

multi versica ACADEMY

Sistemas LLM:

LLaMa 3.1

Optimización Avanzada de Modelos de Lenguaje



IA Avanzada



Procesamiento de Datos



Optimización



Deployment



Sector público LATAM



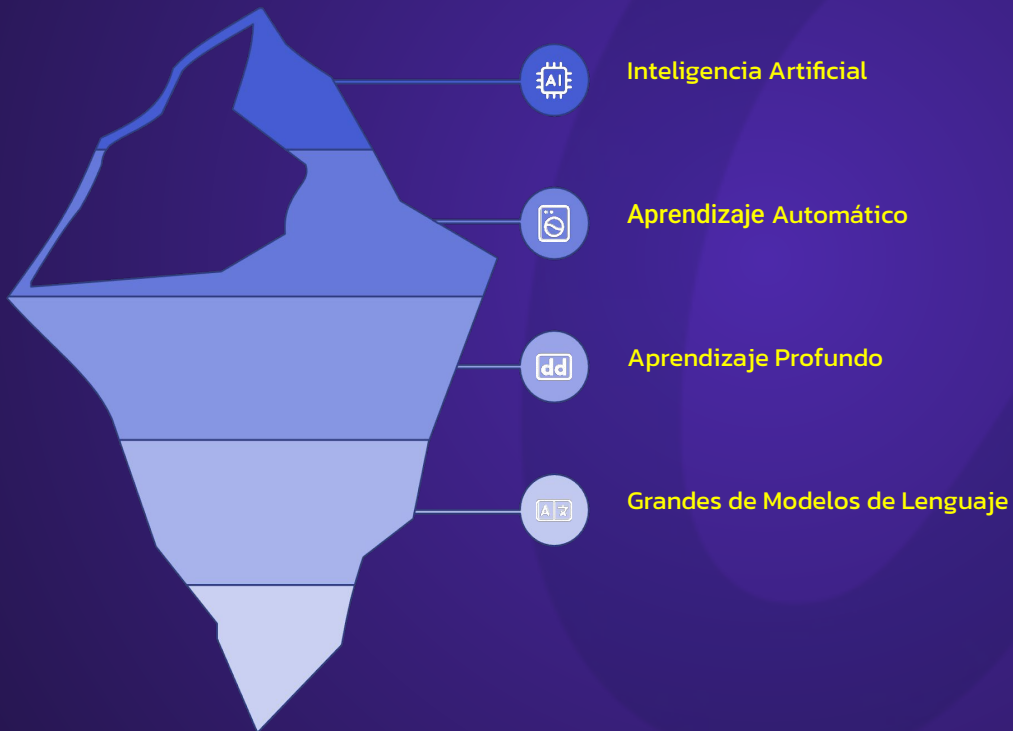
Nivel Medio Alto



Enfoque técnico

Recapitulación

Jerarquía de la IA para LLM



Recapitulación

Transformer

- Imagen del modelo
- Encoder/Decoder
- Posición de las palabras y captura de significado de lenguaje

LLM

- Origen y Evolución
- Contexto
- Tokenización
- Embeddings
- Tipo de prompts (e.g. CoT)
- Uso de LLMs:
 - Contenido
 - Búsqueda
 - Asistentes conversacionales (Chats)
 - Programación
 - Traducción

Historia de LLaMa

	LLama	LLama 2	Code LLama	Code LLama	LLama 3	LLama 3.1	LLama 3.2	LLama 3.3	LLama 4 Scout	LLama 4 Maverick
Año	2023, Feb	2023, Jul	2023, Ago	2024, Ene	2024, Abr	2024, Jul	2024, Sept	2024, Dic	2025, Abr	2025, Abr
Tamaños (B)	13, 65	7, 13, 70	7, 13, 34	70	8, 70	8, 70, 405	1, 3, 11, 90	70	MoE, 109 (total), 17 (activos)	MoE, 400 (total), 17 (activos)
Capacidad	Texto	Texto	Code	Code	Texto	Texto	Input: Texto/Imagen Output: Texto	Texto	Input: Texto/Imagen Output: Texto	Input: Texto/Imagen Output: Texto



Tip: Se puede usar **LLaMa 4** en WhatsApp

Arquitectura LLaMA 3.1

Tamaños de Modelos Disponibles

LLaMA 3.1 8B

Eficiente

Modelo compacto optimizado para despliegue en hardware limitado. Ideal para aplicaciones gubernamentales con restricciones de recursos.

8B parámetros ~16GB RAM

Soporte multilingüe

LLaMA 3.1 70B

Alto Rendimiento

Modelo equilibrado entre rendimiento y recursos. Recomendado para aplicaciones RAG de misión crítica en administraciones públicas.

70B parámetros ~140 GB RAM

Alto rendimiento

LLaMA 3.1 405B

Enterprise

Modelo de máxima capacidad para las tareas más complejas. Óptimo para análisis avanzados y sistemas a escala nacional.

405B parámetros

Cluster necesario Máxima precisión

Ventajas Competitivas



Contexto Extendido

Ventana de contexto de hasta 128 K tokens, ideal para análisis de documentos legales extensos.



Soporte Multilingüe

Optimizado para español y dialectos latinoamericanos con más de 12 idiomas soportados.



Seguridad Integrada

Diseñado con guardrails y herramientas de seguridad específicas para entornos gubernamentales.



Facilidad de Fine-tuning

Arquitectura optimizada para PEFT con adaptadores de bajo rango (LoRA/QLoRA).



Optimización Hardware

Soporte para cuantización a 4/8 bits y deployment eficiente en infraestructura pública.

