NOTAS DE CLASE

INTELIGENCIA ARTIFICIAL - 23 DE OCTUBRE - SEMANA 12

Luis Alfredo González Sánchez
Escuela de Ingeniería en Computación
Instituto Tecnológico de Costa Rica
Cartago, Costa Rica
2021024482 gonzal3z.luis@estudiantec.cr

***Abstract*—Neural network quantization is a vital technique in AI that reduces model size and computational cost by converting weights and activations from floating-point to lower-precision formats, such as integers. This process enables deployment on resource-constrained devices, like mobile or embedded systems, while maintaining high accuracy. Different methods include symmetric and asymmetric quantization, with strategies tailored to specific data distributions and hardware constraints. Dynamic, granular, and post-training quantization further refine this approach by adjusting intervals per layer, per sample, or after training, respectively. These techniques involve calculating scaling factors and zero points to effectively map high-precision values to lower-bit representations, introducing minimal accuracy loss. Overall, quantization enhances efficiency, reduces power consumption, and facilitates real-time AI applications, making it a cornerstone of practical deep learning deployment.**

***Index Terms*—Quantization in neural networks,model compression,QAT quantization techniques,integer representation**

I. Introduction

La cuantización en redes neuronales es una técnica esencial para mejorar la eficiencia del cómputo y reducir el tamaño de los modelos, principalmente transformando los datos de punto flotante a formatos de menor precisión, como enteros. Esta transformación permite que los modelos se ejecuten de manera más rápida y con menor consumo de memoria, lo cual es fundamental para desplegar inteligencia artificial en dispositivos con recursos limitados, como móviles y sistemas embebidos. En el presente documento, se busca resumir la información vista en la clase del 23 de octubre,donde se ha revisado cómo diferentes métodos de cuantización —desde la simétrica y asimétrica hasta la dinámica, granulada y post-entrenamiento— manejan la conversión de pesos, activaciones y sesgos, optimizando el balance entre precisión y eficiencia. Además, se menciona cómo técnicas como la cuantización consciente durante el entrenamiento (QAT) ayudan a mantener la precisión del modelo al considerar el efecto de la cuantización desde el inicio del aprendizaje.

II. Breve definición de Onnix

Para continuar el tema de quantization en supervised learning , es importante entender la herramienta onnix, suponga un modelo llm ya entrenado¿Cómo empieza a funcionar el producto o sistema? La herramienta Onnix permite representar modelos de aprendizaje automático desarrollados en distintos frameworks como PyTorch o TensorFlow en una representación intermedia estándar y eficiente. Esta representación facilita la interoperabilidad y el despliegue de modelos en diferentes plataformas y hardware mediante optimizaciones en C++ u otros lenguajes, asegurando que el mismo modelo pueda ejecutarse con alto rendimiento en entornos variados. Considerando lo anterior, las plataformas poseen diversas limitaciones y rendimiento tanto en software como en hardware, si se entrenan modelos grandes, posiblemente un celular no este adaptado para soportar dicho modelo, para ello se observara el concepto de quantization.

III. Quantization

Suponga que se tiene un modelo de deep learning con muchas capas, por ejemplo , llama 2 , con 70 mil millones de parametros aproximadamente, si cada parametro es de 32 bits, se obtiene un tamaño aproximado de 28 gb para solo almacenar el modelo, ¿Cómo podríamos cargarlo a memoria? Una alternativa es comprar una GPU con dicho tamaño para el procesamiento del modelo, pero GPUs que soporten esos tamaños son costosas , lo que se busca es reducir el tamaño del modelo, una de sus técnicas es quantization

A. definicion

Quantization es una técnica de compresión de modelos de aprendizaje automático que reduce el numero de bits utilizados para representar los parámetros del modelo, transformando los valores de punto flotante a representaciones de menor precisión, generalmente enteros de 8, 5, 2 o incluso 1 bit. Lo que se busca es disminuir el tamaño y la complejidad computacional del modelo, manteniendo una precisión cercana al original. No se debe de confundir como una técnica de redondear pesos, sino de convertir y ajustar los tipos de datos para optimizar el balance entre tamaño, velocidad de inferencia y precisión. Se busca un tradeoff optimo entre capacidades del modelo vs rendimiento.

