dgroove
Tecnologia emozionale.

ICT Training Center

Il tuo partner per la Formazione e la Trasformazione digitale della tua azienda

Spring AI

Generative Artificial Intelligence con Java

Simone Scannapieco

Corso avanzato per Venis S.p.A, Venezia, Italia

Novembre 2025

AUTOMATIC PROMPTING

1. *Automatic prompting*
   - Tecnica per sfruttare il codice genetico del LLM
   - Strategia: scrivere *prompt* che generano *prompt* più allineati con il modo di ragionare del LLM
     - Si chiede al LLM di generare varianti semantiche del *prompt* utente
     - Il LLM analizza tali varianti in base a delle metriche di valutazione del risultato
       - BLEU (BiLingual Evaluation Understudy)
       - ROUGE (Recall-Oriented Understudy for Gisting Evaluation)
     - L'utente ottiene delle varianti semantiche da poter ulteriormente raffinare
   - ⚠ Sgrava l'utente da parte della responsabilità nella scrittura di un buon *prompt*
   - Vantaggi
     - Approccio semi-automatizzato per la creazione di *dataset* mirati
   - Svantaggi
     - Richiede un approccio iterativo per determinare le migliori formulazioni
     - Metriche classiche sono ottimi indicatori ma non specializzate per la *task*

- Sistema di raffinamento di richieste per *chatbot* specializzati
  1. Creazione *prompt template* per generazione varianti semantiche (*system* e *user*)
  2. Creazione modelli `Prompt.java`, `PromptEvaluationRequest.java` e `PromptEvaluationResponse.java` per serializzare e deserializzare richieste al/generazioni dal LLM
  3. Creazione enumeratore `EvaluationMetric.java` con i possibili algoritmi di valutazione
  4. Modifiche ad interfaccia ed implementazione del servizio Gemini
  5. Modifica del controllore MVC per servizio Gemini
  6. *Test* delle funzionalità con Postman/Insomnia

File set-prompt-alternatives-system-prompt.st

```
Sei un generatore automatico di prompt.
Lo '''user''' deve addestrare un chatbot e ha bisogno di diverse varianti
semantiche della '''richiesta''' da usare come dati di addestramento.
Genera una lista puntata di {numero} varianti della richiesta dello '''user'''.
```

File get-prompt-alternatives-user-prompt.st

```
'''Richiesta''': {prompt}
```

File `get-ordered-prompt-alternatives-prompt.st`

```
Applica la metrica di valutazione {metrica} alle seguenti varianti: {lista}.
Ordina e restituisci ciascuna variante e relativo punteggio in ordine decrescente
di punteggio di valutazione.
{formato}
```

Enumeratore per metriche di valutazione

```java
package it.venis.ai.spring.demo.data;

public enum EvaluationMetric {

    BLEU("BLEU (Bilingual Evaluation Understudy)"),
    ROUGE_1("ROUGE-1 (Recall-Oriented Understudy for Gisting Evaluation con sovrapposizione di unigrammi)"),
    ROUGE_2("ROUGE-2 (Recall-Oriented Understudy for Gisting Evaluation con sovrapposizione di bigrammi)"),
    ROUGE_L("ROUGE-L (Recall-Oriented Understudy for Gisting Evaluation basato su Least Common Subsumer)"),
    ROUGE_W("ROUGE-W (Recall-Oriented Understudy for Gisting Evaluation basato su Least Common Subsumer pesato)"),
    ROUGE_S("ROUGE-S (Recall-Oriented Understudy for Gisting Evaluation basato su co-occorrenza di skip-bigrammi)"),
    ROUGE_SU("ROUGE-SU (Recall-Oriented Understudy for Gisting Evaluation basato co-occorrenza di skip-bigrammi e unigrammi)");

    private String metric;

    EvaluationMetric(String metric) {

        this.metric = metric;

    }

    public String getMetric() {

        return metric;

    }

}
```

Modello per prompt

```
package it.venis.ai.spring.demo.model;

import com.fasterxml.jackson.annotation.JsonPropertyDescription;

public record Prompt(
    @JsonPropertyDescription("Il contenuto del prompt") String prompt,
    @JsonPropertyDescription("Il punteggio del prompt secondo la metrica") Double evaluation_score) {
}
```

Modello per richiesta di prompt

```java
package it.venis.ai.spring.demo.model;

import java.util.Objects;
import java.util.UUID;

import it.venis.ai.spring.demo.data.EvaluationMetric;

public record PromptEvaluationRequest(UUID id, String prompt, EvaluationMetric metric) {

    public PromptEvaluationRequest(String prompt, EvaluationMetric metric) {
        this(UUID.randomUUID(), prompt, metric);
    }

    @Override
    public EvaluationMetric metric() {
        return Objects.requireNonNullElse(this.metric, EvaluationMetric.BLEU);
    }

}
```

Modello per lisyta ordinata di varianti di prompt

```
package it.venis.ai.spring.demo.model;

import java.util.List;

public record PromptEvaluationResponse(List<Prompt> evaluated_prompts) {

}
```

