

Arquitectura DeepSeek-V3

Taller • 12 horas • Zoom

El objetivo es implementar en PyTorch los cuatro mecanismos de DeepSeek-V3, atención latente multi-cabeza, mezcla de expertos, balanceo de carga sin pérdida auxiliar y predicción multi-token, integrándolos en un modelo completo y entrenando de principio a fin sobre un corpus real (TinyStories), hasta obtener texto coherente. Siendo honestos con el alcance: el modelo de producción DeepSeek-V3 tiene 671 mil millones de parámetros y se entrenó en 2,048 GPU; el modelo de este taller es una versión compacta que conserva intacto cada mecanismo y se entrena dentro del propio taller, y al cierre trazamos su correspondencia con el original y lo que quedó fuera. Al terminar no habrás leído un diagrama más: habrás escrito cada componente y entenderás por qué funciona.

Lo que harás:

- Entender qué cambia DeepSeek-V3 respecto a un Transformer denso y qué problema de costo o de escala resuelve cada una de sus mejoras.
- Implementar RMSNorm y RoPE: estabilidad numérica bajo precisión mixta, la propiedad de relatividad de RoPE y la visualización de la similitud entre posiciones.
- Implementar la atención latente multicabeza (MLA): compresión de rango bajo de la caché KV, el conflicto con RoPE y su solución con una cabeza rotatoria compartida.
- Medir MLA frente a MHA —tamaño de caché, prellenado y latencia por token al variar contexto y lote— y extrapolar el intercambio memoria/cómputo a 128 K de contexto.
- Construir una capa de mezcla de expertos desde cero: expertos, enrutador top-k con softmax global o sigmoide, y despacho de tokens con máscaras y en versión agrupada.
- Provocar el colapso de enrutado y corregirlo con el balanceo sin pérdida auxiliar: un sesgo por experto que decide la selección, pero nunca el peso de mezcla.
- Incorporar DeepSeekMoE (expertos segmentados, experto compartido, capas densas iniciales) y la predicción multi-token, con su desplazamiento causal y su uso en decodificación especulativa.
- Integrar todo en un modelo completo, entrenarlo sobre TinyStories y diagnosticarlo: términos de la pérdida, reparto de tokens por experto, expertos inactivos y aceptación del módulo MTP (Multi-Token Prediction).

Contenido

Clase 1

- Panorama de DeepSeek-V3: los cuatro mecanismos del modelo y la mejora que aporta cada uno frente a un Transformer denso.
- RMSNorm y RoPE: estabilidad numérica bajo precisión mixta. Formulación de RoPE, su propiedad de relatividad y visualización de la similitud entre posiciones.

Clase 2

- Multi-head Latent Attention: el cuello de botella de la caché KV. Compresión de rango bajo, el conflicto con RoPE y la cabeza rotatoria compartida. Cuadro comparativo con MHA.
- Laboratorio 1 — MLA frente a MHA: medición de caché, prellenado y latencia por token variando contexto y lote. Extrapolación a 128 K y discusión del intercambio memoria/cómputo.

Clase 3

- Fundamento de los Mixture-of-Experts: desacoplar capacidad y cómputo. El experto, el enrutador y la selección top-k: softmax frente a sigmoide.
- Laboratorio 2 — Capa MoE desde cero: reparto de tokens frente a versión agrupada, agregación ponderada y prueba de equivalencia con una MLP densa.

Clase 4

- Balanceo de carga: El sesgo por experto: por qué interviene solo en la selección y nunca en el peso de la mezcla.
- DeepSeekMoE y MTP: segmentación de expertos, experto compartido y capas densas iniciales. Predicción multi-token: desplazamiento causal y decodificación especulativa.

Clase 5

- El modelo completo: bloque con prenormalización, embeddings y MLA, con pruebas de causalidad y de equivalencia de la caché.
- Integración y datos: MoE y MTP dentro del modelo, con prueba de ausencia de fuga. Corpus TinyStories y lanzamiento del entrenamiento.

