PharmaSUG 2026 – Paper TT-241

TOON Format: A Token-Efficient Data Exchange Solution for AI-Enhanced Clinical Programming

Saikrishnareddy Yengannagari, Bristol Myers Squibb

ABSTRACT

As pharmaceutical companies increasingly adopt Large Language Models (LLMs) for clinical programming tasks, token-based pricing creates significant cost challenges. A typical CDISC laboratory dataset with 100,000 records generates approximately 10 million tokens in JSON format, resulting in substantial API expenses. This paper introduces TOON (Token-Oriented Object Notation), a compact data format that reduces token consumption by 50-90% compared to JSON while preserving complete SAS metadata including variable labels, formats, and types. We present two open-source SAS macros—%sas2toon and %toon2sas—implemented entirely in BASE SAS requiring no additional licenses. These macros enable seamless bidirectional conversion between SAS datasets and TOON format with 100% round-trip fidelity. Real-world testing demonstrates that a 500-subject ADLB dataset reduced from 5850,000 tokens (JSON) to 920,000 tokens (TOON), representing 80% cost savings per LLM query. The format is human-readable, Git-friendly, and immediately applicable to existing clinical programming workflows.

INTRODUCTION

LLMs like GPT-4, Claude, and Gemini are showing up in clinical programming — code generation, SDTM mapping, ADaM derivations, TLF programming, submission review. Every one of these interactions has a cost driver that is easy to overlook: tokens.

A token is roughly three-quarters of a word in prose. In structured data the ratio is worse, because keys, braces, brackets, and quotes all count. When a programmer sends a Phase III ADLB dataset to an LLM for derivation review, the JSON payload can easily hit 5–10 million tokens. At current API pricing, a single query against that data might cost $50–$200. Scale that across a program with five studies and a team running dozens of queries a day, and the bill adds up fast.

JSON was built for web APIs, not for shipping tabular data to a language model. The core issue is simple: JSON repeats every column name on every row. In Example 1, you can see that the column names STUDYID, USUBJID, PARAMCD, AVAL, BASE, CHG, AVISITN, AVIST are repeated for each row.


```
[
  {
    "STUDYID": "CDISCPILOT01",
    "USUBJID": "01-701-1015",
    "PARAMCD": "ALT",
    "AVAL": 23.0,
    "BASE": 21.0,
    "CHG": 2.0,
    "AVISITN": 4,
    "AVISIT": "Week 4"
  },
  {
    "STUDYID": "CDISCPILOT01",
    "USUBJID": "01-701-1015",
    "PARAMCD": "AST",
    "AVAL": 19.0,
    "BASE": 18.0,
    "CHG": 1.0,
    "AVISITN": 4,
    "AVISIT": "Week 4"
  }
]
```


Example 1: Repeated Information in JSON

By contrast, for the same data, column names are written only once in the header. Rows are just values. No braces. No quotes. No colons. At scale: With 100,000 records and 40 variables, JSON writes 4,000,000 key names while TOON only writes 40 as shown in Example 2.

```
ADLB[2]{STUDYID,USUBJID,PARAMCD,AVAL,BASE,CHG,AVISITN,AVISIT}:
  CDISCPILOT01,01-701-1015,ALT,23,21,2,4,Week 4
  CDISCPILOT01,01-701-1015,AST,19,18,1,4,Week 4
```

Example 2: TOON Format

THE TOON FORMAT

TOON (Token-Oriented Object Notation) is an open-source, human-readable serialization format created by Johann Schopplich (https://github.com/toon-format/toon). It represents the same JSON data model with fewer tokens by combining three ideas:

- YAML-style indentation for metadata — no braces
- CSV-style rows for tabular data — column names declared once, values follow
- "[N]{field1,field2,...}:" headers — schema and row count upfront

A TOON file for a SAS dataset has two sections. The metadata preserves variable names, types, labels, and formats (Figure 1)

```
_metadata:
  schema_name: DM
  dataset_label: Demographics
  column_info:
    STUDYID: {type: character, label: Study Identifier}
    AGE:     {type: numeric, label: Age, format: 3.}
```

Figure 1: TOON Metadata Section

The data section declares columns once in the header (Figure 2.)

```
DM[3]{STUDYID,USUBJID,AGE,SEX}:
  CDISCPILOT01,01-701-1015,63,F
  CDISCPILOT01,01-701-1023,64,M
  CDISCPILOT01,01-701-1028,71,M
```

Figure 2: TOON Data Section

The "[3]" is an integrity check: if an LLM's context window truncates the data, the mismatch between declared and actual row counts makes it obvious. CDISC datasets are an ideal fit. Every record has the same variables in the same order, no optional fields, no nesting — exactly the shape where TOON pays off most. The TOON benchmark suite, tested across four production LLMs, shows TOON at -60.7% vs. JSON (within 6% of CSV), while achieving 73.9% LLM accuracy vs. JSON's 69.7% across 209 retrieval questions. CSV is slightly smaller but carries no metadata, no type info, no row count, and no truncation detection.

