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

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Description
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

CHAPTER 1

INTRODUCTION

Automatic Question and Answer Generation(AQG) is a challenging task in Natural Language Processing(NLP) and its usage is growing day by day. Question-Answer generating system from a given text can help the students to learn more efficiently and evaluate themselves. As preparing examination paper is another daunting task for teachers and lecturers, they also can get benefited from this system.

1.1 Research Context

Education has become a vital requirement in the current world and exams are used as a common measure of educational achievements and knowledge. Exams are considered important all around the world and in all levels of study from elementary education to postgraduate studies. This will elevate the learning process and help the students to evaluate themselves and enhance their knowledge in different domains. Earlier days students go through the learning materials provided by schools or books published. But nowadays E-learning is widely used in many countries and life of students has become much easier. This enables them to improve their knowledge and learning on their own by accessing materials from internet.

Practicing more questions will make students more prepared for the competitive exams and improve their knowledge in answering questions in examinations. Currently there are several systems and online applications available with question banks in different domain with marking schemes. There are few applications available to generate questions with answers from a given input document or text in English. There are researches carried on in generating exam papers with marking scheme for course materials for different languages and domain.

1.2 Problem

There are plenty of question banks available in English for different domains that are accessible via internet. There are several systems available for providing question and answer generation in English language for course materials in different countries. Many more researches are carried out frequently to generate different type of questions and increase the accuracy levels of correctness of questions and answer generation. There are also few AI platforms available in the internet to generate question and answers when user provides a text input or text or pdf document.

Question Answering system and Question and Answer generation system are two different researches where the first finds the answer from given text when a question

is provided, while the second generates possible questions and answers both from a given text. There are more researches conducted in Question Answering system compared to Question and Answer generation in Tamil. While there are number of researches conducted on developing various NLP tools in Tamil, only very few libraries and tools are publicly available. There are very few researches carried out in generating question and answers in Tamil using the available or developed toolkits. Therefore students studying in Tamil medium have less access for digital materials in Tamil and question banks in internet. This research will help to generate automatic questions and answers from a given text input or document and help them in carrying out their studies easily.

1.3 Motivation

The motivation of this research is to enable the Tamil medium students to learn more efficiently by accessing the question and answers generated by the system and evaluate themselves comparing their answers with the generated answer. In this way it will help them to practice more questions and get prepared for quizzes and examinations. This will also help the examiners to prepare exam papers for class level assessment, preparing quizzes and exam papers.

1.4 Aim and Objectives

Aim - Develop a web based Question and Answer generating system that can generate factoid WH questions from a given history related text in Tamil language. The system accepts input in text or file format and outputs the generated questions and answers in text as well as in downloadable text file format.

Objectives

- Analysis
    - Analyze on the domain and language related challenges.
    - Analyze existing systems & researches and their limitations.
    - Find suitable existing libraries & tools that can be used for NLP techniques.
- Tag a dataset from tamil history documents & publish to be used by future researches.
- Build a Named Entity Recognition(NER) module using the tagged dataset.
- Define gazetteer lists of named entities.
- Build an end-to-end QA generation web application that accepts text or file as input and output Questions & Answers in text or file format.
- System Evaluation
    - Evaluate the questions on grammatical & semantical correctness.
    - Analyze the limitations and improvements to be done for future work.

1.5 Statement of Contribution

A web-based application for Automatic Question and Answer Generation(QAG) is developed as a first attempt for history domain in Tamil language. The system is intended to help the students studying in Tamil medium to learn more efficiently and evaluate themselves for history subject. It also benefits the examiners in preparing quizzes and exam papers using the system.

The system accepts a text or file related to history domain in Tamil language as an input for the system. The uploaded file content is displayed in a text area, so that the user can edit the content before generating questions. It processes the text paragraph sentence by sentence and generates possible WH questions with answers. The system generates simple factoid questions, but it is not capable of generating complex questions like open or discussion questions. The questions & answers generated are displayed as text and can be downloaded as a text file.

The system uses both a Rule-based module as well as Named Entity Recognizer (NER) module for generating questions. Regex patterns and gazetteers are used for the implementation of rule based module. Gazetteer list is defined for few named entities like country, city, kingdom, popular kings, leaders, and so on. NER is built using machine learning technologies as a first attempt for history domain in Tamil language. The dataset used for the NER module was manually tagged from Tamil medium Grade 10 & 11 history textbooks as there are no dataset publicly available for history domain in Tamil. The tagged dataset will be published to be used by future researches. NER module developed can also be used for future researches.

The questions generated from the final system is evaluated on the grammatical and semantical correctness using crowdsourcing review from a number of category of people including experts. The system performs well in producing more meaningful questions despite the domain and language related challenges.

CHAPTER 2

LITERATURE REVIEW

This chapter discusses about the application publicly available in internet for QA generation and the related work carried out for the QA generation in Tamil and other languages using various NLP approaches. There are very few research available for QA generation in Tamil and few researches carried out for other low resource languages. Evaluation techniques used by different researches for QA generation systems are also reviewed. Named Entity recognition (NER) is an important module in the developed system. Hence, this section also discusses the researches carried out for NER in Tamil and other languages and surveys or studies for different NER approaches also discussed.

2.1 Applications and Libraries for QA generation

Quillionz is the first AI-powered platform for automatically generate questions from the textual content in English. The system generates different types of questions like short descriptive & multiple choice questions and output in text, word or pdf documents and flash cards format. The system gives possible keywords from text and user can select keywords to generate questions. The system also identifies lengthy, subjective, incomplete sentences and pronoun replacements from the given text. It allows the user to edit the suggestions based on the suggestions. PrepAI is a test generation AI platform that generates various question types such as multiple choice, fill-in-the-blank, descriptive, true or false and statement based. It accepts inputs in the form of Pdf and word documents, video URLs, video files. Also offers a feature of scraping content directly from Wikipedia. Lumos learning & Questo are few other AI platforms available in internet for QA generation. Questgen is an open source NLP library for Question generation algorithms. It is built using the state-of-art NLP techniques. The library is capable of generating MCQs, Yes/No questions, FAQs, paraphrasing a question and Question Answering in English Language.

2.2 Automatic QA generation in Tamil

Vignesh N and S.Sowmya [25] proposed a system to automatically generate questions in Tamil from a given input text. Words in the sentence are tagged with Noun, Verb, Time, Noun descriptor and Verb descriptor tags. Verb tags have sub tags gender and tense. Initially all words are marked as noun tags. Gender, tense, Noun descriptor and verb descriptor are tagged based on case markers. Words for time marker are defined. After tagging, the question word is replaced in the place of descriptor tags using some defined rules. All possible questions are generated from a sentence for different tags

identified and the questions are optimized by allocating different points to each tags based on precedence.

