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 base per Venis S.p.A, Venezia, Italia

Settembre 2025

🔧 Implicit e zero-shot prompting

- *Implicit prompting*
  1. *Test* su REST API `/ask` per traduzione ITA/ENG, contrari, ...
- *Zero-shot prompting per sentiment analysis*
  1. Creazione enumeratori `Artifact.java` e `Sentiment.java`
  2. Creazione modello richiesta per artefatto `ArtifactRequest.java`
  3. Creazione *prompt template* per *sentiment analysis* di tipo *0-shot* per artefatti
  4. Modifiche ad interfaccia ed implementazione del servizio Gemini
  5. Modifica del controllore MVC per servizio Gemini

Enumeratore per artefatti

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

public enum Artifact {

    LIBRO("Libro"),
    FILM("Film"),
    SPETTACOLO("Spettacolo");

    private String artifact;

    Artifact(String artifact) {
        this.artifact = artifact;
    }

    public String getArtifact() {
        return artifact;
    }

}
```

Enumeratore per *sentiment*

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

public enum Sentiment {

    ESTREMAMENTE_POSITIVO("Estremamente positivo"),
    POSITIVO("Positivo"),
    NEUTRALE("Neutrale"),
    NEGATIVO("Negativo"),
    ESTREMAMENTE_NEGATIVO("Estremamente negativo");

    private String sentiment;

    Sentiment(String sentiment) {
        this.sentiment = sentiment;
    }

    public String getSentiment() {
        return sentiment;
    }

}
```

Modello di richiesta per artefatto

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

import java.util.UUID;

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

public record ArtifactRequest(UUID id, String title, Artifact type, String review) {

    public ArtifactRequest(String title, Artifact type, String review) {
        this(UUID.randomUUID(), title, type, review);
    }

}
```

⚠ Il *path* `resources/templates` viene creato automaticamente da Spring AI in fase di creazione del progetto

File `get-artifact-sentiment-prompt.st`

```
Classifica recensioni di artefatti come ESTREMAMENTE POSITIVO, POSITIVO, NEUTRALE,
NEGATIVO, ESTREMAMENTE NEGATIVO.
Recensione {artefatto}: {recensione}
Sentimento:
```

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);

}
```

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/get-artifact-sentiment-prompt.st")
    private Resource artifactSentimentPrompt;

    @Override
    public Answer getSentimentForArtifact(ArtifactRequest artifactRequest) {
        Sentiment chatResponse = this.chatClient.prompt()
                .options(ChatOptions.builder()
                //.model("gemini-2.0-flash")
                .temperature(0.1)
                .topP(1.0)
                //.topK(30)
                .maxTokens(10)
                //.frequencyPenalty(0.1)
                //.presencePenalty(0.1)
                .build())
                .user(u -> u.text(this.artifactSentimentPrompt)
                        .params(Map.of("recensione", artifactRequest.review(), "artefatto", artifactRequest.type())))
                .templateRenderer(StTemplateRenderer.builder().startDelimiterToken('{')
                        .endDelimiterToken('}')
                        .build())
                .call()
                .entity(Sentiment.class);

        return new Answer(chatResponse.getSentiment());

    }

}
```

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/definition/json/user")
    public Answer getJSONUserFormatDefinition(@RequestBody DefinitionRequest definitionRequest) {

        return this.geminiService.getJSONUserFormatDefinitionFromClient(definitionRequest);

    }

    ...

    @PostMapping("/client/sentiment")
    public Answer getSentiment(@RequestBody ArtifactRequest artifactRequest) {

        return this.geminiService.getSentimentForArtifact(artifactRequest);

    }

}
```

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

Branch: `13-spring-ai-gemini-implicit-zero-shot-prompting`