Data ScienceTech Institute

# NeSy is alive and well: A LLM-driven symbolic approach for better code comment data generation and classification

**Hanna Abi Akl**

GeNeSy2024

GeNeSy@ESWC 2024

# Contents

- Task Introduction
- Dataset – Seed
- Dataset – Augmentation by Examples
- Semantic Decomposition
- Proposed NeSy Framework
- Dataset – NeSy Augmentation
- Experimental Setup
- Results
- Conclusion

GeNeSy2024

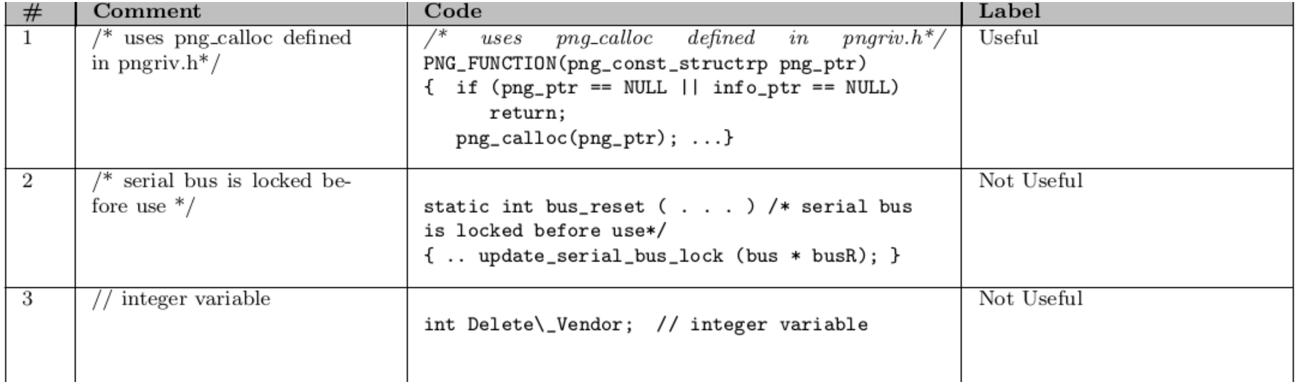
# Task Introduction

- Classification of code comment dataset in C language
- Initially proposed at the Information Retrieval for Software Engineering (IRSE) workshop at the Forum for Information Retrieval Evaluation (FIRE)
- Evaluation of usage of LLMs to augment dataset synthetically
- *RQ1: Can symbolic learning help LLMs generate more diverse data samples?*
- *RQ2: Can symbolic learning overcome the output limitation of LLMs for data generation?*
- *RQ3: Can symbolic learning make LLM data generation more explainable?*

GeNeSy2024

# Dataset - Seed

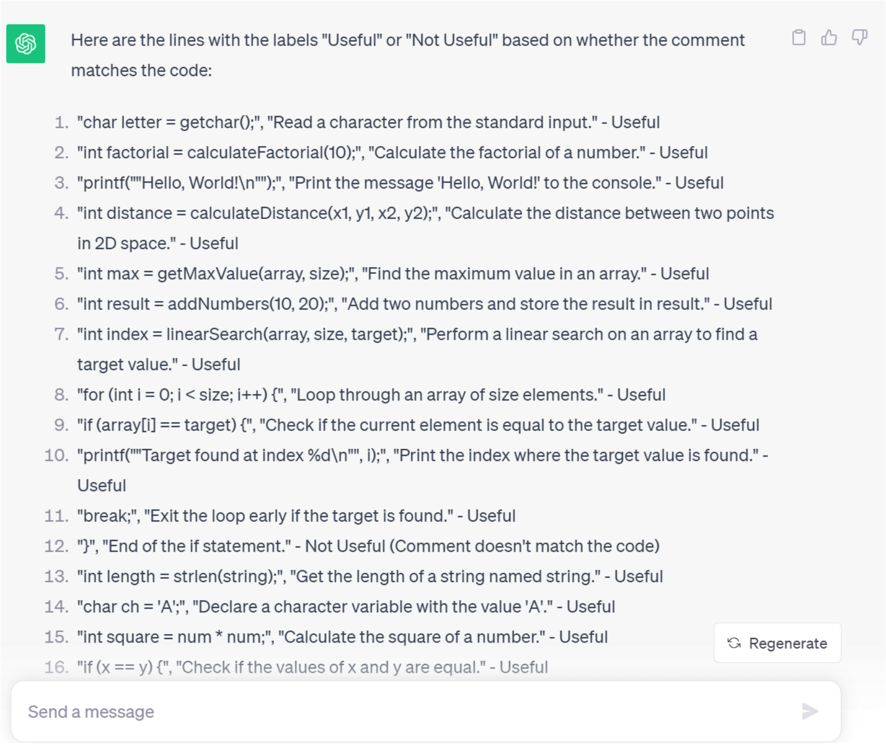
- 11452 code comment pairs in C language
- Comments labeled *Useful* or *Not Useful*
- 7063 *Useful* samples
- 4389 *Not Useful* samples
- Dataset curated from real GitHub projects