2.3 Automatic QA generation in other languages

Automatic QA generators are developed using 3 approaches: syntax based, semantic based and template based. Syntax based approaches focus on the syntax such as POS, NER, syntactic tree, etc. Semantic based approaches focus on a deeper level and mostly uses a knowledge source like taxonomies, Ontologies, etc. Template based approaches use templates that define structure of question using fixed text and placeholders.[57] Rules, templates or statistical methods are used for question construction. Rules are used to select suitable question type and to construct questions. Statistical methods learns question transformation from training data. Recently neural techniques are used to generate questions.

Most of the researches that uses neural techniques follow the sequence-to-sequence framework. Du et al. [58] proposed the first sentence level sequence tagging model using neural techniques. The encoder uses a BiLSTM and an attention mechanism to help decoder focus on most relevant parts of the sentence and a decoder using LSTM to generate question. QG model with this sentence selection model achieved state-of-art paragraph-level question generation performance on the SQUAD data set.

Seq2Seq models struggle to utilize relevant contexts avoiding irrelevant information[69]. Zhao et al. [60] proposes a maxout pointer mechanism with gated self-attention encoder to address the problems of processing long text in previous sequence neural models and achieved a new state-of-art results for the SQUAD dataset. Kim et al. [61] proposed a novel answer-seperated seq2seq architecture with 2 encoders separately for paragraph and answer and answer-seperated decoder. A keyword-net used to obtain the key information from answer. Contextual feature of paragraph from attention mechanism and keyword feature from keyword-net is used by decoder to generate a question. Yuan et al. [59] developed a Tibetan QG system using a Tibetan QG dataset constructed through crowdsourcing. An attention mechanism used to obtain answer-aware paragraph representation during paragraph encoding and copy mechanism to avoid generating unknown or rare words during decoding.

Fabbri et al. [62] proposed a template-based model that uses a retrieval based approach to obtain a sentence from the corpus identical to the current context. A pretrained BERT model was fine-tuned on the data created with question for all context & answer pairs and was evaluated on the SQUAD dataset. Teshani et al.[30] developed a QA Generator for Sinhala using a semantic relationship identifier that defines 8 patterns for Sinhala sentence with the basic subject, object, and verb with pronouns, adjectives, adverbs.

Serban et al.[63] developed a novel neural network architecture for mapping knowledge base into questions. Model is trained with triple: subject, relationship and

object. A template based model was presented with same dataset to compare with neural model and it outperformed the template based model. Sathish et al. [64] proposed an automatic Question Answer pair generator using a knowledge graph. A set of keywords from entities and relationship stored as subject, a predicate and object triple in knowledge graph. A sequence to sequence model using RNN is built using subset of keywords as sequence. Geetanjali et al.[65] developed a QA generator by mapping Abstract Meaning representation (AMR) to question-answer meaning representation (QMR). AMR is a semantic representation of a whole sentence and templates are defined for the relations in AMR. Templates are transformed to suitable questions.

Deepali et.al [31] developed a rule based question generation for Marathi text summarization. POS tagging applied and nouns are further classified as person and location using NER tools. Affix stripping algorithms are used for rule based stemming. Holy Lovenia et.al [28] presented a rule-based automatic QA system for reading comprehension. Sentence selection done using text summarization method Text rank and Latent Semantic Analysis (LSA). The gap selection is done using constituent parsing and Named Entity Recognition. NER was experimented with CRF and Bi-LSTM CRF on generic tags. CRF outperformed deep learning model. The fill in the blank question answer pair is converted to WH questions by converting the sentence to interrogative form. The question word is determined by the named entity type of its answer. Dhaval et.al. [32] developed a rule based QA generation system to generate simple and complex type of questions. Sentence selection is done by feature extraction such as number of nouns and pronouns, length, etc. To generate simple questions, Named Entity Recognizer is used. Complex questions are formed by identifying discourse connective words such as because, since, as a result, etc.

2.4 Evaluation of QA generation system

The most common evaluation approach is expert evaluation. The second most common approach is comparing machine generated questions with human authored questions[57]. BLEU[58, 59, 61, 63, 64], METEOR[58, 61,63] and ROUGE[59,61] are some popular metrics that compute the n-gram similarity between the reference sentence and the generated sentence[66] and used by most of the neural QA generator models.

Jouault et al.[38] have evaluated the semantic ambiguity and fluency of question generated from 12 students and also the coverage of questions generated by expert from the same text is also evaluated. Mazidi et al.[36] and Mazidi et al.[39] has evaluated the question acceptability using 3 point scale and 5 point scale respectively using crowdsourcing review evaluation method. Zhang et al.[40] has evaluated the semantic ambiguity of questions generated using 5 point scale review from 12 students. Huang and He[37] has evaluated the system for question acceptability using binary scale from 4 experts and also difficulty and discrimination is evaluated by

conducting mock exam for students from the generated questions. Flor & Riordan[45] has evaluated the semantical and grammatical correctness with 5 point scale review from 2 experts. Blstˇ ak & Rozinajov ´ a[44] has evaluated the system on question acceptability, semantic and grammatical correctness using more than one evaluation methods: Comparison with another generator and Comparison with human authored questions.

2.5 Surveys on QA generation

Kurdi et al.[57] conducted a survey by reviewing 93 papers related to automatic Question & Answer generation. The study was done for different dimensions like question generation approaches, evaluation, knowledge sources used, different domain, language, question types generated and answer format. The template based approach was the most common method used in researches reviewed. As templates are developed by analyzing set of questions manually or with domain experts, more cost and effort needed. Hence automatically constructing templates is suggested for future works. SQUAD and NewsQA are reading comprehension dataset used by researches. SQUAD consists of question-answer pairs derived from Wikipedia articles on paragraph level whereas NEWSQA contains wh question-answer pairs from CNN news articles. Different evaluation criteria like grammatical correctness, fluency, semantic ambiguity, freeness from errors, distractor readability, educational usefulness, domain relevance, difficulty, discrimination, and cognitive level are used in various researches.

