TAFT: A Transformer-Based Approach for Format Transformation

Erik Schönwälder[1,*], Julius Gonsior[1], Anja Reusch[1], Claudio Hartmann[1] and Wolfgang Lehner[1]

[1]Database Research Group, Technische Universität Dresden, Dresden, Germany

Abstract

The presence of heterogeneous data formats within data lakes poses challenges when attempting to analyze or further process such data. While data cleaning tools can remove heterogeneities within individual documents, they fail to address *global* format heterogeneities across multiple documents. For example, two documents store addresses each in a consistent format, thus not counting as a target for existing data cleaning tools. However, these consistent formats may still differ from each other, thereby posing global format heterogeneities. In order to close this gap, we present the framework TAFT (A **T**ransformer-based **A**pproach for **F**ormat **T**ransformation), designed to remove these global format heterogeneities at scale, without human-in-the-loop involvement. To this end, we leverage a transformer-based model to convert the document columns into a uniform format based on types describing their content, such as Address or Name. With minimal configuration effort, we achieve state-of-the-art results without any further human intervention.

Keywords

format transformation, data preparation, heterogeneity

1. Introduction

Data lakes are widely deployed to collect data in a central repository without undergoing initial processing in its original, raw format [1]. Target use-cases, like analyses, are often unknown during data collection, leading to the majority of stored data being non-standardized, requiring a pre-processing step before further processing [2]. Due to this heterogeneity and particularly owing to the absence of a standard representation throughout the data lake, a substantial portion of the data fails to be directly consumable by downstream applications, such as analytical tools or management systems [2].

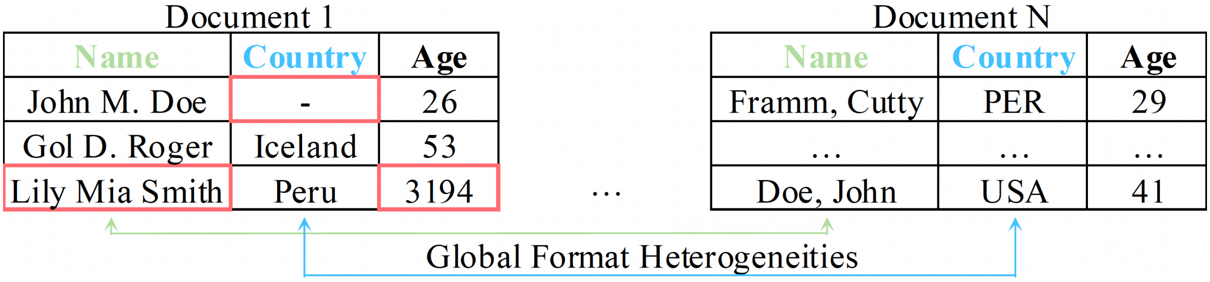
Aiming to make the data consumable for these applications, data scientists undertake a labor-intensive process called data preparation [2]. To enhance productivity, data cleaning tools like Raha [3] and Wrangler [4] assist data scientists in this task, as demonstrated by the typical workflow depicted in Fig. 1. After selecting relevant documents for analysis, the data scientist uses data cleaning tools to clean and adjust the data. Starting with Document 1, the cleaning tool flags *Lily Mia Smith* as a pattern violation due to its differing format and detects missing values in the Country column and outliers in the Age column. After cleaning Document 1, the data scientist moves on to the next document and so forth. While this workflow effectively removes *local* errors within individual documents, it fails to detect *global* errors that become apparent only when considering all documents in a corpus. For example, after correcting a pattern violation in the Name column, Document 1 shows no format heterogeneities. However, inconsistencies may arise when comparing Document 1 with other documents. Document N also has a Name column in a uniform format, but it does not match the format of Document 1 ($John\,M.\,Doe \neq Doe,\ John$). Furthermore, the Country columns do not match in format either

Discovery Science - Late Breaking Contributions 2024

[*]Corresponding author.

