

Discovery Approach Description

The description provided below will be used to evaluate the approach developed by your team to automatically identify public web sources of annual financial data of MNE Groups. This description will be evaluated by the Evaluation panel based on the criteria described in the Evaluation tab of the Discovery Challenge and used for the ranking of your team for the Reusability and Innovativeness Awards.

Methodology (*Data-driven approaches; Availability and quality of documentation*)

Please provide a detailed description of the methodology used for automatically identifying the public web sources of annual financial data of MNE Groups. The description should contain (1) the data processing steps, (2) the methods and models used, (3) references to the scientific papers/sources that present the methods and models used, and (4) the time it took to process the data set.

Bear in mind that the workflow will be also evaluated based on the criteria for the Reusability and Innovativeness Awards.

This section will be evaluated for:

- (1) *Data-driven approaches: The described approach is evaluated based on whether it is data-driven rather than heuristic. More data-driven approaches will receive higher scores. The code will be inspected visually by the evaluation panel. Well documented code which allows evaluators to determine the steps of the approach will yield higher scores.*
- (2) *Availability and quality of documentation: Based on the clarity of the provided documentation describing the approach, the evaluation panel is asked to assess the likeliness that the described approach can successfully reproduce the solution submitted by the team for the Accuracy award.*

Our methodology for automatically identifying public web sources of annual MNE Groups' financial data is a data-driven, multi-stage, robust pipeline which can be scaled up, and reused. It combines web search techniques with natural language processing (NLP) and machine learning principles to locate and prioritize relevant financial documents. The workflow is entirely automated, minimizing manual intervention and enabling efficient processing of large datasets.

(1) Data Processing Steps

The data processing workflow is organized into the following sequential steps, executed programmatically:

Initially, the process takes a structured list of MNE Group names as input, from which an initial set of targeted search queries is programmatically generated for each MNE. These queries are designed to cover various potential avenues for locating financial data, ranging from general investor relations to specific annual report filenames, ensuring a broad initial capture of relevant URLs.

Next, these generated queries are executed against the Google Search API to identify a comprehensive set of initial candidate URLs for each MNE. For URLs identified as potential MNE homepages or investor relations portals, a Python-based web-scraping routine is employed to programmatically navigate the website. This routine identifies links to the website's sub-sections commonly associated with financial reports (e.g., "Investor Relations," "Annual Reports") and specifically filters for PDF documents (.pdf file extension), which are the most common format for financial statements.

Subsequently, all unique URLs gathered for a given MNE are consolidated into a single master list, with duplicate URLs programmatically removed to ensure efficiency. This comprehensive list of potential financial document URLs then undergoes a critical semantic ranking and selection process using a Large Language Model (LLM), such as Gemini 2.0. The LLM evaluates each URL based on its likelihood of being the latest official Annual Financial Statements Report or one of five additional sources providing comprehensive financial data (Net Turnover, Total Assets, Number of Employees, Headquarters Country, NACE Activity). This leverages the LLM's sophisticated understanding of context to assess and prioritize the most likely financial documents.

Finally, the top six ranked URLs for each MNE—one for the latest Annual Financial Statements Report and five others for supplementary financial data—are selected and presented as the solution's output.

(2) Methods and Models Used

Our methodology integrates several well-established computational methods and models:

- Web Scraping (Python): This is a fundamental technique for programmatically extracting data from websites. Libraries like 'Requests' (handles HTTP communication) and 'BeautifulSoup' (provides a powerful and flexible way to parse HTML and XML documents) enable the extraction of specific links and textual content. For highly dynamic websites or those requiring interaction, selenium is additionally employed to handle JavaScript rendering and simulate browser actions.
- Google Search API: Leveraged for broad, initial discovery of relevant web pages and documents based on carefully constructed queries. This API provides programmatic access to Google's search capabilities, crucial for large-scale web data acquisition.
- Large Language Models (LLMs) for Semantic Ranking (e.g., Gemini 2.0): This is the innovative and data-driven core of our ranking mechanism. LLMs are advanced neural networks trained on vast amounts of text data, enabling them to understand, generate, and process human language with remarkable accuracy. In our approach, the LLM is used for:
 - Semantic Understanding: Interpreting the context of URLs, their titles, and surrounding text from search results to infer document content.
 - Relevance Scoring: Assigning a "relevance score" or rank to each URL based on predefined criteria (latest annual financial statement, comprehensive other financial data).
- Heuristics for Initial Filtering: While the LLM provides the primary data-driven ranking, initial heuristics (e.g., .pdf file extension filtering, specific URL patterns like "investor-relations") are