Liangming et al.[69] conducted a survey on latest Neural Question Generation (NQG) systems analyzing dataset, question generation methodology and evaluation methods. QAG researches was reviewed on 3 aspects such as learning paradigm, its input modalities and cognitive level to provide insights on how neural QAG connects to conventional QAG. Traditional QAG architectures are limiting due to the limited to rules & templates used. But neural models motivate an end-to end architectures. Most of the recent NQG systems adopt Seq2Seq framework. SQuAD, NewsQA are shallow cognitive level datasets, MS MARCO, RACE are medium cognitive level datasets and LearningQ, NarrativeQA are deep cognitive level datasets used in NQG researches. Copying mechanism, Linguistic features and policy gradient are common techniques used. Finally three emerging trends in NQG is summarized under 3 aspects such as multi-task learning, wider input modalities and deep question generation.

2.6 NER in Tamil

Kathiravan et al. [27] proposed a methodology for developing a NER system for Tamil social media posts using different Deep Learning Architectures like Gated Recurrent Unit (GRU), Long Short Term Memory (LSTM) and Recurrent Neural Networks (RNN). As RNN cannot efficiently work with very long sequences LSTM and GRU is suggested. Feature extraction done under 3 categories like REGEX, Morphological

Analysis and Content feature extraction. Feature selection done on the features extracted using TF/IDF to find the importance of a word in a given text input. Dimensionality Reduction is done using Singular Value Decomposition (SVD). Hariharan et al. [26] developed a NER system using LSTM and fastText word embedding using a dataset collected from Tamil Wikipedia and FIRE-18. The performance is compared with GloVe word embedding and LSTM with fastText embedding outperformed LSTM with GloVe.

Vijaykrishna & Sobha [1] presents a CRF based Tamil NER for tourism domain using a corpus with 106 nested tag set in 3 hierarchical levels. The test data is tagged with each CRF models built for 3 hierarchical levels separately and outputs from 3 models are merged to get a final output. The system obtained an F1-score of 80.44% as tagging is done only when the context of the present word is similar to the context of the named entities or if the root word is already learned from the training corpus. Malarkodi et al.[2] developed a generic NER system for Tamil language using the CRF and SVM algorithms and performance is compared. After doing an error analysis, some heuristic rules were applied to improve the results. The final system achieved an F1-Score of 70.68% and CRF outperformed SVM. Malarkodi & Sobha [6] developed a fine-grained NER for Tamil using CRF with a dataset of 200k tokens from FIRE 2013 and Cross Lingual Information Access (CLIA) project. POS, chunk information, POS Patterns Preceding and Following NE and few other lexical features are used as features. Extended system generated by eliminating the error driven features and post processing rules were applied to further improve the system. The baseline system achieved an F1-score of 65.07% while extended system achieved 83.68%.

Pranavan et al.[5] developed a NER system in Tamil using MIRA algorithm with a dataset collected from Tamil BBC news and a comparison of performance between MIRA and CRF algorithms is also presented. NE tags are limited to few generic tags. A rule based Morphological Analyzer built using a database of noun & verb stems created from the Tamil WordNet using a web crawler. Features like POS, surefire rules, stem word, gazetteers, orthographic features are used. MIRA algorithm obtained an F1-score of 81.38% while CRF algorithm obtained 79.13% for the same set of features. MIRA classifier out-performs CRF. Srinivasan & Subalalitha [7], implemented a NER system using Naïve Bayes, a probabilistic classifier that assumes each feature is independent. Training data is collected from Fire corpus. Features are grouped under 3 types such as REGEX, Morphological and Context Features. A bootstrapping approach uses a seed set containing the identified features and NEs and it has contributed in the increase of feature. The system achieved an F-measure of 83.54%.

2.7 NER in other Languages

Deep Learning is the current state of the art in NER systems. Norah et al.[70] developed a Classical Arabic NER by fine tuning a pre-trained BERT contextual language model. The BERT model representations are used as input for the two deep learning models, Bidirectional Long Short-Term Memory (BiLSTM) and Bidirectional Gated Recurrent Unit (BiGRU) for further sequence modelling. The output of BiLSTM/BiGRU models is passed to a CRF layer. BERT-BiGRU-CRF model achieved an F-measure of 94.76% on the CANERCorpus and it outperformed other models. Archana et al.[71] proposed a novel bilingual Hindi-Punjabi NER using BiGRU and Convolutional Neural Networks(CNN) with enhanced embeddings. FastText word embeddings concatenated with minimal feature embeddings (POS, word prefix, suffix and word length embeddings) to form Enhanced Word Embeddings(EWE). Bi-GRU and CNN model using EWE improves the performance of NER system and the system achieved an F-score of 91.64%, 93.60%, 93.22% respectively for Hindi, Punjabi and bilingual Hindi-Punjabi. Nazmiye et al.[72] developed a NER on tweets during earthquake disaster to help provide an event location detection & disaster management. LSTM, BiLSTM & GRU were experimented with GloVe word embeddings.

Azeez & Surangika[8] has presented a CRF based fine grained NER system for Sinhala language. NE annotated Sinhala corpus of 70k word tokens with 24 NE tag set divided into 114 nested tags is used. Novel entity tag set like Law, Designation, Product, Event, Work of Art, Facility, Subject, Theory, Language, etc. are used and annotated using markup based annotation scheme. A novel technique is used to improve the gazetteer list by adding inflected words based on 4 case markers such as Nominative, Accusative, Dative and Genitive. Final system produced an F1-score of 84.8% with 5-cross validation. Jinadi & Ruvan[4] presents a NER system in Sinhala language using CRF and Maximum Entropy(ME) models. Context word features and suffix are used in both models. Too much information about the surrounding words did not present good results. System achieved F-measure of 78.95% for CRF and 63.06% for ME. CRF outperformed ME model. Maithilee L. Patawar & M. A. Potey[10] has presented a novel approach for implementing NER for tweets in English and Marathi by combining CRF with KNN classifier to implement a semi supervised algorithm due to lack of tagged training dataset. Normalization process replace ill formed words, misspelled words and abbreviated words with correct words with the help of dictionary and gazetteer. Confidence value for each word is obtained from KNN classifier and used in assigning label and deciding the NE class. NER system for Marathi tweets obtained an F-measure of 50.8%.

