

**DOMAIN SPECIFIC QUESTION AND ANSWER  
GENERATION IN TAMIL**

Rubika Murugathas

199358H

M.Sc. in Computer Science

Department of Computer Science and Engineering

University of Moratuwa

SriLanka

July 2022

**DOMAIN SPECIFIC QUESTION AND ANSWER  
GENERATION IN TAMIL**

Rubika Murugathas

199358H

This dissertation submitted in partial fulfillment of the requirements for the Degree  
of MSc in Computer Science specializing in Software Architecture

Department of Computer Science and Engineering

University of Moratuwa

SriLanka

July 2022

## DECLARATION

I declare that this is my own work and this thesis 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. Also, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my thesis, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

*UOM Verified Signature*

...19.10.2022.....

Rubika Murugathas

Date

The above candidate has carried out research for the Master thesis Dissertation under my supervision. I confirm that the declaration made above by the student is true and correct.

*UOM Verified Signature*

...19.10.2022.....

Dr. Uthayasanker Thayasivam

Date

## ABSTRACT

Automatic Question-Answer generation is a challenging task in natural language processing. A system developed is capable of automatically generating questions and answers from history related text content in Tamil language input by user. The system processes the input text using various NLP techniques and generates questions and answers. The system has four modules namely, Preprocessing module, Rule-based module, Named Entity Recognition (NER) module, Question Answer Generator(QAG) module. Regex patterns and gazetteers are used in rule-based module and machine learning approach is used for NER module. NER module uses Conditional Random Fields (CRF) classifier built with features suitable for the domain and language. Dataset is collected from history textbooks and 23k word tokens are tagged using IOB2 format. Novel entity tag set specific to history domain are tagged. NLP techniques such as Sentence tokenization, POS tagging, Stemming, Unicode conversion uses existing python libraries. Features suitable for the domain and language selected are experimented with multiple combination. POS tag, stem word, gazetteer and clue words are features that contributes more for the performance. The best feature combination produced micro averaged Precision, Recall, F1-score of 87.9%, 67.1% and 76.1% respectively and accuracy of 89.6% on the test dataset. The NER module produced a better results despite the domain & language related challenges. Questions are formed using grammatical and defined rules from the named entities identified from rule-based and NER module. An affix stripping algorithm implemented to find the inflection suffix. A history text from Wikipedia is evaluated by 16 native Tamil speakers under categories such as undergraduates, graduates and experts. According to the evaluation results, 62.22% of total generated questions are grammatically correct and meaningful questions. Questions generated from Rule-based module produces better results compared to NER module.

Keywords: question and answer generation, tamil, NER, CRF, history, domain specific

## **ACKNOWLEDGEMENTS**

My profound gratitude goes to Dr. Uthayasanker Thayasivam, my supervisor for the knowledge, supervision, advice and guidance provided with his expertise, throughout in making this thesis a success.

My appreciation goes to my family for the motivation and support they have provided for my studies. I also would like to thank my colleagues in the MSc batch and at my workplace for the help and support provided in managing my research work.

## **Table of Contents**

DECLARATION	i
ABSTRACT	ii
ACKNOWLEDGEMENTS	iii
INTRODUCTION	1
1.1 Research Context	1
1.2 Problem	1
1.3 Motivation	2
1.4 Aim and Objectives	2
1.5 Statement of Contribution	3
LITERATURE REVIEW	4
2.1 Applications and Libraries for QA generation	4
2.2 Automatic QA generation in Tamil	4
2.3 Automatic QA generation in other languages	5
2.4 Evaluation of QA generation system	6
2.5 Surveys on QA generation	7
2.6 NER in Tamil	7
2.7 NER in other Languages	9
2.8 NER for history domain	10
2.9 Surveys or Studies on NER	10
BACKGROUND	12
3.1 Challenges	12
3.1.1 Domain related challenges	12
3.1.2 Language related challenges	12
3.2 Named Entity Recognition approaches	13
3.2.1 Rule based approach	13
3.2.2 Machine learning approach	14
3.2.3 Hybrid Approach	15
3.2.4 Deep Learning	16
3.3 Feature set	16
3.4 Named Entity Annotation Schema	17
3.5 Analysis on NLP libraries	18

3.5.1 Automatic Input Encoding & Conversion to Unicode	18
3.5.2 Sentence Tokenizer	18
3.5.3 POS tagger	19
3.5.4 Tamil stemmer	20
3.5.5 Morphological Analyzer	21
METHODOLOGY	23
4.1 High Level Architecture	23
4.1.1 Input	24
4.1.2 Process	24
4.1.3 Output	24
4.2 System Overview	24
4.2.1 Preprocessing Module	25
Read file content	25
Unicode conversion	25
Tokenization	25
Sentence Selection	25
4.2.2 Rule based Module	26
Regex pattern matching	26
Gazetteer search	26
4.2.3 Named Entity Recognition Module	27
Named Entity tag set	28
POS Tagging	30
Conditional Random Fields	31
Features	32
Hyper Parameter Tuning	34
4.2.4 Question and Answer Generation Module	34
Steps for Question Generation	34
Extract inflection suffix	36
Question word identification	36
Rules and Assumptions	37
TECHNOLOGY AND RESOURCES	38
5.1 Programming Languages or Frameworks	38

