

# Table of Contents

- Neural Networks, Tensors and other stuff** ..... 1
- Formats** ..... 1
- Quantization ..... 1
- Opensource llm models** ..... 3
- Opensource visual models** ..... 3
- Stable Diffusion 1 ..... 3
- Stable Diffusion XL ..... 3
- Stable Diffusion 3 ..... 4
- Flux.1d ..... 4
- HiDream I1 ..... 4
- Qwen ..... 4
- CLIP (Contrastive Language-Image Pre-training)** ..... 4
- CURL** ..... 4
- No thinking** ..... 4

# Neural Networks, Tensors and other stuff

## Formats

|             |  |
|-------------|--|
| Safetensors | Safe store for tensors   |
| GGUF        | <a href="#">Georgi Gerganov</a> Universal Format (it can mix various precisions) |
| PT          | PyTorch format   |

## Quantization

| Quantization Type | Bits per Weight | Description  |
|-------------------|-----------------|--|
| F64               | 64              | 64-bit IEEE 754 double-precision floating-point. High precision, large memory footprint.   |
| F32               | 32              | 32-bit IEEE 754 single-precision floating-point. Standard for training, high memory usage.   |
| F16               | 16              | 16-bit IEEE 754 half-precision floating-point. Balances precision and efficiency.  |
| I64               | 64              | 64-bit integer. Used for specific metadata or computations.  |
| I32               | 32              | 32-bit integer. Less common for weights.   |
| I16               | 16              | 16-bit integer. Rarely used for model weights.   |
| I8                | 8               | 8-bit integer. Used in some quantization schemes.  |
| Q8_0              | 8               | 8-bit quantization, 32 weights per block, weight = $q * \text{block\_scale}$ . Legacy, close to FP16 accuracy.                           |
| Q8_1              | 8               | 8-bit quantization with block minimum, weight = $q * \text{block\_scale} + \text{block\_minimum}$ . Legacy.                              |
| Q8_K              | 8               | 8-bit quantization, 256 weights per block, used for intermediate results.  |
| Q6_K              | 6.5625          | 6-bit quantization, super-blocks with 16 blocks of 16 weights, weight = $q * \text{block\_scale}$ (8-bit).                               |
| Q5_0              | 5               | 5-bit quantization, 32 weights per block, legacy.  |
| Q5_1              | 5               | 5-bit quantization with block minimum, legacy.   |
| Q5_K              | 5.5             | 5-bit quantization, super-blocks with 8 blocks of 32 weights, weight = $q * \text{block\_scale}$ (6-bit) + $\text{block\_min}$ (6-bit).  |
| Q4_0              | 4               | 4-bit quantization, 32 weights per block, legacy.  |
| Q4_1              | 4               | 4-bit quantization with block minimum, legacy.   |
| Q4_K              | 4.5             | 4-bit quantization, super-blocks with 8 blocks of 32 weights, weight = $q * \text{block\_scale}$ (6-bit) + $\text{block\_min}$ (6-bit).  |
| Q3_K              | 3.4375          | 3-bit quantization, super-blocks with 16 blocks of 16 weights, weight = $q * \text{block\_scale}$ (6-bit).                               |
| Q2_K              | 2.625           | 2-bit quantization, super-blocks with 16 blocks of 16 weights, weight = $q * \text{block\_scale}$ (4-bit) + $\text{block\_min}$ (4-bit). |
| IQ4_NL            | 4               | 4-bit integer quantization, non-linear, super-blocks with 256 weights.   |
| IQ4_XS            | 4.25            | 4-bit integer quantization, extra small, super-blocks with 256 weights.  |
| IQ3_S             | 3               | 3-bit integer quantization, small.   |
| IQ3_XXS           | 3               | 3-bit integer quantization, extra extra small.   |
| IQ2_S             | 2               | 2-bit integer quantization, small.   |
| IQ2_XS            | 2               | 2-bit integer quantization, extra small.   |
| IQ2_XXS           | 2               | 2-bit integer quantization, extra extra small.   |
| IQ1_S             | 1               | 1-bit integer quantization, small.   |
| IQ1_M             | 1               | 1-bit integer quantization, medium.  |

**Approx. loss**

| Quantization  | Correct (%) | Loss of accuracy (%) |
|---------------|-------------|----------------------|
| IQ2_XXS       | 51,5        | 26,0                 |
| Q2_K_S        | 62,4        | 10,3                 |
| IQ3_XXS       | 67,3        | 3,2                  |
| Q3_K_M        | 68,3        | 1,8                  |
| IQ4_XS        | 69,3        | 0,4                  |
| Q4_K_M        | 69,0        | 0,8                  |
| Q6_K          | 69,8        | -                    |
| Q8_0 (static) | 69,6        | -                    |

Legacy Quantizations (Q4\_0, Q4\_1, Q5\_0, Q5\_1, Q8\_0, Q8\_1): These are simpler, faster methods but may have higher quantization error compared to newer types.

K-Quantizations (Q2\_K, Q3\_K, Q4\_K, Q5\_K, Q6\_K): Introduced in llama.cpp PR #1684, these use super-blocks for smarter bit allocation, reducing quantization error.

I-Quantizations (IQ2\_XXS, IQ3\_S, etc.): State-of-the-art for low-bit widths, using lookup tables for improved accuracy but potentially slower on older hardware.

GGUF Q8\_0: Very close to FP16 (perplexity 7.4933), indicating minimal accuracy loss.

GGUF Q4\_K\_M: Slightly higher perplexity (7.5692), still usable for most tasks.

**UD IQ4 (4x performance boost on Blackwell) or IQ4\_NL may be better variant than Q4\_K\_M**

## Opensource llm models

| Model    | Maker      |
|----------|------------|
| Gemma 3  | Google     |
| Nemotron | NVIDIA     |
| LLama 3  | Facebook   |
| DeepSeek | DeepSeek   |
| Qwen     | Alibaba    |
| Mistral  | Mistral AI |

## Opensource visual models

### Stable Diffusion 1

### Stable Diffusion XL

<https://stability.ai/news/stable-diffusion-sdxl-1-announcement>

## Stable Diffusion 3

by Stability AI <https://stability.ai>

## Flux.1d

by Black Forest Labs <https://bfl.ai>

## HiDream I1

by HiDream AI <https://hidream.org>

## Qwen

by Alibaba <https://qwen.ai>

## CLIP (Contrastive Language-Image Pre-training)

[Wikipedia](#)

Self-attention Transformer as a text encoder

## CURL

```
curl https://localhost/v1/chat/completions \  
-H "Content-Type: application/json" \  
-d '{  
  "model": "gemma-3-4b-q4.gguf",  
  "messages": [  
    {  
      "role": "system",  
      "content": "Answer short in czech language"  
    },  
    {  
      "role": "user",  
      "content": "Who is Albert Einstein"  
    }  
  ],  
  "temperature": 0.7,  
  "max_tokens": -1,  
  "stream": false  
'
```

## No thinking

Edit JINJA Set

```
{# Set Instruct mode here #}
```

To

```
{%- set enable_thinking = false %}
```

From:  
<https://wiki.janforman.com/> - **wiki.janforman.com**

Permanent link:  
<https://wiki.janforman.com/nn:index>

Last update: **2026/03/17 13:59**