2.8 NER for history domain

Baptiste et al. [23] experimented 3 models with BERT, CharBERT and LSTM-CRF. CharBERT performs better when the data is noisy and requires less data to learn comparatively. Schweter et al. [24] experimented different word and sub word embeddings with transformer-based language models based on BiLSTM-CRF for historical German with tag set of person, location, organization, product and time. Best performance achieved with FastText embeddings trained on German Wikipedia in combination with a large German Bert language model with an F1-score of 65% on strict setting. Nissim et al. [18] experimented using maximum entropy tagger for recognizing location names. It achieved a good F-Score of 94.2% by performing a binary classification into location non-location. Neudecker et al. [20] trained a Stanford CRF model on newspaper material in French, German and Dutch with additional gazetteers. The 4-fold cross validation was performed and F-scores achieved in 70-80% range for Dutch and French. Ruokolainen et al. [21] trained Stanford CRF on manually corrected OCRed Finnish historical newspapers focusing Location and Person named entities. A large gazetteers with inflected forms are used to improve the performance. The system achieved an F-scores of 87% for location and 80% for person on a test set taken from the same manually corrected OCR, and of 78% and 71% on non-corrected OCR. Kim et al. [22] trained two Stanford NER models using CoNLL-03 data and articles from Trove archive. The model trained on domain data yielded an F-score of 73%, 72% and 37% respectively for Person, Location and Organization entities. Passaro et al. [19] adapted an Italian NER system developed using the Stanford NER to extract entities related to World War I & II texts through automatic creation of a new NE-annotated corpus. Military organizations, Ships and Airplanes were included in addition to traditional entity types. Locations were identified with a good F1 score while other entities achieved a very low F1 score.

2.9 Surveys or Studies on NER

Jing Li et al.[16] conducted survey on existing deep learning NER systems and summarized based on 3 aspects such as distributed representations for input, context encoder and tag decoder. Neural networks like CNN, RNN are used for context encoder to capture the context dependencies. CRF, RNN, pointer networks, etc. are used by Tag decoders to predict tags in input sequence. Input representations plays a main role in the success of NER system. Three types of distributed representations such as word-level, character-level, and hybrid representations are used in researches reviewed. Most common word level embedding used are Google Word2Vec, fastText, GloVe, and SENNA. CNN & RNN based models are widely used for character-level representation extraction. LSTM and GRU are two conventional choices of the basic units for RNN models. BiLSTM-CRF is the most common architecture used for NER using deep learning. Transformer encoder is better than LSTM when pretrained corpus is huge. The output of previous step is needs as the input of current for RNN and

pointer network decoders which is a major disadvantage. CRF is most commonly used as tag decoder as it capture label transition dependencies effectively when adopting non-contextualized embeddings such as Word2vec and GloVe. Training models from scratch using RNN and fine-tuning contextualized language models is a better option when the data is huge. Adopting transfer strategies is a better option when data is limited. Fine-tuning general-purpose contextualized language models with domain specific data could be a better way for domain specific NERs.

Malarkodi and Sobha [9] have performed a deeper study on features used for NER for various languages under the types of Indo-Aryan, Dravidian and European languages. POS patterns around the named entities is used as linguistic features. CRF technique is used to implement the NER system. POS for context window of size 3 is analyzed for named entities and most frequent grammatical and typographical feature that occurred for the three types of languages are identified. The grammatical patterns observed in Dravidian languages are named entity preceding and following Relative Participle verbs or common nouns, named entities occurring after verb or postpositions, verb succeed the named entities and postpositions, adjectives or adverbs following the named entities. The typological patterns observed are named entities at the beginning of the sentence or end of the sentence, Punctuations following named entities and named entities occurring after punctuation. The linguistic feature produced F-measure of 81.58% for Tamil.

CHAPTER 3

BACKGROUND

This chapter discusses about the analysis done on the domain and language related challenges in Question and Answer generation and Named Entity Recognition (NER) with examples, different NER approaches and its pros and cons, different libraries available for Tamil NLP which can be used in the system and the issues or limitations of those libraries and finally the feature sets that can be used in machine learning for NER. The content of this section is obtained from analyzing various resources from internet, summarized from the related work discussed in previous chapter and also from experimenting different libraries to choose the suitable ones to use in the system.

3.1 Challenges

There are domain related as well as language related challenges when building this Question and Answering system.

3.1.1 Domain related challenges

- Polysemic
  Single word referring to different named entities.
  E.g. Word German can refer country or language. Understanding the word context is important to identify the correct named entity.
- Nested Entities
  When a single word is considered, it belongs to one named entity type. But it can actually belong to another named entity when considered as multi word.
  E.g. ஜேர்மன் படைகள் (German force)
  'German' is a country, the whole word 'German force' indicates a troop.
- Context drift
  A country can occur with many prepositions in history context.
  E.g. ஜேர்மனியால் (By Germany)
- Naming conventions
  E.g. Names of Kings - 4 ஆம் லூயி மன்னன் (King Louis IV)

3.1.2 Language related challenges

- Free word order
  The positions of subject, verb and object will not affect the meaning of the sentence. Identification of named entities becomes difficult in such cases.
  E.g., In below sentences subject and object are interchanged. But still it provides the same meaning 'Kuwait was captured by Iraq'.
  ஈராக்கினால் குவைட் ஆக்கிரமிக்கப்பட்டது

குவைட் ஈராக்கினால் ஆக்கிரமிக்கப்பட்டது

- Sentences without subject, object or verb
  It is possible to form a sentence without having subject, object or verb in Tamil language. This generate meaningless questions and the number of features that can be extracted for named entity recognition also reduces.
- No capitalization
  In Tamil there is no capitalization which makes it complicated in identifying the NEs.
- Agglutination
  In Tamil, Words may contain different morphemes as the suffix of word. Tamil language is highly inflected and provide rich linguistic and statistical features. This makes the named entity recognition more complicated.
  E.g. நான் கோவிலுக்கு செல்கிறேன். (I am going to temple)
  'to temple' comes as single word 'கோவிலுக்கு' in Tamil with suffix 'கு'.
- Name or spell variations
  E.g. ஜேர்மன், ஜேர்மனி ஜெர்மனி, ஜெர்மன் – All these words indicate country Germany
- Ambiguity
  E.g. The words 'Haig' (name of a person) and 'Hague' (name of city) both are written in same way in Tamil which make it difficult to identify the correct NE when appear in a sentence.
  டக்ளஸ் ஹேக் – Douglas Haig
  ஹேக் நகரம் – Hague city
- Lack of Resources
  Standard corpora, gazetteer, NLP tools or libraries are rare or not available in Tamil. Even though quite a number of researchers are conducted for Tamil NLP tools very few are publicly available to use.

3.2 Named Entity Recognition approaches

Named Entity Recognition can be implemented using Rule based approach, Machine learning approach, Hybrid approach or Deep learning approach. Deep Learning is considered to be state of the art in NER systems.

