

# **Automatic Code Generation from Graphical User Interface (GUI) Images**

Sabthavi. J

219395V

Master of Science in Computer Science

Department of Computer Science & Engineering  
Faculty of Engineering

University of Moratuwa  
Sri Lanka

March 2024

# **Automatic Code Generation from Graphical User Interface (GUI) Images**

Sabthavi. J

219395V

Thesis/Dissertation submitted in partial fulfillment of the requirements for the degree  
Master of Science in Computer Science

Department of Computer Science & Engineering  
Faculty of Engineering

University of Moratuwa  
Sri Lanka

March 2024

## DECLARATION

I declare that this is my own work, and this thesis/dissertation does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other University or Institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature:

Date:

The above candidate has carried out research for the master's thesis/dissertation under my supervision. I confirm that the declaration made above by the student is true and correct.

Name of Supervisor: Prof. Indika Perera

Signature of the Supervisor:

Date:

## **ACKNOWLEDGEMENT**

First and foremost, I would like to express my deepest gratitude to my project supervisor, Prof. Indika Perera, from the Department of Computer Science & Engineering, Faculty of Engineering, University of Moratuwa, for his invaluable support, guidance, and encouragement throughout the project.

I also take this opportunity to convey my special thanks to all my friends, batch mates and colleagues who supported and encouraged me with their best wishes. Finally, I extend my heartfelt gratitude to my parents and family who supported me in numerous ways, contributing significantly to the success of this project.

## ABSTRACT

In today's software landscape, Graphical User Interfaces (GUIs) are crucial for improving user experience. As the demand for user-friendly software applications grows, the development of GUI-based software becomes more complex. This rapid growth of GUIs demands efficient methods for translating design representations into functional code. This thesis explores the novel approach of employing an Image Captioning model to automatically generate source code from Graphical User Interface (GUI) images. The openly available Pix2Code dataset, which encompasses screenshots of GUIs and their corresponding DSL code for Android, iOS, and web platforms, is utilized. The proposed model employs a standard encoder-decoder architecture. It integrates a ResNet152 model as the image encoder and an LSTM model as the text decoder to convert GUI images into human-readable DSL code sequences. The selection of the ResNet152 model is a strategic choice driven by its exceptional depth and complexity. The architectural design of ResNet152 enables it to adeptly capture the complex visual characteristics found within GUI images, making it well-suited for the task of code synthesis. GUI images often contain complex design elements such as buttons, text fields, and menus, which require a nuanced understanding to accurately translate into code. The depth of ResNet152 enables it to capture these intricate details effectively, contributing to the precision and accuracy of the code generation process. This thesis also investigates the choice of decoding algorithm, opting for a greedy approach over alternatives like beam search. This decision is informed by the need for simplicity and efficiency in the code generation process. Model's performance is assessed using BLEU scores, demonstrating a noteworthy achievement with a score of 0.78. This thesis provides an in-depth exploration of the technical aspects, including model architecture and training processes, along with insights into encountered challenges. This study significantly contributes to the field of automated code generation by introducing an efficient method for translating GUI images into executable code. The implications of this work extend to streamlining software development processes, reducing manual coding efforts, and enhancing overall productivity. The experimental results showcase the model's proficiency in capturing complex GUI designs and generating accurate source code snippets, paving the way for potential applications in GUI-based software development methodologies. This study contributes to the evolving field of automatic code generation and suggests possibilities for future enhancements in code generation for various platforms.

**Keywords:** Graphical User Interfaces (GUIs), Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), Domain Specific Language (DSL), Bilingual Evaluation Understudy (BLEU)

# TABLE OF CONTENTS

Declaration .....	i
Acknowledgement.....	ii
Abstract .....	iii
Table of Contents .....	iv
List of Figures .....	vii
List of Tables.....	ix
List of Abbreviations.....	x
Chapter 1 .....	1
Introduction .....	1
1.1 Introduction .....	1
1.2 Background and Motivation .....	3
1.3 Problem in Brief .....	4
1.4 Aim and Objectives .....	4
1.4.1 Aim.....	4
1.4.2 Objectives.....	5
1.5 Proposed Solution.....	5
1.5.1 Encoder .....	5
1.5.2 Decoder .....	6
1.6 Summary .....	6
Chapter 2 .....	7
Review of Related Work.....	7
2.1 Introduction .....	7
2.2 Other's Work.....	7
2.3 Summary .....	18
Chapter 3 .....	19
Technology Adapted .....	19
3.1 Introduction .....	19
3.2 Technologies Used .....	19
3.2.1 Python 3.11.5 .....	19