✉ erik.schoenwaelder@tu-dresden.de (E. Schönwälder); julius.gonsior@tu-dresden.de (J. Gonsior); anja.reusch@tu-dresden.de (A. Reusch); claudio.hartmann@tu-dresden.de (C. Hartmann); wolfgang.lehner@tu-dresden.de (W. Lehner)

0009-0005-4178-5160 (E. Schönwälder); 0000-0002-5985-4348 (J. Gonsior); 0000-0002-2537-9841 (A. Reusch); 0000-0002-5334-059X (C. Hartmann); 0000-0001-8107-2775 (W. Lehner)



Document 1

Name	Country	Age
John M. Doe	-	26
Gol D. Roger	Iceland	53
Lily Mia Smith	Peru	3194

Document N

Name	Country	Age
Framm, Cutty	PER	29
…	…	…
Doe, John	USA	41

Figure 1: Typical workflow of a data scientist

$(Peru \neq PER)$. As a result, sorting, aggregating, or joining these documents becomes infeasible, leading to an inability to apply data analysis flexibly on file-based data storage.

Existing data cleaning tools, such as Raha [3] or Wrangler [4] cannot detect these *global* errors because they operate on a document-based level and from the perspective of a single document, a uniformly formatted column appears to have no errors. As an alternative, extensive research (e.g. Flashfill [5], UDATA [6]) has been conducted to fix format heterogeneities within the task of *format transformation*. This task refers to the conversion of data of the same logical type into a uniform representation. However, applying these systems globally is not feasible either, as they require user involvement for each transformation, which is impractical and lacks scalability for thousands or millions of documents.

Addressing these issues, we propose TAFT[1], an end-to-end framework designed to detect and correct format heterogeneities in a global manner, with a specific emphasis on achieving this without human-in-the-loop intervention.

2. TAFT: a framework to detect global heterogeneities

The abstract architecture of TAFT is divided into two stages: the *Detection Stage*, which annotates columns with types, and the *Correction Stage*, which converts these columns into a uniform format predefined for each type.

During the *Detection Stage*, columns in documents within a given document collection are annotated with types that classify their content, such as Name, Country, or Address. This annotation is performed by a Column Type Annotation (CTA) model, freely chosen by the data scientist, allowing for the integration of corresponding research and new approaches into TAFT. For our experiments, we selected the state-of-the-art model DODUO [7], as it outperforms other models like SATO [8]. Beyond using pretrained CTA models with a defined type set like DODUO, TAFT also supports the creation of custom types and the training of the chosen CTA model for domain-specific adaptations. For example, a telecommunications company deals with data that cannot be classified using a general type set. Here, specific types such as *IPv4*, *IPv6*, *RGB*, or *MAC addresses* become relevant, as data of these types may be inconsistently formatted. While using a pretrained CTA model is straightforward, tailoring it for domain-specific contexts with custom types requires appropriate data.

To efficiently generate such data, we use fuzzy generators that produce synthetic data samples sharing specific traits, like Names or Addresses. For most types, Python or Java packages can easily build fuzzy generators with minimal effort. Additionally, many packages, like *random_address* or *names*, incorporate real-world data, extending the utility of fuzzy generators beyond synthetic data.

In the *Correction Stage*, annotated columns are transformed into a uniform format predefined for each type. To achieve this, a transformer-based model, specifically *flan-t5-large* from the FLAN-T5 family, is trained to perform the correction. To convert a column into the target format, the model takes the column's values, the desired format, and the task-specific prefix *reshape:* as input. For example, `DD/MM/YYYY reshape: November 01, 2002 [ROW] May 27, 1997 [ROW]. . .` represents the

[1] Code, data, and models are available at https://github.com/goodguyerik/TAFT

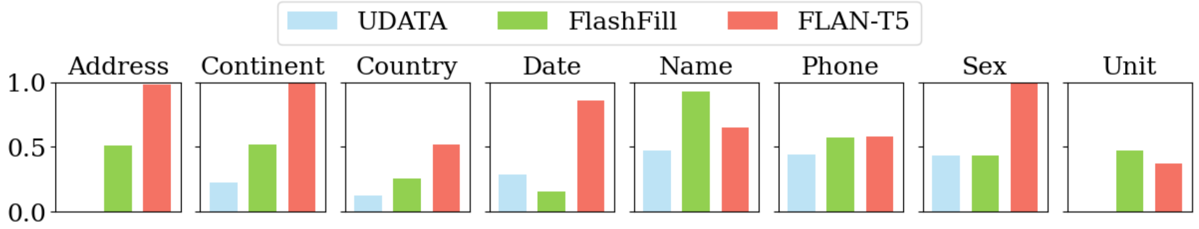


Figure 2: Comparison of Correction Approaches

input for a date column to be formatted as `DD/MM/YYYY`. The special token `[ROW]` separates individual column values. As output, the model produces a text sequence, such as `01/11/2002 [ROW] 27/05/1997 [ROW]. . .` , containing the entered values transformed into the desired format. The generic design of the data generation process from the detection stage can also be utilized to generate data for the correction stage.

