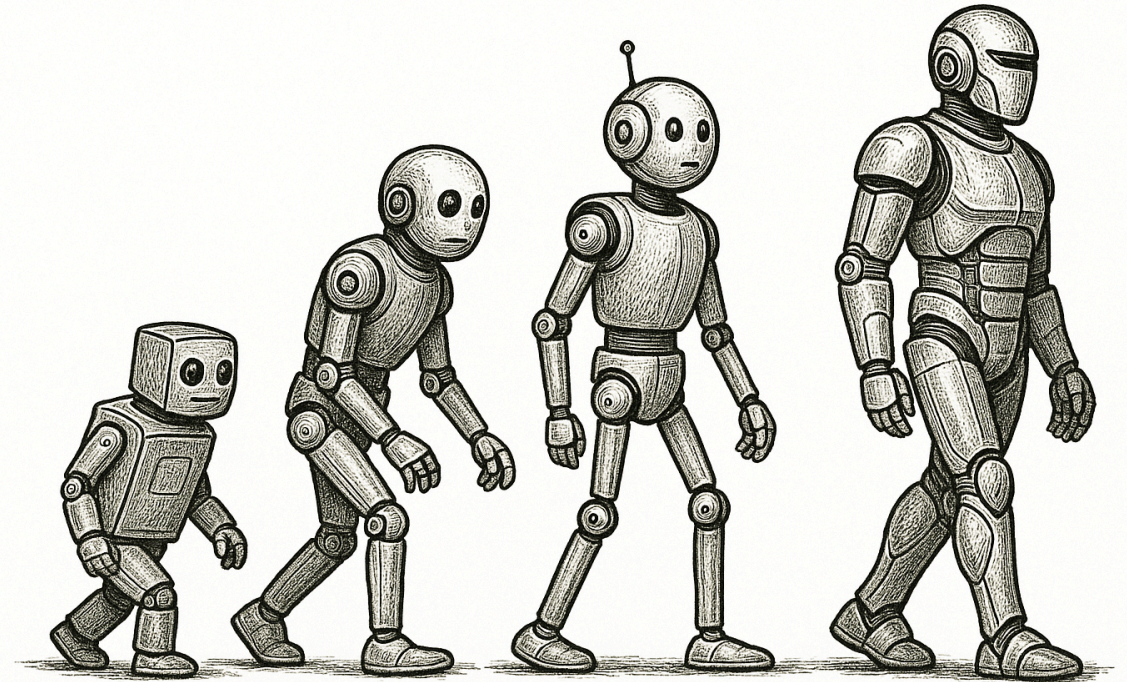
Sergio Julian Zona Moreno




🧠 Mejorando los LLM:


Fine-Tuning y RAG


Enseñando nuevos trucos a modelos
Pre-Entrenados



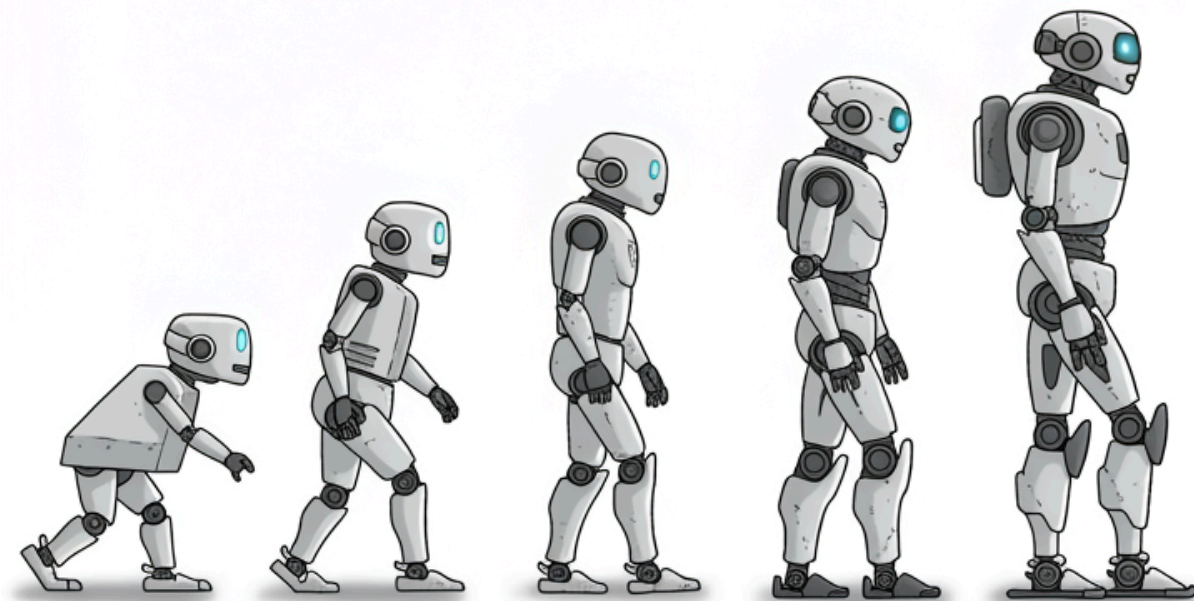
Superando las limitaciones de los LLM

 Los modelos de lenguaje tienen limitaciones inherentes como conocimiento desactualizado, incapacidad para acceder a datos privados, y tendencia a generar información incorrecta o "alucinaciones".


 Existen dos enfoques principales para superar estas barreras: el **Fine-tuning** permite enseñar nuevas habilidades al modelo ajustando sus parámetros internos, mientras que **RAG (Retrieval Augmented Generation)** lo conecta con fuentes externas de información actualizada o especializada.


 Estos métodos se complementan: Fine-tuning modifica el comportamiento intrínseco del modelo, mientras que RAG expande su capacidad para consultar y utilizar conocimiento sin alterar su arquitectura básica.


⚙️ Fine-Tuning



Fine-Tuning: Enseñando nuevos trucos a viejos modelos

 **El Fine-tuning permite adaptar modelos pre-entrenados a tareas específicas,** ajustando sus parámetros para aprender de conjuntos de datos más pequeños y enfocados, logrando así comportamientos especializados que serían imposibles con el modelo base.

 **Este proceso permite personalizar LLMs según necesidades particulares,** como adoptar un tono corporativo específico, dominar jerga técnica de una industria, o aprender a ejecutar flujos de trabajo especializados que no estaban en sus datos de entrenamiento original.

 **El resultado es un modelo que conserva sus capacidades generales pero adquiere experticia en dominios específicos,** similar a un profesional con años de experiencia en un sector particular, capaz de adaptarse más rápidamente a los matices y necesidades del usuario.

Modelos Fundacionales: La base del Fine-Tuning

¿Qué son los modelos fundacionales?

Los **modelos fundacionales** son LLMs pre-entrenados a gran escala con enormes cantidades de datos y recursos computacionales.


Estos modelos sirven como **base reutilizable** para múltiples aplicaciones downstream a través del fine-tuning, evitando entrenar modelos desde cero cada vez.

Ejemplos incluyen GPT-4, Claude, Llama, Mistral y PaLM, que contienen conocimiento general que puede especializarse mediante fine-tuning.


Modelos Fundacionales: La base del Fine-Tuning

Beneficios de la reutilización

 **Sostenibilidad ambiental:** Reduce significativamente la huella de carbono al evitar entrenamientos completos

 **Eficiencia económica:** Reduce costos de entrenamiento hasta en un 99% comparado con entrenar desde cero

 **Ahorro de tiempo:** Reduce semanas o meses de entrenamiento a horas o días de fine-tuning

 **Transferencia de conocimiento:** Aprovecha el conocimiento general ya incorporado en el modelo base

Un estudio de Stanford estimó que entrenar un modelo grande como GPT-3 emite aproximadamente 552 toneladas de CO₂, comparable a 120 autos circulando durante un año. El fine-tuning puede reducir este impacto a menos del 1%.

⚙️ Anatomía del Fine-Tuning: ¿Qué ocurre bajo el capó?

