A Novel Pipeline for Generating Realistic Synthetic CDISC ADaM Datasets Using Large Language Models and Knowledge Graphs

Jaime Yan, Merck & Co., Inc., Rahway, NJ, USA

Chao Su, Merck & Co., Inc., Rahway, NJ, USA

Abstract

This paper presents a systematic pipeline for generating realistic synthetic Clinical Data Interchange Standards Consortium (CDISC) Analysis Data Model (ADaM) datasets. We identify and address two fundamental challenges in direct JSON schema-based generation: the lack of domain-specific knowledge (e.g., laboratory test ranges, MedDRA codes) and insufficient capture of ADaM dataset structures and variable relationships. Our methodology introduces a two-pronged approach to schema enhancement: (1) knowledge enrichment through both a structured knowledge query system leveraging well-organized clinical trial documentation and Large Language Model (LLM) assistance for generating realistic value ranges and distributions, and (2) structure optimization using rule-based LLM prompts to reorganize schemas according to specific ADaM data structures (ADSL, BDS, OCCDS). We further propose two implementation strategies: enhanced JSON schema-based generation and template-based Faker code generation, with the latter ensuring proper handling of cross-dataset relationships by establishing ADSL as the foundation for subsequent dataset generation. Through experimental comparison of datasets generated using direct JSON specifications, template-based generation, and our enhanced schema approach, we demonstrate the effectiveness of our pipeline in producing high-quality synthetic data that maintains both structural integrity and realistic relationships. Our evaluation framework provides comprehensive metrics for assessing synthetic data quality across these different generation approaches.

***K*eywords** Synthetic Data · CDISC · ADaM · Clinical Trials · Large Language Models · Data Generation · Faker

1 Introduction

The generation of synthetic clinical trial data has become increasingly crucial in modern pharmaceutical research and development, driven by growing demands for data privacy protection and efficient software development processes. At its core, the challenge lies in creating synthetic data that faithfully represents real-world clinical trial characteristics while maintaining compliance with Clinical Data Interchange Standards Consortium (CDISC) Analysis Data Model (ADaM) standards. The quality of synthetic data is fundamentally determined by our ability to capture and apply domain knowledge, statistical distributions, and valid value ranges – the more comprehensive and accurate this knowledge, the more realistic the generated data.

Current approaches to synthetic ADaM data generation often fall short in three critical aspects: capturing complex variable relationships, preserving ADaM-specific data structures, and incorporating comprehensive domain knowledge. While existing methods can generate basic variable-level synthetic data, they struggle to replicate the intricate relationships between variables across different datasets (ADSL, BDS, OCCDS) and maintain the hierarchical structures inherent in clinical trial data. This limitation stems primarily from insufficient utilization of available domain knowledge and incomplete understanding of data characteristics.

Our research introduces a novel pipeline for generating high-quality synthetic ADaM datasets, founded on the principle that better knowledge extraction leads to more realistic data generation. We begin with ADaM specifications as our primary knowledge source, being the most relevant documentation for ADaM dataset characteristics. Our approach leverages Large Language Models (LLMs) to extract and synthesize this crucial domain knowledge from both ADaM

specifications and supplementary clinical trial documentation, with the understanding that the utility of additional documentation depends on its relevance to ADaM dataset characteristics.

The innovation in our approach lies in:

1. **Comprehensive Knowledge Acquisition and Integration:**
   - Development of a comprehensive JSON schema conversion process that captures essential ADaM metadata.
   - Novel use of LLMs to extract and synthesize domain knowledge from clinical documentation.
   - Intelligent assessment of documentation relevance to ADaM dataset characteristics.
2. **Dual Enhancement and Validation:**
   - A dual enhancement approach combining structured knowledge querying and LLM-based augmentation.
   - Automated structure optimization aligned with ADaM standards.
   - Template-based generation ensuring consistency across dataset types.
3. **Rigorous Quality Assessment:**
   - Quantitative metrics for evaluating synthetic data realism.
   - Validation of knowledge transfer effectiveness.
   - Cross-dataset relationship verification.

The significance of this work extends beyond ADaM dataset generation. Our approach demonstrates how LLMs can be effectively leveraged to extract and apply domain knowledge in synthetic data generation, a methodology applicable across various domains. The experimental results not only validate our approach's effectiveness in the clinical trial context but also provide insights into the relationship between input documentation quality, knowledge extraction accuracy, and synthetic data realism.

By providing high-quality synthetic datasets that maintain both structural integrity and realistic relationships, our approach enables earlier commencement of statistical programming activities, facilitates software testing, and supports training initiatives while ensuring strict adherence to data privacy requirements. More broadly, our findings contribute to the understanding of how domain knowledge can be effectively extracted and applied in synthetic data generation tasks across different fields.

The remainder of this paper is organized as follows: Section 2 reviews related work in synthetic data generation and clinical data standards. Section 3 details our methodology, emphasizing the role of knowledge extraction and application. Section 4 presents our evaluation framework and experimental results, analyzing the relationship between input knowledge quality and output data realism. Section 5 discusses implications and limitations, and Section 6 concludes with future research directions and broader applications of our approach.