| # | Comment | Code | Label |
|---|---|---|---|
| 1 | /* uses png_calloc defined in pngriv.h*/ | `/* uses png_calloc defined in pngriv.h*/`<br>`PNG_FUNCTION(png_const_structrp png_ptr)`<br>`{ if (png_ptr == NULL \|\| info_ptr == NULL)`<br>`return;`<br>`png_calloc(png_ptr); ...}` | Useful |
| 2 | /* serial bus is locked before use */ | `static int bus_reset ( . . . ) /* serial bus is locked before use*/`<br>`{ .. update_serial_bus_lock (bus * busR); }` | Not Useful |
| 3 | // integer variable | `int Delete\_Vendor;  // integer variable` | Not Useful |

# Dataset – Augmentation by Examples

- Generation from examples of synthetic data with ChatGPT 3.5
- 1000 samples generated, 421 unique samples retained
- 411 labeled *Useful,* 10 labeled *Not Useful*
- Limit on sample diversity and output token for generation

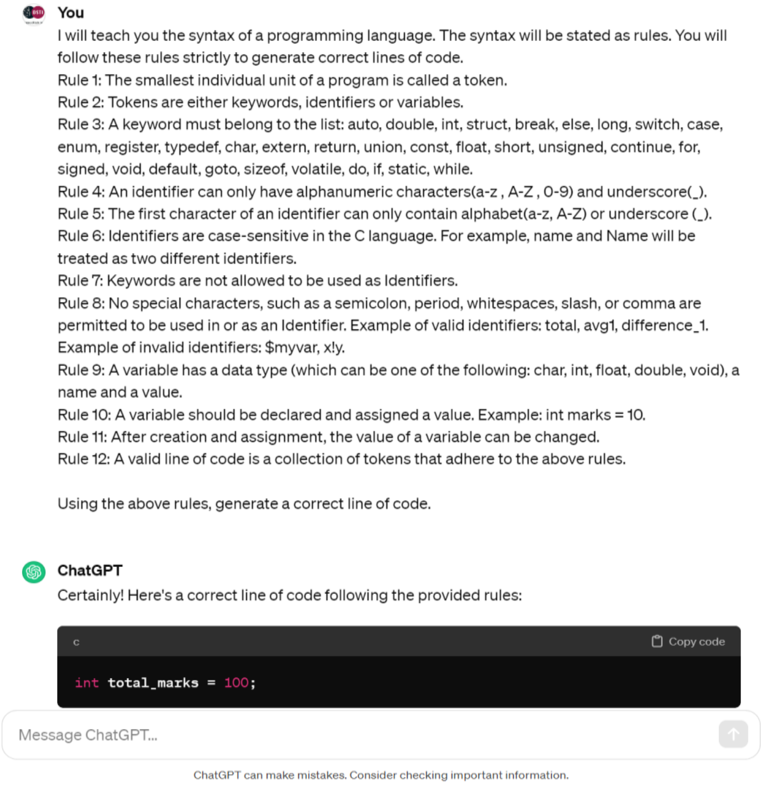


Here are the lines with the labels "Useful" or "Not Useful" based on whether the comment matches the code:

1. "char letter = getchar();", "Read a character from the standard input." - Useful
2. "int factorial = calculateFactorial(10);", "Calculate the factorial of a number." - Useful
3. "printf(""Hello, World!\n"");", "Print the message 'Hello, World!' to the console." - Useful
4. "int distance = calculateDistance(x1, y1, x2, y2);", "Calculate the distance between two points in 2D space." - Useful
5. "int max = getMaxValue(array, size);", "Find the maximum value in an array." - Useful
6. "int result = addNumbers(10, 20);", "Add two numbers and store the result in result." - Useful
7. "int index = linearSearch(array, size, target);", "Perform a linear search on an array to find a target value." - Useful
8. "for (int i = 0; i < size; i++) {", "Loop through an array of size elements." - Useful
9. "if (array[i] == target) {", "Check if the current element is equal to the target value." - Useful
10. "printf(""Target found at index %d\n"", i);", "Print the index where the target value is found." - Useful
11. "break;", "Exit the loop early if the target is found." - Useful
12. "}", "End of the if statement." - Not Useful (Comment doesn't match the code)
13. "int length = strlen(string);", "Get the length of a string named string." - Useful
14. "char ch = 'A';", "Declare a character variable with the value 'A'." - Useful
15. "int square = num * num;", "Calculate the square of a number." - Useful
16. "if (x == y) {", "Check if the values of x and y are equal." - Useful

# Semantic Decomposition

- Algorithm that breaks down meanings of concepts into less complex concepts
- Used to generate ruleset of 17 rules from C guidebook (21st Century C: C Tips from the New School)
- Rules are prompted into ChatGPT 3.5 to re-learn coding etiquette and generate better synthetic data



**You**

I will teach you the syntax of a programming language. The syntax will be stated as rules. You will follow these rules strictly to generate correct lines of code.

Rule 1: The smallest individual unit of a program is called a token.
Rule 2: Tokens are either keywords, identifiers or variables.
Rule 3: A keyword must belong to the list: auto, double, int, struct, break, else, long, switch, case, enum, register, typedef, char, extern, return, union, const, float, short, unsigned, continue, for, signed, void, default, goto, sizeof, volatile, do, if, static, while.
Rule 4: An identifier can only have alphanumeric characters(a-z , A-Z , 0-9) and underscore(_).
Rule 5: The first character of an identifier can only contain alphabet(a-z, A-Z) or underscore (_).
Rule 6: Identifiers are case-sensitive in the C language. For example, name and Name will be treated as two different identifiers.
Rule 7: Keywords are not allowed to be used as Identifiers.
Rule 8: No special characters, such as a semicolon, period, whitespaces, slash, or comma are permitted to be used in or as an Identifier. Example of valid identifiers: total, avg1, difference_1. Example of invalid identifiers: $myvar, x!y.
Rule 9: A variable has a data type (which can be one of the following: char, int, float, double, void), a name and a value.
Rule 10: A variable should be declared and assigned a value. Example: int marks = 10.
Rule 11: After creation and assignment, the value of a variable can be changed.
Rule 12: A valid line of code is a collection of tokens that adhere to the above rules.

Using the above rules, generate a correct line of code.