Proceso técnico

1. **Preparación de datos:** Creación de pares de prompts y respuestas deseadas
2. **Entrenamiento continuo:** Ajuste de pesos del modelo utilizando backpropagation, pero con tasas de aprendizaje más bajas para evitar "catastrophic forgetting"
3. **Validación:** Evaluación continua para evitar sobreajuste

Tipos de Fine-Tuning

- **Full Fine-Tuning:** Ajusta todos los parámetros del modelo
- **Parameter-Efficient Fine-Tuning (PEFT):** Conjunto de técnicas que ajustan solo un pequeño subconjunto de parámetros (0.1-1%) en lugar de todo el modelo. PEFT incluye varios métodos:
 - **LoRA (Low-Rank Adaptation):** El más popular, añade matrices de bajo rango
 - **Adapters:** Inserta pequeñas capas entrenables entre las existentes
- **Instruction Tuning:** Enfocado en seguir instrucciones específicas (Ej: de GPT 3 a ChatGPT)
- **RLHF/DPO:** Alineación con preferencias humanas

Durante el Fine-tuning, los pesos de la red neuronal se modifican gradualmente para minimizar la diferencia entre las predicciones del modelo y las respuestas deseadas, similar a "reescribir" partes del conocimiento del modelo.

¿Cómo funciona LoRA (Low-Rank Adaptation) paso a paso?

1. **Identificación de matrices clave:** Se seleccionan matrices de pesos específicas en el modelo que son cruciales para la tarea objetivo
2. **Congelamiento de parámetros originales:** El 99.9% de los parámetros del modelo base se mantienen intactos y congelados
3. **Creación de "atajos neuronales":**
 - Para cada matriz de pesos W original (ej: 4096×4096)
 - Se crean dos pequeñas matrices A (4096×8) y B (8×4096)
 - El producto $A \times B$ forma una matriz de rango reducido ($\text{rank}=8$)
 - Este producto $A \times B$ captura los cambios necesarios específicos para la tarea

🧩 Cómo funciona LoRA (Low-Rank Adaptation) paso a paso?

4. **Aprendizaje adaptativo:** Durante el entrenamiento, solo se actualizan las pequeñas matrices A y B
5. **Combinación en inferencia:** Las predicciones finales combinan:
 $W_{\text{original}} + \alpha(A \times B)$, donde α es un factor de escala para controlar la influencia de la adaptación

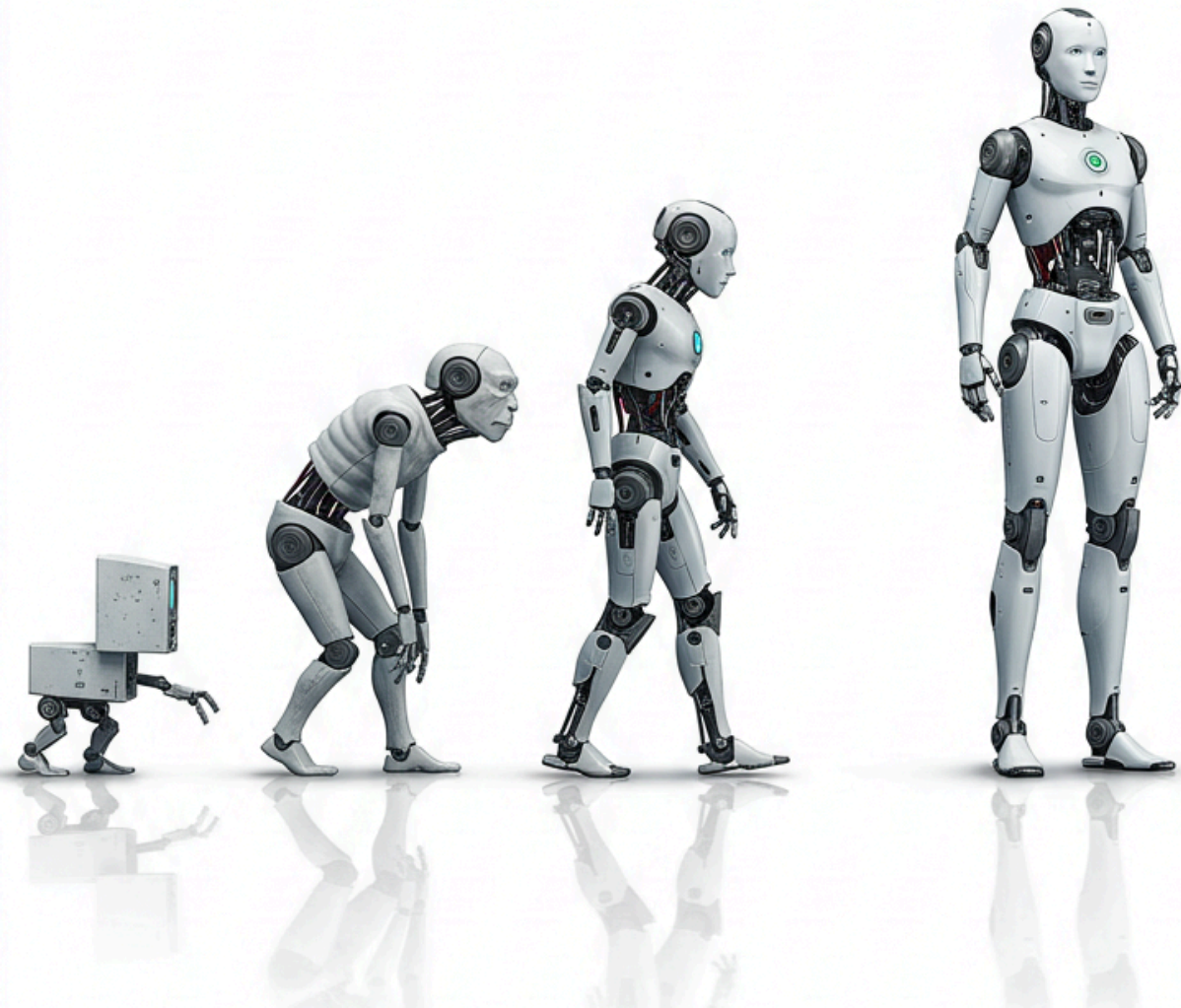
Imagina un enorme panel de control (el modelo) con millones de perillas. En lugar de ajustar todas las perillas individualmente, LoRA añade unos pocos "controles maestros" (matrices A y B) que pueden influir en múltiples perillas simultáneamente, logrando el mismo efecto con mucho menos esfuerzo.

