A multi-stage pipeline with Large Language Model to automated survey generation

Gustavo Bartz Guedes
Leandro Carísio Fernandes

Objetivo

- Geração automática de *survey* utilizando *Large Language Models*.
- Duas abordagens implementadas:
  1. Geração automática:
     - Entrada é apenas o título da *survey*.
  2. Geração semiautomática:
     - Entrada consiste em:
       a) Título da *survey*;
       b) Estrutura de seções;
       c) Lista dos *papers* a serem utilizados em cada seção.

Abordagem 1

Geração automática

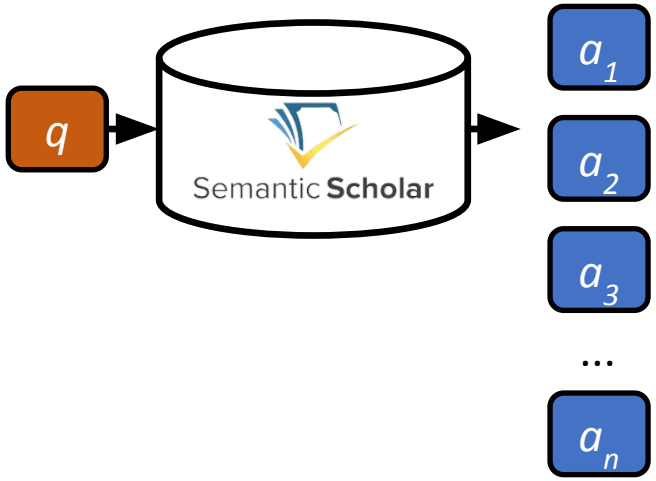
Hiperparâmetros

1. **Busca no semantic scholar**

```
SURVEY_FILTERS = {
  "fields_of_study": 'Computer Science',
  "year": '2020-2023',
  "only_open_access": True,
  "max_number_of_papers": 9999,
}
```

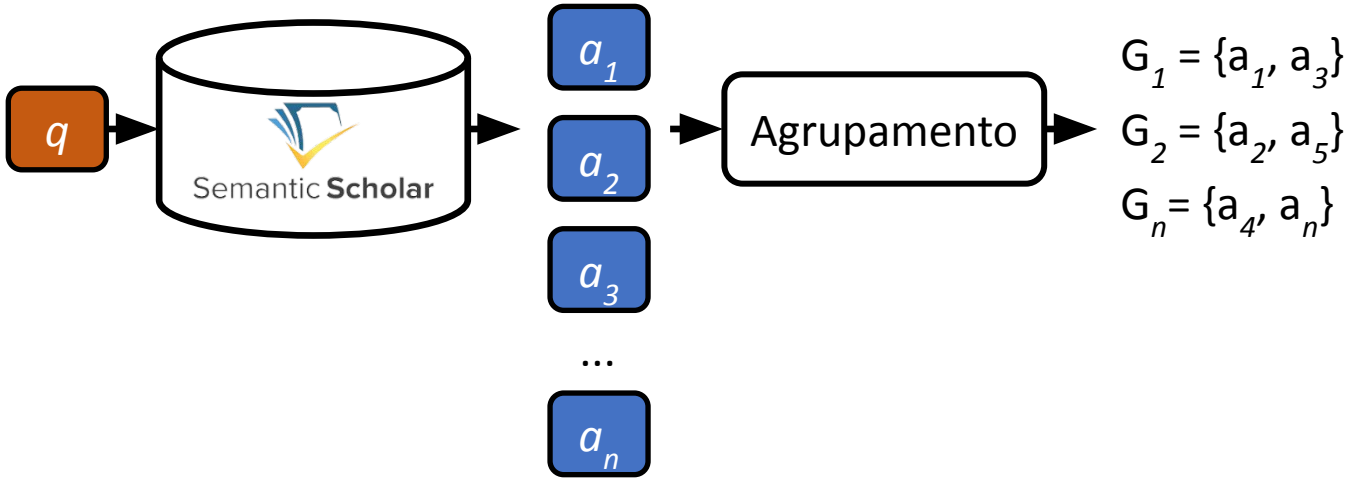
2. **Estrutura da *survey***

```
SURVEY_STRUCTURE = {
  "n_sections": [5, 2], # 5 sections with 3 subsections
  "n_papers_to_suggest_title_section": 10,
  "max_papers_per_section_as_ref": 20,
  "n_papers_to_cluster_in_subsections": 50,
}
```