2 Related Work

The field of synthetic data generation has seen significant advancements, particularly in scenarios involving data augmentation with real data and the creation of entirely synthetic datasets from scratch. In the realm of data augmentation, researchers have focused on generating synthetic data that retains the statistical properties of the original datasets while enhancing privacy and utility. For instance, Barse et al. [1] developed methods for fraud detection using synthetic data, and Lin et al. [2] designed tools for system testing and training. Templ et al. [3] introduced simPop, an open-source tool aimed at statistical disclosure control, while Ma et al. [4] utilized transfer learning to augment small datasets by blending real and synthetic data.

In contrast, the generation of synthetic data from scratch, without any real data input, has been explored using various models and frameworks. Macià et al. [5] generated synthetic data to study classifier behavior, and Capobianco et al. [6] developed DocEmul, a toolkit for producing structured documents. The use of differentially private GANs by Torkzadehmahani et al. [7] aimed at preserving privacy in synthetic data generation. Similarly, Borkman et al. [8] introduced Unity Perception for creating synthetic datasets in computer vision, while McKenna et al. [9] proposed methods to maintain statistical integrity in generated data. Das et al. [10] presented a hybrid model to enhance the robustness of synthetic data generation.

Within clinical trials, synthetic data plays a crucial role in evaluating the safety and efficacy of medical interventions, offering a privacy-preserving alternative to real data. Beaulieu-Jones et al. [11] demonstrated the feasibility of using deep neural networks with differential privacy to generate synthetic participants for clinical studies. Callahan et al.

[12] and Krieg [13] explored the use of synthetic data in assessing new medical technologies and therapies. The work of Harris et al. [14], which compared synthetic and biologic materials in clinical settings, underscores the practical applications and challenges of synthetic data in real-world scenarios.

Despite these advances, challenges remain in ensuring the validity and reliability of synthetic data, particularly regarding its application in clinical trials. Baudry et al. [15] explored the use of LLMs combined with the Faker library for generating synthetic data. However, their approach lacked a comprehensive framework for quality control and automated testing of the generated code.

This paper addresses these limitations by proposing a detailed methodology that integrates LLMs, schema-driven templates, and knowledge graph-based refinement to produce high-quality synthetic data that meets stringent standards for realism and accuracy. This approach is designed to facilitate early-stage analysis and tool development in clinical trials, providing a reliable alternative to real data.

3 Methodology

Our pipeline for generating realistic synthetic CDISC ADaM datasets comprises four primary stages, each addressing specific challenges in synthetic clinical data generation. Figure 1 illustrates the complete workflow, from initial specification conversion to final quality evaluation.

1. **Initial JSON Schema Generation:** Converting ADaM specifications into structured JSON format
2. **Schema Enhancement:** Enriching schemas with domain knowledge and structural relationships
3. **Synthetic Data Generation:** Using enhanced schemas for Faker-based data generation
4. **Quality Evaluation:** Assessing generated data quality through comprehensive metrics

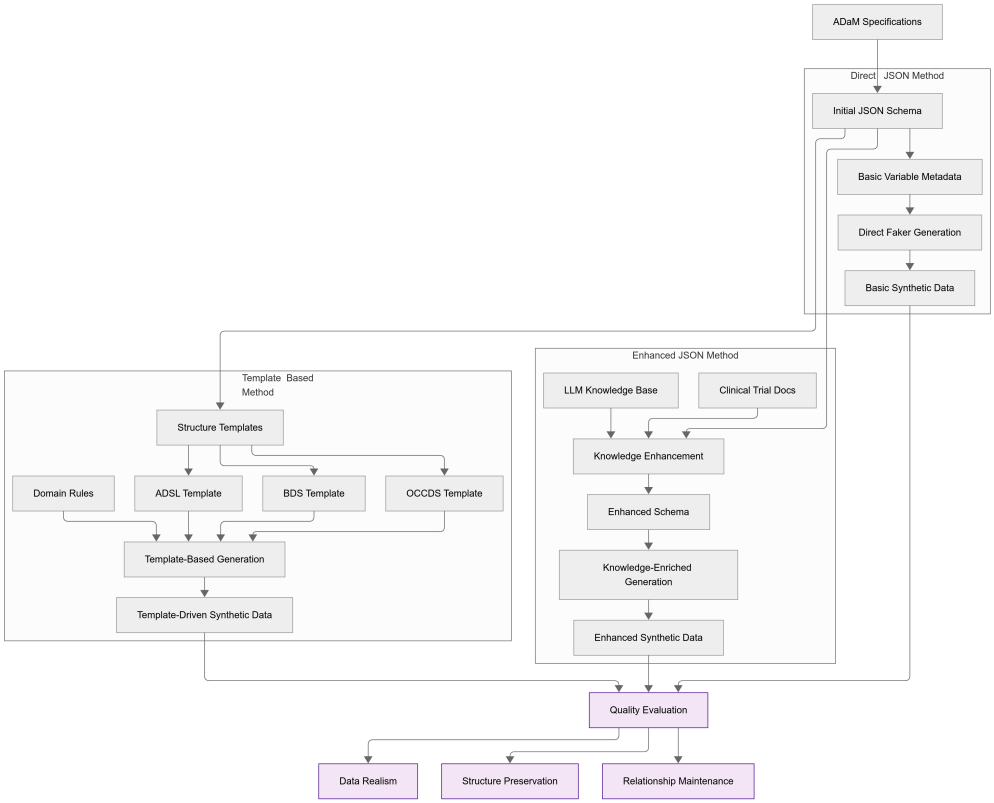

Figure 1: Workflow for Synthetic ADaM Data Generation

3.1 Initial JSON Schema Generation

This initial stage involves transforming ADaM dataset specifications, typically found in spreadsheets or text documents, into a structured, machine-readable JSON schema. This standardized format facilitates subsequent processing steps.

