FlowETL : An Autonomous Example-Driven ETL Pipeline

Mattia Di Profio (author), Dr. Mingjun Zhong (supervisor)

1495 UNIVERSITY OF ABERDEEN

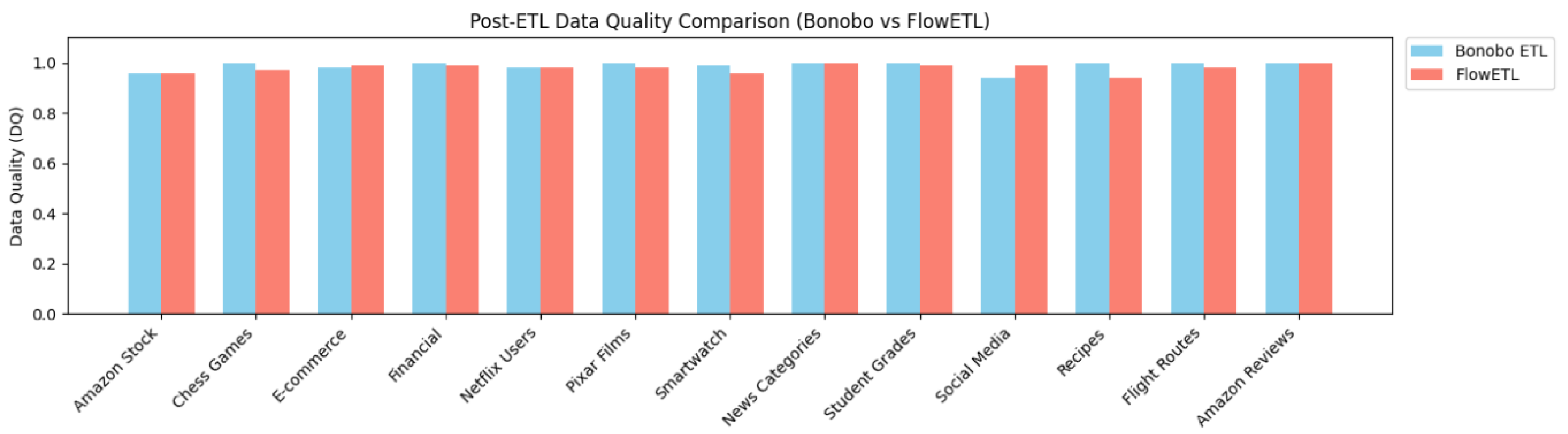
Motivation and Background

- Data analysts spend around *80%* of their time on data wrangling due to the absence of reliable, automated transformation methods, especially for unstructured data [1].
- Traditional ETL pipelines help standardise data but often require manual, one-off transformations that are hard to reuse and generalise [2].
- **FlowETL** proposes an example-based, autonomous ETL architecture that automatically prepares datasets according to a concise, user-defined target.

