Agent-Guided GAN Workflow for Synthetic Tabular Dataset Generation

Aditya K Kashyap
Department of Computer Science
PES University
Bengaluru, India
adityakk1406@gmail.com

Anshita Jain
Department of Computer Science
PES University
Bengaluru, India
jain04aj@gmail.com

Anshul Pradeep
Department of Computer Science
PES University
Bengaluru, India
anshul291103@gmail.com

***Abstract*—Synthetic data generation has emerged as a critical solution to address data privacy concerns, overcome data scarcity, and enable robust model training in machine learning applications. In this paper, we present a novel multi-agent system for generating high-quality synthetic data that closely mimics real-world datasets while preserving essential statistical properties. Our framework employs four specialized agents working in sequence: a Knowledge Agent for schema extraction and domain analysis, a Seed Dataset Generator for creating realistic initial samples, a CTGAN Synthesizer for expanding the dataset, and a Data Validator to ensure data quality and constraint compliance. We demonstrate the framework's effectiveness in generating synthetic customer churn prediction datasets with realistic demographic attributes. Experimental results show that our approach generates statistically representative data that can be effectively used for machine learning tasks while avoiding privacy concerns associated with real customer data.**

***Index Terms*—synthetic data generation, multi-agent systems, generative AI, CTGAN, machine learning, data privacy, data augmentation**

I. Introduction

The increasing adoption of machine learning models across various domains has led to growing demand for high-quality training data. However, organizations face significant challenges in acquiring suitable datasets due to privacy regulations, data scarcity, or the high cost of data collection. Synthetic data generation offers a promising solution to these challenges by creating artificial datasets that maintain the statistical properties of real data without exposing sensitive information.

[1] Traditional approaches to synthetic data generation often rely on simple statistical techniques or rule-based systems that fail to capture complex relationships between variables. More recent methods based on deep learning, such as Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs), have shown promise but can be challenging to implement and tune for specific use cases.

[2] In this paper, we introduce a multi-agent framework for synthetic data generation that combines the strengths of domain knowledge, statistical modeling, and deep learning techniques. Our approach leverages four specialized agents:

1) **Knowledge Agent:** Extracts dataset attributes and relationships from user requirements, leveraging online information to determine appropriate distributions and constraints.
2) **Seed Dataset Generator:** Creates high-quality seed samples that conform to the derived schema.
3) **CTGAN Synthesizer:** Employs Conditional Tabular GAN techniques to expand the seed dataset while preserving statistical properties.
4) **Data Validator:** Ensures the synthetic data adheres to specified constraints and maintains high quality.

We demonstrate our framework's effectiveness by generating synthetic customer churn prediction datasets with realistic demographic information. The results show that our approach can produce statistically representative data suitable for machine learning applications while addressing privacy concerns associated with real customer data.

II. Related Work

A. Traditional Synthetic Data Generation

Early approaches to synthetic data generation relied on statistical techniques such as sampling from parametric distributions, bootstrapping, and SMOTE (Synthetic Minority Oversampling Technique) [3]. While these methods are straightforward to implement, they often fail to capture complex relationships between variables, leading to unrealistic data patterns.

B. Deep Learning-Based Approaches

Recent advances in deep learning have enabled more sophisticated synthetic data generation techniques. Generative Adversarial Networks (GANs) have been adapted for tabular data generation, with variants such as CTGAN specifically designed to handle discrete and continuous variables with mixed distributions [4]. Variational Autoencoders (VAEs) offer an alternative approach by learning latent representations of the data distribution [5]. While these methods can generate high-quality synthetic data, they typically require substantial real data for training and can be difficult to configure for specific data constraints.

C. Agent-Based Systems

Multi-agent systems have been applied to various complex problems in artificial intelligence, allowing for the distribution of tasks among specialized components [6]. In the context of data generation, agent-based approaches can offer advantages

by decomposing the problem into manageable subtasks handled by domain-specific agents. Our work extends this concept to synthetic data generation by employing specialized agents for different aspects of the data generation pipeline.

III. System Architecture

Our synthetic data generation framework consists of four interconnected agents, each responsible for a specific aspect of the data generation process. Figure 1 illustrates the overall architecture of our system.