used in earlier stages to narrow down the search space efficiently before engaging the more computationally intensive LLM.

(3) References to Scientific Papers/Sources

The methods and models employed are grounded in established computer science and artificial intelligence research:

- Web Scraping: While fundamental, the principles are widely discussed in web development and data engineering literature. Good general references include:
 - [Beautiful Soup Documentation](#)
- Large Language Models (LLMs):
 - [Generative Pre-trained Transformers \(GPTs\)](#)

(4) Time Taken to Process the Data Set

For our demonstration dataset, which included **200** MNE Groups, the end-to-end processing time was approximately **10 hours**. This time includes all steps from initial query generation to final URL selection. This performance was achieved using a **standard desktop PC** with 16GB of RAM and was limited primarily by the rate limits of the Google Search API and the latency of the LLM API calls. As discussed in the Architecture section, this approach can be made to work in parallel and thus potentially made a lot faster, especially using the higher tier api limits of the paid versions.

Architecture (*Algorithm reusability and scalability*)

Please provide a description of the architecture of your approach. A diagram of the architecture is considered of additional value. Indicate what modifications would be required to apply the approach to similar datasets on a larger scale.

This section will be evaluated for:

(1) *The described approach is evaluated based on:*

- a. *Are the components well separated and encapsulated to allow for independent modification?*
- b. *What is the degree of modification required to add new companies?*
- c. *Is it possible to implement the approach by running it on parallel machines?*
- d. *What is the extent of effort required to track performance?*
- e. *Is the method robust if scaled to larger number of cases?*
- f. *Is the performance of the approach stable as scale increases?*

Our solution employs a modular, web-scraping-based architecture designed to systematically identify and retrieve the latest financial statements and supplementary financial data for Multinational Enterprises (MNEs). The core objective is to distill a comprehensive set of relevant URLs from a broad initial search.

The architecture is composed of distinct modules, as detailed below and illustrated in the diagram below.

- **Input Module:** The original “discovery.csv” as prepared by the challenge organisers. This file contains the names of the MNE Groups for which we are tasked with finding financial data URLs.
- **URL Discovery Engine (Parallelizable):** This is the core data gathering component. It orchestrates eight distinct search strategies, each leveraging the Google Search API to identify potential URLs for financial information.
 - (1) Homepage Navigation: Initiates a Google search for the official MNE homepage, then performs targeted web scraping (using Python with Beautiful Soup) to navigate to sections like "Investor Relations", "Annual Reports" and "Downloads" and identify and record PDF document URLs. Manual research has shown that these PDF documents will most likely contain the latest annual financial statements (FIN_REP), as well as documents like ESG reports (and others), which can be used as potential OTHER sources of information.
 - (2) Direct PDF Search (Current Year): Executes a Google search for the latest Financial Statements directly, limiting search results to PDF filetypes. The latest statements might be those of 2024 or 2023, so we first carry out the search for 2024.
 - (3) Direct PDF Search (Previous Year): Executes the same Google search as in (2) above but now for the previous year (2023).
 - (4) Generic Investor Relations Search: Uses a broader query for investor relations or financial reports.
 - (5) Key Figures Search: Again, uses a broader query for financial highlights or key figures.

