Paper CM04

Boosting SAS Programming Efficiency with ChatGPT: A Clinical Trials Perspective

Illia Skliar, Intego Group, Kharkiv, Ukraine

ABSTRACT

As the field of clinical trials continues to evolve, there is increasing interest in leveraging artificial intelligence (AI) and natural language processing (NLP) technologies to streamline programming tasks and improve efficiency. One promising technology in this space is ChatGPT®, an advanced language model that has shown impressive results in a variety of applications.

In this session, we will explore the potential of integrating ChatGPT into SAS/STAT® programming workflows for clinical trials. We will begin by providing a simple explanation of what ChatGPT is and how it works. From there, we will investigate how it can be integrated into SAS programming IDEs, including both data-based and non-data-based approaches.

Next, we will examine the specific tasks that ChatGPT can help with, including creating ADaM datasets based on CDISC standards, generating figures and tables using SAS 9.4 programming language, and more. We will also discuss the issue of restrictions related to confidential information and other limitations.

Finally, we will evaluate the accuracy and effectiveness of ChatGPT in programming tasks, drawing on real-world examples. By the end of this session, attendees will have a better understanding of the potential benefits and challenges of integrating ChatGPT into their SAS programming workflows for clinical trials.

INTRODUCTION

In the current landscape of technological innovations, there is growing interest in leveraging artificial intelligence (AI), in particular AI language models like ChatGPT, to streamline and boost various processes in SAS programming for clinical trials. This innovative approach has a potential to solve several of the current problems faced in this area, and can lead to a more efficient level of programming and improvement in data quality.

Clinical trials play a crucial part in medical research and the drug development process. One of the most critical aspects of clinical trials is data management, which involves collecting, storing, and analyzing massive amounts of data. This process is closely related to the use of programming languages such as SAS for data processing and statistical analysis. However, the complexity of clinical trial data and demanding regulatory standards stated by organizations such as FDA and ICH present significant challenges.

SAS programmers need to regularly create and validate datasets, generate tables, listings, and figures (TLFs), as well as perform complex statistical analysis and prepare submission-ready clinical study report (CSR) components. These tasks often require a significant manual effort, making the process time-consuming and likely to produce errors. Moreover, the expansion of global data standards such as CDISC only add complexity to the programming.

Moreover, based on a 2021 study by Tufts Center for the Study of Drug Development, it was estimated that the average cost of developing a prescription drug that successfully gains the market approval is $2.6 billion, where clinical trials accounting is a significant portion of this cost. Any inconsistency in the programming process could lead to delays, overstatement of expenditure, and, in the worst cases, compromise in the integrity of the clinical trial data, eventually blocking the developing of new treatments.

The purpose of this paper is to explore the potential of integrating AI language models, specifically ChatGPT, into SAS programming workflows for clinical trials, aiming to highlight the possible benefits, challenges, and the practical aspects of using this technology in a real-world clinical trial context.

The integration of AI and language models such as ChatGPT in workflow represents a promising solution to enhance the efficiency and accuracy of SAS programming for clinical trials. With an ability to understand human-like text, these models can assist in a wide range of tasks, from generating code fragments to assisting in error handling.
Going deeper into this topic, the paper will present a comprehensive picture of how ChatGPT can potentially revolutionize SAS programming in the clinical trials domain.

UNDERSTANDING CHATGPT

ChatGPT is an advanced language model that uses a deep learning technique known as a transformer neural network. It is part of the GPT (Generative Pre-Trained Transformer) family of models developed by OpenAI. ChatGPT is designed to generate human-like text based on the patterns and knowledge it has gained from huge amounts of training data.

At its core, ChatGPT uses a neural network architecture called the transformer. This architecture enables the model to process and generate text by understanding the relationships between words, phrases, and sentences in a given context. The transformer model employs a mechanism called self-attention, which allows it to weigh the importance of different words or tokens within a sentence while considering the entire input sequence.

The training process of ChatGPT involves exposing the model to a large corpus of text data, such as books, articles, and online content. By predicting the next word or token in a sequence, the model learns the statistical patterns and dependencies in the text. This pre-training phase helps ChatGPT to obtain extensive understanding of the language and enables it to generate consistent and semantically appropriate responses.

To make ChatGPT more applicable to specific tasks or domains, such as SAS or R programming in the field of clinical trials, fine-tuning is performed. During fine-tuning, the model is trained on a narrower dataset related to the target domain. This fine-tuning process helps the model to adjust the responses with the specific language and requirements of the clinical trials domain.

It is important to understand that ChatGPT doesn't possess embedded knowledge of a specific programming languages such as SAS or R. Instead, it learns statistical patterns and associations from training data, which includes a wide range of text from various sources. Consequently, it could generate text that would be presented as a programming code or provide programming-related suggestions based on its acquired patterns.

In simpler terms, ChatGPT works as a highly intelligent text generator that has been trained on a massive amount of text data. It has learned to understand language patterns and can generate responses that are contextually relevant. While it doesn’t have direct knowledge of SAS or R programming, it can still provide assistance by suggesting code fragments, offering guidance, or helping with a general programming task.

In the context of clinical trials, integrating ChatGPT into SAS programming workflows can potentially help rationalize programming tasks, offer insights, and provide efficient code generation. However, it is important to note that ChatGPT should be seen as a tool to boost and assist programmers rather than as a replacement for human expertise. It's crucial to review and validate the generated code or suggestions to ensure accuracy and compliance with regulatory standards.

By leveraging the capabilities of ChatGPT, programmers involved in the field of clinical trials can benefit from its language generation abilities, potentially improving efficiency and productivity in their programming tasks.

KEY FEATURES

- Natural Language Understanding: ChatGPT is able to understand and generate human-like text. It can understand the context, meaning, and nuances of natural language input, making it effective for interacting with SAS programmers.

- Code Generation: ChatGPT can assist SAS programmers by generating code fragments or providing suggestions for programming tasks. It has been trained on a diverse range of text data, including programming-related content or proposing programming solutions.

- Assistance in debugging: SAS programmers often cope with warnings or errors during their programming tasks. ChatGPT can provide assistance in debugging by analyzing error messages, understanding the code context, and suggesting potential solutions.

- Knowledge Repository: ChatGPT can serve as a knowledge repository, storing information on best practices, programming techniques, and commonly used SAS functions or procedures. SAS programmers can query ChatGPT to find relevant programming answers, reducing the need for long manual searching.

- Improved Efficiency and Productivity: By leveraging ChatGPT's features, SAS programmers are able to reduce time and effort spent on routine programming tasks. This can lead to increased efficiency, allowing programmers to focus on more complex or critical aspects of their work.

- Accessibility and User-Friendly Interaction: ChatGPT can be accessed through user-friendly interfaces or chat-based platforms, enabling SAS programmers to interact with it using conversational language.

This makes it easier for programmers, even for those without advanced technical skills, to leverage its assistance effectively.

DEVELOPMENT HISTORY

The development history of ChatGPT and its various versions shows the continuous improvement across various sectors. Here is an overview of its versions:

- GPT-1: The original version of GPT, released by OpenAI in 2018, laid the foundation for subsequent iterations. It introduced the concept of transformer-based models for natural language processing tasks. While it showed promise, it also had limitations in generating coherent and contextually accurate responses.
- GPT-2: Released in 2019, GPT-2 was a significant leap forward in language generation. It was a larger model trained on a massive dataset and exhibited improved capabilities in generating more coherent and contextually relevant text. However, due to concerns about potential misuse, OpenAI initially limited access to the full model.
- GPT-3: Launched in 2020, GPT-3 made substantial strides in language generation and demonstrated remarkable performance on a wide range of tasks. It was trained on an enormous dataset and introduced several innovations, such as multi-modal capabilities and the ability to perform zero-shot and few-shot learning. GPT-3 received extensive attention and adoption in various sectors due to its impressive capabilities.
- ChatGPT: Building upon the successes of GPT-3, OpenAI released ChatGPT in 2021. It was specifically designed to facilitate conversational interactions and provide more interactive and dynamic responses. ChatGPT featured improvements in handling prompts, better context sensitivity, and enhanced control over generated outputs.

Throughout its development OpenAI has fine-tuned the training methodology, increased model sizes, and handled limitations to improve the quality and reliability of generated text.

This evolution points out the significant impact of ChatGPT and its potential to transform various industries, including SAS programming in the field of clinical trials.

As SAS programmers explore the integration of ChatGPT into their workflows, they can benefit from the progress through the development of ChatGPT and its various versions. This continuous improvement paves the way for more effective language models that can assist SAS programmers in clinical trial programming tasks with increased accuracy and efficiency.

INTEGRATION OF CHATGPT IN SAS PROGRAMMING IDE

Efficiently integrating ChatGPT into the SAS programming environment can create a valuable opportunity to boost productivity and programming experience. There are several practical approaches that SAS programmers can explore to incorporate ChatGPT's language generation capabilities into their workflow:

- Chat Interface within IDE: One effective method involves integration a chat interface directly into the SAS programming Integrated Development Environment (IDE). This allows programmers to interact with ChatGPT in real time while they're writing code. This interactive chat functionality lets programmers ask questions, seek code suggestions, or request help on specific coding tasks, all without having to switch between different tools.

- API-Based Integration: Another approach is integrating ChatGPT through APIs (Application Programming Interfaces). This makes API calls to ChatGPT from the SAS programming environment. Programmers can input their queries or coding and receive back generated code pieces or explanations. This approach offers flexibility, enabling programmers to integrate ChatGPT into various SAS tools and interfaces.

- Custom Plugins and Extensions: Developers can create custom plugins or extensions for SAS programming tools that incorporate ChatGPT's capabilities. These plugins could offer features like code completion tailored to context, automated code generation based on requirements, or even explanations for code sections in real time. Such customized extensions improve the SAS programming environment by providing intelligent assistance directly within the IDE.

- Guided Programming Workflow: ChatGPT can serve as a guide throughout the programming process. Programmers can outline their intended logic or coding needs, and ChatGPT can generate a corresponding code, suggest optimization steps, or even identify potential errors. This approach can help both senior and less experienced programmers in creating an effective SAS code.

- Contextual Documentation and Help: Integrating ChatGPT for contextual documentation and assistance is also valuable. When programmers are faced with unfamiliar functions or syntax, they can seek explanations, examples, or best practice suggestions from ChatGPT. This bridges knowledge gaps and offers immediate help within the programming environment.

- Automating Task Generation: SAS programmers can automate the generation of common programming tasks. For instance, programmers can describe the desired analysis or output, and ChatGPT can produce the required code. This approach accelerates the programming process, minimizing manual coding efforts.

- Interactive Debugging Assistance: Programmers can describe encountered errors, and ChatGPT can provide insights into potential causes and solutions. This approach optimizes the debugging process, reducing debugging time.

Selecting the most suitable integration approach depends on such factors as desired interaction levels, depth of assistance, costs of integration and compatibility with existing SAS tools. Despite which approach is used, the ultimate objective remains the same: using ChatGPT's capabilities to efficiently generate code, offer suggestions, and help in problem-solving within the clinical trials field.

EXAMPLES AND CODE SNIPPETS TO ILLUSTRATE THE INTEGRATION PROCESS

In the upcoming sections, we will explore one of the commonly used methods of ChatGPT directly in SAS IDE. We can use both SAS Studio or SAS desktop version. In the example SAS Studio usage will be provided.

To get started, visit the link provided at platform.openai.com. Create an account using your Google or Microsoft email address. Once your account is set up, the most crucial step is acquiring a secret API key that allows access to the API. Make sure to copy and securely store this API key for future use.
In your SAS program, you will be working with two macro variables: **&API_KEY** and **&QUESTION**. The **&API_KEY** macro variable will hold your Secret API key, while the **&QUESTION** macro variable will capture the user's input, which serves as a question or search query.

```
%macro RunGPTquery(query =);
%let API_KEY= sk-XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
%let QUESTION = %str(%"&query.%");

/* Add model to request */
filename in temp;
data _null_;
file in;
put;
put "{";
put  '"model": "gpt-4", "messages": [{"role": "user", "content": '"&question }]";
put "}";
run;
/* Reference the file as in */
filename resp "%sysfunc(getoption(WORK))/echo.json";
proc http
 method="POST"
 url="https://api.openai.com/v1/chat/completions"
 ct="application/json"
 in=in
 out=resp;
 headers "Authorization" = "Bearer &api_key.";
run;
/* Parsing the response and output */
libname response JSON fileref=resp;

proc report data= response.choices_message;
column content;
define content / display "" style(column)=[cellwidth=6in fontsize=10pt asis=ON];
run;
%mend RunGPTquery;
```

Please ensure that the secret API key is not enclosed in quotation marks. Additionally, retain the percentage (%) signs and quotes within the **&QUESTION** macro variable as they are essential components for its proper functioning. If a new model of Chat GPT is released, then change gpt-4 to the new model name in the code above.

The next step is to include the macro code from the external file and call it with a query.

```
%include "/home/u43526594/SAS & ChatGPT/RunGPTquery.sas";
%RunGPTquery(query = SAS code to create boxplot by treatment group and save it to .pdf
format);
```

After executing the code, the output will be generated with the response of the query.

```
Here is an illustrative example of how you might generate the boxplot as requested:
```sas
ods graphics on;
ods pdf file="c:/my_folder/my_boxplot.pdf";
proc sgplot data=my_dataset;
   vbox my_variable / category=treatment_group;
   title "Boxplot by Treatment Group";
run;
ods pdf close;
ods graphics off;
```
```

ChatGPT also provides the information below:

```
In the above code:

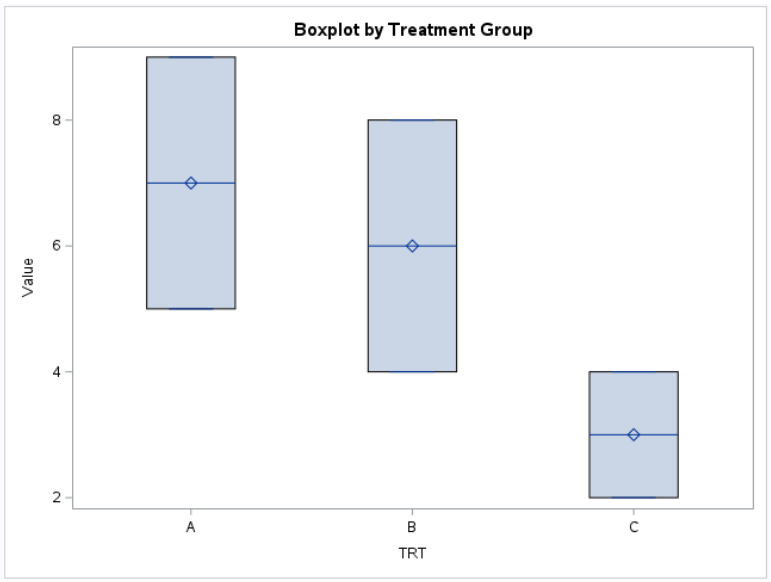
1. `ods graphics on;` enables high-quality graphics
2. `ods pdf file=` starts sending all output to a PDF file
3. `proc sgplot data=my_dataset;` is the start of the SGPLOT procedure where you speci
fy the dataset (`my_dataset`) you would like to use.
4. `vbox my_variable / category=treatment_group;` generates a box plot (`vbox`) for a
variable (`my_variable`) split over categories of another variable (`treatment_group`)
.
5. `title "Boxplot by Treatment Group";` sets a title for the box plot.
6. `run;` indicates the end of the procedure.
7. `ods pdf close;` stops sending output to the PDF file
8. `ods graphics off;` disables high-quality graphics.

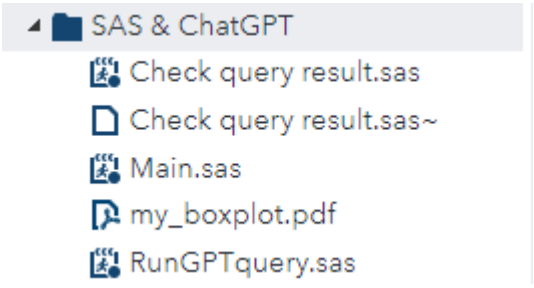
Please replace `my_folder`, `my_boxplot.pdf`, `my_dataset`, `my_variable` and
`treatment_group` with the actual folder's path, pdf's name, dataset name, variable
name and treatment group's name.
```

By executing the ChatGPT suggested code SAS produces boxplot and saves it by the specified path.

```
/* Example data */
data treatment;
    input TRT $ Value;
    datalines;
A 5
A 7
A 9
B 4
B 6
B 8
C 2
C 3
C 4
;
```

```
/*ChatGPT code*/
ods graphics on;
ods pdf file="/home/u43526594/SAS & ChatGPT/my_boxplot.pdf";
proc sgplot data=treatment;
   vbox Value / category=TRT;
   title "Boxplot by Treatment Group";
run;
ods pdf close;
ods graphics off;
```





Integrating ChatGPT into the SAS environment through IDE offers a number of advantages such as real-time assistance, efficiency, and contextual relevance. However, programmers must be aware of accuracy, potential connectivity issues, domain limitations, and data privacy concerns.

By understanding these pros and cons, SAS programmers can make informed decisions about when and how to leverage ChatGPT's capabilities effectively within the clinical trials programming context.

TASK RESOLUTION WITH CHATGPT

The Clinical Data Interchange Standards Consortium (CDISC) standards have long been the key point of clinical research, ensuring consistency, clarity, and streamlined data exchange. Both Analysis Data Model (ADaM) and Study Data Tabulation Model (SDTM) stand out as crucial for representing data, statistical analysis and reporting.

ChatGPT can be used to generate, review, and advise on the creation of SDTM and ADaM datasets, ensuring they remain in line with CDISC standards. The potential of such integration can radically reduce the time, effort, and potential errors associated with manually handled standard datasets.

Below, we examine the interaction between ChatGPT and SAS programming by checking if all the required variables are present in the SDTM LB dataset, as per the SDTM Implementation Guide (two required variables were previously dropped).

```
libname SDTM "/home/u43526594/SDTM";
proc contents data = SDTM.LB(drop = LBSEQ LBDTC)
 out = LB_VARS(keep = name label) noprint;
run;
data _null_;
  set LB_VARS end=last;
  length var_list $2000;
  retain var_list '';
  var_list = catx(' ', var_list, name);
  if last then call symputx('var_list', var_list);
run;
%put &var_list.;
```

```
%RunGPTquery(query = check if all the REQUIRED variables according to SDTM IG v.3.3
are present in my LB dataset. The variables in my LB dataset are as follows:
&var_list.);
```

Here is the result of the query:

Comparing this list to your dataset, it seems you're missing the LBSEQ variable which is crucial for the order of observations.

Also, LBDTC (Date/Time of Observation/Collection) is a required variable and needs to be included even though you have LBDY

The creation of specifications is also the main part of clinical data programming daily routine, whether it is producing Analysis Data Model (ADaM) datasets or Study Data Tabulation Model (SDTM) datasets.
It often requires deep logic and a solid understanding of the purpose and structure of the dataset to create derived variables. The SAS code used to generate these variables can be provided in natural language via ChatGPT. This helps to make sure that both technical as well as non-technical users will understand the specification documentation.

Imagine that, while working with the ADAE specification, you are dealing with this piece of code and wish to create a derivation for the ASTDY variable:

```
/**/
ASTDY=ASTDT-TRTSDT+(ASTDT>=TRTSDT);
/**/
%let code = %nrstr(ASTDY=ASTDT-TRTSDT+(ASTDT>=TRTSDT));

%RunGPTquery(query = Could you create derivation for my specification using this code
&code. and check it with IG compliance);
```

The Result:

```
ASTDY is derived based on the input variables ASTDT (start date of abnormal status)
and TRTSDT (start date of treatment).

If ASTDT is not missing and TRTSDT is not missing, ASTDY is the difference in days
between ASTDT and TRTSDT. If ASTDT is on or after TRTSDT, a day is added to that
difference.

Here is a step-by-step process, and its compliance with CDISC/CDASH standards for
clinical trials (IG = Implementation Guide).

1. First, check that ASTDT (start date of abnormal status) and TRTSDT (start date of
treatment) are not missing. According to CDISC/CDASH standards, dates and times should
be input in ISO 8601 format (YYYY-MM-DD or YYYY-MM-DDThh:mm:ss). The derivation cannot
be performed if either of these variables is missing.

2. Subtract TRTSDT from ASTDT to get the difference in the number of days.

3. Check if ASTDT is on or after TRTSDT. If it is, add one (+) to the difference in
days. If it is not, don't add anything.

4. The result is ASTDY, the number of days from the start date of treatment to the
start date of the abnormal status. If ASTDT is the same as or later than TRTSDT, it is
counted as an additional day.

In terms of the IG compliance, this calculation is generally compliant with the
CDISC/CDASH standards. The calculation is transparent, and the resulting derived
variable (ASTDY) makes sense in the clinical context.

However, it's always important to clearly define all variables and coding in a study
protocol and Statistical Analysis Plan (SAP) to ensure full transparency and
compliance with the IG' recommendations. Performance of this calculation and its
incorporation into analysis should be fully explained in the study documentation.
```

The derivation that ChatGPT earlier presented will be examined in more detail in the following part. The objective is to ensure that the ASTDY derivation we received earlier is both accurate and effective. After that, request ChatGPT to write a SAS code based on the rules that we've established. Before we use our derived definitions in the actual data analysis, we will go through this back-and-forth process, used as a quality check, to ensure that the results are accurate.

```
%let rules = %nrstr(
If ASTDT is not missing and TRTSDT is not missing, ASTDY is the difference in days
between ASTDT and TRTSDT. If ASTDT is on or after TRTSDT, a day is added to that
difference.
Here is a step-by-step process, and its compliance with CDISC/CDASH standards for
clinical trials (IG = Implementation Guide).
1. First, check that ASTDT (start date of abnormal status) and TRTSDT (start date of
treatment) are not missing.
According to CDISC/CDASH standards, dates and times should be input in ISO 8601 format
(YYYY-MM-DD or YYYY-MM-DDThh:mm:ss).
The derivation cannot be performed if either of these variables is missing.
2. Subtract TRTSDT from ASTDT to get the difference in the number of days.
3. Check if ASTDT is on or after TRTSDT. If it is, add one (+) to the difference in
days. If it is not, don't add anything.
4. The result is ASTDY, the number of days from the start date of treatment to the
start date of the abnormal status.
If ASTDT is the same as or later than TRTSDT, it is counted as an additional day.
);
%put &rules.;

%RunGPTquery(query = create SAS code to derive ASTDY variable using this rules:
&rules.);
```

The Result:

````
Following these instructions, SAS code that meets the CDISC/CDASH standards for
clinical trials could look like this:
```SAS
DATA new_dataset;
SET old_dataset;
*Check that ASTDT and TRTSDT are not missing;
IF NOT (MISSING(ASTDT) OR MISSING(TRTSDT)) THEN DO;
    *Calculate the difference between ASTDT and TRTSDT in days;
    ASTDY = ASTDT - TRTSDT;

    *Check if ASTDT is on or after TRTSDT. If it is, add one to the difference in
days;
    IF ASTDT >= TRTSDT THEN ASTDY = ASTDY +1;
END;
RUN;
```
````

An accurate code was successfully produced with ChatGPT. Additionally, it has checked any missing data and added to ASTDY one day if TRTSDT occurred after ASTDT. Furthermore, ChatGPT included comments in the provided code.

This section assessed ChatGPT's ability to support SAS programming in clinical trials. We demonstrated how it is able to create new variables, examine data to make sure it compliance with SDTM and ADaM standards, and even write a code based on the specifications of the study. These features boost our work and can improve the quality of the data we use in clinical trials.

But it is crucial to keep in mind that ChatGPT was created to assist people, not to carry out tasks autonomously or interact with machines.

Even though ChatGPT is useful, you should always be careful with its output. Just like any tool, there might be a chance of error. That's why you must always double-check its work with the official study documents and guidelines.

All the output results need to be checked thoroughly to make sure they meet the strict clinical requirements and CDISC standards.

So SAS programming for clinical trials can greatly benefit from ChatGPT. But rather than completely replacing human effort, its assistance should be supplemented. By being aware of these specifics, SAS programmers may choose carefully how to use ChatGPT to benefit from their work while also taking care of keeping everything accurately and in accordance with the necessary standards.

ADDRESSING RESTRICTIONS AND CONFIDENTIALITY

The protection of sensitive data becomes an important concern in the goal of integrating AI, in particular ChatGPT, into SAS programming for clinical trials. In order to guarantee compliance and achieve greater efficiency, this issue needs to be carefully examined according to the strict rules and regulations that control clinical trials. The following section outlines the limitations, difficulties, and potential safeguards related to handling confidential data when using ChatGPT or other AI language models.

REGULATORY FRAMEWORK

Guidelines for Good Clinical Practice (GCP), ICH E6(R2): These regulations are essential for guaranteeing the safety of human volunteers for testing and the quality of clinical trial data. The confidentiality of trial subjects' data is a crucial requirement which may be difficult to implement when using AI because such models as ChatGPT need data access for processing and analysis.

Working with AI models, programmers need to pay a great amount of attention to the following ICH E6(R2) regulation rules:

1. Data Integrity and Quality: recommendations focus a strong emphasis on the necessity of making sure that data are accurate, comprehensive, and consistent during each step of clinical trials. It is essential to make sure that the language model doesn't compromise the quality and integrity of the data while using AI.

2. Data protection and confidentiality: It is crucial for protecting the confidentiality of trial participants' personal data. Integrating AI must follow the rules for protecting sensitive data, which may include encryption, access restrictions, or anonymization.

3. Audit Trails: Providing audit trails for recording data modifications is required by ICH E6(R2). This requires a method within the AI integration layer to log and keep track of all data interactions.

4. Validation of Computerized Systems: To guarantee accuracy, reliability, and consistent intended performance, the guidelines require the validation of computerized systems used in clinical trials. AI systems related to SAS programming are also subject to this certification.

The Code of Federal Regulations Title 21 (CFR Title 21) which is a significant document of regulations administered by the FDA also adds essential requirements that guarantee safety, efficacy and security of clinical trial.

Below are some additional regulations provided by this document:

1. Record Retention: According to the regulations, electronic records must be preserved in a way that makes it possible to access them at any time within the records retention term. Integrations of AI shouldn't make it harder to store and retrieve data.

2. Ensuring Complete Data: The importance of keeping the complete data collected through all the tests, examinations, and assays is emphasized in order to guarantee compliance with set the specifications and standards.

3. Data Review: It is necessary to regularly check data for completeness, quality, and consistency with the accepted standards. AI tools such as ChatGPT should promote this process rather than hinder it.

There are also other regulations that always should be taken into consideration while integrating any new instruments into a clinical trial perspective: Electronic Records, Electronic Signatures, Protection of Human Subjects, Institutional Review Boards, Quality System Regulation, Investigational Device Exemptions, Financial Disclosure by Clinical Investigators and others.

OTHER INSTRUMENTS TO PROVIDE SECURITY OF CONFEDINTIALITY

The use of data anonymization and pseudonymization procedures, which are essential for hiding identifying information, is key to protecting data confidentiality. Although crucial, these methods come with difficulties, such as the possible loss of data utility and the difficulty of successfully anonymizing data. Strong data encryption and access control procedures are also required to guarantee that sensitive data are safeguarded and only accessible by authorized persons and systems.

Clarifying how the data are used and processed through the implementation of model transparency and auditing systems can help in keeping up with regulatory requirements. Furthermore, if ChatGPT services are hosted on cloud platforms, it is vital to verify the vendor and cloud service compliance. Vendors are required to act in accordance with the rules and regulations controlling clinical trial data.

A legal protection is created by clearly defining the responsibilities, liabilities, and data handling procedures between the parties in contractual agreements. A structure for ongoing monitoring and evaluation supports this legal clarity by ensuring constant compliance to the confidentiality standards and a timely response to any new issues.

Building trust and making sure that the integration of AI technology goes smoothly depend on involving and informing stakeholders about the value of confidentiality and the protective measures in place. Finally, to ensure that the integration of ChatGPT into SAS programming workflows for clinical trials remains compliant and successful, it is crucial to stay up-to-date on future regulatory developments and adapt AI integration approaches accordingly.

CONCLUSION

The efficacy and accuracy of incorporating ChatGPT into SAS 9.4 programming procedures, particularly in the context of clinical trials, have been explored in this research. The results highlight ChatGPT's value in automating routine programming activities, increasing productivity and lowering human error. The model specifically shows significant potential in query creation, data annotation, and even in the complex processes of statistical analysis.

Using ChatGPT in SAS programming procedures for clinical trials could have profound effects. For instance, it could completely change how data are collected and decrease the time spent on the clinical study. Clinical programmers are able to concentrate more on experimental design, data interpretation, and ethical issues by automating some data processing and analysis processes. This increase in productivity is more than simply a convenience; it could also boost the development of medicines and their introduction into the market.

It is important to recognize that ChatGPT has limits even though it shows a lot of potential. These include concerns about data privacy and the model's current inability to properly understand domain-specific nuances. As a result, future research should focus on improving the model's understanding of statistical and medical language and putting in place strong security safeguards.

Taking into account encouraging results and the quickly changing field of AI and NLP technologies, it is crucial for people involved in clinical trials and SAS programming to stay up to date on these developments. For accurate and morally correct AI integrations, a diverse approach of statisticians, data scientists, and healthcare experts may be the solution.

I am glad to encourage the audience to actively explore the integration of AI language models like ChatGPT into their development processes rather than simply take the information in. Clinical research has the potential to reach new heights thanks to the collaboration between human expertise and artificial intelligence.

REFERENCES

1) Drug development workforce in the age of digital transformation, 2021
   URL: csdd.tifts.edu/white-papers/drug_development_workforce_in_the_age_of_digital_transformation

2) Benchmarking and optimizing the process for adopting innovations supporting clinical trial execution, 2022
   URL: csdd.tifts.edu/white-papers/benchmarking_and_optimizing_the_process_for_adopting_innovations

3) OpenAI platform documentation
   URL: platform.openai.com/docs/quickstart/build-your-application

4) E6(R2) Good Clinical Practice: Integrated Addendum to ICH E6(R1) Guidance for Industry, 2018
   URL: fda.gov/media/93884/download

5) CFR – Code of Federal Regulations Title 21, 2023
   URL: accessdata.fda.gov/scripts/cdrh/cfdocs/cfcfr/CFRSearch.cfm

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Please contact the author at:

Illia Skliar
Statistical Programmer/Analyst
Intego Group, LLC
8 Manizer street
Kharkiv, Ukraine / 61000
Work Phone: +38 044 500 7020 ext. 2546
Email: Illia.Skliar@intego-group.com