Ventajas de PEFT


- **Eficiencia de memoria:** Requiere hasta 10x menos memoria GPU
- **Velocidad:** Entrenamiento hasta 5x más rápido
- **Economía:** Reduce costos de computación significativamente
- **Modularidad:** Permite múltiples adaptaciones para diferentes tareas
- **Portabilidad:** Adaptadores pequeños (~10-100MB) vs modelos completos (~30GB)


LoRA (Low-Rank Adaptation) inserta pequeñas matrices de bajo rango que capturan los cambios necesarios sin modificar la mayoría de los parámetros originales, logrando resultados comparables con solo ajustar aproximadamente el 0.1-1% de los parámetros.


Fine-Tuning



RAG: Conectando LLMs con el mundo exterior

 **Retrieval Augmented Generation** permite a los LLMs acceder a información externa, combinando la generación de texto con búsqueda en tiempo real para proporcionar respuestas precisas y actualizadas sin necesidad de reentrenar el modelo completo.

 Este enfoque crea un "cerebro externo" para el modelo, permitiéndole consultar bases de conocimiento especializadas, documentos privados o información actualizada que no estaba disponible durante su entrenamiento original.

 **RAG resuelve el problema de "alucinaciones"** al anclar las respuestas del modelo en fuentes verificables, similar a cómo un investigador consulta referencias antes de emitir una opinión, mejorando significativamente la precisión y confiabilidad.

⚙️ Arquitectura de un sistema RAG: El flujo de la información

Componentes clave

1. **Chunking:** División de documentos en fragmentos manejables
2. **Embeddings:** Transformación de texto en vectores semánticos que capturan su significado
3. **Vector Database:** Almacenamiento optimizado para búsqueda semántica
4. **Retriever:** Sistema que encuentra información relevante basado en similitud semántica
5. **LLM:** Modelo que genera respuestas contextualizadas con la información recuperada

⚙️ Arquitectura de un sistema RAG: El flujo de la información

Flujo de procesamiento

Fase de indexación:

- Los documentos se dividen en chunks
- Cada chunk se convierte en embeddings
- Los vectores se almacenan en la base de datos

Fase de consulta:

- La pregunta del usuario se vectoriza
- Se buscan chunks semánticamente similares
- Se envían al LLM junto con la pregunta
- El LLM genera una respuesta informada