3.2.1 Rule based approach

Rule based approaches are based on set of rules defined by language or domain experts to classify named entities. Rule based approaches may contain syntactic lexical patterns (e.g. POS tags, lemmatization of words), domain specific gazetteer etc.[16]

Pros: Works well when lexicon is exhaustive.[16] Suitable for domain specific systems.

Cons: Developing and maintaining gazetteers and rules is difficult and costly. Cannot be transferred to other domains due to domain specific rules & gazetteers.[16]

3.2.2 Machine learning approach

Machine learning approaches are based on statistical models to make predictions. These models requires large amount of annotated training data to be more effective which is work intensive and costly. Manually annotating data needs an extensive knowledge in the specific domain or language. There are subcategories of machine learning approaches as follows.

Unsupervised Learning

Unsupervised learning approaches learn patterns from unlabeled data using the machine learning algorithms.

Example: Clustering technique groups unlabeled data based on context similarity, corpus statistics like inverse document frequency and shallow syntactic knowledge like noun phrase chunking and so on.

Supervised Learning

Supervised learning approaches build predictive models based on the labelled data. Features are designed using morphological features, word-level features, context features, etc. to represent training data set. Machine learning algorithms are used to learn a model and identify patterns from untrained data.

The following are few widely used supervised learning approaches.

- Support Vector Machines: A data classification algorithm used to group data into two categories.

  Pros: Works best when the data set is small[15]. SVM supports Kernel functions, hence it learns various feature combinations with less computational complexity.[68]
  Cons: Does not perform well for large data set. It does not work well when the target classes are overlapping. State-to-state and feature-to-feature dependencies are not considered. [68]

- Hidden Markov Models: A statistical model that assumes features are independent to each other even though it considers future observations around entities for pattern learning. It identifies and learns the pattern using a sequence modeling algorithm.

Pros: Raw sequence data can be used for learning directly. Future observations are taken into account.[68]
Cons: There is no dependency between the words in input sentence. Due to the joint probability definition, many parameters has to be evaluated. Hence it requires a large data set for training.[68] HMM is not considered as a best technique for entity recognition when considering the performance aspect.

- Maximum Entropy models: A sequence modelling algorithm is used which consider the dependencies between neighboring states and the sequence and it does not assume that features are independent of each other and nor does it take future observations into account in pattern learning.

  Pros: can incorporate many features easily.

  Cons: Label bias cause a model to completely ignore the current observation when predicting the next label. [68] MEMM is not considered as a best technique for entity recognition when considering the performance aspect.

- Conditional Random Fields: A discriminative sequence model which assumes that features are dependent on each other and it also considers the future observations in pattern learning.

  Pros: CRF can accommodate any context information as it does not have strict independence assumptions. Feature design is flexible.[68] CRF provides solution to the label bias problem [68] as it computes the conditional probability of global optimal output nodes.

  Cons: CRF does high computations when training data.[68] When newer data becomes available, it becomes very difficult to retrain the model. CRF is not considered as a best technique for entity recognition when considering the performance aspect.

Semi supervised Learning

A model is trained initially on a set of labeled data and true labels. And then predictions are done on a separate unlabeled dataset. Improved models are created iteratively using predictions of previously generated models.

3.2.3 Hybrid Approach

Hybrid approach is a combination of Rule based and Machine Learning approach to achieve high accuracy. This approach does not need a large set of training data. Rule

based approach leads for accurate analysis by the guidance of human expertise and machine learning helps that analysis scale with ease.

3.2.4 Deep Learning

Deep Learning is based on Artificial Neural Networks which constantly receive learning algorithms and data to improve the efficiency of training.

Pros of Deep Learning (DL) compared to Machine Learning (ML) technique

- ML models create features manually or with domain expertise, which could introduce bias, whereas DL models automatically learn useful representations and underlying factors from raw data.[16]
- DL based models generates non-linear mappings from input to output & capable to learn complex features from data through non-linear transformation.[16]

Pros of Machine Learning (ML) techniques compared to Deep Learning (DL)

- ML model can be built using a small data set while Dl requires large dataset to perform better.[73]
- Computational cost is high. [73]

3.3 Feature set

The following features are summarized from literature review on Named Entity Recognition which is discussed in previous chapter.

- Root of words
- Bigram of Named Entity label - linear chain CRF is formed by binding the consecutive named entity tags
- Contextual length (window size)
- Roots of noun and verbs extracted from morphological analysis
- Surefire rules
    - Title prefix - If a word is prefixed with Mr., Mrs., Prof., Dr. , then it indicates a person named entity.
    - Set of preceding and proceeding words: Organization named entity may start or end with words department, organization, etc.
- Length of word
- Context word features
- First word – Check if the word is the first word of the sentence.
- The NE tags of the previous words - dynamic feature
- Word suffix: A predefined suffix list for each named entity

- Word prefix
- Part Of Speech (POS) tag of the word.
- Patterns – named entities like date, time, percentage, money, etc. follows a pattern.
- Word and POS tag combined - Current word combined with POS of previous two words and next two words.
- Dictionary of Named Entities – Checking if a stem word present in dictionary.
- Capitalization - Checks if a word starts with upper case. This is not applicable for Tamil language as no capitalization exist.
- Gazetteer Lists: List of Organization Suffix, Person Prefix, Action verb, Frequent word, Function words, Designation words, Person name, Location name, Organization name, Month name, Weekdays are used.

3.4 Named Entity Annotation Schema

IO: Simplest annotation scheme where each word from the dataset is assigned using either I tag or O tag. I tag is used for named entities and O tag is for other non-entity words. It is difficult to interpret if the tag belongs to same entity when consecutive entities of same type occurs.

IOB: Assigns B tag to the beginning of a named entity, I tag for the remaining consecutive chunks of the named entity and O tag for other non-entity words. There are two formats in IOB.

- IOB1- B tag is only used to separate two adjacent entities of the same type.
- IOB2 – All entities begin with B tag.

IOE: Assigns I tag for beginning or inside and E tag for the end of the named entity and O tag for other non-entity words. This has 2 formats.

- IOE1- E tag is used to tag the last word of a multi word named entity immediately preceding another named entity of the same named entity type.
- IOE2 – All entities ends with E tag.

IOBES: Assigns B tag for the beginning, I tag for inside, E tag for the end of the multi word named entity, O tag for other non-entity words and S tag to label single word entities.

BI: Similar to IOB. In addition, it labels the beginning of non-entity words with the tag B-O and the others as I-O.

IE: Similar to IOE. In addition, it labels the end of non-entity words with the tag E-O and the others as I-O.