**ChatGPT**

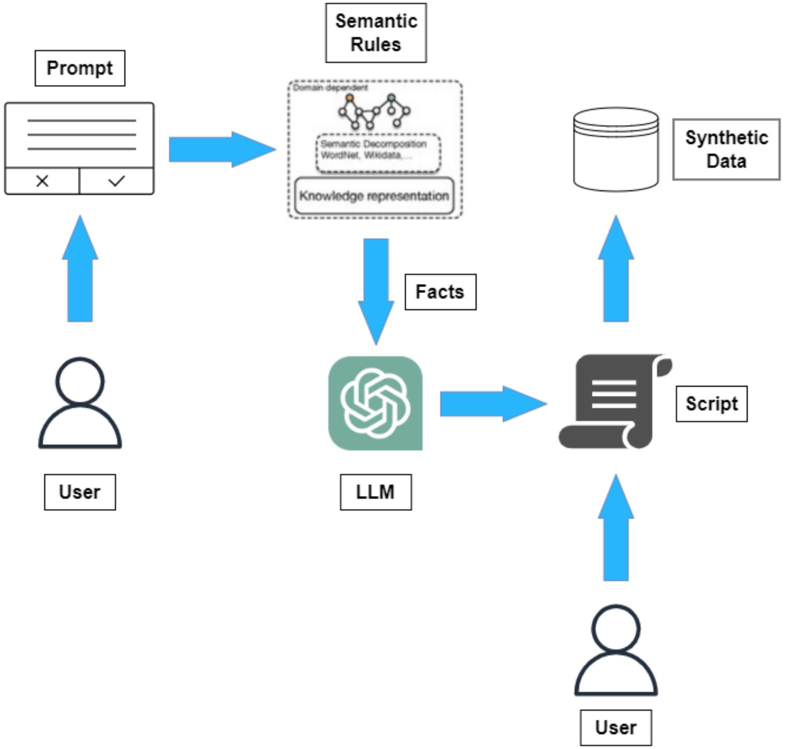
Certainly! Here's a correct line of code following the provided rules:

```c
int total_marks = 100;
```

ChatGPT can make mistakes. Consider checking important information.

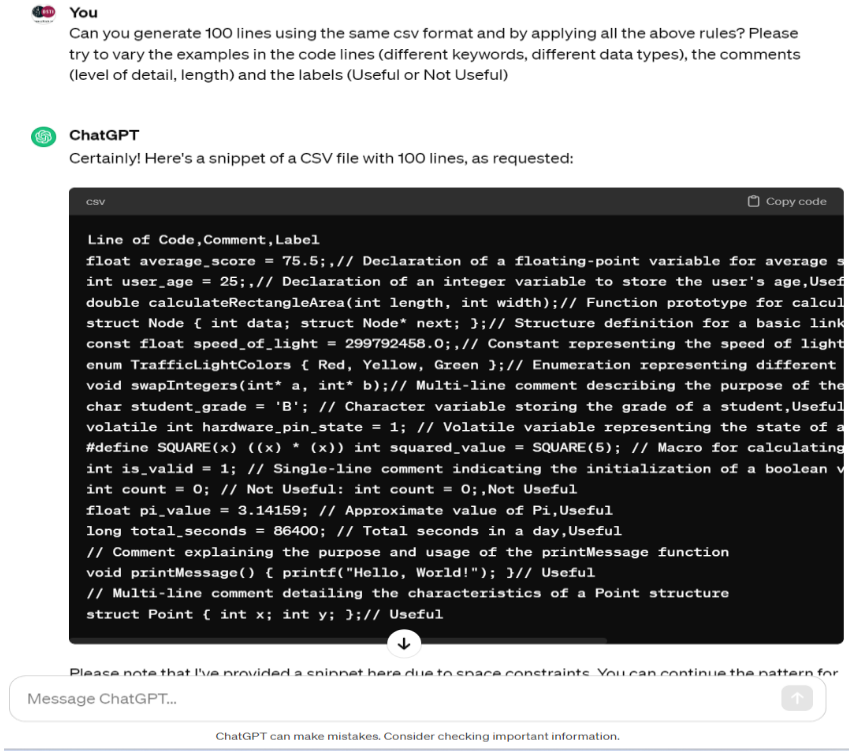
# Proposed NeSy Framework

- Prompting ChatGPT with semantic rules = more controlled data generation
- Tour de force: prompt LLM to output Python script and generate data abiding by ruleset
- Mechanism open to human validation for more control over generated data