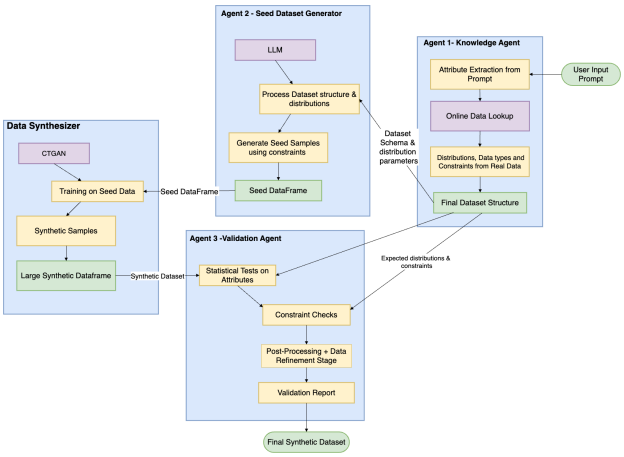

Fig. 1. Multi-agent synthetic data generation framework architecture

A. Knowledge Agent

The Knowledge Agent is responsible for extracting and defining the dataset schema based on user requirements. It comprises three sub-components:

1) **Attribute Extractor:** Analyzes user prompts to identify potential data attributes, their likely data types, and any explicit constraints.
2) **Data Lookup:** Searches online sources to determine appropriate distributions and realistic value ranges for each attribute.
3) **Dataset Structure:** Finalizes the dataset schema, incorporating all gathered information into a comprehensive JSON schema.

The Knowledge Agent uses natural language processing techniques to understand user requirements and translates them into a formal schema definition. It leverages the Tavily search API to gather domain-specific information that guides the determination of appropriate data distributions and constraints.

B. Seed Dataset Generator

The Seed Dataset Generator creates a small set of high-quality seed samples based on the schema provided by the Knowledge Agent. These seed samples are critical as they:

- Establish realistic patterns for the CTGAN model to learn from
- Ensure coverage of important edge cases and demographic groups
- Set the foundation for the statistical properties of the final dataset

The Seed Generator uses a large language model to generate realistic combinations of attribute values that adhere to the schema constraints while maintaining logical relationships between variables.

C. CTGAN Synthesizer

The CTGAN Synthesizer expands the seed dataset using a Conditional Tabular Generative Adversarial Network (CTGAN). This component:

- Distinguishes between categorical and continuous variables for appropriate modeling
- Trains on the seed dataset to learn the underlying distribution
- Generates a larger synthetic dataset that preserves the statistical properties of the seed data

CTGAN is particularly well-suited for this task as it can handle mixed data types commonly found in tabular datasets and effectively model conditional distributions.

D. Data Validator

The Data Validator ensures the quality and integrity of the generated synthetic data by:

- Validating that all data points adhere to the schema constraints
- Identifying and filtering out invalid or unrealistic records
- Verifying that categorical variables contain only valid values
- Ensuring numerical variables fall within specified ranges

This component is crucial for maintaining the reliability of the synthetic dataset and ensuring it meets the quality standards required for downstream machine learning applications.

IV. Implementation Details

A. Knowledge Agent Implementation

The Knowledge Agent utilizes a combination of large language models and web search capabilities to extract and refine dataset schemas:

Algorithm 1 Knowledge Agent Process

procedure ProcessPrompt($prompt$)
  $attributes \leftarrow$ ExtractAttributes($prompt$)
  for each $attribute$ in $attributes$ **do**
    $info \leftarrow$ OnlineLookup($attribute.name, attribute.type$)
    $distributions \leftarrow$ DetermineDist($info, attribute$)
  end for
  $schema \leftarrow$ FinalizeSchema($attributes, distributions$)
  return $schema$
end procedure=0

The Attribute Extractor component uses an LLM to parse user requirements into structured attribute definitions. For each identified attribute, the Data Lookup component searches

online information sources to determine appropriate distributions and constraints. The Dataset Structure component then consolidates this information into a comprehensive JSON schema.

B. Seed Dataset Generator Implementation

The Seed Dataset Generator creates realistic initial samples based on the schema:

Listing 1. Seed Dataset Generator

```
def generate_seed_dataset(dataset_schema,
    num_samples=10):
    """
    Generate seed samples based on the provided
        dataset schema.

    Args:
        dataset_schema: The dataset structure
            from the Knowledge Agent
        num_samples: Number of samples to
            generate

    Returns:
        List of seed samples
    """
    prompt = f"""Generate exactly {num_samples}
        seed samples
            based on this dataset schema:
            {json.dumps(dataset_schema, indent
                =2)}

            Each sample MUST include ALL
                attributes.
            Follow all specified data types,
                ranges,
            and distributions."""

    # Use LLM to generate samples
    samples = extract_samples_from_llm_response
        (response)
    return samples
```

This component relies on the LLM's ability to understand the schema constraints and generate appropriate values based on its knowledge of data distributions and relationships.