BIES: Similar to IOBES. Additionally it uses B-O to tag the beginning of non-entity word, I-O to tag the inside of non-entity words and S-O for single non-entity words.

3.5 Analysis on NLP libraries

There are very few libraries available as open source for Tamil NLP techniques such as POS tagging, Morphological analysis, Stemming, Tamil text font-based Encode to Unicode Converter and other techniques. Three popular NLP libraries which supports many Indo-Aryan and Dravidian languages are iNLTK, Indic NLP Library and StanfordNLP. But there are few other libraries available in Tamil which gives good outcomes. Some of the libraries experimented and its limitations are discussed below in order to choose a better one for the NLP techniques used in the project.

3.5.1 Automatic Input Encoding & Conversion to Unicode

OpenTamil is an Open Source Tamil NLP Library for Python 3. Several NLP techniques are supported by this library such as transliterate, Tamil morse, Tamil stemmer, spell checker, and so on. Conversion from various encodings is also supported by this library. e.g. TSCII to Unicode etc.

The available modules of txt2unicode are below.

1. unicode2encode - This module converts Unicode text into other encodes.
2. encode2unicode – This module converts text into Unicode string. The following Tamil fonts can be converted to Unicode suing this module.

Figure 3.1: Fonts supported by Open Tamil library

3.5.2 Sentence Tokenizer

Indic NLP library supports word level as well as sentence level tokenization for many Indian languages. The below is a python code written to test and compare the sentence

tokenizer of Indic NLP library with normal split function for few sentences where '.' comes in the middle of sentences like dates and abbreviations.

```
from indicnlp.tokenize import sentence_tokenize
import re

if __name__ == '__main__':

    content = " ஐக்கிய அமெரிக்காவின் ஜனாதிபதி வூட்ரோ வில்சன் அதற்காக முனைப்புடன் செயற்பட்டார். " \
              "அதன்படி 1920. 01. 10 ஆந் திகதி அவ்வமைப்பு உருவானது அதன் பெறுபேறே ஆகும். இதன்போது " \
              "ஹிட்லரின் எஸ். எஸ் (S.S) இராணுவமும் கெஸ்டாபோ என்னும் இரகசிய பொலிசாரும் மிலேச்சத்தனமான" \
              " கொலைகளில் ஈடுபட்டனர். இவ்வாறு கி.மு. 1942 அளவில் இலங்கை, இந்தியா ஆகிய நாடுகளைத் தவிர " \
              "முழு தென் மற்றும் தென்கிழக்காசிய நாடுகளையும் ஐப்பான் கைப்பற்றியது."

    sentences = content.split(".")
    print("Sentences split by split function")
    for sentence in sentences:
        sentence = ' '.join(sentence.split())
        sentence = re.sub('\u200c', '', sentence)
        print(sentence)

    sentencesLib = sentence_tokenize.sentence_split(content, lang='tam')
    print("Sentences split by Indic NLP library")
    for sentence in sentencesLib:
        sentence = ' '.join(sentence.split())
        sentence = re.sub('\u200c', '', sentence)
        print(sentence)
```

Figure 3.2: Python code to test Indic NLP library sentence tokenizer

```
Sentences split by split function
ஐக்கிய அமெரிக்காவின் ஜனாதிபதி வூட்ரோ வில்சன் அதற்காக முனைப்புடன் செயற்பட்டார்
அதன்படி 1920
01
10 ஆந் திகதி அவ்வமைப்பு உருவானது அதன் பெறுபேறே ஆகும்
இதன்போது ஹிட்லரின் எஸ்
எஸ் (S
S) இராணுவமும் கெஸ்டாபோ என்னும் இரகசிய பொலிசாரும் மிலேச்சத்தனமான கொலைகளில் ஈடுபட்டனர்
இவ்வாறு கி
மு
1942 அளவில் இலங்கை, இந்தியா ஆகிய நாடுகளைத் தவிர முழு தென் மற்றும் தென்கிழக்காசிய நாடுகளையும் ஐப்பான் கைப்பற்றியது

Sentences split by Indic NLP library
ஐக்கிய அமெரிக்காவின் ஜனாதிபதி வூட்ரோ வில்சன் அதற்காக முனைப்புடன் செயற்பட்டார்.
அதன்படி 1920. 01. 10 ஆந் திகதி அவ்வமைப்பு உருவானது அதன் பெறுபேறே ஆகும்.
இதன்போது ஹிட்லரின் எஸ்.
எஸ் (S.
S) இராணுவமும் கெஸ்டாபோ என்னும் இரகசிய பொலிசாரும் மிலேச்சத்தனமான கொலைகளில் ஈடுபட்டனர்.
இவ்வாறு கி. மு. 1942 அளவில் இலங்கை, இந்தியா ஆகிய நாடுகளைத் தவிர முழு தென் மற்றும் தென்கிழக்காசிய நாடுகளையும் ஐப்ப
```

Figure 3.3: Output of Indic NLP library sentence tokenizer

The library works well for sentences that contain dates and few common abbreviations like B.C, A.D, etc., but for sentences that has domain specific abbreviations like S.S military as in above example, it does not work properly.

3.5.3 POS tagger

An R-based library known as RDRPOSTagger provides support for POS tagging in Tamil language. It is a ripple-down rule-based POS tagger, which comes with pre-trained POS tagging modules. RippleTagger is a slimmed down version of RDRPOSTagger which is python based.

```
from rippletagger.tagger import Tagger

if __name__ == '__main__':
    content ="உலகில் பலம் பொருந்திய நாடுகள் பலவும் இந்த யுத்தத்தில் ஈடுபட்டு முழு உலகிலும் தாக்கத்தை " \
             "ஏற்படுத்தியமையால் இவை உலக மகாயுத்தங்கள் என அழைக்கப்படுகின்றன. ஜேர்மன் பேரரசு ஒன்றை " \
             "உருவாக்குவதற்காக ஜேர்மனி இராணுவத்தை பலப்படுத்த நடவடிக்கை எடுத்தான். பெல்ஜியம், பிரான்ஸ், " \
             "போர்த்துக்கல், ஸ்பெயின் மற்றும் ஒல்லாந்து ஆகிய நாடுகள் வரிசைக்கிரமப்படி அதற்கடுத்த இடங்களை " \
             "வகித்தன. 1870 இல் பிரான்சை யுத்தத்தில் தோற்கடித்த ஜேர்மனியின் பிஸ்மார்க், பிரான்சிடம் இருந்த " \
             "அல்சேஸ் மற்றும் லொரையின் ஆகிய வளமான இரு பிரதேசங்களையும் கைப்பற்றினான். உலக யுத்தத்திற்கு " \
             "அடுத்த மிகப் பெரிய யுத்தம் என்பதால் பேரழிவுகள் ஏற்பட்டன"

    sentences = content.split(".")

    for sentence in sentences:
        sentence = ' '.join(sentence.split())
        tagger = Tagger(language='tam')
        postagger = tagger.tag(sentence)
        print(sentence)
        print('POS tag', postagger)
```

