



Gemma

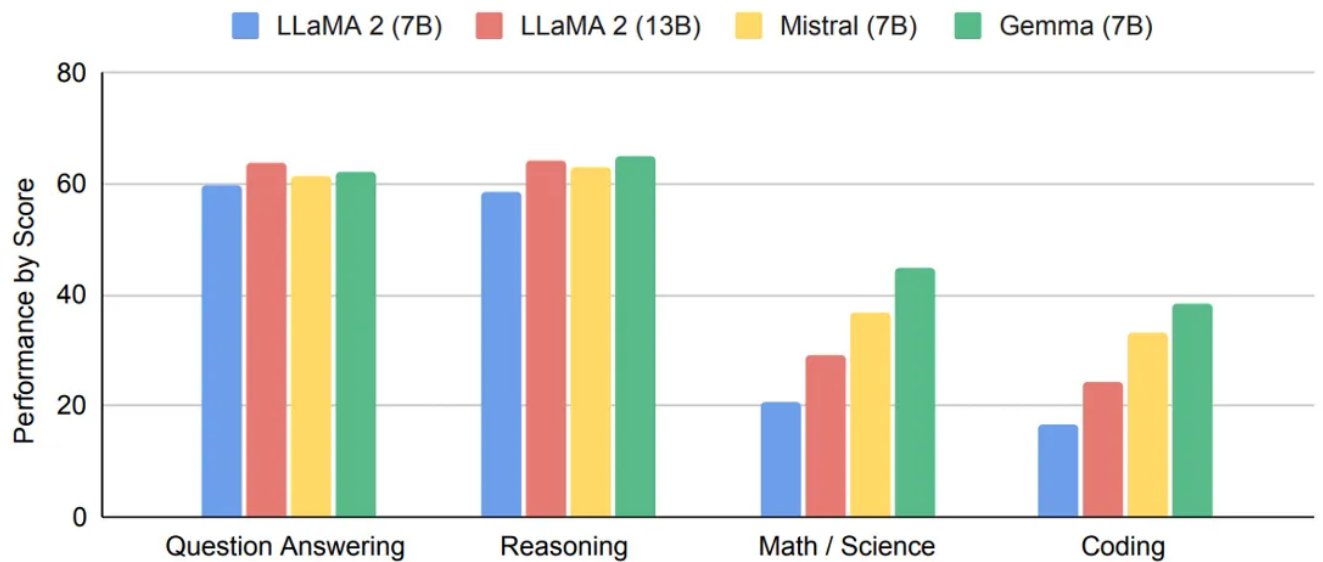


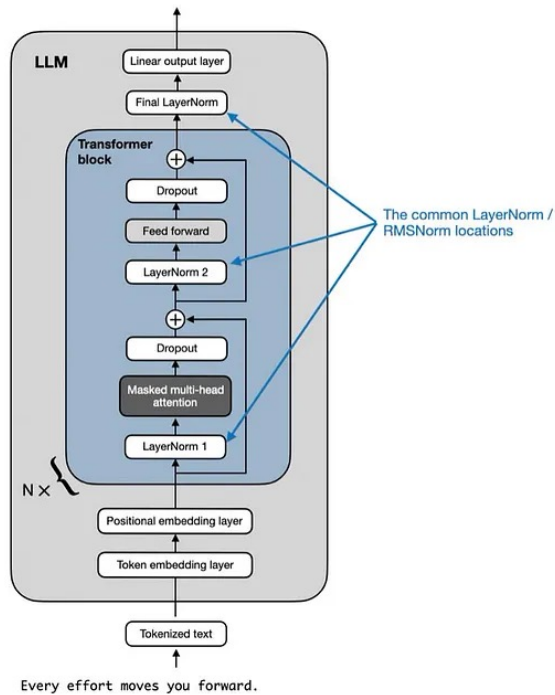
Figure 1 | Language understanding and generation performance of Gemma 7B across different capabilities compared to similarly sized open models. We group together standard academic benchmark evaluations by capability and average the respective scores;

Here is a table showing the relevant formatting control tokens available in Gemma:

Context	Relevant Token
User turn	<code>user</code>
Model turn	<code>model</code>
Start of conversation turn	<code><start_of_turn></code>
End of conversation turn	<code><end_of_turn></code>

You can also use the special control tokens in the context of a multi-turn user prompt as follows:

```
<start_of_turn>user
What is a good place for travel in the US?<end_of_turn>
<start_of_turn>model
California.<end_of_turn>
<start_of_turn>user
What can I do in California?<end_of_turn>
<start_of_turn>model
```