Demo rápido de Ollama



Objetivos de la Presentación

- **Comprender los conceptos de Fine-Tuning y PEFT**
- **Profundizar en LoRA y QLoRA**
- **Entender RAG**
- **Implementar laboratorios guiados con LLaMA 3.1**
- **Buenas prácticas y monitoreo técnico**
- **Evaluación y casos reales**

Fine-Tuning vs RAG



Fine-Tuning

El modelo aprende e incorpora los conocimientos en su base

El sistema obtiene los conocimientos necesarios en el momento para que el modelo los utilice

RAG





Fine-Tuning

Fundamentos de Fine-Tuning

El **fine-tuning** es el proceso de ajustar un modelo de lenguaje previamente entrenado para mejorar su rendimiento en tareas específicas mediante entrenamiento adicional con datos relevantes al dominio.

Propósito del Fine-Tuning

Especializar modelos LLM para:

- Dominios específicos (legal, médico, financiero)
- Tareas particulares (clasificación, generación)
- Adaptación a nuevos idiomas o variantes lingüísticas
- Reducción de sesgos y alineación con valores

Aplicaciones

- Chatbots especializados por sector
- Sistemas con conocimiento contextual
- Asistentes con identidad corporativa
- Optimización de costes operativos
- Mejora en precisión para tareas críticas



Pre-training vs Fine-tuning vs In-context learning

Comparativa de las tres principales estrategias para especializar modelos de lenguaje

Pre-training

Definición

Entrenamiento inicial con grandes volúmenes de datos generales para aprender representaciones y patrones del lenguaje.

Características

- Requiere millones/billones de tokens
- Computacionalmente intensivo
(Llama 3.1 8B: 39.3M GPU hours)
- Desarrolla capacidades generales
- Entrenamiento masivo y autodirigido

Pros y Contras

- + Base robusta
- Alto coste
- + Conocimiento general
- No especializado

Fine-tuning

Definición

Adaptación de un modelo pre-entrenado mediante actualización de pesos en entrenamientos específicos con datasets reducidos.

Características

- Modifica pesos del modelo
- Requiere dataset especializado
- Actualización del modelo
- Balance entre conocimiento general y específico

Pros y Contras

- + Alta especialización
- Recursos GPU
- + Training más rápido

In-context learning

Definición

Adaptación mediante ejemplos incluidos en el prompt, sin modificar parámetros del modelo en tiempo real.

Características

- No actualiza pesos del modelo
- Basado en ejemplos (few-shot)
- Flexibilidad y adaptación inmediata
- Limitado por el contexto disponible






Pros y Contras

- + Sin entrenamiento
- Ventana limitada
- + Adaptabilidad
- Menos preciso

Introducción a PEFT

Parameter Efficient Fine-Tuning (PEFT) es un conjunto de técnicas que permiten ajustar modelos de lenguaje de gran tamaño modificando solo una pequeña fracción de sus parámetros, en lugar de actualizar todos los pesos durante el proceso de fine-tuning.

Ventajas principales

-  Reducción drástica de recursos computacionales
-  Menor consumo de memoria
-  GPU/RAM
-  Tiempos de entrenamiento significativamente menores
-  Prevención del overfitting y catastrophic forgetting

Posibilidad de fine-tuning con datasets más pequeños

Comparativa de Recursos

Fine-tuning Tradicional

Actualiza todos los parámetros del modelo

100%

PEFT

Actualiza solo una fracción de parámetros

< 1%





Aplicaciones Prácticas

- Adaptación de LLMs a dominios específicos
- Personalización de modelos con recursos limitados
- Ajuste rápido para casos de uso corporativos

Low Rank Adaptation (LoRA): Conceptos Clave

LoRA es una técnica de Parameter Efficient Fine-Tuning (PEFT) que reduce significativamente los parámetros a actualizar durante el fine-tuning, congelando la mayoría de los pesos del modelo *pre trained* y añadiendo matrices de bajo rango entrenables.

Ventajas Técnicas

-  Reducción drástica de memoria requerida (hasta 10x menos)
-  Entrenamiento más rápido y eficiente energéticamente
-  Mitigación del "catastrophic forgetting" durante fine-tuning
-  Adaptadores intercambiables para múltiples tareas

Las matrices LoRA pueden ser fusionadas con los pesos originales durante la inferencia, sin aumentar la latencia del modelo.

Principios Matemáticos

LoRA se basa en la hipótesis de que las actualizaciones de los pesos durante el fine-tuning tienen un **rango intrínsecamente bajo**.



Concepto Clave

Descompone las actualizaciones de peso en matrices de bajo rango: $\Delta W = A \times B$

Implementación en Frameworks

LoRA se implementa principalmente en la biblioteca **PEFT** de Hugging Face, permitiendo fine-tuning eficiente de cualquier arquitectura Transformer:

- Compatible con modelos como LLaMA, Mistral, GPT-2/Neo, T5
- Configurable según capas objetivo (attention, MLP)
- Ajustable mediante hiper parámetros como rango (r) y alpha

Quantized LoRA (QLoRA): Eficiencia Mejorada

QLoRA evoluciona de LoRA aplicando cuantización de 4 bits al modelo base mientras entrena adaptadores de rango bajo en precisión completa. Esta técnica permite fine-tuning en recursos limitados con calidad similar.