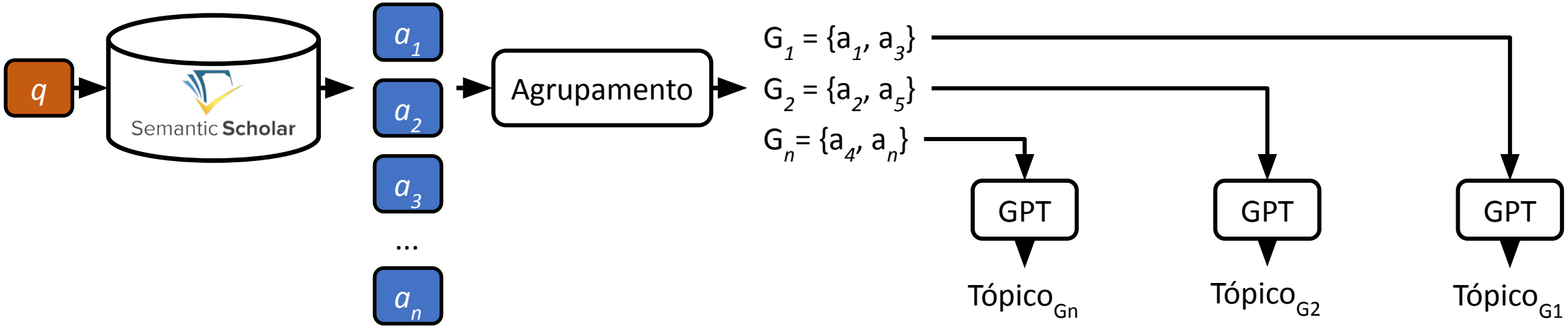
1. Busca Inicial

- Query:
  - survey_topic = 'large language models for textual information retrieval'
- Semantic Scholar API
  - Field: Computer Science
  - Year: 2020-2023
  - openAccessPdf: True