5.2 Development Tools	38
5.3 Version Controlling Systems	38
5.4 Libraries used	38
IMPLEMENTATION	40
6.1 Web Application User Interface	40
6.2 Preprocessing	42
6.3 Rule based approach	42
6.3.1 Regex pattern Matching	43
6.3.2 Gazetteer approach	44
6.4 Named Entity Recognition	44
6.4.1 Corpus tagging	44
6.4.2 CRF implementation	45
6.4.2 Processing predicted tags	48
6.5 Question and Answer Generation	49
6.5.1 Question and Answer generation algorithm	49
6.5.2 Affix Stripping	50
6.5.3 Question Formation	51
EXPERIMENTS AND EVALUATION	53
7.1 NER Module Evaluation	53
7.1.1 Corpus Tagging	53
7.1.2 Performance Evaluation Metrics	53
7.1.3 Experiments	54
7.2 Q&A Generation Module Evaluation	57
7.2.1 Initial Evaluation	57
7.2.2 Error Analysis	57
7.2.3 Post Processing Rules	61
7.2.4 Final Evaluation	62
DISCUSSION	68
8.1 Named Entity Recognition Module	68
8.2 Question and Answer generation Module	68
CONCLUSION	70
FUTURE WORK	71



REFERENCES	72
APPENDIX – A: Question Evaluation Form	79
APPENDIX – B: Implementation Code	97
APPENDIX – C: SPSS Tool Output	99

## LIST OF FIGURES

Figure 3.1: Fonts supported by Open Tamil library .....	18
Figure 3.2: Python code to test Indic NLP library sentence tokenizer .....	19
Figure 3.3: Output of Indic NLP library sentence tokenizer .....	19
Figure 3.4: RippleTagger library testing code.....	20
Figure 3.5: Output of RippleTagger library .....	20
Figure 3.6: Snowball stemmer library testing code .....	21
Figure 3.7: Output of snowball stemmer library .....	21
Figure 3.8: Indic NLP MA library testing code .....	22
Figure 3.9: Indic NLP MA library output .....	22
Figure 4.1: High level Architecture of the system .....	23
Figure 4.2: High Level diagram of NER system .....	27
Figure 6.1: The web application UI.....	40
Figure 6.2: UI after valid file upload.....	41
Figure 6.3: Invalid file upload Error .....	41
Figure 6.4: UI after questions generated .....	42
Figure 6.5: Sample corpus annotation .....	45
Figure 6.6: CRF algorithm .....	45
Figure 6.7: List of numeric word in Tamil.....	46
Figure 6.8: Sample code segment for Inflection suffix identification .....	50
Figure 6.9: Sample Question word list for inflected named entity .....	52
Figure 7.1: Pie chart for profession info of evaluators .....	62
Figure 7.2: Pie chart for higher studies info of evaluators .....	63
Figure 7.3: Pie chart for secondary school studies info of evaluators .....	63
Figure 7.4: Venn diagram for overall Question Evaluation Results.....	65
Figure 7.5: Venn diagram for Question Evaluation Results by experts .....	65

## LIST OF TABLES

Table 4.1: Named Entity tags with examples.....	28
Table 4.2: POS tags supported by Ripple Tagger library.....	30
Table 5.1: List of Libraries and Tools used in system.....	39
Table 6.1: Regex patterns used in System.....	43
Table 6.2: Clue words for Named Entities.....	46
Table 6.3: Prefix Suffix Rule for Named Entity.....	48
Table 6.4: Question words for Named Entities.....	51
Table 7.1: Tag distribution in the dataset.....	54
Table 7.2: Basic feature combination.....	55
Table 7.3: Experiment Results for Different Feature Combination.....	55
Table 7.4: Performance metrics for all Named Entity tags.....	56
Table 7.5: Best feature combination.....	57
Table 7.6: Error Analysis.....	57
Table 7.7: Summary of Evaluation responses.....	64
Table 7.8: Performance metrics for QA generation module evaluation.....	64
Table 7.9: Performance metrics for QA generation module evaluation from experts.....	65
Table 7.10: Performance metrics for QA generation from different approaches.....	66
Table 7.11: Inter-rater Reliability estimate for QA evaluation.....	67

## LIST OF ABBREVIATIONS

<b>Abbreviation</b>	<b>Description</b>
BERT	Bidirectional Encoder Representations from Transformers
BiGRU	Bidirectional Gated Recurrent Unit
BiLSTM	Bidirectional Long Short Term Memory
CNN	Convolutional Neural Networks
CRF	Conditional Random Field
GRU	Gated Recurrent Unit
LSTM	Long Short Term Memory
MA	Morphological Analysis
NE	Named Entity
NER	Named Entity Recognition
NLP	Natural Language Processing
NQG	Neural Question Generation
POS	Part-of-Speech
QA	Question and Answer
QAG	Question and Answer Generation
RNN	Recurrent Neural Networks

## **LIST OF APPENDICES**

<b>Appendix</b>	<b>Description</b>	<b>Page</b>
Appendix-A	Question Evaluation Form	79
Appendix-B	Implementation Code	97
Appendix-C	SPSS Tool Output	99