Key Metadata Captured:

- **Variable Names:** (e.g., USUBJID, AGE, TRT01P).
- **Data Types:** (Character, Numeric, Date).
- **Codelists/Controlled Terms:** Allowed values for categorical variables (e.g., SEX = "M", "F", "U").
- **Dataset Structure:** Subject-Level (ADSL), Basic Data Structure (BDS), Occurrence Data Structure (OCCDS).
- **Relationships and Constraints:** Unique identifiers, foreign key relationships, and logical dependencies between variables.

Example JSON Schema for ADSL

```
{
  "Dataset": "ADSL",
  "Structure": "ONE_RECORD_PER_SUBJECT",
  "Key Variables": ["STUDYID", "USUBJID"],
  "Variables": [
    { "Name": "STUDYID", "Type": "Char", "Length": 8 },
    { "Name": "USUBJID", "Type": "Char", "Length": 20 },
    { "Name": "AGE", "Type": "Num", "Length": 8 },
    { "Name": "SEX", "Type": "Char", "Length": 3, "Codelist": "SEX" }
  ]
}
```

3.2 Schema Enhancement

The initial JSON schema, while structured, may lack explicit representation of the complex relationships and hierarchical structures inherent in ADaM datasets as well as lack the background knowledge. We address two fundamental limitations of direct JSON specifications:

Table 1: Detailed Analysis of Schema Enhancement Approaches

Enhancement Type	Method	Implementation Details
Domain Knowledge	Knowledge Query System	Extracts structured information from clinical trial documents (protocols, SAPs, CRFs) using NLP and pattern matching
	LLM Enhancement	Generates realistic value ranges and distributions based on variable context and relationships
	Knowledge Graph	Maps relationships between variables, datasets, and domain concepts
Structure Optimization	ADSL Templates	One-record-per-subject structure with demographic and treatment patterns
	BDS Templates	Parameter-value relationships with visit patterns and change calculations
	OCCDS Templates	Event-based temporal relationships with sequence validation
Data Generation	Faker Code Generation	Direct translation of enhanced schemas to Faker code with relationship preservation
	Template Application	Specialized templates for each ADaM structure with built-in validations
	Cross-Dataset Integration	ADSL-first approach with consistent subject propagation

3.2.1 Domain Knowledge Integration

Two complementary approaches:

- **Knowledge Query System:** Leveraging well-organized clinical trial documentation (protocols, SAPs, CRFs) to build structured knowledge graphs, the input data continues will highly influence the result of this method, the process is: build the knowledge graph based on collect clinical trial document(the more relevant to ADaM dataset creating the more better result will get, since it will provide extra info for the next Faker code generation process), then use the info from json block by block as input to this knowledge graph query system, we extract the knowledge like variable range, derivation equation etc. info, let LLM insert the extract info to exist json follow the predefined structure, then the enhancement work done.
- **LLM-Based Enhancement:** Leveraging Large Language Models to extract and synthesize domain-specific knowledge, including realistic value ranges, statistical distributions, and variable relationships. The effectiveness of this approach correlates with the model's capabilities in understanding clinical trial contexts and statistical patterns. Our experiments demonstrate that advanced LLMs can identify nuanced relationships between variables and suggest appropriate value constraints based on their comprehensive understanding of clinical trial protocols and ADaM specifications.

3.2.2 Structure Optimization

This enhancement step aimsis to reorganize the schema based on the specific ADaM data structure (ADSL, BDS, OCCDS) to make relationships explicit and would be catch by the LLM.

Different ADaM dataset types adhere to distinct structural patterns. Table 2 summarizes these structures and the corresponding enhancement strategies.

Table 2: ADaM Dataset Structure and Relationship Organization

Dataset Type	Structure	Enhancement Approach
ADSL	One Record Per Subject	Ensure subject-level integrity (one record per USUBJID). Maintain hierarchical relationships between key identifiers (USUBJID, STUDYID, SITEID). Align treatment variables (TRT01P, TRT01A) and clearly structure demographic/baseline covariates (AGE, SEX, RACE).
BDS	One Record Per Parameter Per Visit	Align PARAMCD with AVAL, BASE, CHG, and other related variables within each parameter and visit. Ensure each parameter has a clear mapping to corresponding analysis variables.
OCCDS	One Record Per Occurrence	Preserve event-based relationships, ensuring each record represents a distinct occurrence (e.g., adverse events, concomitant medications). Clarify relationships between variables like AEDECOD, AESTDY, AEENDY, and severity gradings.