Interfaccia servizio Gemini

```
package it.venis.ai.spring.demo.services;

import it.venis.ai.spring.demo.model.Answer;
import it.venis.ai.spring.demo.model.ArtifactRequest;
import it.venis.ai.spring.demo.model.DefinitionRequest;
import it.venis.ai.spring.demo.model.DefinitionResponse;
import it.venis.ai.spring.demo.model.Question;

public interface GeminiFromClientService {

    String getAnswerFromClient(String question);

    Answer getAnswerFromClient(Question question);

    Answer getDefinitionFromClient(DefinitionRequest definitionRequest);

    Answer getCustomFormatDefinitionFromClient(DefinitionRequest definitionRequest);

    Answer getJSONUserFormatDefinitionFromClient(DefinitionRequest definitionRequest);

    DefinitionResponse getJSONOutputConverterFormatDefinitionFromClient(DefinitionRequest definitionRequest);

    Answer getSentimentForArtifact(ArtifactRequest artifactRequest);

    Answer getNERinYAMLForArtifact(ArtifactRequest artifactRequest);

    Answer getSuggestionForArtifact(ArtifactRequest artifactRequest);

    Artifact getGeneratedArtifact(ArtifactRequest artifactRequest, Integer numChoices, Integer numParagraphs);

    Answer getGenreForArtifact(ArtifactRequest artifactRequest);

    PromptEvaluationResponse getEvaluatedPrompts(PromptEvaluationRequest promptEvaluationRequest);
}
```

Implementazione servizio Gemini - I

```
package it.venis.ai.spring.demo.services;

import java.util.Map;
import org.springframework.ai.chat.client.ChatClient;
import org.springframework.ai.template.st.StTemplateRenderer;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.core.io.Resource;
import org.springframework.stereotype.Service;
import it.venis.ai.spring.demo.model.Answer;
import it.venis.ai.spring.demo.model.DefinitionRequest;
import it.venis.ai.spring.demo.model.Question;

@Service
public class GeminiFromClientServiceImpl implements GeminiFromClientService {

    private final ChatClient chatClient;

    public GeminiFromClientServiceImpl(ChatClient.Builder chatClientBuilder) {
        this.chatClient = chatClientBuilder.build();
    }

    @Override
    public String getAnswerFromClient(String question) {
        return this.chatClient.prompt()
                .user(question)
                .call()
                .content();
    }
    ...
```

Implementazione servizio Gemini - II

```
...
@Value("classpath:templates/set-prompt-alternatives-system-prompt.st")
private Resource promptAlternativesSystemPrompt;

@Value("classpath:templates/get-prompt-alternatives-user-prompt.st")
private Resource promptAlternativesUserPrompt;

@Value("classpath:templates/get-ordered-prompt-alternatives-prompt.st")
private Resource orderedPromptAlternativesPrompt;

@Override
public PromptEvaluationResponse getEvaluatedPrompts(PromptEvaluationRequest promptEvaluationRequest) {
    BeanOutputConverter<PromptEvaluationResponse> converter =
                        new BeanOutputConverter<>(PromptEvaluationResponse.class);
    String listResponse = this.chatClient.prompt()
            .options(ChatOptions.builder()
            .model("gemini-2.0-flash")
            .temperature(2.0)
            //.topP(1.0)
            //.topK(30)
            .maxTokens(2000)
            //.frequencyPenalty(0.1)
            //.presencePenalty(0.1)
            .build())
            .system(s -> s.text(this.promptAlternativesSystemPrompt)
                    .params(Map.of("numero", 10)))
            .user(u -> u.text(this.promptAlternativesUserPrompt)
                    .params(Map.of("prompt", promptEvaluationRequest.prompt())))
            .templateRenderer(StTemplateRenderer.builder().startDelimiterToken('{')
                    .endDelimiterToken('}')
                    .build())
            .call()
            .content();

...
```

Implementazione servizio Gemini - III

```
    ...

        String orderedListResponse = this.chatClient.prompt()
                .options(ChatOptions.builder()
                .model("gemini-2.0-flash")
                .temperature(0.0)
                //.topP(1.0)
                //.topK(30)
                .maxTokens(2000)
                //.frequencyPenalty(0.1)
                //.presencePenalty(0.1)
                .build())
                .user(u -> u.text(this.orderedPromptAlternativesPrompt)
                        .params(Map.of("metrica", promptEvaluationRequest.metric().getMetric(),
                                       "lista", listResponse,
                                       "formato", converter.getFormat())))
                .templateRenderer(StTemplateRenderer.builder().startDelimiterToken('{')
                        .endDelimiterToken('}')
                        .build())
                .call()
                .content();

        return converter.convert(orderedListResponse);

    }

}
```

Implementazione controllore REST

```
package it.venis.ai.spring.demo.controllers;

import org.springframework.web.bind.annotation.RestController;
...
import it.venis.ai.spring.demo.services.GeminiFromClientService;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;

@RestController
public class QuestionFromClientController {

    private final GeminiFromClientService geminiService;

    ...

    @PostMapping("/client/genre")
    public Answer getGeneratedArtifact(@RequestBody ArtifactRequest artifactRequest) {

        return this.geminiService.getGenreForArtifact(artifactRequest);

    }

    @PostMapping("/client/evaluate")
    public PromptEvaluationResponse getEvaluatedPrompts(@RequestBody PromptEvaluationRequest promptEvaluationRequest) {

        return this.geminiService.getEvaluatedPrompts(promptEvaluationRequest);

    }

}
```

https://github.com/simonescannapieco/spring-ai-base-dgroove-venis-code.git

Branch: 18-spring-ai-gemini-automatic-prompting