- (6) Other Reports Search: Targets non-FIN_REP reports that might contain additional financial insights (e.g., ESG, sustainability reports).
- (7) Annual Reports/Financial Results: Look for the phrases 'annual reports' or 'financial results' in 2023 or 2024
- (8) Domain-Specific Keyword Search: Constructs a query targeting site:[company name].com with keywords like "investor" or "downloads."
- **URL Consolidation Module:** Collects all identified URLs from the eight search strategies above for each input MNE Group.
- **URL Ranking and Selection Module (LLM-based):** Utilizes a Large Language Model (LLM), specifically Gemini 2.0 (at the time of results submission), to rank the gathered URLs based on their relevance and likelihood of containing the most recent Annual Financial Statements Report, plus five additional sources of financial data (containing MNE Group Headquarters Country, Number of Employees, Net Turnover, Total Assets). This module outputs a prioritized list of at most six URLs per MNE Group.
- **Output Module:** Presents the final list of top-ranked URLs for each MNE.

(a) Are the components well separated and encapsulated to allow for independent modification?

Yes, the architecture is designed with strong modularity. The "URL Discovery Engine", "URL Consolidation Module" and "URL Ranking and Selection Module" are distinct layers. For instance, new search strategies can be added to the URL Discovery Engine, or a different LLM or ranking algorithm can be swapped into the URL Ranking and Selection Module without affecting the core scraping or output functionalities. This modularity significantly enhances **reusability** as individual components can be swapped or updated without redesigning the entire system.

(b) What is the degree of modification required to add new companies?

Minimal. The system is designed to be highly scalable in terms of adding new companies. Adding new MNEs simply requires appending their names to the initial input list used in the "Input Module." No code modifications are needed for the core processing logic when new companies are added. The entire pipeline runs iteratively for each MNE Group provided.

(c) Is it possible to implement the approach by running it on parallel machines?

Yes, the approach can be parallelized. The URL Discovery Engine, in particular, can be amended to process multiple MNEs concurrently across different machines or processing units. Each MNE's URL discovery and subsequent ranking process is independent of others. This allows for significant speedup when processing a large number of MNEs. A single machine was used by the DOUBLE_A team for the discovery challenge, but given more time and resources parallelisation is certainly possible.

(d) What is the extent of effort required to track performance?

Performance tracking is managed by monitoring and logging key code metrics. If deemed necessary the "logger.py" module can be extended to track more metrics, as per the user's requirements.