Example: ADSL Schema Enhancement *Problem: The corresponding variables for the same subject might be treated as individual variables, such as the subject identifier* `USUBJID` *and* `STUDY`*, the* `USUBJID` *always contains the info from* `STUDY`*.*

JSON Structure (Before Enhancement)

```
{
  "Variables": {
    "USUBJID": { "Label": "Unique Subject Identifier", "Type": "Char", "Length": 20 },
    "SITEID": { "Label": "Site Identifier", "Type": "Char", "Length": 10 },
    "TRT01P": { "Label": "Planned Treatment", "Type": "Char", "Length": 20 }
  }
}
```

Enhanced JSON Structure (After Reorganization)


```
{
  "Dataset": "ADSL",
  "Structure": "ONE_RECORD_PER_SUBJECT",
  "Key Variables": ["USUBJID", "STUDYID"],
  "Variables": [
    { "Name": "USUBJID", "Label": "Unique Subject Identifier", "Type": "Char", "Length": 20 },
    { "Name": "SITEID", "Label": "Site Identifier", "Type": "Char", "Length": 10 },
    { "Name": "TRT01P", "Label": "Planned Treatment", "Type": "Char", "Length": 20 }
  ]
}
```


Key Improvements in ADSL (Analysis Data Structure for Subjects):

- Explicitly defines the dataset structure as "ONE_RECORD_PER_SUBJECT". The quotes are crucial for representing this as a string literal value. Using ensures a consistent, monospaced font, which is standard for representing code or data values. Using ONE_RECORD_PER_SUBJECT describes the dataset structure more accurately.
- Ensures hierarchical relationships (e.g., between subject, visits, and events) are preserved and clearly defined within the data structure. The original text was vague; this adds crucial context.

Example: BDS (Basic Data Structure) Schema Enhancement *Problem: PARAMCD is a list of codes, but there is no explicit mapping between a PARAMCD and the corresponding AVAL, BASE, CHG. The LLM cannot understand how each PARAMCD relates to different values.*

Original JSON Structure (Before Enhancement)


```
{
  "PARAMCD": { "Values": ["SYSBP", "DIABP"], "Label": "Parameter Code" },
  "AVAL": { "Label": "Analysis Value", "Type": "Num" },
  "BASE": { "Label": "Baseline Value", "Type": "Num" }
}
```


Enhanced JSON Structure (After Reorganization)


```
{
  "Dataset": "BDS",
  "Structure": "ONE_RECORD_PER_SUBJECT_PER_PARAMETER_PER_VISIT",
  "Parameters": [
    {
      "PARAMCD": "SYSBP",
      "PARAM": "Systolic Blood Pressure",
      "Analysis Variables": [
        { "Name": "AVAL", "Label": "Analysis Value", "Type": "Num" },
        { "Name": "BASE", "Label": "Baseline Value", "Type": "Num" },
        { "Name": "CHG", "Label": "Change from Baseline", "Type": "Num" }
      ]
    },
    {
      "PARAMCD": "DIABP",
      "PARAM": "Diastolic Blood Pressure",
      "Analysis Variables": [
        { "Name": "AVAL", "Label": "Analysis Value", "Type": "Num" },
        { "Name": "BASE", "Label": "Baseline Value", "Type": "Num" },
        { "Name": "CHG", "Label": "Change from Baseline", "Type": "Num" }
      ]
    }
  ]
}
```


Key Improvements in BDS (Basic Data Structure):

- Ensures that each PARAMCD has a clear and unambiguous mapping to its corresponding variables (e.g., PARAM, AVAL, AVALC). This improves data traceability and understandability. Adding specific examples of related variables is crucial for clarity.

- Adheres to the BDS standard structure of "ONE_RECORD_PER_SUBJECT_PER_PARAMETER_PER_VISIT". This ensures consistency and facilitates data integration. The use of quotes is important here, indicating a specific data structure string.
- Facilitates structured synthetic data generation by providing a well-defined schema. This is essential for testing and development purposes. Improved wording for clarity and conciseness.

Example: OCCDS (Occurrence Data Structure) Schema Enhancement *Problem: The original schema did not show the occurrence relationship between the occurrence variables.*

Original JSON Structure (Before Enhancement)

```
{
  "AEDECOD": { "Label": "Adverse Event", "Type": "Char", "Length": 40 },
  "AESTDY": { "Label": "Start Day", "Type": "Num" },
  "AEENDY": { "Label": "End Day", "Type": "Num" }
}
```

Enhanced JSON Structure (After Reorganization)

```
{
  "Dataset": "OCCDS",
  "Structure": "ONE_RECORD_PER_SUBJECT_PER_OCCURRENCE",
  "Occurrence Variables": [
    { "Name": "AEDECOD", "Label": "Adverse Event", "Type": "Char", "Length": 40 },
    { "Name": "AESTDY", "Label": "Start Day", "Type": "Num" },
    { "Name": "AEENDY", "Label": "End Day", "Type": "Num" },
    { "Name": "AESER", "Label": "Serious Event Flag", "Type": "Char", "Allowed Values": ["Y", "N"] }
  ]
}
```

Key Improvement in OCCDS (Occurrence Data Structure):

- Explicitly states the dataset follows a "ONE_RECORD_PER_SUBJECT_PER_OCCURRENCE" structure. This clearly defines the granularity of the data. The use of quotes and is essential for representing this as a string literal.
- Ensures each event (e.g., Adverse Event - AE) is stored as a separate row. This facilitates data analysis and reporting. Added "e.g., Adverse Event - AE" for clarification. Using the unabbreviated and abbreviation provides better clarity.

