Apuntes de la clase del 23 de octubre de 2025

Curso de Inteligencia Artificial

Nelson Rojas Obando
Estudiante Ingeniería en Computación
nelson.rojas@estudiantec.cr

***Resumen*—Este informe presenta una síntesis de los temas abordados en la sesión del 23 de octubre del curso de Inteligencia Artificial, centrada en el cierre del aprendizaje supervisado y la introducción al proceso de quantization como técnica de optimización de modelos de aprendizaje profundo.**

***Index Terms*—Inteligencia Artificial, Quantization**

I. Introducción

Durante la clase se abordaron temas de actualidad relacionados con la evolución de los modelos de lenguaje y su impacto en el futuro del internet. Con esta sesión concluye la sección del curso dedicada al aprendizaje supervisado. A partir de este punto, los contenidos se centran en métodos de aprendizaje no supervisado, es decir, aquellos que no dependen de etiquetas o resultados predeterminados para evaluar la calidad del aprendizaje del modelo.

II. Aspectos administrativos

Se mencionó la integración de herramientas como ChatGPT Atlas para Chrome, que reflejan cómo las empresas están orientando sus estrategias hacia la adopción de Modelos de Lenguaje Extensos (LLMs) como núcleo de sus servicios digitales.

Asimismo, se compartieron noticias institucionales sobre la rama IEEE del Tecnológico de Costa Rica, que organiza reuniones periódicas entre distintas universidades. El propósito principal es identificar fuentes de financiamiento para eventos tecnológicos, especialmente aquellos destinados a llevar conocimiento a zonas rurales o con menor acceso. También se anunció la realización de un taller de team building el domingo 9 de noviembre, con un costo de $20, que incluye almuerzo y transporte.

III. Tema principal: Quantization

Quantization es una técnica de optimización de modelos de aprendizaje profundo que busca reducir el tamaño y el consumo de recursos computacionales de un modelo sin comprometer significativamente su precisión. La idea es convertir los parámetros del modelo (usualmente almacenados en formato de punto flotante de 32 bits, float32) a representaciones de menor precisión, como int8, int4 o incluso int1, dependiendo del nivel de compresión deseado.

Esto permite ejecutar modelos de gran tamaño en hardware con recursos limitados (por ejemplo, dispositivos móviles o microcontroladores).

III-A. Ejemplo contextual

Un modelo como LLaMA 2 posee más de 70 mil millones de parámetros, lo que equivale a aproximadamente 28 GB solo para almacenarlos en disco. Cargar ese modelo en memoria sería inviable sin una GPU especializada, por lo que la quantization se convierte en una alternativa para reducir el tamaño y mantener la funcionalidad.

IV. Representación numérica

IV-A. Números enteros

Los computadores representan los números utilizando secuencias de bits. Con N bits se pueden representar 2^n valores distintos.

Por ejemplo, con 3 bits se pueden representar los números del 0 al 7.

El formato más común para representar números enteros con signo en CPUs es el complemento a dos, donde:

El bit más significativo indica el signo (0 = positivo, 1 = negativo). Los demás bits representan el valor absoluto.

4.2. Números de punto flotante

Los números de punto flotante se utilizan para representar valores reales que no pueden expresarse de manera exacta con enteros. En la norma **IEEE 754**, un número flotante se representa mediante tres componentes principales: el *signo*, el *exponente* y la *mantissa* (también conocida como fracción o significando).

Parte	Descripción	Bits típicos (float32)
Signo (s)	Indica si el número es positivo o negativo	1
Exponente (e)	Determina la escala o rango del número	8
Mantissa (m)	Define la precisión o parte fraccionaria	23

Cuadro I
Estructura del formato IEEE 754 de 32 bits.

El valor real que representa el número en punto flotante se calcula mediante la siguiente ecuación:

$$x = (-1)^s \times (1 + m) \times 2^{(e-127)}$$

Donde:

- s es el bit de signo.
- m es la fracción o mantissa normalizada.
- e es el exponente con un sesgo de 127 (en el caso de `float32`).

Este formato permite representar números muy grandes o muy pequeños, aunque implica un mayor uso de memoria y recursos computacionales en comparación con representaciones de menor precisión.

V. Quantization de redes neuronales

En redes neuronales, las matrices de pesos y sesgos están representadas como flotantes. El proceso de quantization busca convertir esos valores a enteros para reducir memoria y acelerar la inferencia.

V-A. Etapas del proceso

- **Quantize:** Los valores en punto flotante se transforman a enteros.
- **Inferencia** El modelo realiza sus cálculos con aritmética entera.
- **Dequantize** Los resultados se transforman nuevamente a flotantes para la siguiente capa.

El desafío está en mantener la precisión del modelo. Los hardware modernos (GPU, TPU, CPU vectoriales) incluyen soporte para operaciones de baja precisión (por ejemplo, int8) para facilitar este proceso.

VI. Tipos de quantization

VI-A. Quantization simétrica

Usa un rango centrado en cero:

VI-B. Quantization asimétrica

Utiliza un rango desplazado $[\alpha, \beta]$:

VII. Estrategias y variantes

VII-A. Dynamic Quantization

La escala y el rango se calculan en tiempo de inferencia. Se aplican factores estadísticos derivados del conjunto de datos de prueba ("calibration set").

VII-B. Post-Training Quantization (PTQ)

Después del entrenamiento, se insertan observadores (observers) en el modelo para analizar las salidas de cada capa y determinar los mejores parámetros de escala y punto cero. Este proceso no requiere reentrenamiento y es rápido, aunque puede perder algo de precisión.

VII-C. Quantization-Aware Training (QAT)

Simula la quantization durante el entrenamiento. El modelo aprende a compensar los errores introducidos por la reducción de precisión, por lo que mantiene un rendimiento superior tras el proceso.

VIII. Ventajas del Quantization

- Menor consumo de memoria: los modelos comprimidos se cargan más rápido.
- Menor tiempo de inferencia: cálculos más simples.
- Menor consumo energético: ideal para dispositivos embebidos o móviles.
- Portabilidad: permite ejecutar modelos complejos en hardware limitado.

IX. Conclusiones

El estudio del quantization permite comprender cómo los modelos de inteligencia artificial pueden adaptarse a las limitaciones del hardware sin comprometer significativamente su desempeño. Esta técnica representa un punto de conexión entre el desarrollo teórico de los algoritmos y su aplicación real en sistemas de producción, donde los recursos computacionales, la energía y el tiempo de inferencia son factores determinantes.