Fine-tuning `gemini-2.0-flash` (Vertex AI)

1. **Сформируйте обучающий JSONL**

   Экспортируем 100 последних реплик пользователя из MongoDB при помощи `ConversationManager.get_user_conversation_history`[1] и превращаем их в требуемый формат[2]:

   Листинг 1: `make_dataset.py`

```
import asyncio, json, random
from conversation_manager import ConversationManager

async def build_jsonl(uid: int, out="train.jsonl", n=100):
    cm = ConversationManager("mongodb://localhost:27017")
    history = await cm.get_user_conversation_history(uid, n)

    #                                                  user->bot
    pairs, buf = [], None
    for msg in history:
        if msg["sender"] == "user":
            buf = msg["text"]
        elif buf:                              #  b o t reply
            pairs.append((buf, msg["text"]))
            buf = None

    random.shuffle(pairs)
    with open(out, "w", encoding="utf-8") as f:
        for user, bot in pairs:
            sample = {
              "contents": [
                {"role": "user",  "parts": [{"text": user}]},
                {"role": "model", "parts": [{"text": bot}]}
              ]
            }
            f.write(json.dumps(sample, ensure_ascii=False) + "
               \n")

asyncio.run(build_jsonl(123456789))
```

2. **Загрузите датасет на Cloud Storage**

```
gsutil mb -l us-central1 gs://namazapp-tuning
gsutil cp train.jsonl gs://namazapp-tuning/
```

   Важно: регион `us-central1` обязателен для Gemini-тюнинга[3].

3. **Создайте supervised-tuning job (Python SDK)**

[1] См. исходник `conversation_manager.py`.

[2] Каждая строка — одна пара `user`→`model` внутри ключа `contents`; формат описан в документации Vertex AI Supervised Tuning.

[3] Указано в разделе *"Gemini models you can fine-tune"* официальной документации Vertex AI.

```
from vertexai.preview.tuning.sft import SupervisedTuningJob
from google.cloud import aiplatform

PROJECT  = "your-gcp-project"
REGION   = "us-central1"
DATA_URI = "gs://namazapp-tuning/train.jsonl"

aiplatform.init(project=PROJECT, location=REGION)

job = SupervisedTuningJob.create(
    display_name    = "namazapp-gemini-sft",
    source_model    = "gemini-2.0-flash",
    tuning_data_uri = DATA_URI,
    train_steps     = 2000,
    learning_rate   = 1e-5,
)
job.wait()                                  #         3060
print("Endpoint:", job.tuned_model_endpoint_name)
```

Модели, поддерживающие тюнинг (Gemini 2.0 Flash, Flash-Lite, 2.5 Flash), перечислены в официальных справочных таблицах Vertex AI[4].

4. **Проверьте результат**

```
from vertexai.generative_models import GenerativeModel

tuned = GenerativeModel(job.tuned_model_endpoint_name)
print(tuned.generate_content("          ,␣      ␣          ␣
            ?").text)
```

5. **Интегрируйте в существующий клиент**

В `PerfectGPTClient` поле `llm_model` сейчас жёстко равно `"gemini-2.0-flash"`[5]. Замените его на ID эндпоинта:

```
- "llm_model": "gemini-2.0-flash",
+ "llm_model": "projects/   /endpoints/ENDPOINT_ID",
```

6. **(Опция) REST-запрос без SDK**

```
echo '{
␣␣"contents":[{"role":"USER","parts":{"text":"              ␣
        ␣                ?"}}]
}' > request.json

curl -X POST \
  -H "Authorization:␣Bearer␣$(gcloud␣auth␣print-access-token)"
      \
  -H "Content-Type:␣application/json;␣charset=utf-8" \
```

[4] Документ *"Fine-tune Gemini models"*, раздел «Supported base models».
[5] См. `perfect_gpt_client.py`.

```
  -d @request.json \
  "https://us-central1-aiplatform.googleapis.com/v1/projects/$
     {PROJECT}/locations/us-central1/endpoints/${ENDPOINT_ID}:
     generateContent"
```

Формат запроса приведён в REST-примере официального гайда Vertex AI.