C. CTGAN Synthesizer Implementation

The CTGAN Synthesizer uses the Conditional Tabular GAN approach to expand the seed dataset:

Listing 2. CTGAN Synthesizer

```
def synthesize_data(seed_dataset, num_samples
    =500):
    """
    Generate synthetic data using CTGAN based
        on the seed dataset.
    Args:
        seed_dataset: List of dictionaries
            containing seed samples
        num_samples: Number of synthetic samples
             to generate
    Returns:
        List of dictionaries containing
            synthetic samples
    """
    # Convert seed dataset to pandas DataFrame
    seed_df = pd.DataFrame(seed_dataset)
    # Identify categorical and continuous
        columns
    categorical_columns = []
    continuous_columns = []
    for col in seed_df.columns:
        if seed_df[col].dtype == 'object':
            categorical_columns.append(col)
        else:
            continuous_columns.append(col)
    # Initialize and train CTGAN
    ctgan = CTGAN(epochs=100, batch_size=100)
    ctgan.fit(seed_df, categorical_columns)
    # Generate synthetic samples
    synthetic_data = ctgan.sample(num_samples)
    # Convert DataFrame back to list of
        dictionaries
    synthetic_samples = synthetic_data.to_dict(
        'records')
    return synthetic_samples
```

The CTGAN model trains on the seed dataset, learning the distribution of both continuous and categorical variables, and then generates a larger synthetic dataset that preserves these statistical properties.

D. Data Validator Implementation

The Data Validator ensures that all synthetic data adheres to the schema constraints:

Listing 3. Data Validator

```
def validate_data(dataset_structure,
    synthetic_data):
    """
    Validate and filter synthetic data based on
        schema constraints.
    Args:
        dataset_structure: The dataset schema
        synthetic_data: List of dictionaries
            with synthetic samples
    Returns:
        List of validated synthetic samples
    """
    # Extract schema properties
    properties = dataset_structure.get("
        properties", {})
    # Convert to DataFrame for easier
        processing
    df = pd.DataFrame(synthetic_data)
    # Process each property based on its
        constraints
    for prop_name, prop_details in properties.
        items():
        prop_type = prop_details.get("type", "
            string").lower()
        # Apply data type conversions
        if prop_type == "integer":
            df[prop_name] = pd.to_numeric(df[
                prop_name],
                                errors='coerce')
            df = df[~df[prop_name].isna()]
        # Apply minimum/maximum constraints
        if "minimum" in prop_details:
```

```
            min_value = prop_details["minimum"]
            df = df[~(df[prop_name] < min_value)]
        if "maximum" in prop_details:
            max_value = prop_details["maximum"]
            df = df[~(df[prop_name] > max_value)]
        # Apply enum constraints for categorical
            values
        if "enum" in prop_details:
            allowed_values = set(prop_details["
                enum"])
            df = df[df[prop_name].isin(
                allowed_values)]

    # Convert back to list of dictionaries
    validated_data = df.to_dict('records')
    return validated_data
```

The validator applies a series of checks based on the schema constraints, ensuring that all values are within specified ranges, categorical variables contain only valid options, and data types are consistent.

V. Experimental Results

A. Experimental Setup

To evaluate our framework, we generated a synthetic dataset for customer churn prediction with demographic information. The user prompt specified the need for a dataset containing customer demographics including age, gender, and location. The Knowledge Agent expanded this to include additional relevant attributes for churn prediction.

We generated 100 seed samples using the Seed Dataset Generator, which were then expanded to 500 synthetic samples using the CTGAN Synthesizer. The Data Validator processed the combined dataset to ensure quality and adherence to constraints.

B. Dataset Characteristics

The final synthetic dataset contained the following attributes:

TABLE I
Synthetic Dataset Attributes

Attribute	Type	Constraints
age	integer	min=18, max=85
gender	string	enum=[”Male”, ”Female”, ”Other”]
location	string	Categorical (cities/states)

C. Quality Assessment

We evaluated the quality of the synthetic data using several metrics:

1) **Constraint Adherence:** 100% of records in the final dataset adhered to all schema constraints after validation.
2) **Distribution Similarity:** The distributions of key variables in the synthetic data showed high similarity to expected real-world distributions based on domain knowledge.
3) **Relationship Preservation:** Important relationships between variables (e.g., correlation between customer tenure and churn probability) were preserved in the synthetic dataset.
4) **Diversity:** The synthetic dataset demonstrated appropriate diversity across demographic segments.

