Large Language Models: Llama 2

Arquitectura del Modelo

Llama 2 adopta la mayoría de las configuraciones de preentrenamiento y la arquitectura del modelo de Llama 1. Utiliza la conocida arquitectura Transformer, aplica RMSNorm para prenormalización, utiliza la función de activación SwiGLU y emplea incrustaciones de posición rotadas (RoPE). Las principales diferencias con Llama 1 incluyen la ampliación de la longitud del contexto de 2048 a 4096 y el uso de Atención Agrupada de Consulta (GQA).

Llama 2 vs. Llama 1:

	Training Data	Params	Context Length	GQA	Tokens	LR
Llama 1	*See Touvron et al. (2023)*	7B	2k	✗	1.0T	3.0×10^{-4}
		13B	2k	✗	1.0T	3.0×10^{-4}
		33B	2k	✗	1.4T	1.5×10^{-4}
		65B	2k	✗	1.4T	1.5×10^{-4}
Llama 2	*A new mix of publicly available online data*	7B	4k	✗	2.0T	3.0×10^{-4}
		13B	4k	✗	2.0T	3.0×10^{-4}
		34B	4k	✓	2.0T	1.5×10^{-4}
		70B	4k	✓	2.0T	1.5×10^{-4}

Table 1: Llama 2 family of models. Token counts refer to pretraining data only. All models are trained with a global batch-size of 4M tokens. Bigger models — 34B and 70B — use Grouped-Query Attention (GQA) for improved inference scalability.

Atención Agrupada de Consulta (GQA)

Este mecanismo de atención mejora la escalabilidad de los modelos grandes al compartir la proyección de claves y valores entre múltiples cabezas sin degradación significativa del rendimiento. Puede utilizarse el formato original de Atención Multi-Consulta (MQA) con una única proyección de KV o la variante de Atención Agrupada de Consulta (GQA) con 8 proyecciones KV.

Tamaño del Modelo

Llama 2 viene en 4 tamaños de modelo diferentes: 7B, 13B, 34B y 70B (la versión de 34B aún no ha sido lanzada).

Preentrenamiento

Los datos de preentrenamiento para el modelo Llama 2 incluyen 2 billones de tokens, casi un 40% más grande que Llama 1. Se utiliza el optimizador AdamW para el entrenamiento con β1=0.9, β2=0.95 y eps=10^-5. Se aplica un programa de tasa de aprendizaje coseno con un calentamiento de 2000 pasos y una reducción de la tasa de aprendizaje final al 10% de la tasa de aprendizaje máxima. Se aplican una decadencia de peso de 0.1 y un recorte de gradiente de 1.0.

Fine Tuning

Una versión inicial de Llama-2-chat se crea mediante ajuste fino supervisado (SFT). Luego, Llama-2-chat se somete a un refinamiento iterativo utilizando retroalimentación de refuerzo humano (RLHF), que incluye muestreo de rechazo y optimización de política proximal (PPO).