Figure 3.4: RippleTagger library testing code

```
உலகில் பலம் பொருந்திய நாடுகள் பலவும் இந்த யுத்தத்தில் ஈடுபட்டு முழு உலகிலும் தாக்கத்தை ஏற்படுத்தியமையால் இவை உலக மகாயுத்தங்கள் என
அழைக்கப்படுகின்றன.
POS tag [('உலகில்', 'NOUN'), ('பலம்', 'NOUN'), ('பொருந்திய', 'ADJ'), ('நாடுகள்', 'NOUN'), ('பலவும்', 'PRON'), ('இந்த', 'DET'), ('யுத்தத்தில்', 'NOUN'),
('ஈடுபட்டு', 'VERB'), ('முழு', 'ADJ'), ('உலகிலும்', 'ADJ'), ('தாக்கத்தை', 'NOUN'), ('ஏற்படுத்தியமையால்', 'NOUN'), ('இவை', 'NOUN'), ('உலக', 'PROPN'),
('மகாயுத்தங்கள்', 'NOUN'), ('என', 'PART'), ('அழைக்கப்படுகின்றன', 'VERB')]

ஜேர்மன் பேரரசு ஒன்றை உருவாக்குவதற்காக ஜேர்மனி இராணுவத்தை பலப்படுத்த நடவடிக்கை எடுத்தான்.
POS tag [('ஜேர்மன்', 'NOUN'), ('பேரரசு', 'NOUN'), ('ஒன்றை', 'NUM'), ('உருவாக்குவதற்காக', 'ADV'), ('ஜேர்மனி', 'PROPN'), ('இராணுவத்தை', 'NOUN'),
('பலப்படுத்த', 'ADJ'), ('நடவடிக்கை', 'VERB'), ('எடுத்தான்', 'VERB')]

பெல்ஜியம், பிரான்ஸ், போர்த்துக்கல், ஸ்பெயின் மற்றும் ஒல்லாந்து ஆகிய நாடுகள் வரிசைக்கிரமப்படி அதற்கடுத்த இடங்களை வகித்தன
POS tag [('பெல்ஜியம்,', 'NOUN'), ('பிரான்ஸ்,', 'NOUN'), ('போர்த்துக்கல்,', 'NOUN'), ('ஸ்பெயின்', 'NOUN'), ('மற்றும்', 'CONJ'), ('ஒல்லாந்து', '_'), ('ஆகிய', 'ADJ'),
('நாடுகள்', 'NOUN'), ('வரிசைக்கிரமப்படி', 'ADV'), ('அதற்கடுத்த', 'ADJ'), ('இடங்களை', 'VERB'), ('வகித்தன', 'VERB')]

1870 இல் பிரான்சை யுத்தத்தில் தோற்கடித்த ஜேர்மனியின் பிஸ்மார்க், பிரான்சிடம் இருந்த அல்சேஸ் மற்றும் லொரையின் ஆகிய வளமான இரு
பிரதேசங்களையும் கைப்பற்றினான்
POS tag [('1870', 'NUM'), ('இல்', 'NOUN'), ('பிரான்சை', 'NOUN'), ('யுத்தத்தில்', 'NOUN'), ('தோற்கடித்த', 'ADJ'), ('ஜேர்மனியின்', 'NOUN'),
('பிஸ்மார்க்,', 'NOUN'), ('பிரான்சிடம்', 'NOUN'), ('இருந்த', 'ADJ'), ('அல்சேஸ்', 'PROPN'), ('மற்றும்', 'CONJ'), ('லொரையின்', 'NOUN'), ('ஆகிய', 'ADJ'),
('வளமான', 'ADJ'), ('இரு', 'NUM'), ('பிரதேசங்களையும்', 'VERB'), ('கைப்பற்றினான்', 'VERB')]

உலக யுத்தத்திற்கு அடுத்த மிகப் பெரிய யுத்தம் என்பதால் பேரழிவுகள் ஏற்பட்டன
POS tag [('உலக', 'PROPN'), ('யுத்தத்திற்கு', 'NOUN'), ('அடுத்த', 'ADJ'), ('மிகப்', 'DET'), ('பெரிய', 'ADJ'), ('யுத்தம்', 'NOUN'), ('என்பதால்', 'PART'),
('பேரழிவுகள்', 'VERB'), ('ஏற்பட்டன', 'VERB')]
```

Figure 3.5: Output of RippleTagger library

For complicated sentences as in above examples (i.e. sentence with compound verbs, lengthy sentences, multiple sentences joined together as one sentence, etc.) the results are not accurate. But for simple sentences this pos tagger gives very good results.

3.5.4 Tamil stemmer

Stemming is a process where words are reduced to a root by removing inflection. Snowball stemmer is a library which supports for stemming in Tamil language. Python 2 and Python 3 (>= 3.3) are supported.

```
import snowballstemmer

if __name__ == '__main__':
    words = ["ஜேர்மனியும்", "ஜேர்மனியினை", "சேர்பியாவுக்கு", "பிரதேசத்தில்", "இத்தாலி", "நாட்டில்", "இராணுவத்தை",
             "யுத்தங்களினால்", "நூற்றாண்டின்", "நாடுகளான", "பிஸ்மார்க்", "பிஸ்மார்க்கின்","சரஜிவோ", "நகரத்தில்",
             "ஆஸ்திரியாவின்", "நாடாகிய",  "ரஷ்யாவுக்கும்", "பிரான்சுக்கும்", "ஐக்கிய", "அமெரிக்கா", "லூசிடானியா",
             "இத்தாலியின்", "ரஷ்யா", "நேசநாடுகள்", "உலகில்", "ஐரோப்பாவில்", "ஆபிரிக்கப்", "பிரதேசங்களில்",
             "ஜேர்மனி", "இராணுவம்"]

    for word in words:
        stemmer = snowballstemmer.stemmer('tamil')
        stemWord = stemmer.stemWords(word.split())
        print(word + " => " + stemWord[0])
```