Evaluation Methodology

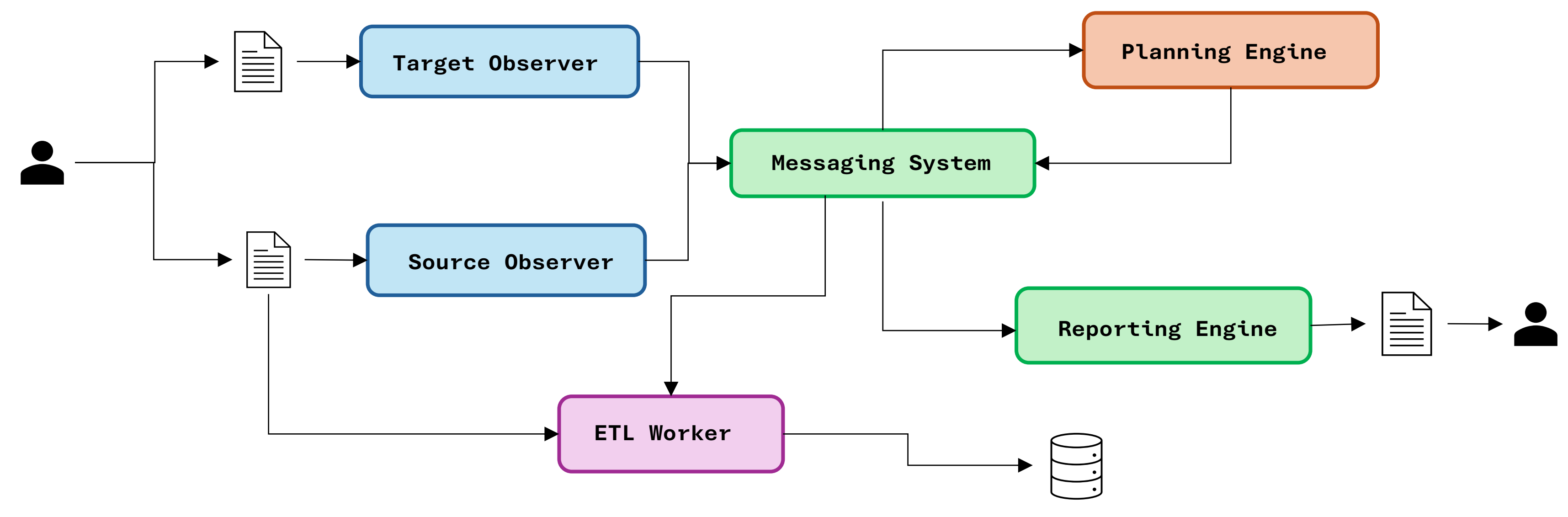
- An evaluation corpus of *13* diverse datasets was created to assess FlowETL's generalisation against human-defined targets and ground truth transformation plans.
- *40%* missing values, *20%* duplicate rows, and *10%* outliers were artificially introduced, simulating real-world data issues.
- The Planning Engine was evaluated using a new metric – **PlanEval** – inspired by the Success Rate for Data Transformation (SRDT) [3], measuring the percentage of correctly generated transformations.
- **Bonobo**, an open-source Python framework, was selected as the main tool for comparison

Evaluation Results



- FlowETL achieved post-ETL data quality scores between *0.96-1.00* across all datasets.
- FlowETL consistently produced high-quality outputs and PlanEval scores by autonomously inferring and executing transformations.
- The overall data quality across all datasets was slightly lower for FlowETL, showing *0.5–4%* more data wrangling issues compared to Bonobo.
- FlowETL achieved constant runtime on the plan generation step for all datasets, including CSV files with *600,000+* rows and JSON files with *50,000+* objects.

System Architecture



- **Observers** are responsible for detecting the source file to be standardized and the target file that guides the data wrangling plan generation.
- The **Planning Engine** uses **LLM** inference to create an executable plan for standardising the source file and improving data quality against missing values, duplicates, and outliers.
- The **ETL Worker** ingests the source file and applies the transformation plan.
- The **Messaging System** and **Report Engine** serve as supporting components, enabling communication between components and logging runtime metrics, respectively.

Conclusion and Future Work

- FlowETL has been evaluated across 13 datasets, including structured and unstructured data from various domains, showing strong performance and generalisation capabilities.
- Future work will focus on extending the data wrangling capabilities and conducting a more in-depth evaluation of schema matching and transformation logic inference.
- Deploying the pipeline in an enterprise environment could offer valuable insights into the strengths and weaknesses of the architecture.

References

- [1] Anthony Mbata, Yaji Sripada, and Mingjun Zhong. A survey of pipeline tools for data engineering. arXiv preprint arXiv:2406.08335, 2024.
- [2] Sara B Dakrory, Tarek M Mahmoud, Abdelmgeid A Ali, et al. Automated etl testing on the data quality of a data warehouse. International Journal of Computer Applications, 131(16): 9–16, 2015.
- [3] Tengjun Jin, Yuxuan Zhu, and Daniel Kang. Elt-bench: An end-to-end benchmark for evaluating ai agents on elt pipelines. arXiv preprint arXiv:2504.04808, 2025.