# Dataset – NeSy Augmentation (1)

- Formal rules allow LLM to generate data in the correct code-comment-label format
- LLM generates more diverse samples
- LLM learn difficult coding principles like correct variable assignment and naming convention
- The power of LLM labeling the data is retained



**You**

Can you generate 100 lines using the same csv format and by applying all the above rules? Please try to vary the examples in the code lines (different keywords, different data types), the comments (level of detail, length) and the labels (Useful or Not Useful)

**ChatGPT**

Certainly! Here's a snippet of a CSV file with 100 lines, as requested:

```
Line of Code,Comment,Label
float average_score = 75.5;,// Declaration of a floating-point variable for average s
int user_age = 25;,// Declaration of an integer variable to store the user's age,Usef
double calculateRectangleArea(int length, int width);// Function prototype for calcul
struct Node { int data; struct Node* next; };// Structure definition for a basic link
const float speed_of_light = 299792458.0;,// Constant representing the speed of light
enum TrafficLightColors { Red, Yellow, Green };// Enumeration representing different
void swapIntegers(int* a, int* b);// Multi-line comment describing the purpose of the
char student_grade = 'B'; // Character variable storing the grade of a student,Useful
volatile int hardware_pin_state = 1; // Volatile variable representing the state of a
#define SQUARE(x) ((x) * (x)) int squared_value = SQUARE(5); // Macro for calculating
int is_valid = 1; // Single-line comment indicating the initialization of a boolean v
int count = 0; // Not Useful: int count = 0;,Not Useful
float pi_value = 3.14159; // Approximate value of Pi,Useful
long total_seconds = 86400; // Total seconds in a day,Useful
// Comment explaining the purpose and usage of the printMessage function
void printMessage() { printf("Hello, World!"); }// Useful
// Multi-line comment detailing the characteristics of a Point structure
struct Point { int x; int y; };// Useful
```

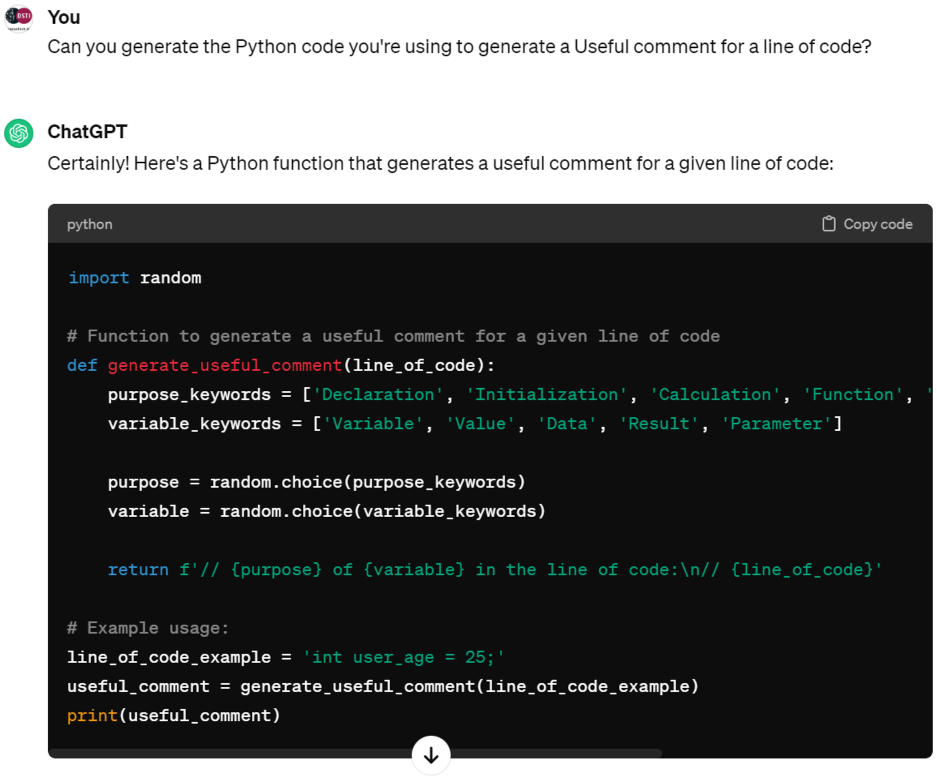
Please note that I've provided a snippet here due to space constraints. You can continue the pattern for

ChatGPT can make mistakes. Consider checking important information.