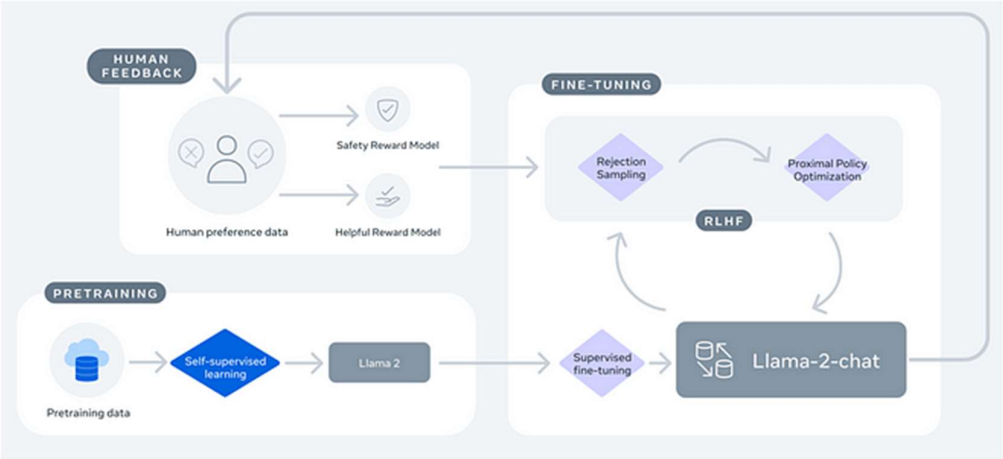
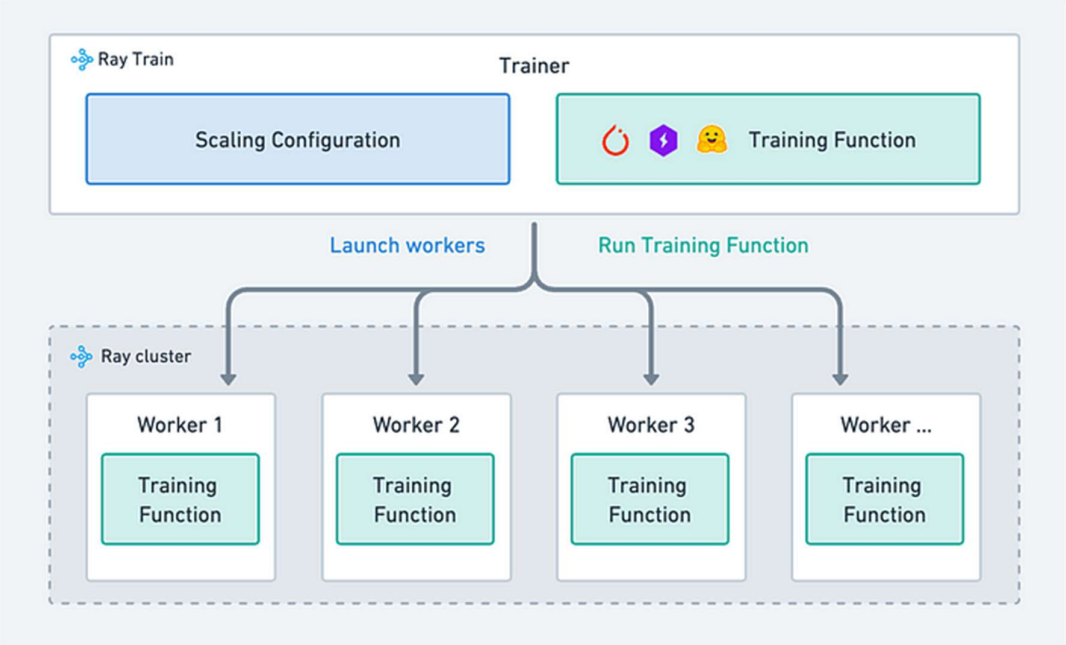


Figure 4: Training of Llama 2-Chat: This process begins with the **pretraining** of Llama 2 using publicly available online sources. Following this, we create an initial version of Llama 2-Chat through the application of **supervised fine-tuning**. Subsequently, the model is iteratively refined using Reinforcement Learning with Human Feedback (**RLHF**) methodologies, specifically through rejection sampling and Proximal Policy Optimization (PPO). Throughout the RLHF stage, the accumulation of **iterative reward modeling data** in parallel with model enhancements is crucial to ensure the reward models remain within distribution.

SFT (Supervised Fine-Tuning):

En esta etapa, Meta introduce una forma novedosa de organizar la información, clasificándola en instrucciones relacionadas con la utilidad y la seguridad. El proceso comenzó utilizando datos de ajuste de instrucciones públicamente disponibles, anotando cuidadosamente alrededor de 27,540 ejemplos con un fuerte enfoque en la calidad de los datos. Durante el ajuste fino supervisado, se aplicó un método de ajuste de la velocidad de aprendizaje que comenzó con una velocidad inicial de 2*10^-5. El proceso involucró ajustar cuidadosamente algunos valores (llamados hiperparámetros) durante un período de dos ciclos de entrenamiento. El objetivo del entrenamiento siguió un enfoque donde la pérdida asociada con las indicaciones del usuario se ajustó y la retroalimentación se aplicó únicamente a las respuestas generadas.



RLHF :

Meta ha establecido un procedimiento preciso para los anotadores durante el proceso de recolección de datos. Inicialmente, los anotadores creaban una solicitud, después de lo cual se les presentaban dos respuestas generadas por el modelo. Su tarea era evaluar estas respuestas según criterios predefinidos. Para mejorar la diversidad, las dos respuestas para cada solicitud se extraían de dos variantes diferentes del modelo, cada una utilizando diferentes hiperparámetros de temperatura. Como se ilustró anteriormente, los datos recopilados se clasificaron según dimensiones de seguridad y utilidad, formando la base para el Modelo de Recompensa. Meta desarrolló múltiples iteraciones de RLHF, desde la V1 hasta la V5, utilizando un enfoque de Ajuste Fino Instruido (IFT) respaldado por dos algoritmos distintos

Optimización de Política Proximal (PPO): Este método se alinea con el enfoque de OpenAI, utilizando el modelo de recompensa como una estimación para la función de recompensa genuina, que refleja las preferencias humanas. El modelo de lenguaje preentrenado sirve como la política, sujeta a optimización.