El proceso RAG paso a paso

1. Preprocesamiento:

- Carga de documentos (PDFs, sitios web, bases de datos)
- Limpieza y preprocesamiento de los datos (stops words, caracteres especiales ...)
- Segmentación en chunks (párrafos, secciones)
- Generación de embeddings mediante modelos como BERT, Ada o E5

2. Almacenamiento:

- Indexación de vectores en bases como Pinecone, Weaviate o Chroma
- Mantenimiento de metadatos para contexto adicional

El proceso RAG paso a paso

3. Recuperación:

- Conversión de la consulta del usuario a vector
- Búsqueda de similitud (distancia coseno)
- Selección de los k fragmentos más relevantes

4. Generación aumentada:

- Construcción de prompt enriquecido con contexto recuperado
- Generación de respuesta que sintetiza la información
- Citación de fuentes para transparencia

Comparación: Fine-Tuning vs RAG

Fine-Tuning

Ventajas:

- Mejora comportamientos intrínsecos
- Menor latencia (no requiere búsquedas)
- Aprende patrones específicos

Desventajas:

- Requiere ejemplos de calidad
- Costoso computacionalmente
- Se desactualiza con el tiempo
- Limitado al tamaño del contexto

RAG

Ventajas:

- Información siempre actualizable
- Mayor transparencia (citas)
- Reduce alucinaciones
- Acceso a conocimiento privado

Desventajas:

- Mayor complejidad técnica
- Latencia más alta
- Calidad depende de la recuperación

Comparación: Fine-Tuning vs RAG

Fine-Tuning

Ideal para:

- Adaptación de estilo o formato
- Tareas especializadas recurrentes
- Mejora de instrucciones específicas

RAG

Ideal para:

- Sistemas que requieren información actualizada
- Consulta a documentación privada o especializada
- Aplicaciones donde la precisión factual es crucial

Muchos sistemas modernos combinan ambos enfoques: Fine-tuning para mejorar el rendimiento base y RAG para proporcionar información actualizada y precisa en tiempo real.



Soluciones híbridas: Lo mejor de ambos mundos

Sistemas avanzados que combinan Fine-tuning y RAG:

1. **RAG con retriever fine-tuneado:** El componente de recuperación se ajusta específicamente al dominio
2. **Fine-tuning para mejorar la integración de contexto:** El modelo se entrena para utilizar mejor la información recuperada
3. **Sistemas adaptativos:** Utilizan RAG cuando la información es nueva o cambiante, y conocimiento interno cuando es estable
4. **LLMs fine-tuneados como routers de información:** Modelos especializados que deciden qué fuentes consultar para cada consulta



Conclusiones: Ampliando horizontes de los LLMs

- **El Fine-tuning personaliza el comportamiento intrínseco** de los modelos para adaptarlos a necesidades específicas, enseñándoles nuevas "habilidades"
- **RAG expande el conocimiento accesible** al conectar LLMs con información externa sin alterar su estructura interna
- **Los sistemas híbridos representan el estado del arte**, combinando la especialización del fine-tuning con la flexibilidad y actualización del RAG

Estos enfoques transforman LLMs generalistas en asistentes especializados adaptados a contextos profesionales específicos, mejorando significativamente su precisión, utilidad y confiabilidad en entornos donde el error no es una opción.



Material Complementario: Recursos Interactivos



Para explorar en profundidad

- **Guía interactiva de Fine-tuning:**
Explicación paso a paso del proceso
[Hugging Face - Fine-tuning guides](#)
- **RAG en acción:**
Implementaciones prácticas con diferentes LLMs
[LangChain Documentation](#)
- **Herramientas empresariales:**
Plataformas que facilitan implementación
[LlamaIndex Documentation](#)



Herramientas recomendadas

- **Para Fine-Tuning:**
 - [OpenAI Fine-tuning API](#)
 - [PEFT Library](#)
 - [TRL \(Transformer Reinforcement Learning\)](#)
- **Para RAG:**
 - [LangChain](#)
 - [LlamaIndex](#)
 - [Weaviate Vector Database](#)
 - [Pinecone](#)

Recursos del Curso

Plataformas y Enlaces Principales

GitHub del curso

 github.com/CamiloVga/IA_Aplicada

Asistente IA para el curso

 [Google Notebook LLM](#)