2. Agrupamento

- SpecterV2:
  - Título [SEP] Abstract
  - Adapter: specter2_proximity



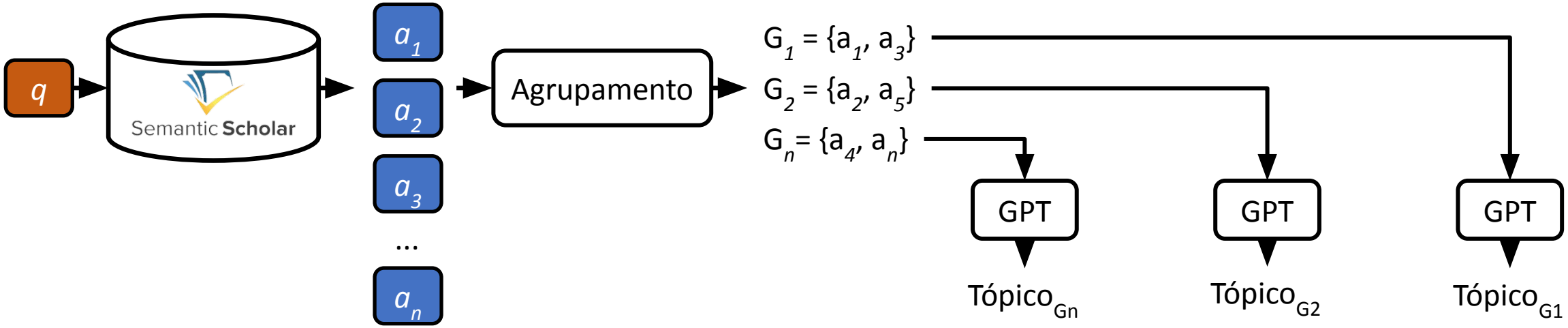
3. Tópicos/Seções

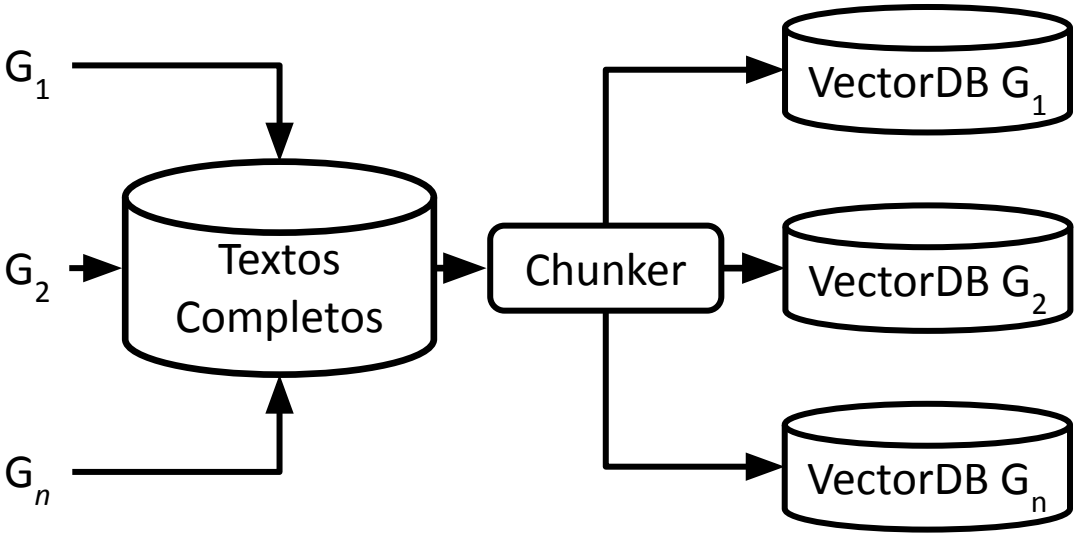
```
You are a renowned scientist who is writing a survey on '{survey_topic}'.
You are currently writing a section about '{query_section}'
[…]
I will send you a list of title and abstract of scientific articles. Most
of them cover a specific topic about the section '{query_section}'
Your task is to find out what this topic is and suggest a good title for a
section in a scientific survey that addresses it.
You should also explain your reasoning.
```

survey_topic = ‘neural information retrieval'
query_section = chamadas recursivas de acordo com o total de seções/subseções definidas