Cuantización a 4 bits

- Compresión de pesos del LLM base de FP16/FP32 a INT4
- Técnicas especializadas: double quantization
- Vectorización por bloques para precisión
- Paginación para gestión de memoria eficiente

Ventajas sobre LoRA

- Reducción drástica del consumo de memoria (hasta 75%)
- Posibilita fine-tuning en una sola GPU de gama media
- Mantiene calidad cercana al fine-tuning completo
- Evita la degradación de la calidad del modelo
- Compatible con arquitecturas de transformers

Comparativa LoRA vs QLoRA

Característica	LoRA	QLoRA
Precisión base	16-bit	4-bit -75%
Uso de memoria	Alto	Muy bajo
Hardware	A100/H100	RTX series
Calidad	Alta	Comparable

Memoria GPU requerida para modelo



Pipeline de Entrenamiento QLoRA

Proceso paso a paso para aplicar fine-tuning eficiente con LLaMA 3.1

1

Carga del modelo cuantizado

Configuración de BitsAndBytes para cuantización a 4 bits

2

Aplicación de LoRA

Configuración de adaptadores de bajo rango para entrenamiento eficiente

3

Entrenamiento con Trainer

Configuración del loop de entrenamiento con HuggingFace Trainer

Monitoreo y Evaluación



 Tracking con W&B

 Métricas de rendimiento

 Consumo de memoria



Ventajas clave

- ♦ Entrenamiento en GPU estándar
- ♦ Reducción de 75% en memoria
- ♦ Preservación de conocimiento base
- ♦ Adaptadores intercambiables

Tiempo estimado de
entrenamiento:

~3 horas (A100)

~10 horas (V100)

Troubleshooting y Buenas Prácticas QLoRA



Prevención de Overfitting

- Configurar **LoRA dropout** entre 0.05–0.1 para modelos gubernamentales
- Implementar early stopping basado en pérdida de validación
- Utilizar regularización de pesos específica para adaptadores



Gestión de Checkpoints

- Guardar checkpoints cada 100–200 pasos de entrenamiento Implementar
- estrategia **keep_last_n** para optimizar almacenamiento Guardar solo
- adaptadores LoRA (~15MB) vs modelo completo (~15GB)



Monitoreo Efectivo

- Integrar Weights & Biases para visualización en tiempo real
- Monitorear **training/validation loss, perplexity** y tiempos
- Implementar callbacks para alertas automáticas durante entrenamiento

Benchmark: 5.2 horas para fine-tuning LLaMA 3.1 8B en RTX 4090

Problemas Comunes y Soluciones

Explosión de gradientes
Implementar gradient clipping (`max_grad_norm=1.0`)

Lenta convergencia
Ajustar el learning rate entre $1e-4$ y $3e-4$ con warmup

Error CUDA OOM
Reducir batch size, incrementar gradient accumulation steps

Sesgos en outputs
Balancear datasets de entrenamiento, evaluar con métricas de fairness

Recomendación para entornos gubernamentales

Documentar exhaustivamente la configuración de entrenamiento para garantizar reproducibilidad y cumplimiento normativo

Métricas y Comparativas en Clasificación

Modelo	Precision	Recall	F1-Score	Mejora %
LLaMA 3.1 (base)	0.68	0.72	0.70	-
Fine-tuning completo	0.84	0.86	0.85	↑ 21.4%
LoRA	0.82	0.83	0.82	↑ 17.1%
QLoRA	0.80	0.82	0.81	↑ 15.7%



Análisis Pre- Post Fine- Tuning

- Reducción significativa de falsos positivos en clasificación
- Mayor precisión en tareas de sentiment analysis gubernamental
- Mejor adherencia al contexto proporcionado por RAG

Impactos Operativos

- Tiempo de respuesta reducido en 28% en sistemas de producción
- Reducción de escalamientos manuales en atención ciudadana
- Clasificación precisa de documentos administrativos

Implementación Práctica: Configuración de QLoRA

La configuración adecuada de QLoRA requiere definir varios parámetros clave que determinan el equilibrio entre eficiencia computacional y calidad del fine-tuning.

Parámetros Fundamentales

- **r**: Rango de las matrices de adaptación (4-64)
- **alpha**: Escala de inicialización, típicamente $r \times 2$
- **dropout**: Regularización (0.0-0.2)
- **bits**: Precisión de cuantización (4-bit para QLoRA)

Selección de Capas

La elección de módulos a adaptar impacta significativamente en:

- Calidad del fine-tuning
- Velocidad de entrenamiento
- Tamaño del adaptador resultante
- Comportamiento en inferencia

Recomendaciones 💡

- Adaptar todas las capas lineales mejora resultados
- Para modelos más pequeños: $r=8$, $\alpha=16$
- Para fine-tuning: enfoque en capas de atención
- Monitorear convergencia para ajustar parámetros

Manejo de los **Datos** en Fine-Tuning

Mejores Prácticas para **Datasets** de Entrenamiento

La calidad de los datos es esencial para el éxito de cualquier proyecto de fine-tuning. Un dataset óptimo debe ser representativo, balanceado y adecuado para el caso de uso específico.

Calidad y Diversidad

- **Precisión:** Datos actualizados y verificados
- **Compleitud:** Sin omisiones críticas
- **Relevancia:** Adaptados al dominio específico
- **Diversidad:** Representación de casos de uso variados

Estrategias de Curación

- Filtrado automático de ruido y redundancia
- Normalización estructural y semántica
- Anotaciones de calidad por expertos
- Balanceo estratégico de categorías y ejemplos



Balance y Representatividad

Distribución equilibrada que represente la realidad del problema y evite sesgos sistemáticos. Incluir casos de borde y excepciones comunes.

Dataset Balanceado



Validación y División

División estratégica en conjuntos de:

- ✓ **Train (70-80%):** Entrenamiento principal
- ✓ **Validation (10-15%):** Ajuste de hiperparámetros
- ✓ **Test (10-15%):** Evaluación final de rendimiento



Mantenimiento y Actualización

Revisión periódica para mantener la relevancia temporal. Incorporación de nuevos ejemplos y casos emergentes. Proceso iterativo de mejora continua.

Técnicas de Limpieza y Preprocesamiento

La **calidad de los datos** es crucial para el éxito de los sistemas RAG. El preprocesamiento asegura que la información sea precisa, consistente y estructurada para optimizar la recuperación.