B. ventajas

- menor consumo de memoria al cargar los modelos en memoria
- Permite insertar el modelo en sistemas con recursos limitados / con propósito especifico, como celulares o embebidos

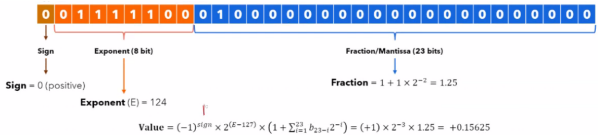

Fig. 1. Partes de un número punto flotante

- Genera un menor tiempo para hacer las inferencias, sus datos son más simples
- Menor consumo energía debido a menor complejidad de computación

IV. Breve repaso a las operaciones con bits

Se dará un breve repaso a la manipulación de bits en sistemas computacionales para entender mejor el proceso de quantization

Con 2^n bits se pueden representar 2^n valores distintos. Esto significa que, por ejemplo, con 3 bits es posible representar $2^3 = 8$ números diferentes.

Un ejemplo básico que se vió en clase es de conversión de binario a decimal es el número 6, que en binario se escribe como 110. La conversión se realiza sumando las potencias de 2 correspondientes a los bits activos (1) según su posición:

$$6 = 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 4 + 2 + 0 = 6$$

Cada dígito binario representa una potencia de 2, comenzando desde la derecha con la potencia 0.

Los números enteros en sistemas digitales se representan normalmente en complemento a 2, donde el bit más significativo indica el signo: 0 para positivo y 1 para negativo. Esto facilita realizar operaciones aritméticas con números negativos usando operaciones binarias estándar.

Para números en punto flotante, la representación se divide en tres partes: signo, exponente y mantisa (fracción). El valor decimal se calcula aproximadamente como:

$$\text{valor} = (-1)^{\text{signo}} \times (1 + \text{mantisa}) \times 2^{\text{exponente} - \text{bias}}$$

Esta técnica de representación permite expresar un amplio rango de números reales con precisión limitada y eficiencia en almacenamiento mediante manipulación de bits.

Observe la figura 1 donde se puede observar las partes del numero flotante de 32 bits Ahora bien , considerando las partes del numero punto flotante, es importante detallar que **La precisión dada por la mantiza se va a disminuir con quantization** .

V. Procedimiento de Quantization en Modelos de Redes Neuronales

Pasos generales del procedimiento:

- **Transformación de pesos:** Los pesos de la red, originalmente en formato de punto flotante (float), se convierten a valores enteros mediante mapas de cuantización que asignan rangos de valores flotantes a niveles discretos enteros.
- **Cuantización de entradas:** Las entradas a cada capa también se convierten a enteros para mantener la coherencia en la representación y facilitar operaciones eficientes.
- **Cuantización del sesgo (bias):** Los términos de sesgo, que son sumados en cada neurona, se transforman de float.
- **Normalización del rango:** Se definen valores máximos y mínimos para pesos, entradas y sesgos, que corresponden a los valores límite de la representación entera (por ejemplo, el rango de int8). Esto asegura que los valores cuantizados estén dentro de rangos representables.
- **Cálculo en espacio entero:** Las operaciones de la capa (multiplicación y suma) se realizan en enteros, generando un vector cuantizado.
- **Des-cuantización:** Después de la capa, los valores enteros se convierten nuevamente a punto flotante para continuar con el procesamiento de modo que las capas siguientes no requieren conocer el esquema de cuantización aplicado.

Durante la **dequantization** es donde puede ocurrir pérdida de precisión, ya que la conversión entre representaciones introduce aproximaciones. Sin embargo, el objetivo es que la salida cuantizada sea lo suficientemente cercana a la original para no afectar el rendimiento del modelo. Este proceso es beneficioso ya que permite que modelos originalmente pesados funcionen eficientemente con menor consumo de memoria y tiempo de cómputo, esencial sistemas embebidos o con capacidad limitada. Es importante aclara que las capas que siguen a una capa cuantizada generalmente no requieren modificaciones ni conocen directamente la cuantización aplicada, manteniendo transparencia en la mayoría de frameworks.

VI. Tipos de Quantization

Los tipos de consonantizan son los siguientes:

- **Quantization simétrica:** Mapea valores positivos y negativos de un rango *mínimo a máximo* que incluye el cero. Aquí, el valor cero real se mapea exactamente a cero entero. Esto simplifica el manejo de pesos y activaciones con signo, aplicando la misma escala en ambos lados del cero.
- **Quantization asimétrica:** Mapea valores entre 0 y un valor máximo entero, pero el valor mínimo real no se mapea a cero, sino a un valor entero llamado *zero point* (Z), que representa el valor neutro o "offset" de la cuantización. Esto permite representar valores con un desplazamiento, útil cuando los valores no están centrados en cero.

