**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.*

|  |
| --- |
| **Data Processing Steps:**   1. **Company Name Normalization**: Input company names are normalized by removing common corporate suffixes (PLC, AG, NV, AB, etc.) and special characters to improve search accuracy. 2. **Multi-Query Search Strategy**: For each company, we generate multiple search queries with varying specificity:    * Basic annual report searches    * European-focused queries (adding "Europe", "EU" terms)    * Company-specific queries for known entities    * Fallback queries with alternative terminology 3. **Intelligent Web Search**: Using Google Custom Search API for reliable and scalable search capabilities. The system implements:    * Geographic filtering to prioritize European sources    * Keyword relevance scoring for investor relations pages    * Duplicate detection and URL deduplication    * API key management for sustained search operations 4. **PDF Link Extraction**: From identified investor relations pages, the system extracts potential PDF links using:    * Direct PDF link detection (URLs ending in .pdf)    * Download button/link analysis with text pattern matching    * Company-specific URL pattern matching for known entities 5. **Reference Year Extraction**: Automated extraction of reference years from URLs and link text using regex patterns to identify 4-digit years (20XX format). 6. **Link Verification**: All identified URLs are verified for accessibility using HTTP HEAD/GET requests with extended retry logic to handle rate limiting.   **Methods and Models Used:**   * **Web Scraping**: BeautifulSoup4 for HTML parsing and link extraction * **Search API Integration**: Google Custom Search API for reliable web search * **Pattern Recognition**: Regular expressions for year extraction and URL pattern matching * **Scoring Algorithm**: Custom relevance scoring based on keyword presence, URL patterns, and recency * **Rate Limit Handling**: API quota management with intelligent request spacing   **Processing Time:**  The complete dataset of 200 companies was processed in approximately 30-60 minutes, including all retry attempts and delays or blocks on the scraped company webpages. |

|  |
| --- |
| **References**   * Google Custom Search API Documentation: <https://developers.google.com/custom-search/v1/overview> * BeautifulSoup Documentation: <https://www.crummy.com/software/BeautifulSoup/> * Requests Library: <https://docs.python-requests.org/> * Web scraping best practices for API integration |

**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:*
   1. *Are the components well separated and encapsulated to allow for independent modification?*
   2. *What is the degree of modification required to add new companies?*
   3. *Is it possible to implement the approach by running it on parallel machines?*
   4. *What is the extent of effort required to track performance?*
   5. *Is the method robust if scaled to larger number of cases?*
   6. *Is the performance of the approach stable as scale increases?*

|  |
| --- |
| **System Architecture:**  Input CSV (Company Names)  ↓  Company Name Normalizer  ↓  Multi-Query Generator  ↓  Google Custom Search API Interface  ↓  Results Filter (Geographic & Relevance)  ↓  PDF Link Extractor  ↓  URL Verification Engine  ↓  Reference Year Extractor  ↓  Results Formatter  ↓  Output CSV (Discovery Format) |

|  |
| --- |
| **Component Separation and Encapsulation:**  The architecture is designed with well-separated, modular components:   * **FinancialReportFinder Class**: Main orchestrator with clear method separation * **Google Search Interface**: Isolated API interaction logic with quota management * **PDF Extraction Module**: Standalone link extraction and verification * **Pattern Matching Engine**: Company-specific URL patterns in dictionary format * **Request Handler**: Separate retry logic and rate limiting management   **Scalability Considerations:**  **Adding New Companies**:   * Zero modification required for standard companies * New company-specific patterns can be added to the company\_patterns dictionary * No core logic changes needed   **Parallel Processing**:   * Each company processing is independent * Can be easily parallelized using Python's multiprocessing * Rate limiting logic supports distributed processing with shared state   **Performance Tracking**:   * Built-in logging for each processing step * Intermediate CSV saves for progress tracking * Success/failure metrics automatically captured   **Large-Scale Robustness**:   * Batch processing capability with configurable batch sizes * Automatic pause mechanisms between batches * Memory-efficient streaming for large datasets * Graceful error handling and recovery   **Performance Stability**:   * Linear scaling with number of companies * API quota management prevents performance degradation * No memory leaks or accumulation issues |

**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*
   1. *Is it possible to implement the approach by running it on parallel machines?*
   2. *Is the method robust if scaled to larger number of cases?*
   3. *Is the performance of the approach stable as scale increases?*

**Machine 1**

|  |  |
| --- | --- |
| CPUs | Apple M2 Pro, 16GB shared memory |
| GPUs | Apple M2 Pro, 16GB shared memory |
| TPUs | TPU name and capacity |
| Disk space | 5 MB |

**Machine 2**

|  |  |
| --- | --- |
| CPUs | CPU name and capacity |
| GPUs | GPU name and capacity |
| TPUs | TPU name and capacity |
| Disk space | The space required to calculate and store the data |

**Libraries**

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

|  |
| --- |
| **Core Libraries Used:**   1. **requests (v2.31.0+)**: HTTP library for web requests and API calls    * Link: <https://pypi.org/project/requests/>    * Used for: Google API calls, URL verification, PDF accessibility checks 2. **google-api-python-client**: Official Google API client library    * Link: <https://pypi.org/project/google-api-python-client/>    * Used for: Google Custom Search API integration 3. **beautifulsoup4 (v4.12.0+)**: HTML parsing and web scraping    * Link: <https://pypi.org/project/beautifulsoup4/>    * Used for: Parsing search results, extracting PDF links from web pages 4. **pandas (v1.5.0+)**: Data manipulation and analysis    * Link: <https://pypi.org/project/pandas/>    * Used for: CSV file handling, data structure management 5. **urllib.parse**: Built-in Python module for URL manipulation    * Used for: URL encoding, parsing, and joining relative/absolute URLs 6. **re**: Built-in Python regular expressions module    * Used for: Pattern matching, year extraction, company name normalization 7. **csv**: Built-in Python CSV handling module    * Used for: Reading input files and writing output discovery.csv 8. **time/random**: Built-in Python modules    * Used for: API request spacing, random delays to avoid quota exhaustion |