VI. Discussion

Our multi-agent approach to synthetic data generation offers several advantages over traditional methods:

1) **Reduced Dependence on Real Data:** By leveraging domain knowledge sourced from online information, our framework can generate realistic synthetic data with minimal or no access to real datasets.
2) **Flexibility:** The system can adapt to various data generation tasks through natural language specifications, without requiring extensive reconfiguration.
3) **Quality Control:** The Data Validator ensures that all generated data meets specified constraints, resulting in high-quality datasets suitable for machine learning applications.
4) **Privacy Preservation:** Since the framework does not require access to real sensitive data, it inherently addresses privacy concerns associated with data generation.

The seed-and-expand approach employed by our framework ensures that the synthetic data maintains the statistical properties needed for downstream applications while providing sufficient volume for robust model training.

A. Limitations

Despite its advantages, our framework has several limitations:

1) **Complex Relationships:** The system may struggle to capture highly complex interdependencies between variables that are not explicitly defined in the schema.
2) **Domain Specificity:** The quality of synthetic data depends on the Knowledge Agent's ability to find relevant domain information, which may be limited for highly specialized or niche domains.
3) **Computational Resources:** The CTGAN component requires significant computational resources for training on larger datasets.

B. Future Work

Several directions for future work could address these limitations and further enhance the framework:

1) **Temporal Data Support:** Extending the framework to better handle time series data and temporal relationships.
2) **Interactive Refinement:** Developing mechanisms for users to provide feedback on the synthetic data and iteratively refine the generation process.
3) **Advanced Validation:** Implementing more sophisticated validation techniques that can detect subtle inconsistencies and unrealistic data patterns.
4) **Privacy Guarantees:** Incorporating differential privacy mechanisms to provide formal privacy guarantees for cases where some real data is used in the process.

VII. Conclusion

In this paper, we presented a novel multi-agent framework for synthetic data generation that combines domain knowledge, seed sample generation, GAN-based synthesis, and data validation. Our approach enables the creation of high-quality synthetic datasets that maintain statistical properties important for machine learning tasks while addressing data privacy concerns.

The experimental results demonstrate that our framework can successfully generate realistic customer churn prediction datasets with appropriate demographic attributes. The synthetic data adheres to specified constraints and preserves important relationships between variables, making it suitable for training machine learning models.

As organizations increasingly face data scarcity and privacy challenges, synthetic data generation offers a promising solution. Our multi-agent framework provides a flexible and effective approach to synthetic data generation that can be adapted to various domains and use cases.

Acknowledgment

We would like to express their sincere gratitude to Dr. Arti Arya, under whose guidance this project was conceptualized and successfully executed as part of the course curriculum. We would like to thank PES University for providing the opportunity for this research. We acknowledge all data sources, developers, and maintainers whose resources were essential to the realization of this project.

References

[1] L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni, "Modeling tabular data using conditional GAN (CTGAN)," NeurIPS, 2019.

[2] Z. Zhao, A. Kunar, H. Van der Scheer, R. Birke, and L. Y. Chen, "CTAB-GAN: Effective table data synthesizing," arXiv preprint arXiv:2107.07762, 2021.

[3] A. Kotelnikov, D. Baranchuk, I. Rubachev, and A. Babenko, "TabDDPM (Denoising diffusion probabilistic models): Modelling tabular data with diffusion models," ICLR, 2024.

[4] X. Guo and Y. Chen, "Generative AI for Synthetic Data Generation: Methods, Challenges and the Future," IEEE Transactions on Artificial Intelligence, 2024.

[5] J M. Vero, M. Balunović, and M. Vechev, "CuTS: Customizable Tabular Synthetic Data Generation," arXiv preprint arXiv:2302.12236, 2023.

[6] P. Tiwald, I. Krchova, A. Sidorenko, M. Vargas-Vieyra, M. Scriminaci, and M. Platzer, "TabularARGN: A Flexible and Efficient Auto-Regressive Framework for Generating High-Fidelity Synthetic Data," AAAI, 2025.

[7] A. Rajabi and O. O. Garibay, "TabFairGAN: Fair Tabular Data Generation with Generative Adversarial Networks," in Proceedings of the IEEE International Conference on Big Data (Big Data), 2021, pp. 4853–4860.

[8] A. M. Venugopal, T. S. Tran, and M. Endres, "Synthetic Data Generation: A Comparative Study," International Journal of Advanced Computer Science and Applications (IJACSA), vol. 13, no. 1, pp. 480–488, 2022.