Las fórmulas para la cuantización simetrica y asimétricas se describen a continuación : se toma un rango [b,a] y se mapea a un rango de salida, se calcula el parámetro de escalado y por ultimo se calcula el número neutro del mapeo.

- $x_q = \text{clamp}\left(\left\lfloor \frac{x_f}{s} \right\rfloor + z;\ 0;\ 2^n - 1\right)$
  - x_f = Valor flotante

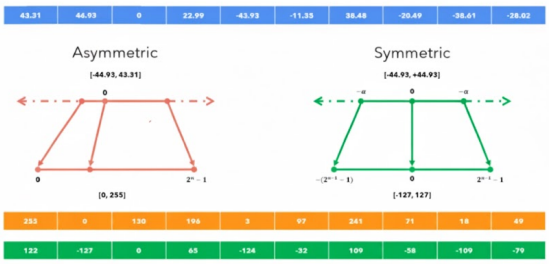

Fig. 2. Tipos de quantizaton : simétrica vs asimétrica

- **Parámetro de escalado** s:

$$s = \frac{\alpha - \beta}{2^n - 1}$$

$$2^n - 1 = \text{El rango de salida}$$

- **Parámetro neutro** z:

$$z = \left\lfloor -1 \cdot \frac{\beta}{s} \right\rceil$$

- n es el número de bits.

y su respectiva des-cuantización :

$$x_f = s \times (x_q - z)$$

Para la cuantización simétrica :

- $x_q = \text{clamp}\left(\left\lfloor \frac{x_f}{s} \right\rceil ;\ -\left(2^{n-1} - 1\right);\ 2^{n-1} - 1\right)$
- **Parámetro de escalado** s:

$$s = \frac{\text{abs}(\alpha)}{2^{n-1} - 1}$$

- n es el número de bits.

y su respectiva descuantización se brinda por la siguiente formula

$$X_f = S \times X_q$$

VII. Otros tipos de cuantizaciones

Cuantización dinámica: La cuantización dinámica se enfoca en cuantizar las activaciones de las neuronas seleccionando apropiadamente los valores mínimos (a) y máximos (b) para el mapeo de cuantización de cada tensor. La estrategia de selección del intervalo a, b es la siguiente:

- Para cuantización **asimétrica**, se seleccionan los valores extremos reales del tensor, es decir, b y a corresponden al máximo y mínimo del tensor respectivamente.
- Para cuantización **simétrica**, se toma el mayor valor en términos absolutos y se define el intervalo como $[-a, a]$, centrado en cero.
- Esta técnica puede inducir un mayor error debido a la sensibilidad a valores atípicos (outliers). Una solución es utilizar **percentiles** basados en la distribución del tensor, excluyendo los outliers y reduciendo el error cuadrático medio (MSE).

Cuantización granulada en convoluciones:

- Las convoluciones se realizan con múltiples filtros que aprenden valores y distribuciones distintas.
- Cada filtro detecta diferentes características (features) de la imagen.
- No es posible aplicar el mismo intervalo a, b para todos los filtros, por lo que se calcula un intervalo a, b específico para cada filtro respetando su distribución.

Quantization post-training:

- Se realiza después del entrenamiento, utilizando datos nunca antes vistos por el modelo.
- Introduce un componente llamado *observer*, que obtiene estadísticas de cada capa para calibrar las salidas y calcular los parámetros de cuantización como la escala (s) y punto cero (z).
- Permite cuantizar el modelo sin necesidad de re-entrenamiento completo.

Quantization aware training (QAT):

- Método avanzado donde la cuantización se simula durante el entrenamiento.
- El modelo aprende a compensar la pérdida de precisión por la cuantización al utilizar la función de perdida para actualizar los pesos que constantemente sufren de este efecto.
- Mejora el rendimiento en modelos cuantizados para entornos de baja precisión.

VIII. Conclusión

La información presentada demostró la importancia para la optimización del uso de recursos en modelos de redes neuronales, especialmente si se desea implementar en sistemas embebidos o con recursos limitados, en esta clase se aprendio que : tex

- La cuantización consiste en transformar pesos, activaciones y sesgos de punto flotante a representaciones de menor precisión, principalmente enteros, con el fin de reducir tamaño y acelerar la inferencia.
- Existen distintos tipos de cuantización: **simétrica**, **asimétrica**, **dinámica**, **granulada** y **post-entrenamiento**, cada una con estrategias específicas para mapear y convertir datos.