**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 to State-of-the-Art:**   1. **Web Scraping Fundamentals**: Similar to standard web scraping approaches using requests + BeautifulSoup combination 2. **Search API Integration**: Common pattern of using commercial search APIs for data discovery 3. **Pattern Recognition**: Standard use of regex for data extraction (years, URLs) 4. **API Quota Management**: Industry-standard approaches to managing API usage limits   **Key Differences and Innovations:**   1. **API-First Approach**: Strategic use of Google Custom Search API for reliability, moving away from unstable free search engines 2. **Multi-Tier Search Strategy**: Unique combination of general → European-focused → company-specific search progression 3. **Company-Specific Pattern Libraries**: Pre-built URL pattern dictionaries for major European companies, reducing search dependency 4. **Geographic Filtering**: Explicit filtering for European domains to improve relevance for EU-based MNEs 5. **Hybrid Verification Approach**: Combination of HEAD and GET requests with intelligent fallback mechanisms 6. **Adaptive Query Generation**: Dynamic query modification based on company characteristics and previous attempt failures   **Improvements Over Existing Approaches:**   * **Reliability**: Superior stability through commercial API usage vs. free search engines * **Accuracy**: Multi-tier approach significantly improves success rates * **Efficiency**: Company-specific patterns reduce unnecessary API calls * **Scalability**: Modular design allows easy extension and parallel processing |

**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.*

|  |
| --- |
| **Scientific Contributions:**   1. **Automated Financial Data Discovery**: Demonstrates feasibility of fully automated annual report discovery for regulatory compliance and research purposes 2. **Rate Limiting Strategy Innovation**: The extended retry mechanism with hour-long waits provides a template for handling aggressive rate limiting in academic/research contexts 3. **Multi-Modal Search Optimization**: The progressive search refinement approach (general → specific → pattern-based) offers a replicable framework for web data discovery 4. **European Corporate Data Access**: Specifically addresses challenges in accessing European MNE financial data, contributing to regulatory technology research   **Potential Impact:**   * **Regulatory Compliance**: Automated data collection for tax transparency and regulatory reporting * **Academic Research**: Enabling large-scale financial analysis studies across European corporations * **Cost Reduction**: Eliminating manual data collection efforts, reducing research costs by estimated 80-90% * **Standardization**: Providing a replicable methodology for financial data discovery across jurisdictions   **Future Work Applications:**   * Extension to other regulatory requirements (ESG reporting, sustainability reports) * Integration with automated data extraction from discovered PDFs * Multi-language support for non-English European companies * Real-time monitoring systems for newly published financial reports * Integration with machine learning for improved pattern recognition |

**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.*

|  |
| --- |
| **Technical Lessons:**   1. **Commercial APIs Provide Reliability**: Free search engines like DuckDuckGo can suddenly restrict access or shut down free tiers. Investing in commercial APIs (Google Custom Search) provides more reliable and sustainable automation. 2. **API Quota Management is Essential**: Even commercial APIs have usage limits. Intelligent quota management and request spacing prevent service interruptions and ensure consistent performance. 3. **Company-Specific Patterns Matter**: Pre-built URL patterns for major companies significantly improve success rates and reduce API calls, suggesting the value of hybrid approaches combining web search with known patterns. 4. **Geographic Filtering is Essential**: For European MNE data, explicit filtering for European domains improves accuracy and reduces false positives from US/other jurisdiction entities. 5. **Multiple Search Strategies Required**: No single search approach works for all companies; progressive refinement from general to specific queries improves overall success rates.   **Methodological Lessons:**   1. **Modular Design Enables Scalability**: Clear separation of concerns (search, extraction, verification) makes the system easier to debug, modify, and parallelize. 2. **Intermediate Result Saving is Crucial**: For long-running processes with potential failures, saving intermediate results prevents data loss and enables recovery. 3. **Error Handling Must Be Comprehensive**: Web scraping faces numerous failure modes; comprehensive error handling with graceful degradation is essential. 4. **Free Services Are Unreliable for Production**: The experience with DuckDuckGo shutting down free access highlighted the importance of planning for commercial-grade services from the start.   **Future Improvements:**   1. **Machine Learning Integration**: Training models on successful search patterns could improve query generation and result ranking 2. **Caching Mechanisms**: Implementing intelligent caching to reduce redundant requests and improve performance 3. **Multi-Language Support**: Extending the approach to handle non-English company websites and documentation 4. **Real-Time Monitoring**: Developing systems to monitor for newly published reports automatically 5. **Content Validation**: Adding PDF content analysis to verify that discovered documents are actually annual financial reports 6. **API Integration**: Where available, integrating with official company APIs or regulatory databases for more reliable data access |

|  |
| --- |
| The competition highlighted that while automation is possible, the diversity of corporate web structures and varying API limitations require flexible, multi-faceted approaches rather than single-strategy solutions. The shift from free to commercial APIs proved essential for reliable results. |

**Short description of the Team – area of expertise**

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

|  |
| --- |
| **Christian Url, Data Scientist and Statistician, [christian.url@protonmail.com](mailto:christian.url@protonmail.com)** |