Procedimientos Esenciales

Eliminación de Ruido



- Eliminar caracteres especiales y HTML
- Filtrar información irrelevante
- Remover elementos decorativos

Normalización



- Unificación de formato (mayúsculas/minúsculas)
- Normalización de caracteres (UTF-8)
- Estandarización de términos específicos

Flujo de Preprocesamiento

```
# Documento original doc = """ Producto:
Servidor X500
Características PRINCIPALES: * CPU: Intel i9 12900K (3.2 Ghz) * RAM:
64GB DDR5 * Detalles adicionales en www.empresa.com """
```



1

Extracción de Texto

Elimina etiquetas HTML, normaliza espacios



2

Estructuración

Extrae metadatos (producto, características)



```
# Documento procesado { "producto": "Servidor X500",
"características": { "cpu": "Intel i9 12900K 3.2 Ghz", "ram":
"64GB DDR5" } }
```

Creación de Plantillas de Conversación

Los **plantillas de conversación** son patrones estructurados que definen cómo se formatea la interacción entre usuario y modelo para el entrenamiento de LLMs.

Principios de Diseño

- **Consistencia:** Mantener estructura uniforme
- **Claridad:** Separar claramente instrucción/contexto/respuesta
- **Especificidad:** Incluir metadatos relevantes
- **Naturalidad:** Reflejar conversaciones reales

Impacto en los Resultados

- Mejora la coherencia entre consulta y respuesta
- Reduce ambigüedad en instrucciones
- Facilita la incorporación de conocimiento RAG
- Permite control sobre formato y tono de respuestas
- Aumenta la consistencia entre distintas interacciones

Formato Alpaca

Instructivo

```
### Instrucción:  
Crea un resumen técnico del siguiente documento sobre servidores.  
  
### Contexto:  
{documento_recuperado_por_RAG}  
  
### Respuesta:  
{respuesta_modelo}
```

Formato ChatML

Conversacional

```
<|im_start|>system  
Eres un asistente especializado en documentación técnica.  
<|im_end|>  
  
<|im_start|>user  
¿Qué dice este documento sobre arquitecturas de  
microservicios?  
{contexto_recuperado}  
<|im_end|>  
  
<|im_start|>assistant  
{respuesta_modelo}  
<|im_end|>
```


Validación y División de Datasets

La **división adecuada** del dataset es crucial para garantizar que el modelo pueda generalizar correctamente y evitar problemas como el sobreajuste.

División Estratégica

- **Training (70-80%):** Datos para entrenamiento directo del modelo
- **Validation (10-15%):** Ajuste de hiperparámetros y monitoreo
- **Test (10-15%):** Evaluación final del rendimiento

Buenas Prácticas

- Mantener la distribución de clases en todas las particiones
- Evitar fugas de datos entre conjuntos
- Estratificar en casos de desbalanceo
- Validación cruzada para datasets pequeños
- Asegurar representatividad en cada partición

Training Set

- Optimiza parámetros del modelo
- Mayor volumen de ejemplos
- Debe representar todo el dominio

70%

Validation Set

- Ajuste de hiperparámetros
- Detección de early stopping
- Monitoreo de overfitting

15%

Test Set

- Evaluación final
- Simula datos reales en producción

15%

Métricas de Calidad

Perplexity

Mide incertidumbre del
modelo al predecir texto

BLEU/ROUGE

Similitud entre respuestas
y referencias

Coherencia

Lógica y consistencia
de respuestas

Validación humana

Evaluación experta de
calidad

Monitoreo del Fine-Tuning

Monitoreo con Weights & Biases

Weights & Biases (W&B) es una herramienta de MLOps que permite visualizar y comparar experimentos de fine-tuning en tiempo real, facilitando el seguimiento de métricas, hiperparámetros y artefactos.

Configuración para Fine-Tuning

```
import wandb
wandb.init(project="llama-qlora",

            name="experimento-01",
            config={
                "r": 8,
                "lora_alpha": 16,
                "epochs": 3
            })
```

Beneficios del Monitoreo

- **Detección temprana** de problemas (overfitting, exploding gradients)
- **Comparación de múltiples** ejecuciones y configuraciones
- **Coaboración** en equipo y reproducibilidad
- **Optimización de hiperparámetros** con datos objetivos

Métricas de Entrenamiento



Training/validation loss, accuracy, perplexity
Velocidad de entrenamiento (ejemplos/segundo)

Seguimiento de Hiperparámetros



Rango, alpha, learning rate, batch size, target modules
Matrices de barrido paramétrico

Dashboard: Fine-Tuning LLaMA 3.1 con QLoRA

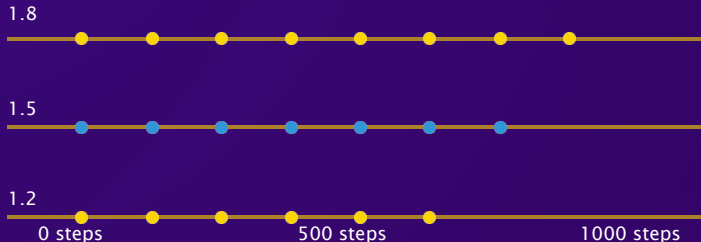


Métricas

Hiperparámetros

Artefactos

Training Loss



Configuración de Tracking de Experimentos

El seguimiento sistemático de experimentos de fine-tuning permite **comparar resultados**, **reproducir éxitos** y **tomar decisiones basadas en datos**.

Organización en W&B

- **Proyectos:** Agrupar experimentos relacionados
- **Runs:** Ejecutar instancias de entrenamiento
- **Artifacts:** Guardar modelos, datasets y resultados
- **Tags:** Etiquetar para búsqueda y filtrado rápido

Toma de decisiones

La configuración del seguimiento debe optimizarse para:

1

Comparar distintas configuraciones de hiperparámetros

2

Identificar señales tempranas de overfitting

3

Seleccionar modelos óptimos para producción

Dashboard de W&B

LLaMa 3.1 QLoRA

12 experimentos

Training Loss



BLEU Score

34.2

ROUGE-L

0.67

Training Time

2.3h

GPU Memory

14.6GB

● llama3_qlora_r8_alllinear

Completado

○ llama3_qlora_r16_alllinear

Ejecutando

Evaluación de Resultados y Comparativas

El análisis **antes/después** del fine-tuning es esencial para validar la efectividad de la adaptación del modelo.