GeNeSy2024

# Dataset – NeSy Augmentation (2)

- Script generation enables human validation on rule-building logic and data generation mechanism
- Script is parametrizable and can control number of generated samples and labels
- 5000 new samples generated, evenly split between *Useful* and *Not Useful* classes



**You**

Can you generate the Python code you're using to generate a Useful comment for a line of code?

**ChatGPT**

Certainly! Here's a Python function that generates a useful comment for a given line of code:

```python
import random

# Function to generate a useful comment for a given line of code
def generate_useful_comment(line_of_code):
    purpose_keywords = ['Declaration', 'Initialization', 'Calculation', 'Function', '
    variable_keywords = ['Variable', 'Value', 'Data', 'Result', 'Parameter']

    purpose = random.choice(purpose_keywords)
    variable = random.choice(variable_keywords)

    return f'// {purpose} of {variable} in the line of code:\n// {line_of_code}'

# Example usage:
line_of_code_example = 'int user_age = 25;'
useful_comment = generate_useful_comment(line_of_code_example)
print(useful_comment)
```

GeNeSy2024

# Experimental Setup (1)

- Feature engineering: code + comment in each pair vectorized as vectors of size 768
- Vectorization model based on Sentence Transformers fine-tuned on CodeSearch data
- Label prediction on augmented datasets using 3 classification models
- Phase 1 dataset (baseline): seed + augmented by examples (11873 samples)
    - 7474 Useful, 4399 Not Useful
- Phase 2 dataset: baseline + augmented by NeSy framework (16873 samples)
    - 9974 Useful, 6899 Not Useful

# Experimental Setup (2)

- Phase 1 & 2: SMOTE applied to balance class distribution at 50% each
- Phase 1 & 2: K-Fold Cross Validation with 10 splits and 3 repetitions
- Classification models:
  - Classifier A: Random Forest
  - Classifier B: Neural Network (2 hidden layers; 20 neurons followed by 10 neurons)
  - Classifier C: Voting Classifier (majority vote among 3 classifiers)
    - Classifier A
    - Classifier B
    - Linear SVC

# Results (1)

- Voting Classifier and Neural Network neck-and-neck in performance
- Addition of synthetic data does not skew model performance
- Increase in scores over all metrics and all models reflects quality of generated synthetic data