3.2.3 LLM's Role in Reorganization

The Large Language Model (LLM) assists by:

- Converting unstructured JSON data into the new, standardized format based on predefined templates. This ensures consistency and reduces manual effort.
- Extracting and populating missing metadata, including standard derivations such as `CHG` (Change from Baseline) and `BASE` (Baseline Value). Providing specific examples of the metadata (CHG, BASE) with clear explanations greatly improves clarity.
- Applying naming consistency across variables, ensuring adherence to established naming conventions. This is crucial for data quality and interoperability. Adding context about *why* naming consistency is important.

3.3 Synthetic Data Generation

The enhanced JSON schema serves as the blueprint for generating synthetic data. We utilize the Python `Faker` library, employing two complementary approaches:

1. **Direct Faker Code Generation:** The JSON schema is automatically translated into Python code that directly calls `Faker` functions to generate data for each variable. This is suitable for simpler ADaM structures.
2. **Template-Based Code Generation:** For more complex ADaM datasets (e.g., BDS, OCCDS), predefined Python code templates are used. These templates are populated with variable-specific information from the enhanced JSON schema, allowing for greater control over data generation and relationship preservation.

Table 3: Comparison of ADaM Dataset Generation Methods

Aspect	Direct JSON Spec	Enhanced JSON Spec	Template-Based
Knowledge Source	Basic variable metadata only	Domain knowledge + LLM suggestions	Predefined templates + Domain rules
Structural Info	Limited variable relationships	Explicit ADaM structure	Built-in structural patterns
Variable Relationships	Individual variable constraints	Cross-variable dependencies	Pre-designed relationship flows
Dataset Integration	Independent generation	Knowledge graph-based links	ADSL-centered generation
Scalability	High	Medium	Medium
Maintenance	Simple updates	Requires knowledge base updates	Template updates needed
Use Cases	Simple datasets	Complex relationships	Production environments

3. **Enhanced Schema-Based Generation:** Use LLM's ability to either add extra knowledge to the json spec, or reorganize the json structure to insert relationship info to the spec json file.

Example Faker-Based Data Generation for ADSL

```
from faker import Faker
fake = Faker()

def generate_adsl(num_subjects):
    data = []
    for _ in range(num_subjects):
        record = {
            "STUDYID": fake.pystr(max_chars=8),
            "USUBJID": fake.pystr(max_chars=20),
            "AGE": fake.random_int(min=18, max=90),
            "SEX": fake.random_element(elements=("M", "F", "U"))
        }
        data.append(record)
    return data
```

4 Evaluation and Results

4.1 Evaluation Metrics Formalization

4.1.1 General Quality Score

The overall quality score combines three components:

$$Q = w_1 \cdot DS + w_2 \cdot SS + w_3 \cdot RS \tag{1}$$

where $w_1 + w_2 + w_3 = 1$ and:

- DS: Data Structure Score
- SS: Subject-level Score
- RS: Relationship Score

4.1.2 ADSL-Specific Patterns

1. **Record Uniqueness Score:**

$$RU_{ADSL} = \frac{|\text{unique}(USUBJID_{gen})|}{|\text{unique}(USUBJID_{real})|} \tag{2}$$

2. **Demographic Pattern Score:**

$$DP_{ADSL} = \frac{1}{|V_d|} \sum_{v \in V_d} MI(v_{real}, v_{gen}) \tag{3}$$

where $V_d = \{AGE, SEX, RACE, ...\}$ are demographic variables

3. **Treatment Pattern Score:**

$$TP_{ADSL} = \frac{|\text{unique}(TRT01P_{gen}) \cap \text{unique}(TRT01P_{real})|}{|\text{unique}(TRT01P_{real})|} \tag{4}$$

4.1.3 BDS-Specific Patterns

1. **Parameter-Value Relationship Score:**

$$PV_{BDS} = \frac{1}{|P|} \sum_{p \in P} \frac{|\text{valid\_pairs}(p_{gen})|}{|\text{valid\_pairs}(p_{real})|} \tag{5}$$

where P is the set of PARAMCD values

2. **Visit Pattern Score:**

$$VP_{BDS} = \frac{1}{|S|} \sum_{s \in S} \frac{|\text{visits}(s_{gen}) \cap \text{visits}(s_{real})|}{|\text{visits}(s_{real})|} \tag{6}$$

where S is the set of subjects

3. **Analysis Variable Consistency:**

$$AV_{BDS} = \frac{1}{|C|} \sum_{c \in C} |corr(AVAL_{c,real}, CHG_{c,real}) - corr(AVAL_{c,gen}, CHG_{c,gen})| \tag{7}$$

where C is the set of valid PARAMCD-VISIT combinations

4.1.4 OCCDS-Specific Patterns

1. **Temporal Sequence Score:**

$$TS_{OCCDS} = \frac{|\{e \in E_{gen} : AESTDY_e \leq AEENDY_e\}|}{|E_{gen}|} \tag{8}$$

where E_{gen} is the set of events in generated data

2. **Event Distribution Score:**

$$ED_{OCCDS} = 1 - JS(P_{real}(AEDECOD), P_{gen}(AEDECOD)) \tag{9}$$

where JS is Jensen-Shannon divergence

3. **Subject Event Pattern Score:**

$$SE_{OCCDS} = \frac{1}{|S|} \sum_{s \in S} \left| \frac{|events(s_{gen})|}{|events(s_{real})|} - 1 \right| \tag{10}$$