3. Evaluation

We evaluated the FLAN-T5 model, which is designed to transform columns into a predefined uniform format, by comparing it to two promising alternatives: UDATA [6] and FlashFill [5]. For our experimental setup, we generated 1,000 test columns, each in both an input and output format, for eight distinct column types (Address, Continent, Country, Date, Name, Phone, Sex, Unit) using our data generation method. Fig. 2 shows the percentage of correctly transformed columns for each type and correction approach. As demonstrated, our FLAN-T5 model remarkably outperforms UDATA and FlashFill, except in the Name and Unit types. For the Continent, Country, and Sex types, which necessitate semantic understanding for transformation, FLAN-T5 considerably outperforms both FlashFill and UDATA. For instance, converting a country code into its corresponding country name requires semantic understanding of the correct mapping, as seen with `Germany` → `DEU`. FLAN-T5 learns these mappings during training without access to an external knowledge base, whereas UDATA and FlashFill rely on observing patterns using provided examples, which do not exist. For the other types, which only require syntactic operations such as changing the order of components, adding delimiters, or deleting components or parts thereof, FLAN-T5 achieves results comparable to FlashFill, while UDATA either fails or only moderately performs the desired transformations.

4. Conclusion

We proposed the two-stage framework TAFT, which addresses global format heterogeneities in data lakes without human intervention. Our evaluation shows that the language model-based correction approach, especially due to its semantic understanding, outperforms state-of-the-art systems while being highly customizable. The framework frees data scientists from manually resolving global format heterogeneities, allowing them to focus on other data preparation tasks.

Acknowledgments

The authors gratefully acknowledge the computing time made available to them on the high-performance computer at the NHR Center of TU Dresden. This center is jointly supported by the Federal Ministry of Education and Research and the state governments participating in the NHR (www.nhr-verein.de/unsere-partner).

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] A. Nambiar, D. Mundra, An overview of data warehouse and data lake in modern enterprise data management, Big Data and Cognitive Computing 6 (2022) 132. doi:10.3390/bdcc6040132.

[2] M. Hameed, F. Naumann, Data preparation: A survey of commercial tools, SIGMOD Rec. 49 (2020) 18–29. doi:10.1145/3444831.3444835.

[3] M. Mahdavi, Z. Abedjan, R. Castro Fernandez, S. Madden, M. Ouzzani, M. Stonebraker, N. Tang, Raha: A configuration-free error detection system, in: Proceedings of the 2019 International Conference on Management of Data, SIGMOD '19, Association for Computing Machinery, New York, NY, USA, 2019, p. 865–882. URL: https://doi.org/10.1145/3299869.3324956. doi:10.1145/3299869.3324956.

[4] S. Kandel, A. Paepcke, J. Hellerstein, J. Heer, Wrangler: interactive visual specification of data transformation scripts, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '11, Association for Computing Machinery, New York, NY, USA, 2011, p. 3363–3372. URL: https://doi.org/10.1145/1978942.1979444. doi:10.1145/1978942.1979444.

[5] S. Gulwani, Automating string processing in spreadsheets using input-output examples, SIGPLAN Not. 46 (2011) 317–330. URL: https://doi.org/10.1145/1925844.1926423. doi:10.1145/1925844.1926423.

[6] M. Pham, C. A. Knoblock, J. Pujara, Learning data transformations with minimal user effort, in: 2019 IEEE International Conference on Big Data (Big Data), 2019, pp. 657–664. doi:10.1109/BigData47090.2019.9006350.

[7] Y. Suhara, J. Li, Y. Li, D. Zhang, c. Demiralp, C. Chen, W.-C. Tan, Annotating columns with pre-trained language models, in: Proceedings of the 2022 International Conference on Management of Data, SIGMOD '22, Association for Computing Machinery, New York, NY, USA, 2022, p. 1493–1503. URL: https://doi.org/10.1145/3514221.3517906. doi:10.1145/3514221.3517906.

[8] D. Zhang, M. Hulsebos, Y. Suhara, c. Demiralp, J. Li, W.-C. Tan, Sato: contextual semantic type detection in tables, Proc. VLDB Endow. 13 (2020) 1835–1848. URL: https://doi.org/10.14778/3407790.3407793. doi:10.14778/3407790.3407793.