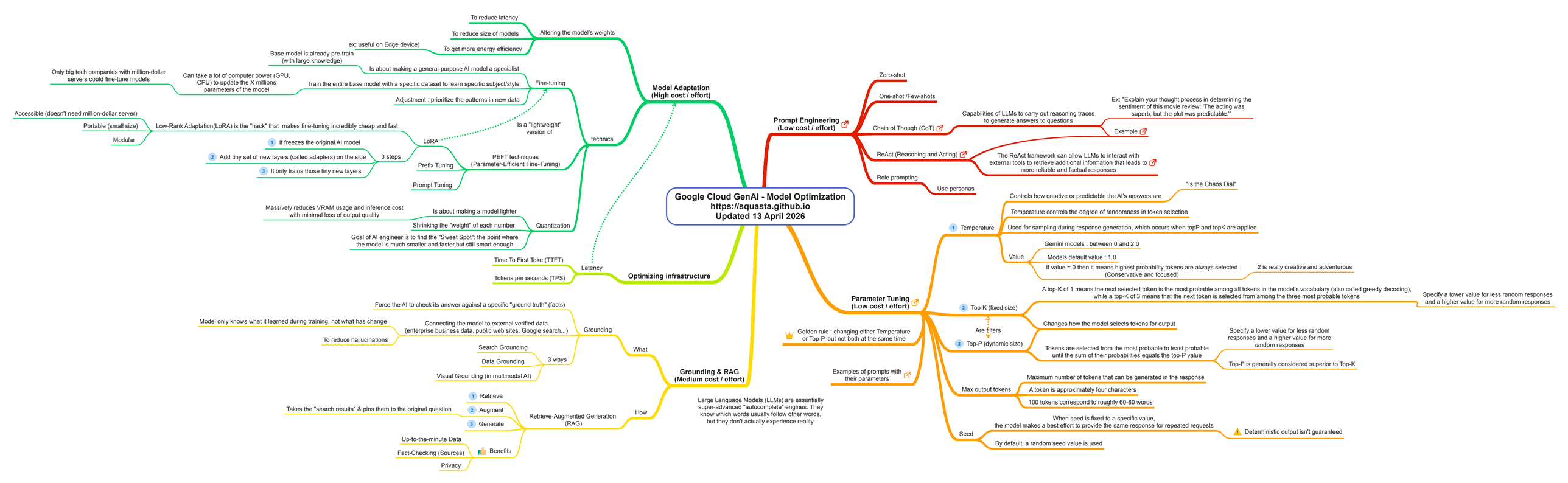
Google Cloud GenAI - Model Optimization
https://squasta.github.io
Updated 13 April 2026
Model Adaptation
(High cost / effort)
Altering the model's weights
To reduce latency
To reduce size of models
To get more energy efficiency
ex: useful on Edge device)
Fine-tuning
Base model is already pre-train
(with large knowledge)
Is about making a general-purpose AI model a specialist
Train the entire base model with a specific dataset to learn specific subject/style
Can take a lot of computer power (GPU,
CPU) to update the X millions
parameters of the model
Only big tech companies with million-dollar
servers could fine-tune models
Adjustment : prioritize the patterns in new data
technics
Is a "lightweight"
version of
PEFT techniques
(Parameter-Efficient Fine-Tuning)
LoRA
Low-Rank Adaptation(LoRA) is the "hack" that makes fine-tuning incredibly cheap and fast
Accessible (doesn't need million-dollar server)
Portable (small size)
Modular
3 steps
1 It freezes the original AI model
2 Add tiny set of new layers (called adapters) on the side
3 It only trains those tiny new layers
Prefix Tuning
Prompt Tuning
Quantization
Massively reduces VRAM usage and inference cost
with minimal loss of output quality
Is about making a model lighter
Shrinking the "weight" of each number
Goal of AI engineer is to find the "Sweet Spot": the point where
the model is much smaller and faster,but still smart enough
Optimizing infrastructure
Latency
Time To First Toke (TTFT)
Tokens per seconds (TPS)
Grounding & RAG
(Medium cost / effort)
What
Grounding
Connecting the model to external verified data
(enterprise business data, public web sites, Google search...)
Force the AI to check its answer against a specific "ground truth" (facts)
Model only knows what it learned during training, not what has change
To reduce hallucinations
3 ways
Search Grounding
Data Grounding
Visual Grounding (in multimodal AI)
How
Retrieve-Augmented Generation
(RAG)
1 Retrieve
2 Augment
3 Generate
Takes the "search results" & pins them to the original question
Benefits
Up-to-the-minute Data
Fact-Checking (Sources)
Privacy
Large Language Models (LLMs) are essentially
super-advanced "autocomplete" engines. They
know which words usually follow other words,
but they don't actually experience reality.
Prompt Engineering
(Low cost / effort)
Zero-shot
One-shot /Few-shots
Chain of Though (CoT)
Capabilities of LLMs to carry out reasoning traces
to generate answers to questions
Example
Ex: "Explain your thought process in determining the
sentiment of this movie review: 'The acting was
superb, but the plot was predictable.'"
ReAct (Reasoning and Acting)
The ReAct framework can allow LLMs to interact with
external tools to retrieve additional information that leads to
more reliable and factual responses
Role prompting
Use personas
Parameter Tuning
(Low cost / effort)
Golden rule : changing either Temperature
or Top-P, but not both at the same time
Examples of prompts with
their parameters
1 Temperature
Controls how creative or predictable the AI's answers are
"Is the Chaos Dial"
Temperature controls the degree of randomness in token selection
Used for sampling during response generation, which occurs when topP and topK are applied
Value
Gemini models : between 0 and 2.0
Models default value : 1.0
If value = 0 then it means highest probability tokens are always selected
(Conservative and focused)
2 is really creative and adventurous
2 Top-K (fixed size)
A top-K of 1 means the next selected token is the most probable among all tokens in the model's vocabulary (also called greedy decoding),
while a top-K of 3 means that the next token is selected from among the three most probable tokens
Specify a lower value for less random responses
and a higher value for more random responses
Are filters
3 Top-P (dynamic size)
Changes how the model selects tokens for output
Tokens are selected from the most probable to least probable
until the sum of their probabilities equals the top-P value
Specify a lower value for less random
responses and a higher value for more
random responses
Top-P is generally considered superior to Top-K
Max output tokens
Maximum number of tokens that can be generated in the response
A token is approximately four characters
100 tokens correspond to roughly 60-80 words
Seed
When seed is fixed to a specific value,
the model makes a best effort to provide the same response for repeated requests
Deterministic output isn't guaranteed
By default, a random seed value is used