4.1.5 Cross-Dataset Relationships

1. **Subject Consistency Score:**

$$SC = \frac{|\bigcap_{d \in D} USUBJID_d^{gen}|}{|\bigcap_{d \in D} USUBJID_d^{real}|} \tag{11}$$

where D is the set of all datasets

2. **Treatment Alignment Score:**

$$TA = \frac{1}{|S|} \sum_{s \in S} I(TRT01P_s^{ADSL} = TRT01P_s^{other}) \tag{12}$$

where I is the indicator function

3. **Visit Consistency Score:**

$$VC = \frac{1}{|V|} \sum_{v \in V} \frac{|subjects(v_{gen})|}{|subjects(v_{real})|} \tag{13}$$

where V is the set of all visits

4.1.6 Final Component Scores

1. **Data Structure Score (DS):**

$$DS = \alpha_1 DS_{ADSL} + \alpha_2 DS_{BDS} + \alpha_3 DS_{OCCDS} \tag{14}$$

2. **Subject-level Score (SS):**

$$SS = \beta_1 SS_{demo} + \beta_2 SS_{visit} + \beta_3 SS_{event} \tag{15}$$

3. **Relationship Score (RS):**

$$RS = \gamma_1 RS_{within} + \gamma_2 RS_{cross} + \gamma_3 RS_{temporal} \tag{16}$$

where all weight combinations (α_i, β_i, γ_i) sum to 1.

4.2 Experimental Setup

4.2.1 Dataset Characteristics

Reference datasets:

- Phase III trial ADSL (n=500)
- Corresponding BDS datasets (ADVS, ADLB)
- Associated OCCDS dataset (ADAE)

4.2.2 Evaluation Metrics Details

Table 4: Detailed Evaluation Metrics

Category	Specific Metrics	Measurement Method
Structure	Record count ratios	Count matching records per structure type
	Key variable presence	Verify required variables exist
	Format compliance	Check data type and length requirements
Relationships	Value dependencies	Correlation between related variables
	Cross-dataset links	USUBJID consistency across datasets
	Temporal patterns	Visit sequence validity
Data Quality	Value distributions	KS test for continuous variables
	Category proportions	Chi-square test for categorical variables
	Missing patterns	Compare missing data patterns

4.3 Results

Table 5 presents a performance comparison across the three methods evaluated: Direct JSON, Enhanced JSON, and Template-Based. We assessed performance using several metrics, including Data Structure Score, Subject-level Score, and Relationship Score. An Overall Quality score was also calculated.

As shown in the table, the Template-Based method consistently outperforms both the Direct JSON and Enhanced JSON methods across all metrics. The Template-Based approach achieved the highest Overall Quality score (0.70), followed by Enhanced JSON (0.63) and then Direct JSON (0.45). Specifically, the Template-Based method shows significant improvements in Data Structure Score (0.75), Subject-level score (0.70), and Relationship score(0.65) compared to the other methods. The Enhanced JSON method shows moderate improvements over the Direct JSON approach in all evaluated metrics.

Table 5: Performance Comparison Across Methods

Metric	Direct JSON	Enhanced JSON	Template-Based
Data Structure Score	0.52	0.68	0.75
Subject-level Score	0.45	0.63	0.70
Relationship Score	0.38	0.58	0.65
Overall Quality	0.45	0.63	0.70

4.4 Detailed Analysis

4.4.1 Structure Preservation

- **ADSL Structure:**
  - One-record-per-subject maintained: 98% (expected due to simple structure)
  - Key variable completeness: 85-92%
- **BDS Structure:**
  - Parameter-value pair validity: 65-75%
  - Visit pattern consistency: 55-70%
- **OCCDS Structure:**
  - Temporal sequence validity: 50-65%
  - Event relationship preservation: 45-60%

4.4.2 Method-Specific Findings

1. **Direct JSON Specification:** - Strong in basic variable generation - Weak in maintaining relationships - Limited complex structure preservation

2. **Enhanced JSON Specification:** - Improved relationship maintenance - Better handling of domain constraints - Moderate structure preservation

3. **Template-Based Generation:** - Best structure preservation - Stronger relationship maintenance - More consistent subject-level patterns

4.5 Comparison of Methods

Table 6 presents a comprehensive comparison of the three synthetic data generation approaches evaluated in this study. Our analysis reveals distinct characteristics and capabilities across multiple dimensions, from basic functionality to advanced features specific to ADaM dataset generation.

The comparison highlights several key distinctions:

- **Knowledge Integration:** While the Direct JSON approach relies solely on basic variable metadata, both Enhanced JSON and Template-Based methods incorporate sophisticated domain knowledge, with the Template-Based approach providing the most comprehensive integration through predefined patterns.