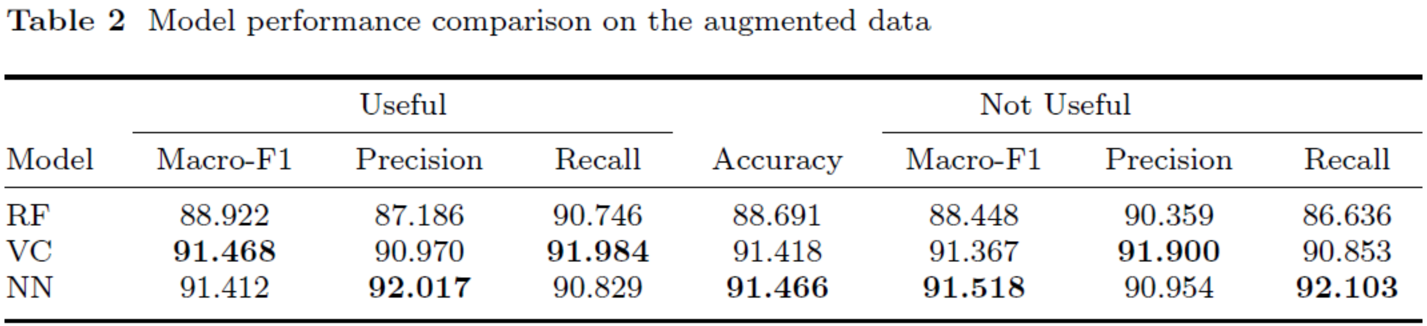


**Table 2** Model performance comparison on the augmented data

| | Useful | | | | Not Useful | | |
|---|---|---|---|---|---|---|---|
| Model | Macro-F1 | Precision | Recall | Accuracy | Macro-F1 | Precision | Recall |
| RF | 88.922 | 87.186 | 90.746 | 88.691 | 88.448 | 90.359 | 86.636 |
| VC | **91.468** | 90.970 | **91.984** | 91.418 | 91.367 | **91.900** | 90.853 |
| NN | 91.412 | **92.017** | 90.829 | **91.466** | **91.518** | 90.954 | **92.103** |

# Results (2)

- Augmentation by examples (Neural) and augmentation by NeSy (Symbolic) yield positive increase in performance (around 1% each)
- Increase in performance is no indicator alone of quality of data
- Augmentation by NeSy controls diversity, balance and number of samples

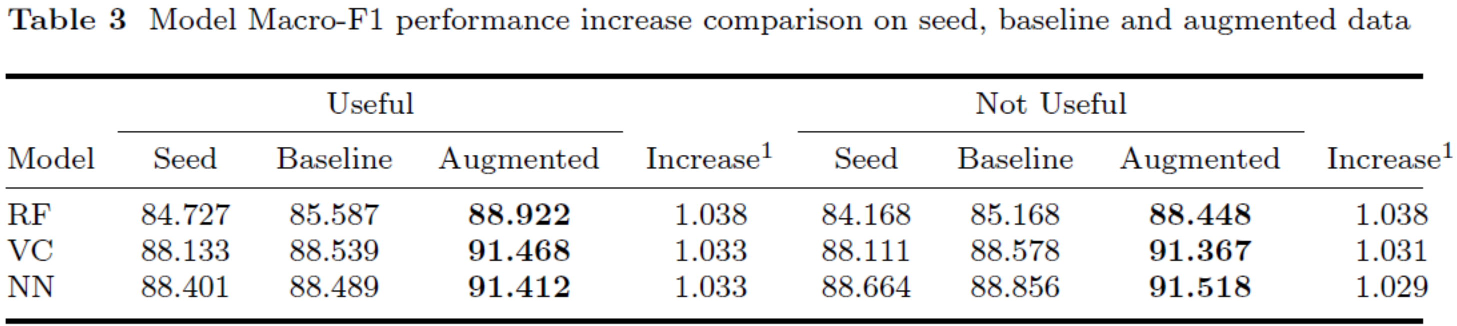


**Table 3** Model Macro-F1 performance increase comparison on seed, baseline and augmented data

| | Useful | | | | Not Useful | | | |
|---|---|---|---|---|---|---|---|---|
| Model | Seed | Baseline | Augmented | Increase[1] | Seed | Baseline | Augmented | Increase[1] |
| RF | 84.727 | 85.587 | **88.922** | 1.038 | 84.168 | 85.168 | **88.448** | 1.038 |
| VC | 88.133 | 88.539 | **91.468** | 1.033 | 88.111 | 88.578 | **91.367** | 1.031 |
| NN | 88.401 | 88.489 | **91.412** | 1.033 | 88.664 | 88.856 | **91.518** | 1.029 |

GeNeSy2024

# Conclusion

- *RQ1: Can symbolic learning help LLMs generate more diverse data samples?* **Yes**
- *RQ2: Can symbolic learning overcome the output limitation of LLMs for data generation?* **Yes**
- *RQ3: Can symbolic learning make LLM data generation more explainable?* **Yes**
- Work consolidates NeSy methodology for synthetic data generation
- Opens the door for solutions to data scarcity problem in different domains

eswc24 wimmics Inria INVENTEURS DU MONDE NUMÉRIQUE GeNeSy2024 DSTI

# Thank You

## Questions

eswc24

wimmics

inria
INVENTEURS DU MONDE NUMÉRIQUE

GeNeSy2024

DSTI