	Gemma		Llama 2 7B	OLMo 7B
Parameters	2B	7B		
d_{model}	2048	3072	4096	4096
Layers	18	28	32	32
Feedforward hidden dims	32768	49152	11008	8192
Num heads	8	16	32	32
Num KV heads	1	16	16	16
Head size	256	256	128	128
Vocab size	256128	256128	32000	50280
Exact parameter number	9,324,112,896		6,738,415,616	6,888,628,224
Accounting for weight tying	8,537,680,896			

Extremely large vocabulary size

Sigmoidal part

Linear projection

$$\text{GEGLU}(x, W, V, b, c) = \text{GELU}(xW + b) \otimes (xV + c)$$
$$\text{SwiGLU}(x, W, V, b, c, \beta) = \text{Swish}_{\beta}(xW + b) \otimes (xV + c)$$

Gemma benefits from many learnings of the Gemini model program including code, data, architecture, instruction tuning, reinforcement learning from human feedback, and evaluations.

Here's a quick breakdown of their technical paper:

👉 Model Architecture

Based on the original transformer decoder architecture (From “Attention Is All You Need” paper) with the below improvements:

- 🚩 Multi-Query Attention instead of the original multi-head attention.
- 🚩 RoPE Embeddings in each layer, sharing them across inputs and outputs to reduce model size.
- 🚩 GeGLU Activations instead of ReLU
- 🚩 Normalizer Location: Normalizes both input and output of each transformer sub-layer using RMSNorm.

👉 Training Data

- 🚩 Trained on English data from web documents, mathematics, and code, using a modified subset of the SentencePiece tokenizer from Gemini. Filtering is done to improve alignment.

👉 Instruction Tuning

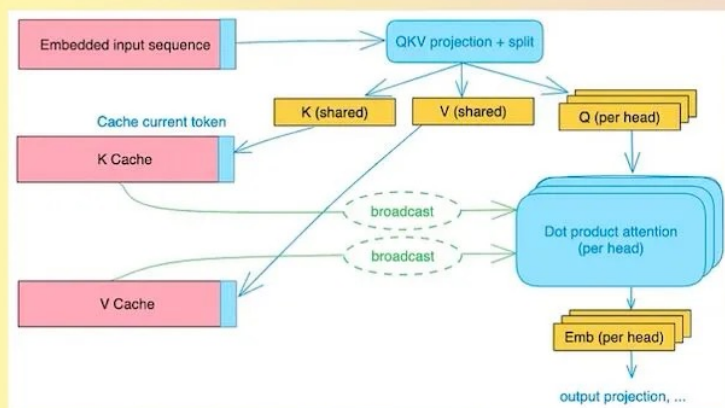
Uses SFT(Supervised Fine-Tuning) and RLHF(Reinforcement Learning from Human Feedback)

- 🚩 SFT: Data mixes are chosen by comparing responses from different models based on human preference. Prompts focus on following instructions and being safe and use automatic judges.
- 🚩 RLHF: Further fine-tuning is conducted by gathering human preferences and training a policy.

The core parameters of the architecture are summarized in Table 1. Models are trained on a context length of 8192 tokens.

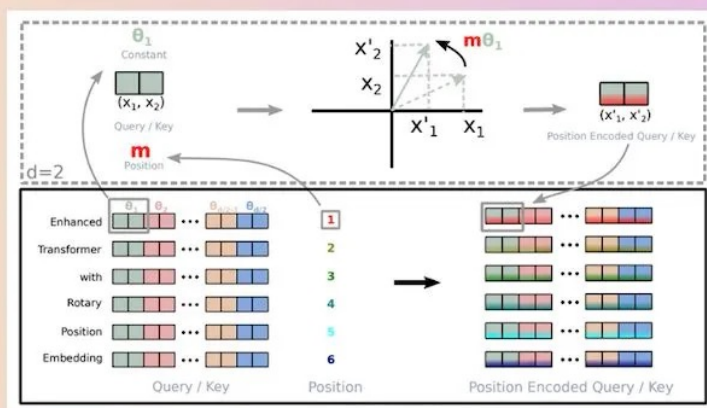
Google Gemma Architecture Deep-dive

Created By: Aishwarya Naresh Reganti



Multi-Query Attention used in Gemma

(Image Source: <https://blog.fireworks.ai/multi-query-attention-is-all-you-need-db072e758055>)



RoPE embeddings used in Gemma

(Image Source: <https://arxiv.org/pdf/2104.09864.pdf>)

Model Architecture

- Gemma model architecture is based on the transformer decoder by Vaswani et al. (2017).
- Improvements:
 - Multi-Query Attention:** Utilizes multi-query attention (Shazeer, 2019) instead of the original multi-head attention.
 - RoPE Embeddings:** Integrates rotary positional embeddings (Su et al., 2021) in each layer, sharing them across inputs and outputs to reduce model size.
 - GeGLU Activations:** Replaces ReLU with the GeGLU activation function (Shazeer, 2020).
 - Normalizer Location:** Normalizes both input and output of each transformer sub-layer using RMSNorm, deviating from the standard practice.