Table 6: Comparison of Synthetic Data Generation Methods

Feature	Direct JSON	Enhanced JSON	Template-Based
Knowledge Source	Basic variable metadata only	Domain knowledge + LLM suggestions	Predefined templates + Domain rules
Structural Info	Limited variable relationships	Explicit ADaM structure	Built-in structural patterns
Variable Relationships	Individual variable constraints	Cross-variable dependencies	Pre-designed relationship flows
Dataset Integration	Independent generation	Knowledge graph-based links	ADSL-centered generation
Data Realism	Moderate	High	Very High
Scalability	High	Medium	Medium
Maintenance	Simple updates	Requires knowledge base updates	Template updates needed
Use Cases	Simple datasets	Complex relationships	Production environments

- **Structural Handling:** Template-Based generation excels in maintaining ADaM-specific structures, particularly for complex patterns in BDS and OCCDS datasets. The Enhanced JSON approach shows significant improvement over Direct JSON but may require additional validation steps.
- **Dataset Integration:** The Template-Based approach's ADSL-centered generation strategy provides superior cross-dataset consistency, while Enhanced JSON leverages knowledge graphs for relationship maintenance. Direct JSON generation treats datasets independently, limiting relationship preservation.
- **Practical Considerations:** Each method presents distinct trade-offs between implementation complexity, maintenance requirements, and scalability. The Direct JSON approach offers simplicity but limited capabilities, while Template-Based generation provides comprehensive features at the cost of increased maintenance overhead.

This comparative analysis informs our subsequent detailed evaluation of each method's performance across specific metrics and use cases.

4.6 Limitations

While our approach demonstrates significant improvements in synthetic ADaM data generation, several important limitations warrant discussion:

1. **Temporal Relationship Complexity:**
   - Current methods achieve only 50-65% accuracy in temporal sequence validation
   - Complex event sequences, particularly in OCCDS datasets, show reduced consistency
   - Handling of time-dependent variable relationships remains particularly challenging in BDS datasets
2. **Cross-Dataset Consistency Challenges:**
   - Integration accuracy decreases notably as dataset complexity increases
   - Subject-level consistency drops from 98% in ADSL to 70-75% in complex BDS relationships
   - Maintaining treatment arm consistency across multiple datasets remains problematic
3. **Rare Event Pattern Generation:**
   - Current templates struggle to realistically reproduce low-frequency events
   - Risk of over- or under-representation of rare adverse events
   - Limited ability to maintain proper distribution of uncommon demographic combinations
4. **Flexibility vs. Constraint Trade-offs:**
   - Strict enforcement of ADaM rules can limit generation of edge cases

- Template rigidity may restrict adaptation to novel study designs
- Balance between maintaining structure and allowing necessary variations remains challenging

These limitations suggest several promising directions for future research, particularly in improving temporal relationship handling and rare event generation. Additionally, they highlight the need for careful consideration when applying these methods in different clinical trial contexts.

5 Discussion

The results of our experimental evaluation demonstrate several key findings that warrant detailed discussion. Our analysis reveals important insights into the effectiveness and limitations of different synthetic data generation approaches for ADaM datasets.

5.1 Template-Based Superiority

The template-based approach demonstrated consistent superiority across all evaluation metrics, with particularly strong performance in:

- Data structure maintenance (0.75 on our composite score)
- Subject-level pattern preservation (0.70)
- Cross-dataset relationship consistency (0.65)

This superior performance can be attributed to the structured nature of templates, which encode domain knowledge and relationships explicitly rather than relying on implicit connections in the data model.

5.2 Structural Integrity Analysis

Our analysis revealed varying degrees of success in maintaining structural integrity across different ADaM dataset types:

- **ADSL Structure:** All methods maintained basic subject-level structure effectively (> 90%), with the template-based approach achieving near-perfect preservation (98%)
- **BDS Structure:** Moderate success in maintaining parameter-value relationships (65-75%) and visit patterns (55-70%)
- **OCCDS Structure:** More challenging, with temporal sequence validity ranging from 50-65% and event relationship preservation at 45-60%

5.3 Performance Trade-offs

Each method presents distinct trade-offs that should be considered in implementation:

1. **Direct JSON Approach:**
   - *Advantages:* Simple implementation, rapid deployment
   - *Limitations:* Poor relationship preservation, limited structural complexity
2. **Enhanced JSON Approach:**
   - *Advantages:* Improved relationship handling, better domain alignment
   - *Limitations:* Increased implementation complexity, moderate scalability
3. **Template-Based Approach:**
   - *Advantages:* Superior structure preservation, robust relationship handling
   - *Limitations:* Higher initial setup cost, template maintenance overhead

5.4 Implementation Considerations

Several key factors emerged as critical for successful implementation:

- **Knowledge Integration:** Domain-specific knowledge significantly improves data quality and relationship preservation
- **Structure Optimization:** Careful attention to ADaM-specific structures is crucial for maintaining data integrity
- **Cross-Dataset Consistency:** Template design must explicitly account for inter-dataset relationships

6 Conclusion and Future Work

This paper has presented a comprehensive pipeline for generating synthetic CDISC ADaM datasets, successfully addressing key challenges in maintaining structural integrity and variable relationships. Our methodology introduces novel approaches to schema enhancement and template-based generation, resulting in significantly improved synthetic data quality.