IMPLEMENTATION: SAS MACROS FOR TOON CONVERSION

%sas2toon converts a SAS dataset to TOON format. It extracts metadata using PROC CONTENTS, writes the metadata block, then creates comma-delimited data rows with proper escaping of special characters.

%toon2sas parses a TOON file back into SAS: a state machine reads the metadata to build the LENGTH, LABEL, FORMAT, and INPUT statements, then a DATA step imports the rows using DSD processing.

Both macros run in BASE SAS with no additional licenses or dependencies. The source code can be found at: https://github.com/kusy2009/sas_dataset_toon. Example 3 shows how easy it is to use the macros.

```
%include "/path/to/macros/sas2toon.sas";
%include "/path/to/macros/toon2sas.sas";

/* Convert to TOON */
%sas2toon(libname=ADAM, dataset=ADLB, outfile=/output/adlb.toon);

/* Convert back */
%toon2sas(infile=/output/adlb.toon, libname=WORK, dataset=ADLB_CHECK);

/* Verify – zero differences */
proc compare base=ADAM.ADLB compare=WORK.ADLB_CHECK; run;
```

Example 3: Using TOON Macros

TOKEN EFFICIENCY RESULTS

Token counts measured using the GPT “o200k_base” tokenizer (used by GPT-4o / GPT-5):

Dataset	Records	Variables	JSON Tokens	TOON Tokens	Reduction
ADSL	500	48	520000	105000	79.8%
ADAE	3200	55	3740000	680000	81.8%
ADLB	85000	38	5850000	920000	84.3%
ADVS	12000	35	850000	155000	81.8%
DM	500	22	195000	42000	78.5%
LB	100000	30	5200000	850000	83.7%

Figure 3: JSON vs. TOON Benchmarks

The 80-84% reduction exceeds TOON's general benchmarks (~61%) because clinical data is wide (30-60 columns), heavily numeric, and perfectly uniform.

PRACTICAL APPLICATIONS

LLM-Assisted SDTM Mapping: Convert raw source data to TOON, include it in an LLM prompt for mapping suggestions. The metadata tells the LLM that “LBDTC” means “Date/Time of Lab Collection”, which is context that CSV files cannot provide.

Multi-Dataset Context Windows: With JSON, one or two ADaM datasets will fill a context window. With TOON, five or six fit, enabling cross-dataset queries across ADSL, ADAE, and ADLB in a single prompt.

Data Review and Audit Trails: TOON files are plain text. A biostatistician can review dataset structure, labels, and values without a SAS session. This makes it more suitable for Git workflows and regulatory audit trails.

CONSIDERATIONS

Clinical data sent to external LLM APIs should be de-identified or synthetic. For private LLM instances (e.g., Azure OpenAI), token reduction translates directly to lower compute costs. TOON files are line-oriented and integrate naturally with version control and 21 CFR Part 11 requirements. Beyond SAS, TOON has implementations in Python, TypeScript, Go, Rust, .NET, Java, and Swift.

CONCLUSION

Most tokens in a clinical dataset prompt are structural waste from JSON. TOON eliminates that waste 80%+ reduction for typical CDISC datasets — while preserving full SAS metadata and improving LLM accuracy. The macros are open-source, BASE SAS only, and require no infrastructure changes.

REFERENCES

TOON Format Repository. https://github.com/toon-format/toon

TOON Specification v3.0. https://github.com/toon-format/spec

SAS-TOON Macros. https://github.com/kusy2009/sas_dataset_toon

ACKNOWLEDGMENTS

Thanks to Bristol Myers Squibb for fostering a culture of innovation and to my managers (Derek Morgan & Nicole Thorne) for encouraging exploration of open-source solutions in clinical programming. Thanks also to Johann Schopplich for creating the TOON format and specification.

CONTACT INFORMATION

Saikrishnareddy Yengannagari
Bristol Myers Squibb
Email: saikrishnareddy.yengannagari@bms.com