Mini Fiche : Accélération des réponses LLM via Cache Redis

Ahmed Bassoul

November 26, 2025

1 Introduction

L'objectif de ce travail est de réduire le temps de réponse des requêtes posées à un LLM dans une application de recettes culinaires. Lorsqu'une question similaire est posée plusieurs fois, il est plus efficace d'utiliser un cache pour retourner la réponse instantanément plutôt que de recalculer le résultat avec le modèle.

2 Stratégie de Cache

Nous avons utilisé Redis pour stocker :

- La question posée.
- L'embedding de la question (vecteur SentenceTransformer).
- La réponse générée par le LLM.

Pour détecter des questions similaires, nous utilisons la **cosine similarity** entre l'embedding de la question en cours et les embeddings des questions stockées. Un seuil de similarité (ici 0.85) permet de réutiliser la réponse existante si la question est jugée similaire.

3 Fonctionnement

3.1 Première requête

1. L'utilisateur pose une question.
2. Le LLM traite la question et génère la réponse.
3. La réponse et son embedding sont stockés dans Redis.
4. Temps de réponse : longue (200 sec pour notre exemple).

3.2 Requête similaire ultérieure

1. L'utilisateur pose une question similaire.
2. Redis détecte une question proche via embedding et cosine similarity.
3. La réponse est renvoyée instantanément depuis le cache.
4. Temps de réponse : très court (5 sec).

4 Code principal Python

```
import redis, json, torch, requests
from sentence_transformers import SentenceTransformer, util

r = redis.Redis(host='localhost', port=6379, db=0)
EMBEDDING_MODEL_NAME = "multi-qa-mpnet-base-dot-v1"
SIMILARITY_THRESHOLD = 0.85
sim_model = SentenceTransformer(EMBEDDING_MODEL_NAME, device="cpu")

def get_embedding(text):
    return sim_model.encode(text, convert_to_tensor=True)

def find_similar_question(query_emb):
    for key in r.keys():
        raw = r.get(key)
        if not raw: continue
        try:
            cached_data = json.loads(raw)
        except json.JSONDecodeError:
            continue
        cached_emb = torch.tensor(cached_data["embedding"])
        if util.cos_sim(query_emb, cached_emb).item() >=
           SIMILARITY_THRESHOLD:
            return key.decode(), cached_data["response"]
    return None, None

def ask_llm_with_redis_smart(question):
    query_emb = get_embedding(question)
    similar_key, cached_response = find_similar_question(query_emb)
    if cached_response:
        from_cache = True
        response_text = cached_response
    else:
        # automate_asking = fonction qui appelle le LLM
        response_text = automate_asking(question, ollama_model_name="
           llama3.2")
        data_to_cache = {
            "embedding": query_emb.tolist(),
            "response": response_text
        }
        r.set(question, json.dumps(data_to_cache))
        from_cache = False
    return response_text, from_cache
```

5 Résultats

Exemple de test avec deux questions similaires :

```
start = time.time()
res, from_cache = ask_llm_with_redis_smart("can you tell me a Dessert
    recette Banana Pancakes")
print("The LLM Resposne is :\n")
display(Markdown(res))
print("Cache:", from_cache)
print("Temps:", time.time() - start)

print("\n\n\n##################################\n\n\n")

start = time.time()
res, from_cache = ask_llm_with_redis_smart("i want a Dessert recette
    Banana Pancakes")
print("The LLM Resposne is :\n")
display(Markdown(res))
print("Cache:", from_cache)
print("Temps:", time.time() - start)
```

Resultat :

```
The LLM Resposne is :
A delicious dessert! According to our database, I have just the recipe
    for you: Banana Pancakes!

Here's how to make it:

Ingredients:
- 1 large banana
- 2 medium eggs
- A pinch of baking powder
- A sprinkling of vanilla extract
- 1 tsp oil
- 25g pecan nuts
- 125g raspberries

Instructions:
1. Mash the banana with a fork until it resembles a thick pur e.
2. Stir in the eggs, baking powder, and vanilla extract.
3. Heat a large non-stick frying pan or pancake pan over medium heat and
     brush with half the oil.
4. Using half the batter, spoon two pancakes into the pan, cook for 1-2
    mins each side, then tip onto a plate.
5. Repeat the process with the remaining oil and batter.
6. Top the pancakes with pecans and raspberries.

Enjoy your Banana Pancakes!

(Note: I couldn't find any additional information about this recipe in
    our database. If you'd like more guidance or variations, feel free to
     ask!)
Cache: False
Temps: 204.81155562400818
```

```
############################################################



Using cached response for a similar question!
The LLM Resposne is :
A delicious dessert! According to our database, I have just the recipe
   for you: Banana Pancakes!

Here's how to make it:

Ingredients:
- 1 large banana
- 2 medium eggs
- A pinch of baking powder
- A sprinkling of vanilla extract
- 1 tsp oil
- 25g pecan nuts
- 125g raspberries

Instructions:
1. Mash the banana with a fork until it resembles a thick pur e.
2. Stir in the eggs, baking powder, and vanilla extract.
3. Heat a large non-stick frying pan or pancake pan over medium heat and
    brush with half the oil.
4. Using half the batter, spoon two pancakes into the pan, cook for 1-2
   mins each side, then tip onto a plate.
5. Repeat the process with the remaining oil and batter.
6. Top the pancakes with pecans and raspberries.

Enjoy your Banana Pancakes!

(Note: I couldn't find any additional information about this recipe in
   our database. If you'd like more guidance or variations, feel free to
    ask!)
Cache: True
Temps: 5.0426716804504395
```

- Question 1 : "can you tell me a Dessert recette Banana Pancakes"
  - Cache : False
  - Temps : 204 sec
- Question 2 : "i want a Dessert recette Banana Pancakes"
  - Cache : True
  - Temps : 5 sec

6 Utilité pour l'application de recettes

- Réduction massive du temps de réponse.
- Détection des questions formulées différemment mais similaires.
- Meilleure expérience utilisateur.

- Économie de ressources CPU/GPU lors de l'utilisation d'un LLM lourd.

7 Conclusion

L'utilisation d'un cache basé sur embeddings permet d'accélérer de manière significative le traitement des questions récurrentes et améliore l'efficacité globale de l'application de recettes.