Clase 6

- Laboratorio 3 — Implementación y monitoreo del MoE: composición de la pérdida y seguimiento independiente de sus tres términos mientras el modelo entrena.
- Diagnóstico y escalado: reparto de tokens por experto, expertos inactivos y aceptación del módulo MTP.

Al terminar tendrás:

- Un modelo estilo DeepSeek-V3 escrito por ti en PyTorch (MLA, DeepSeekMoE con balanceo sin pérdida auxiliar y MTP) entrenado de principio a fin hasta generar texto coherente.
- Un set de pruebas que respalda cada pieza: causalidad, equivalencia de la caché, equivalencia de la capa MoE con una MLP densa y ausencia de fuga en MTP.
- La comprensión real de cada mecanismo (RMSNorm, RoPE, MLA, enrutado top-k, sesgo de balanceo, experto compartido, MTP), no un diagrama que repites sin saber qué hace.

- Tus propias mediciones del intercambio memoria/cómputo entre MLA y MHA, y el criterio para extrapolarlas a contextos de 128 K.
- Las herramientas para diagnosticar el entrenamiento de un MoE: cada término de la pérdida por separado, reparto de tokens por experto, expertos inactivos y aceptación del módulo MTP.
- Un mapa claro de cómo tu implementación corresponde al modelo de producción y de qué componentes quedan fuera de alcance.
- Constancia de participación verificable en línea.

Ya viste lo que vas a construir.

Asegura tu lugar

Inscríbete ahora · \$1,200 MXN →

Constancia con registro STPS · Grabaciones por 1 año · Acceso a Zoom al instante

[¿Dudas antes de inscribirte? Escríbenos por WhatsApp](#)

¿Para quién es este taller?

- Personas que ya usan PyTorch y conocen el Transformer, pero quieren entender por dentro las ideas que hacen eficientes a los grandes modelos de lenguaje actuales: MLA, MoE y MTP.
- Ingenieros e investigadores de ML que leen reportes técnicos como el de DeepSeek-V3 y quieren pasar de "entender el diagrama" a "poder implementarlo y medirlo".
- Quienes entrenan, sirven u optimizan modelos de lenguaje y necesitan criterio sobre memoria de caché, latencia y balanceo de expertos para tomar decisiones de arquitectura.

Este taller NO es para ti si:

- Buscas un curso introductorio de Python, de deep learning o del Transformer.
- Quieres descargar, ajustar (fine-tunear) o desplegar los pesos oficiales de DeepSeek. Aquí construyes la arquitectura sin pesos preentrenados ni bibliotecas de modelado.
- Esperas reproducir el modelo de producción. Entrenas una versión compacta sobre TinyStories.

Requisitos

Python y PyTorch, fundamentos de redes neuronales y entrenamiento (retropropagación, optimizadores, sobre/subajuste) y familiaridad con la arquitectura Transformer y el mecanismo de atención. Se recomienda contar con una GPU, propia o en la nube (Gratis: Colab, Kaggle, LightningAI), para seguir los laboratorios en vivo.

Clases

3 semanas, 3 sesiones de 4 horas (12 hrs en total): sábados 31 de octubre, 7 y 14 de noviembre de 2026, de 10:00 a 14:00 (hora de la Ciudad de México). Todo se imparte por Zoom y queda grabado; tendrás acceso a las grabaciones durante 1 año para repasar a tu ritmo.

Instructor

M. en C. Israel Prado — [LinkedIn](#)

Testimonios

Hemos capacitado a más de 2,000 personas en IA — visita actumlogos.com

Medios de pago

Tarjetas de crédito/débito, Link, PayPal, transferencia SPEI, Mercado Pago, Apple Pay, Google Pay. Ver [checkout](#)

Lista de Espera

Constancia de participación con folio único, verificable en línea

Al terminar recibirás una constancia de participación firmada por el instructor y el director de Actumlogos — agente capacitador registrado ante la Secretaría del Trabajo y Previsión Social. Cada constancia incluye un folio único con el que cualquier persona puede verificar su autenticidad en línea, y podrás compartirla directamente en LinkedIn.

Registro STPS: ZAGE-810930-FW2-0005

Este es un ejemplo para ilustrar qué esperar:

