PharmaSUG China 2022 - Paper AD-121

A Point-and-Click MedDRA Leveling Tool

Yanze Zhang, IMPACT Therapeutics

ABSTRACT

With fast development of clinical trials around the globe, there has been an increasing demand of integration of Adverse Event (AE) datasets to serve different purposes of safety analysis, such as Integrated Summary of Safety (ISS), Investigator Brochure (IB) update, and Development Safety Update Report (DSUR). Usually, the MedDRA versions used in pooled study data are different. Before pooling all AE datasets together, we need to upgrade the MedDRA version of the AE dataset from previous studies to the latest version or certain version, which could be called “MedDRA Leveling”. The owner of this process varies from company to company. For a small biotech company, due to the resource constraint, this mission may rest on the shoulders of talented statistical programmers. In this paper, a SAS macro decorated with a Point-and-Click Graphical User Interface (GUI) built by Python is introduced to complete this mission.

INTRODUCTION

MedDRA or Medical Dictionary for Regulatory Activities is developed by International Council for Harmonization of Technical Requirements for Pharmaceuticals for Human Use (ICH) and is widely used by regulatory authorities and pharmaceutical industry for safety monitoring and registration. MedDRA is maintained by Maintenance and Support Services Organization (MSSO) and updated twice a year (May and November). Thus, studies from different periods may use different versions of MedDRA. Within the same study the MedDRA version could also changes at long time span.

To understand the safety of a medical product better we are often required to pool safety data from different studies together. The most important requirement comes from regulatory authorities, such as Integrated Summary of Safety (ISS). It is crucial to use the same data standards and dictionary for those pooled data so that the integrated analysis leads to meaningful results. Among those standards and dictionaries, MedDRA is a key dictionary that needs to be leveled. MedDRA coding is commonly applied in following datasets following CDISC standards: SDTM dataset – AE, PR and their ADaM counterparts – ADAE, ADPR. Adverse Event (AE) terms are always coded using the latest MedDRA version. Due to the complexity of MedDRA hierarchy and size of the data, it is cumbersome for medical coder to update the coded terms manually. For a small-sized biotech if this task is outsourced it will bring extra financial burden to the company. Whether this mission can be accomplished by statistical programming with user-friendly interface is worthy of exploring.

The plan is complete the mission in two steps:

- Step 1: Develop a SAS macro to update coded terms in AE SAS dataset to a new MedDRA version
- Step 2: Build GUI to pass entries to SAS macro variables with the help of Python packages

But before that, the first key step is to understand MedDRA hierarchy.

MEDDRA HIERARCHY

MedDRA terminology has five levels of structural hierarchy: Lowest Level Terms (LLT), Preferred Terms (PT), High Level Terms (HLT), High Level Group Terms (HLGT) and System Organ Class (SOC). The relationship between different levels is illustrated in Figure 1. MedDRA Terminology Hierarchy. An LLT can be linked to only one PT. a PT can be linked to more than one HLTs. An HLT can only be linked to a particular SOC via one route (i.e., linked to only one HLGT per SOC). An HLGT can be linked to different SOCs. The relationship leads to multiple paths from an LLT to an SOC. One of them is the primary path which corresponds to the primary SOC linked to a PT. As Figure 1 shows, PT2 is linked to SOC1 and SOC2 via different paths. The path highlighted in red is the primary path from staring point of LLT3 to the ending point of SOC2.

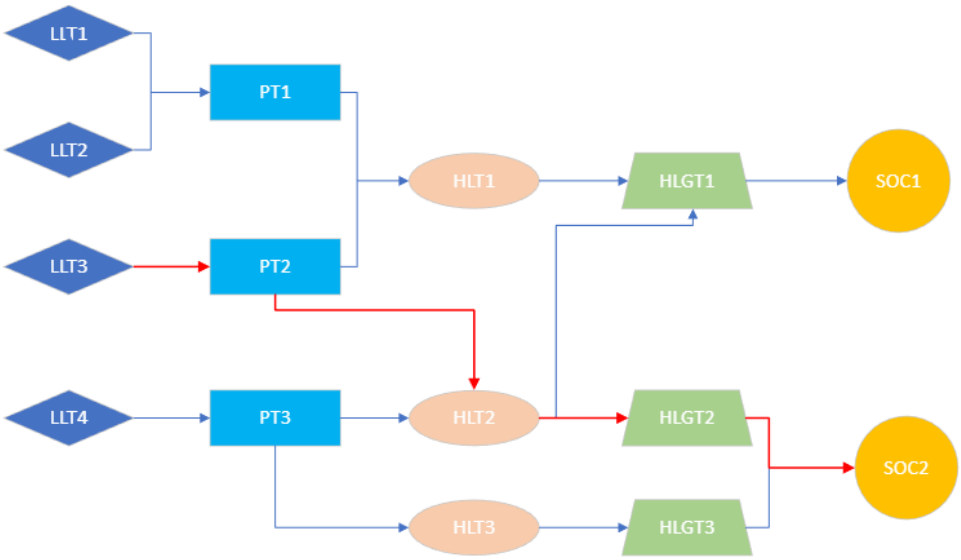


Figure 1. MedDRA Terminology Hierarchy

With the MedDRA hierarchy in mind, there are naturally two options to perform MedDRA leveling: using only the primary SOC path or not. Let's assume LLTs are not going to change and other terms (PT, HLT, HLGT and SOC) need to be updated, since AE verbatim terms are directly coded with LLTs, and the chance of changing LLT to a new one is very low. If only the primary SOC path is selected, LLTs in dataset of starting version of MedDRA can be easily joined with the ending version MedDRA dictionary and terms of the other levels are populated with the new terms along the primary path. This is how MedDRA coding is done in most cases. Especially for a small-sized biotech company, SOCs preferred by study team are usually not defined, and the primary SOCs by MedDRA are followed. Even if all paths are considered, the ones beyond the primary path are rare and manageable from programming's perspective.

A SAS MACRO TO UPDATE CODED TERMS TO A NEW MEDDRA VERSION

The flow char of developing a SAS macro to perform MedDRA leveling is in Figure 2.

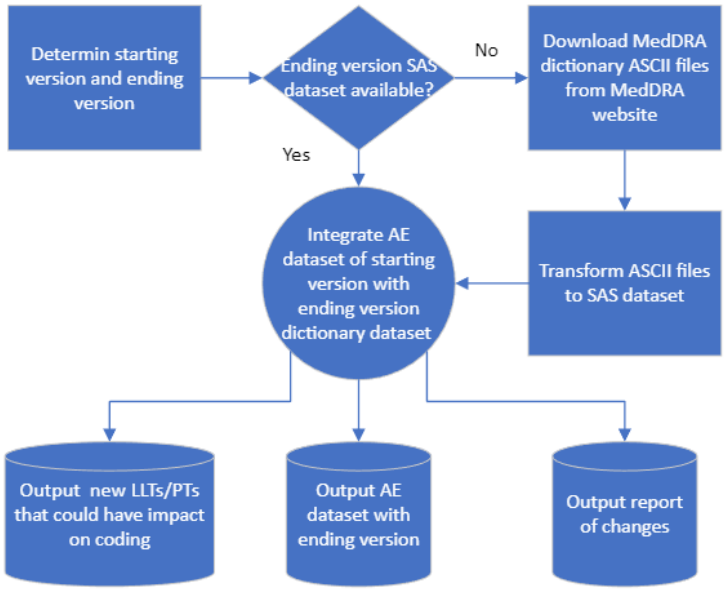


Figure 2. Flow Chart of Step 1

The macro expects the following entries:

1. Input path and dataset name
2. Output path and dataset name
3. Starting version of MedDRA dictionary
4. Ending version of MedDRA dictionary
5. If only primary SOCs are considered (Y/N)
6. If new LLTs/PTs and AE verbatim terms coded with them are output for reference (Y/N)
7. Reference dataset path and name
8. Language (CN/EN)

PREPARE MEDDRA DICTIONARY DATASET

MedDRA ASCII files are the source of MedDRA dictionary datasets. They can be downloaded from MedDRA website. The following two files are imported to SAS:

- “llt.asc” is the file of LLT including all LLT codes, LLTs and linked PT codes
- “mdhier.asc” is the file of hierarchy including all PT codes, PTs, HLT codes, HLTs, HLGT codes, HLGTs, SOC codes, SOCs and Primary SOC flag (PRIMARY_SOC_FG).

After imported to SAS, they are joined by PT code and archived by MedDRA version and language (English/Chinese). If coded terms need to be translated before leveling, it can also be conveniently achieved via programming by joining the tables of different language by corresponding code.

INTEGRATE AE DATASET WITH ENDING VERSION DICTIONARY DATASET

Only primary SOCs are considered

LLTs of starting version of MedDRA are joined with the ending version MedDRA dictionary dataset and terms of the other levels are populated with the terms in new dictionary with PRIMARY_SOC_FG='Y'.

All SOCs are considered

AE of staring version of MedDRA is joined with ending version MedDRA dictionary dataset by LLT code and SOC code. There are two scenarios after joining:

1. If LLT codes and SOC codes match, populate coded terms other than LLT and SOC with new terms. This has nothing to do with the primary SOC anymore.
2. If LLT codes match and SOC codes don’t match, no matter if PT codes match or not, populate coded terms other than LLT with new terms of PRIMARY_SOC_FG='Y', as there is no way to tell which path should be selected unless it is the primary path.

OUTPUT REPORT

The changes made to the dataset of staring version of MedDRA are output to an excel file for further check. Besides, since new LLTs may have impact on existing coding, a reference dataset of ending version of MedDRA with new LLTs/PTs is joined with updated dataset by verbatim term. If the verbatim terms match between those two datasets, it is reasonable to update the existing LLT with the new LLT as well. But this needs to be reviewed by medical/safety professionals for confirmation. Once confirmed this output can be used as input to further update the AE dataset.

VALIDATION

The datasets used to test the macro are summarized in Table 1.

Dataset	Number of Unique AEs	Language	Starting Version	Ending Version
AE	15	CN	22.0	24.1
AE	646	CN	23.0	24.1
AE	22	CN	23.1	24.1
AE	59	CN	24.0	24.1
Reference	4055	CN	24.1	

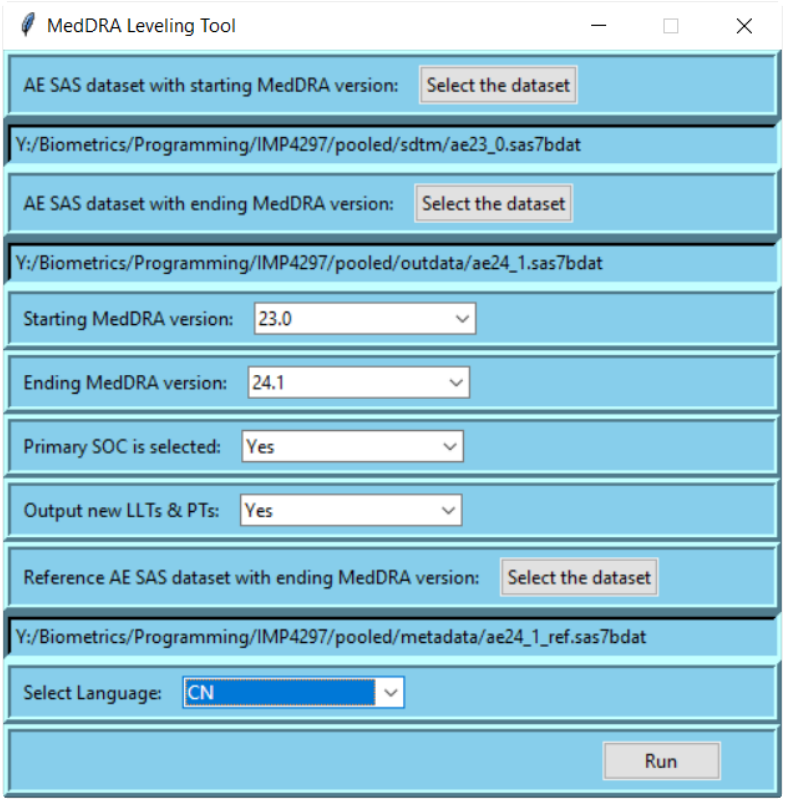
Table 1. Testing Dataset Information

Due to the limited number of AEs collected so far by the company, only a preliminary testing of the macro has been completed. In the future, with more AE data accumulated and more MedDRA versions coming out, or a dedicated dummy dataset is created, the macro will be tested to the largest extent.

BUILD GUI WITH PYTHON PACKAGES

Although SAS/AF® does provide a solution of point-and-click application in development environment, it must be purchased as an addition to SAS installation. Overall, it is not SAS's greatest strength to create a user-friendly interface to run the macro. Meanwhile, there are a lot of other tools to create nice looking Graphical User Interface (GUI), such as Java, Python, PowerShell, and even R. In this paper a Python package called "tkinter" is applied to create GUI. Compared with new Python packages like ipywidgets, Tkinter is a little dated, but it is the only framework built into the Python standard library, and it suits the need of a Python beginner like the author of this paper.

Tkinter widgets used in this tool include Label, Button, Frame, Combobox. The interface is demonstrated in Display 1.



Display 1. GUI built by Python package Tkinter

Input dataset, output dataset and reference dataset can be entered by clicking the button “Select the dataset”. Once selected the full path name will be displayed below. The starting version, ending version, if only primary SOCs are selected, if new LLTs/PTs are output and language can be selected in the pulldown list. Once all entries are in place, clicking the “Run” button on the bottom will trigger the function to pass all entries to SAS macro variables of the MedDRA leveling macro and then run the macro.

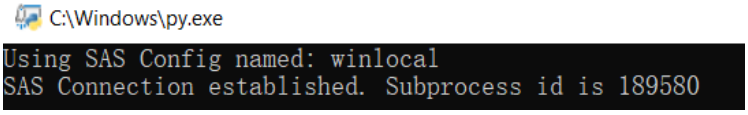
The function imports another Python package called “saspy”. SASPy is a module developed by SAS Institute for the Python programming language. With SASPy, SAS procedures can be executed in Python scripts, and data can be transferred between SAS datasets and their Python DataFrame equivalent. The Python code to transfer a Python value, such as the entries to the Tkinter GUI, to a SAS session is very simple. Firstly, the function to run SAS is defined:

```
def run_sas():
    import saspy
    sas = saspy.SASsession(results='html')
    …
```

After a series of Python string manipulation on the Tkinter entries, assign the value to SAS macro variables using the code below:

```
sas.symput('inpath', indata_filename1)
sas.symput('inname', indata_filename2)
…
```

The value of ‘indata_filename1’ is assigned to SAS macro variable &inpath, etc. Before SASPy is put into use, a configuration file needs to be configured so that the connection between Python and SAS session can be established successfully. Please note if the dataset encoding is “utf8” (Chinese characters) the same encoding should be assigned in the configuration file. Once the connection is established, the prompt window shows the message in Display 2.



Display 2. Connection between Python and SAS session

After the MedDRA leveling macro run is completed, a message box will pop up as shown in Display 3.

Display 3. Message Box after SAS macro run

Then SAS logs and outputs are checked to make sure the macro has been run successfully.

CONCLUSION

MedDRA leveling is a crucial step for AE dataset pooling and integrated analysis. Ideally this step should be completed in an integrated system with other dictionaries such as WHO-DD, but in this paper a simple Point-and-Click tool dedicated to MedDRA leveling works very efficiently and provides some flexibility for remediation. SAS code is the kernel of the tool, and it completes the most important task of updating coded terms. The GUI built by Python is like the shell of it, but it also requires extensible knowledge in

Python package “tkinter” and “saspy”. It is worth pointing out that the output needs to be reviewed by safety/medical staff to confirm the accuracy.

REFERENCES

Lankham, Isaiah. (2019) “Everything is better with friends: Executing SAS® code in Python scripts with SASPy.” *Proceedings of the SAS Global Forum 2019 Conference*, Dallas, TX. Available at https://www.sas.com/content/dam/SAS/support/en/sas-global-forum-proceedings/2019/3189-2019.pdf

Amos, David. “Python GUI Programming With Tkinter.” Real Python. Accessed Jun 30, 2022. Available at https://realpython.com/python-gui-tkinter/#building-your-first-python-gui-application-with-tkinter

SAS Institute. "SASPy – A Python interface to the SAS system." Accessed Jul 10, 2022. Available at https://sassoftware.github.io/saspy/index.html

CONTACT INFORMATION

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

Yanze Zhang
IMPACT Therapeutics
12F, New Bund Times Square, 399 Haiyang West Rd.
Shanghai, China
Phone: +86 18017898826
E-mail: yanze.zhang@impacttherapeutics.com