Seções sugeridas

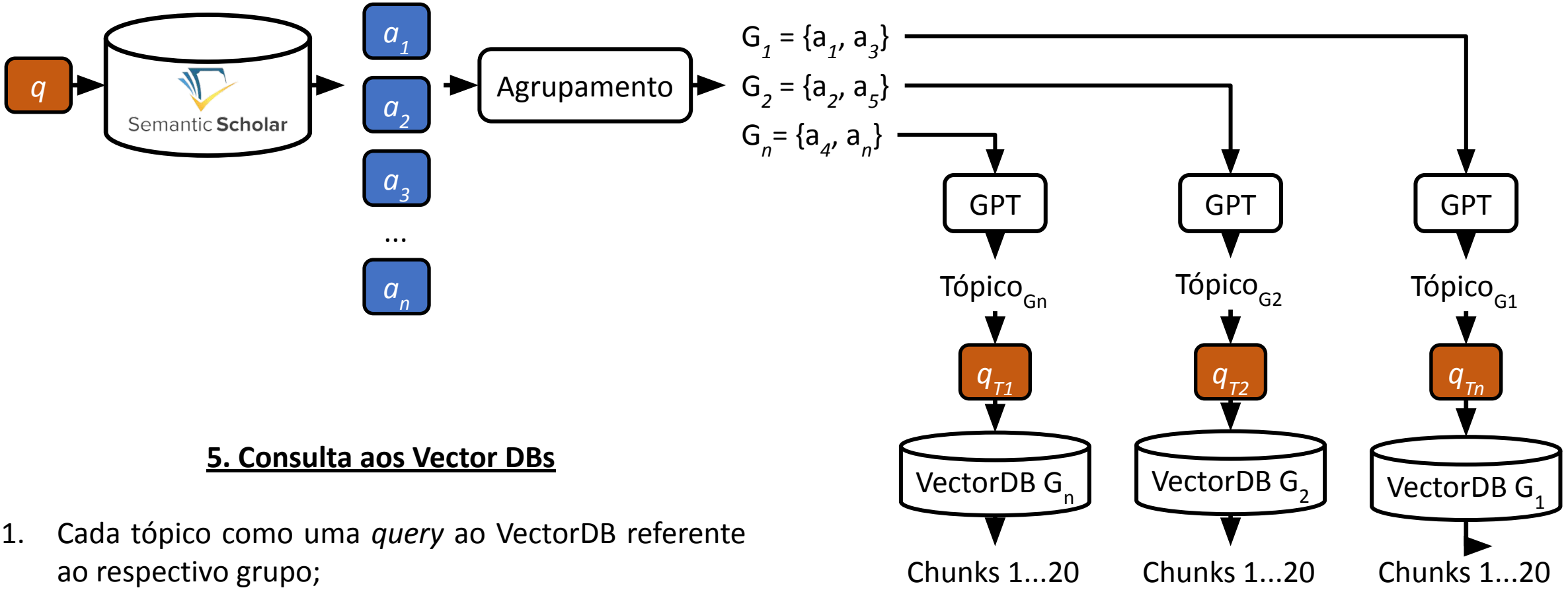
1 Enhancing Text Information Retrieval with Neural Models
    1.1 Incorporating Neural Models for Text Information Retrieval
    1.2 Enhancing Text Information Retrieval with Neural Models: A Comprehensive Survey
    1.3 Applying Neural Models for Text Information Retrieval
2 Deep Learning Approaches for Text Classification in Information Retrieval
    2.1 Leveraging Cross-Document Interactions for Learning-to-Rank in Deep Learning Framework
    2.2 Specialized Interfaces for Domain-specific Information Retrieval in Deep Learning Approaches
    2.3 Deep Learning Approaches for Text Classification in Information Retrieval
3 Advancements in Neural Network-based Text Analysis and Retrieval
    3.1 Advancements in Multimodal Feature Fusion for User Preference Prediction in Social Media
    3.2 Advancements in Neural Network-based Text Mining and Information Retrieval
    3.3 Exploring Multimodal Clues for Text Analysis and Retrieval
4 Deep Learning Approaches for Text Information Retrieval
    4.1 Efficient Content-Based Image Retrieval using Deep Learning
    4.2 Enriching Text with Semantic Information from Ontologies for Deep Learning Approaches
    4.3 Deep Learning Models for Text Classification and Information Extraction
5 Advancements in Neural Network-based Text Summarization and Information Extraction
    5.1 Advancements in Text Similarity Calculation and Content Extraction using Neural Networks
    5.2 Enhancing Local Feature Extraction with Global Representation for Neural Text Classification
    5.3 Advancements in Neural Network-based Text Summarization and Information Extraction: A Comprehensive Review