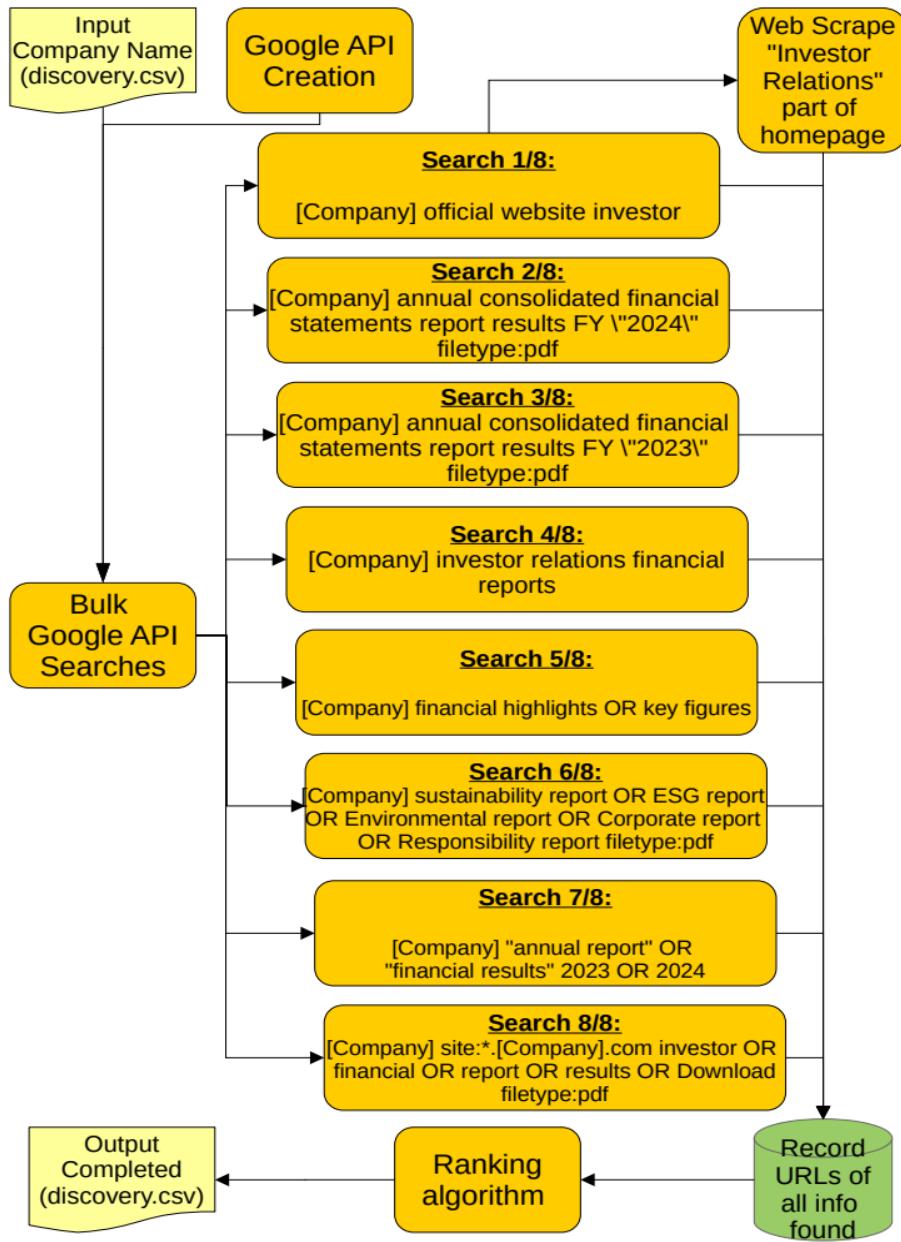
(e) Is the method robust if scaled to larger number of cases?

Yes, the method is robust for scaling. The algorithm might take longer to run but if it is parallelised then it can be made much faster.

(f) Is the performance of the approach stable as scale increases?

The stability of performance primarily depends on the rate limits and responsiveness of the Google

Search API and the target MNE websites. If for example Google imposes stricter rate limits or if numerous MNE websites become unresponsive or change their structure frequently, this could impact overall throughput.



Hardware Specifications (*Algorithm reusability and scalability*)

Please describe the hardware specifications of the machines that were used to run the methodology.

This section will be evaluated for:

(1) Algorithm reusability and scalability

- a. Is it possible to implement the approach by running it on parallel machines?
- b. Is the method robust if scaled to larger number of cases?
- c. Is the performance of the approach stable as scale increases?

Machine 1

CPUs	Intel i3-12100f
GPUs	Not Used (except when using a cloud LLM in ranking URLs)
TPUs	Not Used
Disk space	Minimal (Less than 1GB)

Libraries

Please provide the libraries used for approach, if any, as well as the links to these libraries, if available.

Core web scraping and data processing:

- requests==2.32.3 (<https://pypi.org/project/requests/>)
- beautifulsoup4==4.13.4 (<https://pypi.org/project/beautifulsoup4/>)
- selenium==4.32.0 (<https://pypi.org/project/selenium/>)
- pandas==2.2.3 (<https://pypi.org/project/pandas/>)
- numpy==2.2.6 (<https://pypi.org/project/numpy/>)

Google API and LLM integration:

- google-api-python-client==2.169.0 (<https://pypi.org/project/google-api-python-client/>)
- google-genai==1.16.1 (<https://pypi.org/project/google-genai/>)

HTTP clients and utilities:

- httpx==0.28.1 (<https://pypi.org/project/httpx/>)
- curl_cffi==0.11.1 (<https://pypi.org/project/curl-cffi/>)

PDF processing:

- PyPDF2==3.0.1 (<https://pypi.org/project/PyPDF2/>)

Configuration and Logging:

- python-dotenv==1.1.0 (<https://pypi.org/project/python-dotenv/>)
- colorama==0.4.6 (<https://pypi.org/project/colorama/>)

Web driver management:

- webdriver-manager==4.0.2 (<https://pypi.org/project/webdriver-manager/>)

Similarities/differences to State-of-the-Art techniques (*Originality of the approach*)

Please provide a list of similarities and differences between the used methodology and to the state-of-the-art techniques.

This section will be evaluated for:

(1) the Originality of the approach criterion: compare the approach used to the state of the art, i.e. currently published approaches that are closest to the approach applied for the submission, and the extent to which the submission represents an improvement over these approaches. The submission will be evaluated based on the degree to which it is new or unique. This could be in terms of technology, methodology, or application.

Similarities:

- Multi-strategy web scraping approach common in academic literature
- Use of search APIs for initial URL discovery
- PDF filtering heuristics widely used in document retrieval systems
- Modular pipeline architecture follows established design patterns

Differences:

- Novel application of LLMs for semantic URL ranking in financial document discovery
- Hybrid approach combining traditional web scraping with modern AI for relevance assessment
- Eight-strategy search methodology specifically tailored for MNE financial documents
- Integration of Google Search API with LLM-based filtering represents advancement over purely heuristic approaches
- Domain-specific optimization for financial statement identification vs. general document retrieval

Contribution to scientific field (*Future orientation*)

Please describe how your submission contributed to the scientific field, what impact it could have and what could potentially be future work to improve the solution.

This section will be evaluated for:

(1) the Future orientation and impact criterion: The potential effect of the approach used will be evaluated. This includes the scale of impact it has on the problem of discovering sources of financial data from the Internet. The impact will be evaluated based on potential efficiency improvements and cost reductions.

On account of how difficult it is to navigate each individual MNE's website and download its financial statements, perhaps this work will help in making such reports more easily available. The algorithm can be generalised and made to identify other or all useful PDF documents found for each MNE Group, besides the Financial Statements (for example the Environmental Sustainability and/or the Tax Transparency reports). This demonstrates the **extensibility and adaptability** of the core methodology for broader document discovery tasks beyond financial statements, such as ESG reports or tax transparency data.

Lessons Learned (*Future orientation*)

Please state any lessons learned during the competition.

This section will be evaluated for:

(1) the Future orientation and impact criterion: what were the lessons learnt during the competition, and what could potentially be future work to improve the solution.

Key lessons from the competition:

1. **Website Variability:** MNE websites have highly inconsistent structures, validating our multi-strategy approach over single-method solutions.
2. **LLM Effectiveness:** Semantic ranking via LLMs significantly outperformed keyword-based heuristics for financial document relevance assessment.
3. **Scalability Considerations:** Single-threaded processing adequate for competition scale, but parallelization essential for production deployment.
4. **PDF Prevalence:** Targeting PDF documents proved highly effective as financial statements are predominantly published in this format.

Future Improvements:

- Implement request pooling and caching for API efficiency
- Add fallback strategies for websites blocking automated access
- Enhance error handling for unreachable URLs
- Develop confidence scoring for ranked results

Short description of the Team – area of expertise

Please provide a description of the team, your area of expertise and contact information.

The DOUBLE_A team consists of Andreas and Andreas, two data enthusiasts who try their best to find patterns in data and look for solutions that would benefit people's lives.

- Andreas Soteriou – asoteriou@cystat.mof.gov.cy
Data Scientist, Python programmer and AI & ML (Artificial Intelligence & Machine Learning) expert.
- Andreas Hadjittofis – ahadjitofis@cystat.mof.gov.cy
Data Analyst, R programmer and process automation specialist.