- **Métricas objetivas:** Precisión, recall, F1-score
- **Métricas perceptuales:** Relevancia, coherencia, fidelidad
- **Métricas de negocio:** Tiempo de respuesta, costes operativos

Criterios de éxito

- Reducción de **alucinaciones** en respuestas
- Mayor adherencia a la información de **contexto**
- Respuestas **concretas y pertinentes**
- Latencia de respuesta adecuada
- **Mejora en la satisfacción** del usuario

Outputs comparativos

Antes: "La memoria VRAM necesaria para fine-tuning depende del tamaño del modelo. Para un modelo grande, necesitas aproximadamente 24GB."

Después: "Para fine-tuning del modelo LLaMA 3.1 8B con QLoRA, se requieren 16GB de VRAM con cuantización a 4-bits, mientras que el fine-tuning completo requeriría 32GB."

Comparativa de rendimiento

Precisión en recuperación	↑ +28%
Relevancia contextual	↑ +34%
Tasa de alucinaciones	↓ -65%
Tiempo de inferencia	↑ +5%

Sin Fine-tuning 40%

Fine-tuning estándar 68%

QLoRA + RAG 88%

Casos de uso y buenas prácticas en Fine-Tuning

Casos de Estudio y Buenas Prácticas

Aplicaciones del **Fine-Tuning** en diferentes sectores empresariales y recomendaciones para un despliegue exitoso.

Aplicaciones Sectoriales



Sector Salud

- Especializado en literatura médica
- Reducción de hallucinations críticas en diagnósticos
- Mejora del 85% en precisión sobre información clínica



Sector Gobierno

- Adaptación de modelos a normativas específicas
- Fine-tuning para detectar riesgos en documentos
- Procesamiento eficiente de informes trimestrales

Errores Comunes y Soluciones

⚠ Overfitting en Datasets Pequeños

Uso de conjuntos de datos insuficientes que llevan a modelos con poca generalización.

✅ **Solución:** Aumentación de datos, regularización y monitoreo constante de validación.

⚠ Desalineación Prompt-Retrieval

Fine-tuning sin considerar el formato real de los prompts en producción.

✅ **Solución:** Dataset de entrenamiento que refleje casos de uso reales.

Recomendaciones Finales

- ✅ Establecer métricas claras de evaluación **antes** del fine-tuning
- ✅ Documentar configuraciones y datasets para reproducibilidad
- ✅ Implementar sistemas de evaluación continua post-despliegue
- ✅ Balancear ejemplos positivos y negativos en datasets RAG

Aprendizajes Clave

- ✓ Las técnicas **PEFT** reducen significativamente los recursos necesarios para fine-tuning.
- ✓ **QLoRA** permite fine-tuning en hardware más modesto sin sacrificar calidad.
- ✓ La calidad de los datos es fundamental para el éxito de sistemas.
- ✓ El monitoreo y evaluación continua son clave para la mejora iterativa.
- ✓ La adaptación de modelos específicos por dominio supera a los modelos generalistas en tareas especializadas.

Checklist para Fine-Tuning exitoso

- ✓ Seleccionar un modelo base apropiado para la tarea
- ✓ Preparar datasets balanceados y representativos
- ✓ Definir los parámetros adecuados de LoRA/QLoRA
- ✓ Implementar evaluación automática y humana
- ✓ Monitorizar métricas clave durante el entrenamiento
- ✓ Realizar pruebas con casos reales antes de despliegue

Retrieval Augmented Generation (RAG)

Laboratorio: Preparación de Dataset RAG en Español



Flujo de Trabajo RAG



Documentos en Español

PDF, TXT, DOCX, HTML



Limpieza y Normalización

Acentos, Espacios, Metadatos



Chunking

800-1000 caracteres por fragmento



Template Conversacional

Formato pregunta-respuesta

Preparación de Datos para Sistemas RAG

La calidad y estructura de los datos son factores críticos para el éxito de un sistema RAG efectivo. La preparación adecuada permite mejor recuperación contextual y respuestas más precisas.



Extracción



Limpieza



Chunking



Embedding

Impacto en la Calidad RAG

- Mejora 68% en precisión de recuperación
- Reducción 75% en hallucinations
- Contexto relevante en 93% de consultas



Extracción y Conversión

Transformación desde formatos diversos (PDF, HTML, DOCX, bases de datos) a textos procesables. Preservación de estructura semántica y metadata relevante.



Limpieza y Normalización

Eliminación de ruido, redundancias y elementos decorativos. Normalización de caracteres especiales y estandarización de terminología específica del dominio gubernamental.



Chunking Semántico

Segmentación inteligente que preserva la coherencia temática. Definición de tamaño óptimo (800-1000 caracteres) con solapamiento controlado para mantener contexto.



Bases Vectoriales

Generación de embeddings e indexación en bases vectoriales (Pinecone, Faiss, Weaviate) para búsqueda semántica eficiente en grandes volúmenes de datos regulatorios.

Herramientas Recomendadas



LangChain



Faiss



NLTK



SpaCy

Formatos y Plantillas de Datos

La estructura de los datos es fundamental para el rendimiento de sistemas RAG y chatbots conversacionales. Los formatos adecuados optimizan la recuperación y generación.