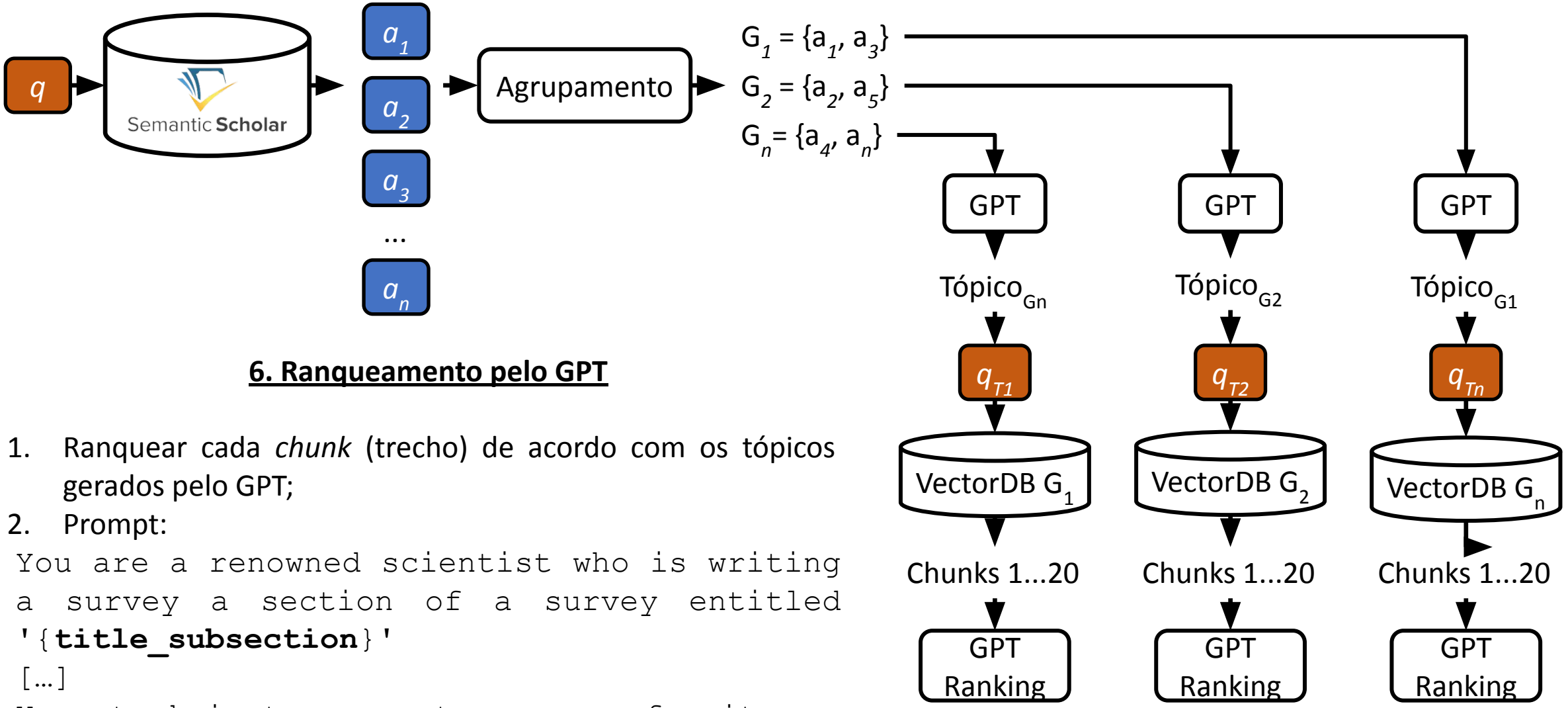
4. Indexação (textos completos)

1. SpecterV2
2. Grobid para geração de dict do *paper*.
   - Recuperar apenas texto do corpo do *paper*.
   - Caso grobid falhe, usa o PDFium (texto entre *introduction* e *reference*).
3. Chunks (trechos) do texto completo.
   - *SpacyTextSplitter* modificado para sentenças
   - 7 sentenças com *overlap* (*stride*) 2



