Modelos Generativos

Nahuel Costa

Grado en Ciencia e Ingeniería de Datos

Universidad de Oviedo
Universidá d'Uviéu
University of Oviedo

Reducción Computacional

A pesar de que el avance de los modelos generativos ha permitido grandes avances en el campo de la IA, estos modelos requieren una cantidad significativa de recursos computacionales para entrenar y ejecutar.

En esta sección se presentan varias técnicas para reducir el coste computacional de los modelos, tanto para entrenamiento como para inferencia.

Low-Rank Adaptation (LoRA)

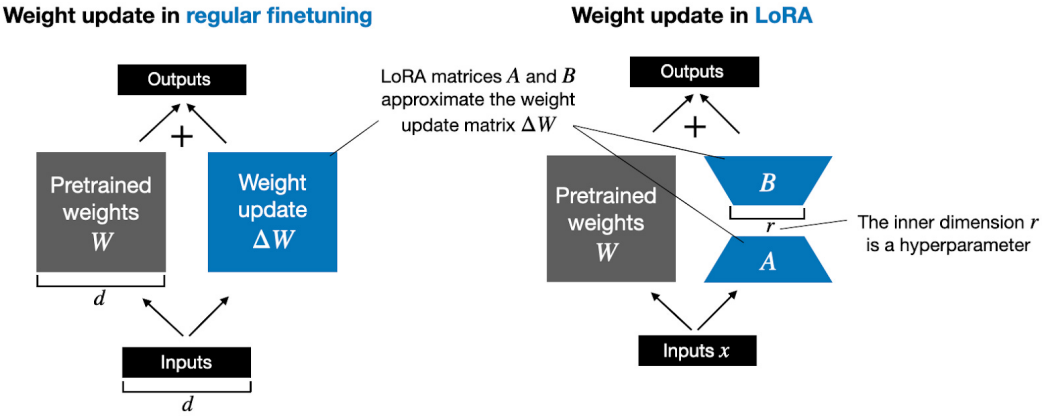
Low-Rank Adaptation (LoRA) [1] es una técnica de reentrenamiento de grandes modelos generativos. A diferencia del fine-tuning tradicional, que requiere reentrenar todo el modelo o las capas seleccionadas para el ajuste, LoRA selecciona sólo partes específicas de la red. De esta manera, se pueden conseguir mejoras específicas sin necesidad de un reentrenamiento exhaustivo, que puede llevar tiempo y consumir muchos recursos.

Low-Rank Adaptation (LoRA)

Cuando se entrenan grandes modelos generativos, ajustar todos los parámetros puede ser muy costoso, especialmente cuando solo se quiere adaptar el modelo a una tarea específica con datos limitados. LoRA busca reducir esta carga utilizando una técnica llamada descomposición de bajo rango. La idea central es descomponer las matrices de pesos (algunas o todas) del modelo original en matrices de bajo rango y entrenarlas en su lugar.

Al entrenar el modelo solo se ajustan esas matrices. Esto reduce significativamente la cantidad de parámetros que se necesitan ajustar, lo que hace que el proceso sea más eficiente.

Low-Rank Adaptation (LoRA)



Fuente

Low-Rank Adaptation (LoRA)

Por ejemplo, supongamos que tenemos un LLM con 7B de parámetros representados en una matriz de pesos W. Durante el backpropagation, se aprende una matriz ΔW, que contiene información sobre cuánto se quiere actualizar los pesos originales para minimizar la función de pérdida durante el entrenamiento.

La actualización de pesos entonces es: $W_{actualizado} + \Delta$

Low-Rank Adaptation (LoRA)

Si la matriz de pesos W contiene 7B parámetros, entonces la matriz de actualización de pesos ΔW también contiene 7B parámetros. El cálculo de la matriz ΔW puede ser muy intensivo en términos de computación y memoria.

LoRA propone descomponer los cambios de pesos, ΔW, en una representación de rango inferior. Es decir, no requiere calcular explícitamente ΔW, sino que se aprende la representación descompuesta de ΔW directamente durante el entrenamiento.

Low-Rank Adaptation (LoRA)

Beneficios/aportaciones de LoRA:

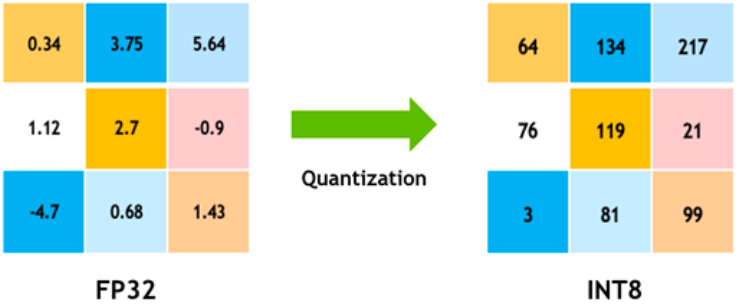
1. Eficacia: permite mejorar modelos sin necesidad de un reentrenamiento exhaustivo, lo que supone ahorro de tiempo y recursos computacionales.
2. Personalización: Con LoRA, los modelos pueden ajustarse para satisfacer necesidades o aplicaciones específicas.
3. Accesibilidad: La reducción de los requisitos computacionales hace más factible que investigadores y organizaciones con recursos limitados puedan utilizar y mejorar grandes modelos.

Cuantización

La cuantización es una técnica para reducir los costes computacionales y de memoria en inferencia mediante la representación de los pesos y las activaciones con tipos de datos de baja precisión, como enteros de 8 bits (int8) en lugar de los habituales de coma flotante de 32 bits (float32).

Reducir el número de bits significa que el modelo resultante requiere menos almacenamiento en memoria, consume menos energía (en teoría) y operaciones como la multiplicación de matrices pueden realizarse mucho más rápido.

Low-Rank Adaptation (LoRA)



Fuente

Cuantización

La idea básica es bastante sencilla: pasar de una representación de alta precisión (normalmente 32 bits en coma flotante) a un tipo de datos de menor precisión. Los tipos de datos de menor precisión más comunes son:

1. float16, tipo de datos de acumulación float16
2. bfloat16, tipo de datos de acumulación float32
3. int16, tipo de datos de acumulación int32
4. int8, tipo de datos de acumulación int32

Cuantización

El "tipo de datos de acumulación" especifica el tipo del resultado de acumular (sumar, multiplicar, etc) valores del tipo de datos en cuestión.

Por ejemplo, consideremos dos valores int8: $A = 127, B = 127$, y definamos C como la suma de A y B: $C = A + B$

Aquí el resultado es mucho mayor que el mayor valor representable en int8, que es 127. De ahí la necesidad de un tipo de datos de mayor precisión para evitar una enorme pérdida de precisión que haría inútil todo el proceso de cuantización.
Los dos casos de cuantización más comunes son float32 -> float16 y float32 -> int8

Cuantización - Más recursos

1. Guía de HuggingFace
2. DOTCSV Bitnet
3. Practical Quantization in PyTorch
4. Sebastian Raschka - Accelerating Large Language Models with Mixed-Precision Techniques

[1] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "Lora: Low-rank adaptation of large language models," *arXiv preprint arXiv:2106.09685*, 2021.