3.2.2	PyTorch.....	20
3.2.3	Machine Learning .....	20
3.2.4	Deep Learning.....	21
3.2.5	Image Processing .....	22
3.2.6	Natural Language Processing (NLP) .....	22
3.2.7	Convolutional Neural Network (CNN).....	23
3.2.8	ResNet 152.....	24
3.2.9	Recurrent Neural Network (RNN).....	24
3.2.10	Long Short-Term Memory.....	25
3.3	Summary .....	26
Chapter 4.....		27
Methodology and Design .....		27
4.1	Introduction .....	27
4.2	Dataset Selection .....	27
4.3	Data Preprocessing .....	28
4.4	Model Architecture.....	28
4.4.1	Encoder .....	29
4.4.2	Decoder .....	32
4.5	Summary .....	36
Chapter 5.....		37
Implementation and Evaluation .....		37
5.1	Introduction .....	37
5.2	Implementation.....	37
5.2.1	Dataset Preprocessing .....	37
5.2.2	Dataset Splitting .....	38
5.2.3	Building Vocabulary .....	38
5.2.4	Model Definition.....	40
5.2.5	Model Training.....	42
5.2.6	DSL to Executable Code Conversion .....	44
5.2.7	Final Results.....	45
5.3	Evaluation.....	48
5.3.1	Evaluation Metrics - BLEU Score .....	48

5.3.2	Selection of Encoder Model.....	48
5.3.3	Selection of Decoder Searching Algorithm .....	55
5.3.4	Visual Comparison of Results.....	57
5.3.5	Training Loss Summary.....	60
5.4	Summary .....	61
Chapter 6	.....	62
Conclusion and Future Work	.....	62
6.1	Conclusion.....	62
6.2	Future Work .....	65
References	.....	66

## LIST OF FIGURES

<b>Fig</b>	<b>Description</b>	<b>Page</b>
Fig. 1.1	Overview of Software Development Life Cycle	2
Fig. 2.1	Overview of Proposed Approach for REMAUI	7
Fig. 2.2	Overview of Proposed Approach for REDRAW	8
Fig. 2.3	Overview of Proposed Approach for image2emmet	10
Fig. 2.4	Reverse engineering from single input image	12
Fig. 2.5	Overview of Proposed Approach for pix2code	12
Fig. 2.6	Distribution of Quantity of articles per year	13
Fig. 2.7	Distribution of Approaches per platform	13
Fig. 2.8	Distribution of Approaches per input	13
Fig. 2.9	Distribution of Approaches per output	14
Fig. 2.10	Distribution of Frequencies of consideration of UI elements	14
Fig. 2.11	Overview of Proposed Approach of Aşıroğlu et al	15
Fig. 2.12	Overview of ORBIT Tool	15
Fig. 2.13	Overview of Object Detection	16
Fig. 2.14	Overview of Code Generation Process of Latent Prediction Network	17
Fig. 4.1	High Level Architecture of Automatic Code Generator Model	29
Fig. 4.2	Basic Architecture of ResNet-152 model	30
Fig. 4.3	Basic Architecture of LSTM	32
Fig. 4.4	Detailed Architecture of Automatic Code Generator Model	35
Fig. 5.1	Resulting Folder Structure after Data Splitting	38
Fig. 5.2	Unique Tokens of Generated Vocabulary	38
Fig. 5.3	Code Snippet of Encoder Model	40
Fig. 5.4	Code Snippet of Decoder Model Part 1	41
Fig. 5.5	Code Snippet of Decoder Model Part 2	42
Fig. 5.6	Code Snippet of Adam Optimizer and Cross Entropy Loss Function	42
Fig. 5.7	Code Snippet of Model Training	43
Fig. 5.8	Model Training Results	44
Fig. 5.9	Partial Screenshot of DSL-to-HTML Mapping File of Web Platform	45
Fig. 5.10	Input Image (Set 1)	46

Fig. 5.11	Resulting DSL Code (Set 1)	46
Fig. 5.12	Resulting HTML Code (Set 1)	46
Fig. 5.13	Input Image (Set 2)	47
Fig. 5.14	Resulting DSL Code (Set 2)	47
Fig. 5.15	Resulting HTML Code (Set 2)	47
Fig. 5.16	BLEU Score of proposed Model	48
Fig. 5.17	Training Results of Model using VGG16 Encoder	49
Fig. 5.18	Training Results of Model using VGG19 Encoder	50
Fig. 5.19	Training Results of Model using ResNet 101 Encoder	51
Fig. 5.20	Training Results of Model using ResNet 152 Encoder	52
Fig. 5.21	Comparison of BLEU Scores of different Encoder Models at Epochs	53
Fig. 5.22	Code Snippet of Greedy Search Algorithm	56
Fig. 5.23	Code Snippet of Compare_Visual_Results Function	57
Fig. 5.24	Sample 1 Result Set	58
Fig. 5.25	Sample 2 Result Set	59
Fig. 5.26	Training Loss Trend Graph (Loss vs. Epoch)	60

## LIST OF TABLES

<b>Table</b>	<b>Description</b>	<b>Page</b>
Table 5.1	Comparison of BLEU Scores of different Encoder Models at Epochs	53

## LIST OF ABBREVIATIONS

<b>Abbreviation</b>	<b>Description</b>
GUI	Graphical User Interface
CNN	Convolutional Neural Network
LSTM	Long Short-Term Memory
DSL	Domain Specific Language
BLEU	Bilingual Evaluation Understudy
NLP	Natural Language Processing
PIL	Python Imaging Library
HTML	Hyper Text Markup Language
RQ	Research Question