JSON Estructurado

Estructura jerárquica para datos RAG que facilita indexación y búsqueda por campos.

```
{
  "producto": "LLaMA 3.1 8B", "categoría":
  "Large Language Model", "descripción":
  "Modelo optimizado...", "capacidades":
  ["RAG", "Fine-tuning"], "requisitos": {
    "memoria": "16GB",
    "disco": "20GB"
  }
}
```

- ✓ Ideal para catálogos y FAQs
- ✓ Fácil indexación por campos

ChatML Multi-turno

Estructura conversacional que mantiene el contexto entre múltiples intercambios usuario-asistente.

```
{ "role": "system", "content": "Asistente
RAG..." }
{ "role": "user", "content": "¿Qué es LLaMA
3.1?" }
{ "role": "assistant", "content": "LLaMA 3.1
es..." }
{ "role": "user", "content": "¿Puedo hacer
fine-tuning?" }
{ "role": "assistant", "content": "Sí,
```

- ✓ Mantiene contexto completo
- ✓ Ideal para asistentes interactivos

Alpaca Instructivo

Template popular con estructura clara para instrucciones y contexto, ideal para fine-tuning.

Instrucción: Genera una descripción técnica para este producto.

Entrada: Modelo: LLaMA 3.1 8B
Categoría: Large Language Model

Respuesta: LLaMA 3.1 8B es un modelo de lenguaje de 8 billones de

- ✓ Simple y eficaz para fine-tuning
- ✓ Estructura clara para instrucciones

 La elección del formato depende del caso de uso específico de RAG y las capacidades del modelo

Técnicas de Chunking y Preprocesado



Segmentación de Textos (Chunking)

División estratégica de documentos en fragmentos procesables

- ♦ **Chunking semántico:** preserva contexto y significado
- ♦ **Chunking por longitud:** fragmentos de tamaño consistente
- ♦ **Chunking recursivo:** división jerárquica por separadores

```
text_splitter = RecursiveCharacterTextSplitter( chunk_size=800,
chunk_overlap=100, separators=["\n\n", "\n", ".", " ", ""])

```



Limpieza de Ruido

Eliminación de elementos que reducen la calidad de datos

Normalización de caracteres y espacios

- ♦ Eliminación de etiquetas HTML y código decorativos y repetitivos
- ♦ Filtrado de elementos decorativos y repetitivos
- ♦ Corrección de caracteres especiales y codificación

Extracción de Metadatos

Captura de información contextual para mejorar recuperación

- ♦ Identificación de entidades y relaciones
- ♦ Extracción de fechas, autores y categorías
- ♦ Generación de tags temáticos y referencias



Herramientas Recomendadas



NLTK

Tokenización, análisis morfológico y filtrado de stopwords



SpaCy

Reconocimiento de entidades, análisis sintáctico avanzado



LangChain

Loaders, text splitters y procesadores especializados para RAG



BeautifulSoup

Extracción y limpieza de contenido HTML/XML

Flujo de Preprocesamiento Recomendado

1. Extracción de texto raw desde documentos
2. Limpieza y normalización
3. Segmentación semántica (chunking)
4. Enriquecimiento con metadatos
5. Validación de calidad

Llama en Gobierno

Análisis de Texto con LLMs en Gobierno



Clasificación de Documentos

Categorización automática de documentos oficiales, solicitudes ciudadanas y trámites según su temática, urgencia y departamento responsable. Reduce en un 65% el tiempo de procesamiento manual.



Análisis de Sentimiento

Monitoreo del tono y satisfacción ciudadana en comunicaciones, quejas y solicitudes. Identificación de áreas de mejora y detección temprana de problemas prioritarios.



Extracción de Entidades

Identificación de nombres, organizaciones, ubicaciones, fechas y otros datos relevantes en comunicaciones para facilitar el procesamiento administrativo.



Casos de Uso en Gobierno

Atención Ciudadana Eficiente

Reducción del 78% en tiempo de respuesta mediante clasificación automática de consultas y generación de respuestas asistidas.

Análisis de Retroalimentación

Procesamiento de más de 10,000 comentarios ciudadanos mensuales, identificando tendencias y áreas de mejora con 92% de precisión.

Priorización de Solicitudes

Detección de solicitudes urgentes y casos críticos mediante análisis de texto en tiempo real, mejorando la asignación de recursos.

Seguridad y Compliance en Modelos IA



Principios Éticos en IA Gubernamental

Pilares fundamentales para el uso responsable de IA en el sector público:

- Transparencia y explicabilidad en decisiones automatizadas
- Respeto a la privacidad y protección de datos personales
- Inclusión y accesibilidad universal



Evaluación y Mitigación de Sesgos

Estrategias para identificar y reducir sesgos en modelos LLM:

- Datasets representativos y balanceados
- Métricas específicas: Regard Score y Honest Benchmark
- Evaluación continua post-implementación



Llama Guard: Protección Integrada

Suite de seguridad específica para modelos LLaMA:

- **Llama Guard 4:** Protección unificada texto-imagen
- **Prompt Guard 2:** Detección de jailbreaks e inyecciones
- **LlamaFirewall:** Orquestación de guardrails

Normativas Clave

- ✓ Leyes de protección de datos
- ✓ Requisitos de accesibilidad
- ✓ Marcos de auditoría IA
- ✓ Estándares internacionales



Seguridad por Diseño

Enfoque proactivo integrado desde la concepción del modelo

Caso Destacado

Implementación de Llama Guard en sistemas judiciales: reducción de **92%** en hallucinations y respuestas inapropiadas

Ecosistema Llama

Ecosistema LLaMA 3.1: **Suite Integral de IA**

Modelos Especializados

Llama Guard 3

Modelo especializado en seguridad que filtra contenido dañino y garantiza interacciones seguras para usuarios vulnerables

Prompt Guard

Protección contra ataques de prompt injection y manipulación maliciosa

Infraestructura Completa

Llama Stack

Interfaces estandarizadas para componentes canónicos: fine-tuning, generación de datos sintéticos, agentes

Integración Nativa

Ecosistema integrado con WhatsApp y Meta AI para implementación empresarial

Ventaja Competitiva: LLaMA es el único ecosistema open-source que ofrece herramientas de seguridad integradas y compatibilidad nativa con