5. Consulta aos Vector DBs

1. Cada tópico como uma *query* ao VectorDB referente ao respectivo grupo;
2. LangChain
   - *similarity_search* (k=20)

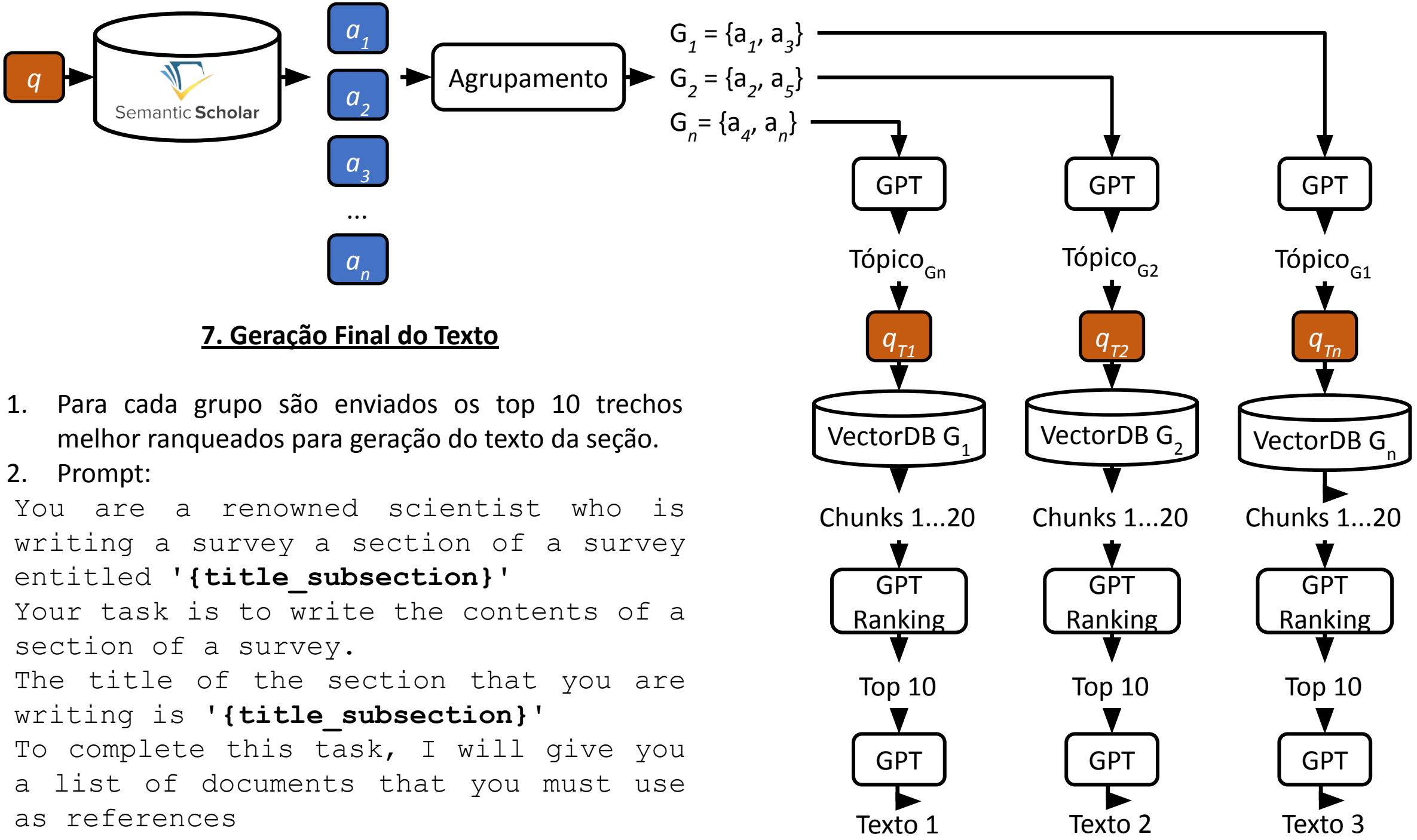


6. Ranqueamento pelo GPT

1. Ranquear cada *chunk* (trecho) de acordo com os tópicos gerados pelo GPT;
2. Prompt:

```
You are a renowned scientist who is writing
a survey a section of a survey entitled
'{title_subsection}'
[…]
Your task is to generate a score for it
ranging from 0 to 5 indicating its
importance to the section you are writing.
```