Fine Tuning de Muestreo de Rechazo: Este enfoque implica el muestreo de K salidas del modelo y la selección del candidato más prometedor basado en un puntaje de recompensa. Las salidas elegidas forman un nuevo estándar de oro para un ajuste fino adicional del modelo. Este proceso refuerza el mecanismo de recompensa, mejorando iterativamente el rendimiento del modelo. El enfoque de Muestreo de Rechazo utilizado en el modelo de 70B se considera intuitivo y más fácil de entender con fines educativos. Ayuda a mantener una brecha creciente entre el rendimiento promedio y máximo, indicando un progreso general.

Meta entrenó dos modelos de recompensa distintos, **el modelo de recompensa de seguridad (R_s)** y **el modelo de recompensa de utilidad (R_h**). Para priorizar la seguridad, se identificaron las solicitudes con potencial para respuestas inseguras y las respuestas se filtraron utilizando un umbral de 0.15, resultando en una precisión del 0.89 y una recuperación del 0.55 según la evaluación con el conjunto de pruebas de seguridad de Meta. El proceso de entrenamiento empleó el optimizador AdamW con un decaimiento de peso de 0.1 y un límite de recorte de gradiente de 1.0. Se utilizó una tasa de aprendizaje constante de 10^-6 durante el entrenamiento. Las iteraciones de Proximal Policy Optimization (PPO) utilizaron un tamaño de lote de 5

Tokenizador

Llama 2 utiliza el mismo tokenizador que Llama 1. Ambos emplean el algoritmo de Codificación de Par de Bytes (BPE) implementado con SentencePiece. Al igual que en Llama 1, todos los números se dividen en dígitos separados y los caracteres UTF-8 desconocidos se descomponen en bytes. El tamaño total del vocabulario es de 32k tokens.

Evaluación

La evaluación en el artículo muestra que Llama 2 supera a otros modelos de lenguaje de código abierto en muchos benchmarks externos, incluyendo pruebas de razonamiento, codificación, competencia y conocimiento.

Benchmark (Higher is better)	MPT (7B)	Falcon (7B)	Llama-2 (7B)	Llama-2 (13B)	MPT (30B)	Falcon (40B)	Llama-1 (65B)	Llama-2 (70B)
MMLU	26.8	26.2	45.3	54.8	46.9	55.4	63.4	68.9
TriviaQA	59.6	56.8	68.9	77.2	71.3	78.6	84.5	85.0
Natural Questions	17.8	18.1	22.7	28.0	23.0	29.5	31.0	33.0
GSM8K	6.8	6.8	14.6	28.7	15.2	19.6	50.9	56.8
HumanEval	18.3	N/A	12.8	18.3	25.0	N/A	23.7	29.9
AGIEval (English tasks only)	23.5	21.2	29.3	39.1	33.8	37.0	47.6	54.2
BoolQ	75.0	67.5	77.4	81.7	79.0	83.1	85.3	85.0
HellaSwag	76.4	74.1	77.2	80.7	79.9	83.6	84.2	85.3
OpenBookQA	51.4	51.6	58.6	57.0	52.0	56.6	60.2	60.2
QuAC	37.7	18.8	39.7	44.8	41.1	43.3	39.8	49.3
Winogrande	68.3	66.3	69.2	72.8	71.0	76.9	77.0	80.2

Seguridad

Se evalúa la seguridad de Llama 2 utilizando tres benchmarks comúnmente utilizados, centrándose en tres dimensiones clave: "Veracidad", que se refiere a si el modelo de lenguaje produce información incorrecta, utilizando el benchmark TruthfulQA; "Toxicidad", que se refiere a si el modelo de lenguaje genera contenido "tóxico", grosero o perjudicial, utilizando el benchmark ToxiGen; "Bias", que se refiere a si el modelo de lenguaje produce contenido sesgado, utilizando el benchmark BOLD.

Conclusión

Meta ha presentado Llama 2, un modelo de lenguaje avanzado que marca la pauta en personalización y transparencia. En un contraste significativo con modelos cerrados como GPT-3 y GPT-4, Llama 2 se destaca por su enfoque abierto, permitiendo a usuarios interesados acceder a documentación detallada que explica su creación y entrenamiento. Llama 2-Chat, una variante especializada, ha sido afinada para destacarse en diálogos, ofreciendo respuestas más contextualmente ricas durante conversaciones. Esto se traduce en mejoras notables en aplicaciones como atención al cliente y chatbots, donde la comprensión y relevancia contextual son cruciales. Estos modelos no solo brindan oportunidades para grandes empresas, sino también para desarrolladores y startups. Su despliegue en la nube, compatible con plataformas como Microsoft Azure y Amazon Web Services a través de Hugging Face, facilita su implementación. Además, la posibilidad de entrenamiento personalizado permite a las empresas adaptar estos modelos a sus necesidades específicas, generando texto personalizado según sus requisitos.