Ventajas del ecosistema LLaMA

Características	LLaMA (Meta)	Mistral AI	Falcon	Otros Open Source
Suite de Seguridad	✓ Completa	✗ Limitada	✗ No	✗ No
Integración WhatsApp	✓ Nativa	✗ No	✗ No	✗ No
Protección de contenido	✓ Avanzada	⚡ Básica	⚡ Básica	⚡ Variable
Soporte multimodal	✓ Completo	⚡ Parcial	✗ No	⚡ Variable



Suite de Seguridad Integrada

Llama Guard 3 y 4 para protección de contenido
Prompt Guard contra ataques de jailbreak Llama
Firewall para protección en tiempo real Evaluación
continua de privacidad y sesgos



Integración Segura con WhatsApp

Canal prioritario para servicios gubernamentales
Cifrado de extremo a extremo nativo Procesamiento
privado de mensajes
APIs oficiales para desarrollo gubernamental



Ecosistema Meta AI

Meta AI como complemento para chatbots
Herramientas multimodales (imagen, audio, texto)
Procesamiento de consultas complejas
Compatibilidad con sistemas existentes

Patrones de Implementación **Exitosos**

Patrones de Implementación **Exitosos**

Diseño Centrado en el Usuario

- ▶ **Co-creación:** Involucrar a usuarios con discapacidades desde el diseño
- ▶ **Iteración continua:** Mejoras basadas en feedback real
- ▶ **Múltiples modalidades:** Voz, texto, táctil
- ▶ **Simplicidad:** Interfaces intuitivas y naturales

Integración Tecnológica

- ▶ **Multiplataforma:** WhatsApp, Web, móvil
- ▶ **APIs abiertas:** Interoperabilidad con tecnologías asistivas
- ▶ **Escalabilidad:** Arquitectura distribuida
- ▶ **Redundancia:** Múltiples canales de acceso

Factores de Éxito Críticos

Tiempo de Respuesta < 3 segundos

Esencial para mantener engagement

Disponibilidad 99.9%

Confiabilidad crítica para servicios públicos

Comprensión > 80%

Procesamiento efectivo de consultas

Métricas de Impacto

- ▶ **Reducción de tiempo de gestión: 75%**
- ▶ **Aumento de accesibilidad: 300%**

Lecciones Aprendidas y Errores Comunes

✓ Lo que SÍ funciona

Enfoque gradual: Implementación por fases permite refinamiento continuo

Feedback temprano: Involucrar usuarios con discapacidades en pruebas beta

Redundancia de canales: Múltiples vías de acceso evitan puntos de falla únicos

Monitoreo continuo: Métricas de usabilidad específicas para accesibilidad

Capacitación de equipo: Personal entrenado en principios de accesibilidad

✗ Errores Comunes a Evitar

Consultar discapacidad solo al final del proceso

Sobre-engineering: Soluciones complejas cuando simples son más efectivas

Falta de contingencia: No planificar para limitaciones técnicas

Pasos futuros

Recursos Adicionales de Aprendizaje

Recursos Técnicos

Documentación Oficial

- ▶ LLaMA 3.1 Model Cards y guías técnicas
- ▶ Llama Stack APIs y ejemplos
- ▶ Fine-tuning guides para aplicaciones legales
- ▶ Herramientas de evaluación de sesgo

Datasets y Modelos

- ▶ **LLaMandement dataset** - 15K pares FR
- ▶ Legal fine-tuned models en HuggingFace
- ▶ Datasets de evaluación judicial
- ▶ Benchmarks de neutralidad legal

Formación Especializada

Cursos Recomendados

- ▶ NLP for Legal Applications (Stanford)
- ▶ AI Ethics in Judicial Systems
- ▶ Fine-tuning LLMs for Domain Adaptation
- ▶ Legal Data Science y Analytics

Certificaciones

- ▶ AI in Legal Technology (MIT)
- ▶ Judicial AI Ethics Certification
- ▶ LLaMA Specialization Programs

Herramientas Prácticas

- ▶ HuggingFace
- ▶ Transformers
- ▶ LangChain
- ▶ Legal Weights & Biases
- ▶ Legal-BERT

Roadmap de Aprendizaje Continuo

Fase 1: Fundamentos (2-3 meses)

Conocimientos Base

- ▶ Fundamentos de NLP y transformers
- ▶ Arquitectura LLaMA y técnicas de fine-tuning
- ▶ Principios de ética en IA
- ▶ Marcos legales y regulatorios

Habilidades Prácticas

- ▶ Implementación básica de LLaMA
- ▶ Técnicas de LoRA y QLoRA
- ▶ Evaluación y métricas de calidad
- ▶ Manejo de datasets legales

Fase 2: Especialización (3-4 meses)

Desarrollo Avanzado

- ▶ Fine-tuning especializado para dominio legal
- ▶ Implementación de sistemas de seguridad
- ▶ Optimización de rendimiento
- ▶ Integración con sistemas existentes

Gestión y Operaciones

- ▶ MLOps para sistemas judiciales
- ▶ Monitoreo y auditoría continua
- ▶ Gestión de cambios organizacionales
- ▶ Compliance y regulación

Fase 3: Liderazgo e Innovación (Continuo)

Investigación

- ▶ Desarrollo de nuevas aplicaciones
- ▶ Contribución a papers académicos
- ▶ Participación en consorcios

Mentorship

- ▶ Formación de equipos
- ▶ Transferencia de conocimiento
- ▶ Desarrollo de estándares

Innovación

- ▶ Exploración de fronteras tecnológicas
- ▶ Colaboración interinstitucional
- ▶ Liderazgo en transformación digital