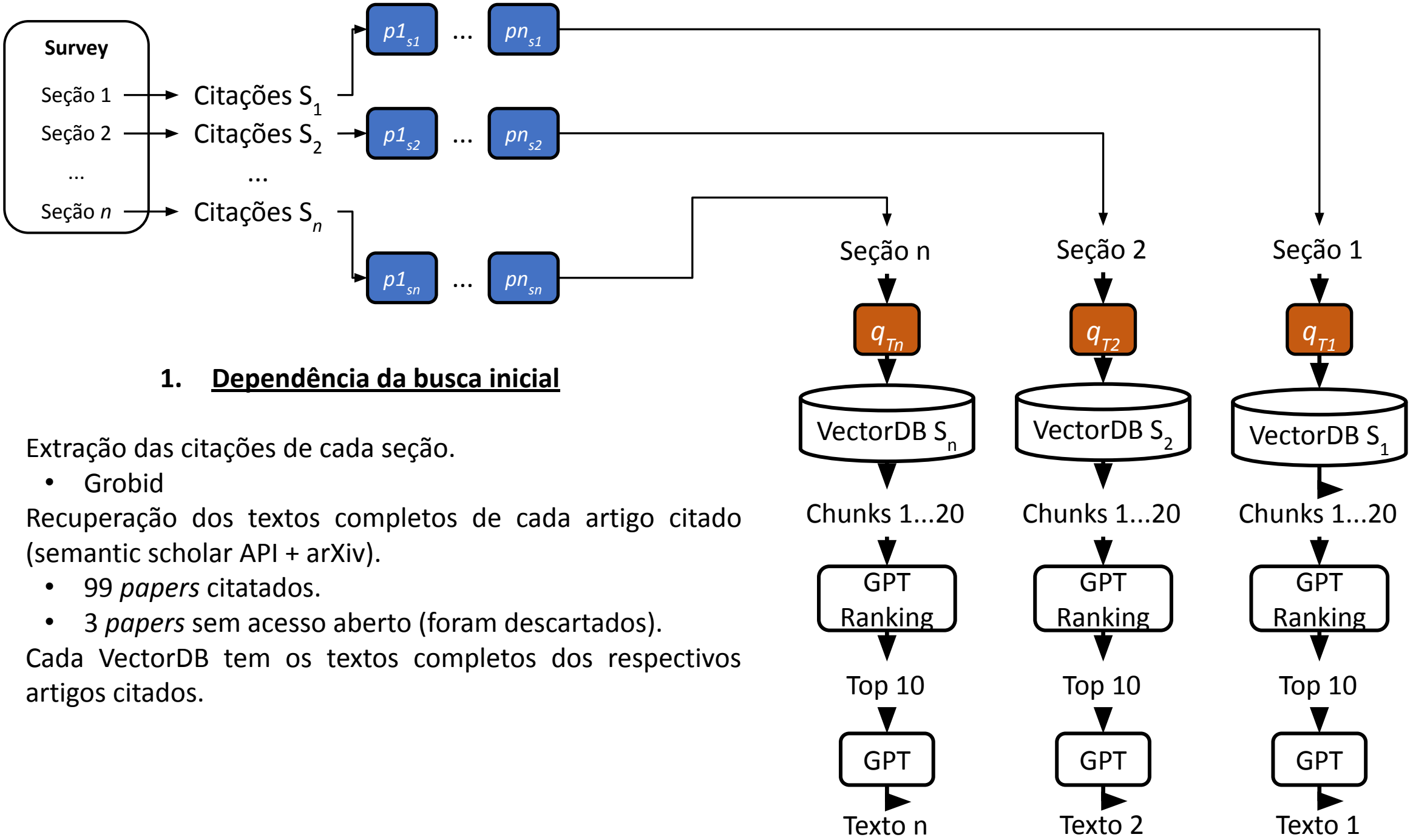
title_subsection = {$Tópico_{G1}$ | $Tópico_{G2}$ | $Tópico_{GN}$}