Training Infra & Data

- Training Infrastructure:** Gemma models train on TPUv5e chips, with the 7B model using 4096 chips across 16 pods and the 2B model using 512 chips across 2 pods, utilizing sharding and replication techniques.
- Pretraining:** Gemma 2B and 7B models are trained on English data from web documents, mathematics, and code, using a modified subset of the SentencePiece tokenizer from Gemini. The dataset undergoes filtering to remove undesirable or unsafe content and personal information.
- Carbon Footprint:** Pretraining Gemma models emits about 131 tons of CO2 equivalent, based on TPU datacenter energy usage, which is offset by Google's carbon-neutral data centers.

Instruction Tuning

- Gemma models undergo two fine-tuning methods: **SFT** (Supervised Fine-Tuning) and **RLHF** (Reinforcement Learning from Human Feedback).
- SFT:** Data mixes are chosen by comparing responses from different models to see which ones people prefer. Different prompts focus on things like following instructions and being safe, with automatic judges using methods that match what people like.
- Formatting:** Models are trained with a specific formatter annotating examples with extra information, indicating conversation roles and turns.
- RLHF:** Further fine-tuning is conducted by gathering human preferences and training a policy.

Parameters	2B	7B
d_{model}	2048	3072
Layers	18	28
Feedforward hidden dims	32768	49152
Num heads	8	16
Num KV heads	1	16
Head size	256	256
Vocab size	256128	256128

Table 1 | Key model parameters.

Model	Embedding Parameters	Non-embedding Parameters
2B	524,550,144	1,981,884,416
7B	786,825,216	7,751,248,896

Table 2 | Parameter counts for both sizes of Gemma models.

What are some of the interesting design choices behind Gemma?

- 1) Gemma uses a really large vocabulary and consequently embedding weight matrices.
- 2) Like OLMo and GPT-2, Gemma also uses weight tying.
- 3) It even uses GeLU (in the form of GeGLU), similar to GPT-2, unlike Llama 2 and other LLMs.
- 4) Interestingly, Gemma normalizes the embeddings by the square root of the hidden layer dimension.
- 5) The RMSNorm layers are in the usual location, different from what was hinted at in the technical paper.
- 6) However, the RMSNorm implementation comes with the unit offset that wasn't mentioned in the technical paper.

As mentioned above, its vocabulary size (and consequently the embedding matrix size) is very large. The table below shows an architecture overview comparing Gemma to LLama 2 7B and OLMo 7B.

7 LLM Generation Parameters

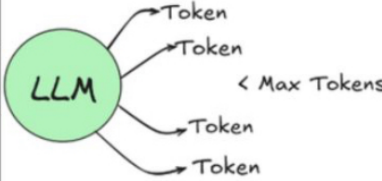
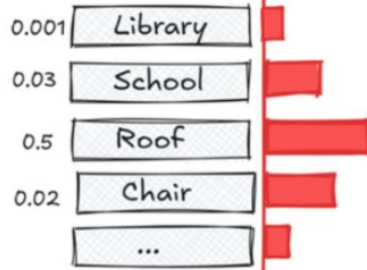
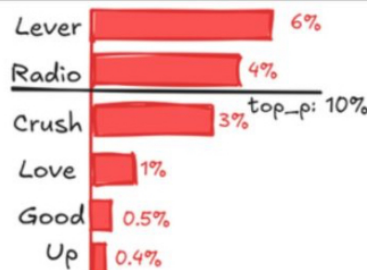
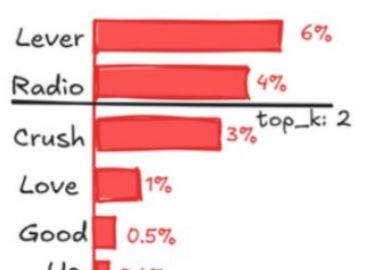
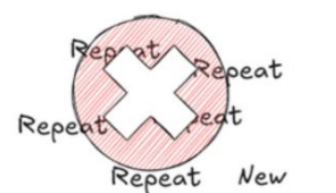


NON
BRAND
DATA

Non-Brand Data

To Know

Explanation

Value Range

max_tokens		Upper limit for the number of tokens the model generates	1 to infinity
temperature		Controls randomness in output. A higher temperature makes more creative and diverse	0 to 2 (Common Range)
top_p		Controls probability distribution is considered when sampling tokens	0 to 1
top_k		Limits the number of top probable tokens to sample from	1 to infinity
frequency penalty		Penalizes token repetition based on frequency. Positive values reduce repetition	-2 to 2
presence penalty		Encourages the model to use new tokens that haven't been generated	-2 to 2
stop		A list of tokens where the model will stop generating further tokens	Custom List

LLM Hyperparameters