6.1 Key Contributions

Our work has made several significant contributions to the field:

1. Development of a scalable pipeline for synthetic ADaM data generation incorporating domain knowledge and structural optimization
2. Introduction of novel metrics for evaluating synthetic clinical data quality, providing a comprehensive framework for assessment
3. Implementation of template-based generation strategies capable of handling complex data structures and relationships
4. Empirical validation of the superiority of template-based approaches for maintaining data integrity

6.2 Future Research Directions

Several promising areas for future research have emerged from this work:

1. **Template Automation:** Development of automated methods for template creation and validation for new ADaM structures
2. **Temporal Relationships:** Enhancement of temporal pattern preservation in complex longitudinal data
3. **Rare Event Patterns:** Improved methods for generating and validating rare event sequences
4. **Cross-Dataset Validation:** Development of more sophisticated methods for ensuring consistency across multiple related datasets

6.3 Practical Implications

The proposed pipeline provides a robust foundation for generating synthetic clinical trial data that maintains both structural integrity and realistic relationships. This advancement enables:

- Earlier commencement of software development and testing in clinical research
- Improved validation of statistical analysis programs
- Enhanced training capabilities for clinical data analysts
- Better support for privacy-preserving data sharing initiatives

The methodologies and findings presented in this paper represent a significant step forward in synthetic clinical data generation and highlight important areas for continued research and development in this critical field.

References

[1] Emilie Lundin Barse, Håkan Kvarnström, and Erland Jonsson. A synthetic fraud data generation methodology. 2002.

[2] Pengyue J. Lin, Behrokh Samadi, Alan Cipolone, Daniel R. Jeske, Sean Cox, Carlos Rendón, Douglas Holt, and Rui Xiao. Development of a synthetic data set generator for building and testing information discovery systems. In *Third International Conference on Information Technology*, 2006.

[3] Matthias Templ, Bernhard Meindl, Alexander Kowarik, and Olivier Dupriez. Simulation of synthetic complex data: The r package simpop. *Journal of Statistical Software*, 2017.

[4] Boyuan Ma, Xiaoyan Wei, Chuni Liu, Xiaojuan Ban, Haiyou Huang, Hao Wang, Weihua Xue, Stephen Wu, Mingfei Gao, Qing Shen, Adnan Omer Abuassba, Haokai Shen, and Yanjing Su. Data augmentation in microscopic images for material data mining. *arXiv preprint arXiv:1909.05824*, 2019.

[5] Núria Macià, Albert Orriols-Puig, and Ester Bernadó-Mansilla. Genetic-based synthetic data sets for the analysis of classifiers behavior. In *Eighth International Conference on Hybrid Intelligent*, 2008.

[6] Samuele Capobianco and Simone Marinai. Docemul: A toolkit to generate structured historical documents. *arXiv preprint arXiv:1709.08782*, 2017.

[7] Reihaneh Torkzadehmahani, Peter Kairouz, and Benedict Paten. Dp-cgan: Differentially private synthetic data and label generation. *arXiv preprint arXiv:2001.09700*, 2020.

[8] Steve Borkman, Adam Crespi, Saurav Dhakad, Sujoy Ganguly, Jonathan Hogins, You-Cyuan Jhang, Mohsen Kamalzadeh, Bowen Li, Steven Leal, Pete Parisi, Cesar Romero, Wesley Smith, Alex Thaman, Samuel Warren, and Nupur Yadav. Unity perception: Generate synthetic data for computer vision. *arXiv preprint arXiv:2109.11978*, 2021.

[9] Ryan McKenna, Gerome Miklau, and Daniel Sheldon. Winning the nist contest: A scalable and general approach to differentially private synthetic data. *arXiv preprint arXiv:2107.11873*, 2021.

[10] Hari Prasanna Das, Ryan Tran, Japjot Singh, Xiangyu Yue, Geoff Tison, Alberto Sangiovanni-Vincentelli, and Costas J. Spanos. Conditional synthetic data generation for robust machine learning applications with limited pandemic data. *arXiv preprint arXiv:2104.05309*, 2021.

[11] B. K. Beaulieu-Jones, Z. S. Wu, C. Williams, R. Lee, S. P. Bhavnani, J. B. Byrd, and C. S. Greene. Privacy-preserving generative deep neural networks support clinical data sharing. *bioRxiv*, 2018.

[12] R J Callahan, A Bogdanov, A J Fischman, T J Brady, and R Weissleder. Preclinical evaluation and phase i clinical trial of a 99mtc-labeled synthetic polymer used in blood pool imaging. *AJR. American Journal of Roentgenology*, 1998.

[13] Arthur M. Krieg. Therapeutic potential of toll-like receptor 9 activation. *Nature Reviews Drug Discovery*, 2006.

[14] Hobart W Harris, Frank Primus, Charlotte Young, Jonathan T Carter, Matthew Lin, Rita A Mukhtar, Benjamin Yeh, Isabel E Allen, Chris Freise, Esther Kim, Hani Sbitany, David M Young, and Scott Hansen. Preventing recurrence in clean and contaminated hernias using biologic versus synthetic mesh in ventral hernia repair: The price randomized clinical trial. *Annals of Surgery*, 2021.

[15] Benoit Baudry, Khashayar Etemadi, Sen Fang, Yogya Gamage, Yi Liu, Yuxin Liu, Martin Monperrus, Javier Ron, André Silva, and Deepika Tiwari. Generative ai to generate test data generators. *IEEE Software*, page 1–9, 2024.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

Jaime Yan
mingyu.yan1@merck.com

Chao Su
chao.su@merck.com