Conclusiones y Próximos Pasos

Síntesis Final

Puntos Clave Demostrados

- ▶ **Viabilidad técnica:** LLaMandement alcanzó 91.5% de precisión humana
- ▶ **Eficiencia operativa:** 600x más rápido que procesamiento manual
- ▶ **Escalabilidad probada:** 5,400 documentos en 10 minutos
- ▶ **Neutralidad verificada:** Métricas de sesgo satisfactorias

Ventajas de LLaMA 3.1

- ▶ **Ecosistema completo:** Herramientas integradas
- ▶ **Seguridad nativa:** Llama Guard y Prompt Guard
- ▶ **Open source:** Transparencia y auditabilidad
- ▶ **Soporte empresarial:** Integración Meta

Plan de Acción Recomendado

Pasos Inmediatos (1-3 meses)

- ▶ Evaluación de procesos actuales
- ▶ Definición de casos de uso piloto
- ▶ Formación del equipo técnico-legal
- ▶ Preparación de datasets iniciales

Implementación (3-12 meses)

- ▶ Desarrollo de MVP con LLaMA 3.1
- ▶ Fine-tuning y/o RAG para casos específicos
- ▶ Pruebas piloto controladas
- ▶ Validación y refinamiento

Escalamiento (12+ meses)

- ▶ Despliegue en producción
- ▶ Expansión a nuevos procesos
- ▶ Integración sistémica completa
- ▶ Mejora continua y evolución

MÓDULO 7

CASOS DE USO Y MEJORES PRÁCTICAS

Aplicaciones de IA/LLM en el Sistema Judicial

Automatización y Eficiencia en Procesos Judiciales

Enfoque técnico especializado para ingenieros de sistemas judiciales

Índice de Contenidos

Casos de Uso Reales

- ▶ LLaMandement: Parlamento Francés
- ▶ Sistemas de IA judicial en Brasil (STJ)
- ▶ Automatización en Singapur
- ▶ Casos europeos de procesamiento legal
- ▶ Patrones de implementación exitosos
- ▶ Lecciones aprendidas y errores comunes

Mejores Prácticas y Próximos Pasos

- ▶ Ventajas del ecosistema LLaMA 3.1
- ▶ Resumen de mejores prácticas técnicas
- ▶ Recursos adicionales de aprendizaje
- ▶ Roadmap de aprendizaje continuo
- ▶ Oportunidades futuras

Casos de Uso Reales

Contenido Teórico: Contexto Global

Desafíos del Sector Judicial

- ▶ Volumen creciente de documentos legales
- ▶ Procesamiento manual lento y costoso
- ▶ Necesidad de análisis neutral y preciso
- ▶ Demanda de mayor eficiencia operativa

Oportunidades con IA/LLM

- ▶ Automatización de tareas repetitivas
- ▶ Análisis inteligente de documentos
- ▶ Resumen automatizado de casos
- ▶ Clasificación y distribución eficiente

Contexto técnico: Los sistemas judiciales procesan miles de documentos diariamente. La implementación de LLMs permite automatizar hasta el 70% de tareas de procesamiento documental, manteniendo precisión y neutralidad.

Caso de Estudio: LLaMandement

Francia - Parlamento Nacional

Características Técnicas

- ▶ Basado en **LLaMA 2** con técnica LoRA
- ▶ **15,000 pares** de datos de entrenamiento
- ▶ Automatización de distribución de enmiendas
- ▶ Generación de resúmenes neutrales

Modelo: LLaMA 13B fine-tuned

Técnica: Low-Rank Adaptation (LoRA)

Datos: Enmiendas parlamentarias francesas

Resultados Cuantitativos

15.1/20

Puntuación LLaMandement

16.5/20

Redactores Humanos

94%

Tasa Auto-Asignación

5,400 enmiendas procesadas en **menos de 10 minutos**

Casos de Estudio: Implementaciones Internacionales

Brasil – STJ

Sistema: IA para clasificación de casos

Impacto: Reducción 40% en tiempo de procesamiento

Enfoque: Análisis estadístico y selección automática

- ▶ Preparación automática de borradores
- ▶ Asistencia en decisiones judiciales
- ▶ Análisis de precedentes legales

Singapur

Enfoque: Integración gradual y estructurada

Herramientas: LLMs para resúmenes de casos

Resultado: Mayor precisión que abogados expertos

- ▶ Oficina centralizada para coordinación
- ▶ Guías para uso de IA generativa
- ▶ Predicción de resultados de casos

Casos Europeos

Reino Unido: Mediadores robóticos

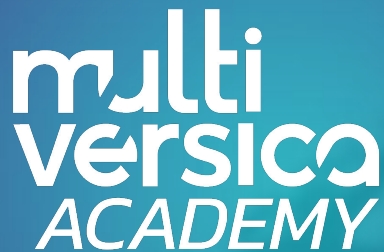
Estonia: Herramientas de apoyo judicial

Países Bajos: Prevención de fraude

Patrones Comunes Identificados

- ▶ **Implementación gradual:** Inicio con tareas de apoyo
- ▶ **Supervisión humana:** IA como asistente, no juez
- ▶ **Transparencia:** Procesos auditables y explicables
- ▶ **Especialización:** Modelos adaptados al dominio legal

Tendencia clave: Todos los casos exitosos utilizan modelos fine-tuned en datos legales específicos, similar al enfoque de LLaMandement



¡Tu transformación comienza ahora!

Únete a los líderes gubernamentales que ya están revolucionando el sector público con IA responsable

- 8 módulos especializados con metodología MPE
- Entorno inmersivo VR/WebVR incluido
- Asistente IA 24/7 y soporte (6 meses)

Inscripción abierta

Reserva tu lugar en la próxima cohorte de líderes en IA gubernamental

Inversión completa

(Incluye todas las características especiales y soporte)



Inscribirse ahora

Garantía de satisfacción de 30 días

Información y contacto



multiversica.academy



educacion@augexp.com