7. Geração Final do Texto

1. Para cada grupo são enviados os top 10 trechos melhor ranqueados para geração do texto da seção.
2. Prompt:

```
You are a renowned scientist who is writing a survey a section of a survey entitled '{title_subsection}'
Your task is to write the contents of a section of a survey.
The title of the section that you are writing is '{title_subsection}'
To complete this task, I will give you a list of documents that you must use as references
```



1. **<u>Dependência da busca inicial</u>**

2. Extração das citações de cada seção.
   - Grobid
3. Recuperação dos textos completos de cada artigo citado (semantic scholar API + arXiv).
   - 99 *papers* citatados.
   - 3 *papers* sem acesso aberto (foram descartados).
4. Cada VectorDB tem os textos completos dos respectivos artigos citados.

Abordagem 2

Geração semiautomática

Avaliação

- Modelo utilizado nas gerações
  - gpt-3.5-turbo-0613
  - zero-shot (*instructions only*)
- Na geração semiautomática, com o *survey* de referência, é possível comparar seção a seção.
- Na geração automática (*pipeline* completo) somente é possível comparar o *survey* como um todo.
  - Dificuldade em induzir o modelo a gerar a mesma estrutura do *survey* de referência.

Avaliação
Geração semiautomática

- Uso do BERTScore
  - Similaridade entre os *tokens* contextualizados de um candidato e uma referência.
- Modelo: roberta-large
  - Truncamento em 512 *tokens*
- Resultados
  - Precision = 0.782084
  - Recall = 0.814449
  - F1 = 0.797784

survey	mean	max	min	median
Reference	670.45	768	595	675.0
Generated	698.75	1480	328	600.5

Exemplos gerados
Text Representations for Ranking - BOW Encodings

The BOW encoding scheme assumes that the order of terms in a document does not convey important information for retrieval purposes. Instead, it focuses on the frequency of occurrence of terms within a document.

Exemplos gerados
Text Representations for Ranking - Word Embeddings

Word embeddings are dense vector representations that capture semantic and syntactic relationships between words. These representations have shown promising results in various natural language processing tasks, including information retrieval and ranking. One common method for generating word embeddings is through the use of **word2vec** models [REF5]. Word2vec models, such as skip-gram and continuous bag-of-words (CBOW), learn word embeddings by predicting the context words given a target word or vice versa.

Exemplos gerados
Learned Sparse Retrieval - Sparse representation learning

One approach to learned sparse retrieval is the use of sparse representation models such as BOW (Bag-of-Words) and **SPLADE (Sparse Lexical Adaptive Document Embeddings)** [REF0]. These models aim to represent documents and queries in a sparse manner, where only a subset of the features or terms are considered relevant for retrieval.

This sparsity enables efficient computation and storage, making it suitable for large-scale retrieval tasks.

Sparse models, such as SPLADE, provide interpretable representations where the dimensions directly correspond to the terms in the lexicon [REF1].

To improve the efficiency of sparse representation learning, regularization techniques are often employed. Regularization helps in reducing the computational cost by promoting sparsity in the learned representations. For example, FLOPS regularization has been shown to be advantageous over $\ell 1$ regularization in terms of decreasing computing cost [REF4].

Exemplos gerados
Learned Sparse Retrieval - Sparse representation learning

- Exemplo de Chunk

  [REF5] In addition, we observe the advantage of the FLOPS regularization over ℓ 1 in order to decrease the computing cost […]

Trabalhos Futuros

1. Dificuldade em medir a contribuição:
   - In-contexto Learning vs Dados vistos pelo LLM
   - Na semiautomática apenas um *paper* de 2022, todos os demais de 2021 para trás.
2. Testar modelos open-source.
   a) OpenLLaMa 7B
   b) MPT
   c) Vicuna
4. Few-shots
   - Melhorar a correspondência no texto fluido gerado dos *chunks* (referências) usados.
5. Avaliação com revisores (humanos)