Figure 3.6: Snowball stemmer library testing code

```
ஜேர்மனியும் => ஜேர்மனி, ஜேர்மனியினை => ஜேர்மனியி, சேர்பியாவுக்கு => சேர்பி, பிரதேசத்தில் => பிரதேசம், இத்தாலி => தாலி,
நாட்டில் => நாடு, இராணுவத்தை => இராணுவம், யுத்தங்களினால் => யுத்தம், நூற்றாண்டின் => நூற்றாண், நாடுகளான => நாடுகள்,
பிஸ்மார்க் => பிஸ், பிஸ்மார்க்கின் => பிஸ், சரஜிவோ => சரஜி, நகரத்தில் => நகரம், ஆஸ்திரியாவின் => ஆஸ்திரி, நாடாகிய => நா,
ரஷ்யாவுக்கும் => ரஷ்யா, பிரான்சுக்கும் => பிரான், ஐக்கிய => ஐக்கி, அமெரிக்கா => அமெரி, லூசிடானியா => லூசிடானி,
இத்தாலியின் => தாலி, ரஷ்யா => ரஷ், நேசநாடுகள் => நேசநாடு, உலகில் => உல, ஐரோப்பாவில் => ஐரோ, ஆபிரிக்கப் => ஆபிரி,
பிரதேசங்களில் => பிரதேசம், ஜேர்மனி => ஜேர்மனி, இராணுவம் => இராணுவம்
```

Figure 3.7: Output of snowball stemmer library

This stemmer library find the root word for nouns with inflections correctly most of the time. But this library does not stem proper nouns correctly. It also perform stemming in noun without inflection in below cases.

- if the word ends with suffix which is used in inflected word கு, உக்கு, இல், இன், ஐ, ஆல்
- if the word starts with prefix which is used in determiner words இ, அ similar to ‘this’, ‘that’.

3.5.5 Morphological Analyzer

Morphological Analysis is more suitable for morphologically rich languages like Tamil which has more inflections in the words. This library finds the root words correctly when inflected word does not modify the root word, but only ends with inflection suffix. But if the root word is modified in inflected word, it does not properly identify the root word. The library does not stem the non-inflected word correctly in most of the cases unlike in Snowball stemmer discussed previously. E.g. non-inflected country names and person names. Also the performance of this library is very low. It process for a very long time to find the morphemes of even a single word.

Below is the testing code and output of Indic NLP library which supports morphological analysis for Tamil.

```python
from indicnlp.tokenize import sentence_tokenize
import re

if __name__ == '__main__':

    content = " ஐக்கிய அமெரிக்காவின் ஜனாதிபதி ரூட்ரோ வில்சன் அதற்காக முனைப்புடன் செயற்பட்டார். " \
              "அதன்படி 1920. 01. 10 ஆந் திகதி அவ்வமைப்பு உருவானது அதன் பெறுபேறே ஆகும். இதன்போது " \
              "ஹிட்லரின் எஸ். எஸ் (S.S) இராணுவமும் கெஸ்டாபோ என்னும் இரகசிய பொலிசாரும் மிலேச்சத்தனமான" \
              " கொலைகளில் ஈடுபட்டனர். இவ்வாறு கி.மு. 1942 அளவில் இலங்கை, இந்தியா ஆகிய நாடுகளைத் தவிர " \
              "முழு தென் மற்றும் தென்கிழக்காசிய நாடுகளையும் ஜப்பான் கைப்பற்றியது."

    sentences = content.split(".")
    print("Sentences split by split function")
    for sentence in sentences:
        sentence = ' '.join(sentence.split())
        sentence = re.sub('\u200c', '', sentence)
        print(sentence)

    sentencesLib = sentence_tokenize.sentence_split(content, lang='tam')
    print("Sentences split by Indic NLP library")
    for sentence in sentencesLib:
        sentence = ' '.join(sentence.split())
        sentence = re.sub('\u200c', '', sentence)
        print(sentence)
```

Figure 3.8: Indic NLP MA library testing code

```
['ஜேர்மனி', 'யும்'],     ஜேர்மனியும் => ஜேர்மனி
['ஜேர்மனி', 'யினை'], ஜேர்மனியினை => ஜேர்மனி
['சேர்', 'பியா', 'வுக்கு'],  சேர்பியாவுக்கு => சேர்
['பிரதேசத்தில்'],    பிரதேசத்தில் => பிரதேசத்தில்
['இத்தாலி'],    இத்தாலி => இத்தாலி
['நாட்டில்'],    நாட்டில் => நாட்டில்
['இராணுவ', 'த்தை'],  இராணுவத்தை => இராணுவ
['யுத்த', 'ங்களினால்'], யுத்தங்களினால் => யுத்த
['நூற்றாண்டி', 'ன்'],    நூற்றாண்டின் => நூற்றாண்டி
['நாடு', 'களான'], நாடுகளான => நாடு
['பிஸ்மார்க்'],  பிஸ்மார்க் => பிஸ்மார்க்
['பிஸ்மார்க்', 'கின்'],  பிஸ்மார்க்கின் => பிஸ்மார்க்
['சர', 'ஜி', 'வோ'],  சரஜிவோ => சர
['நகர', 'த்தில்'],   நகரத்தில் => நகர
['ஆஸ்திரிய', 'ாவின்'],     ஆஸ்திரியாவின் => ஆஸ்திரிய
['நாடாகிய'],   நாடாகிய => நாடாகிய
['ரஷ்யா', 'வுக்கும்'],   ரஷ்யாவுக்கும் => ரஷ்யா
['பிரான்சு', 'க்கும்'],   பிரான்சுக்கும் => பிரான்சு
['ஐக்கிய'], ஐக்கிய => ஐக்கிய
['அமெரிக்கா'], அமெரிக்கா => அமெரிக்கா
['லூசி', 'டா', 'னியா'],    லூசிடானியா => லூசி
['இத்தாலி', 'யின்'],    இத்தாலியின் => இத்தாலி
['ரஷ்யா'], ரஷ்யா => ரஷ்யா
['நேச', 'நாடுகள்'],     நேசநாடுகள் => நேச
['உலகில்'],     உலகில் => உலகில்
['ஐரோப்பா', 'வில்'],   ஐரோப்பாவில் => ஐரோப்பா
['ஆபிரிக்க', 'ப்'], ஆபிரிக்கப் => ஆபிரிக்க
['பிரதேசங்களில்'], பிரதேசங்களில் => பிரதேசங்களில்
['ஜேர்மனி'],    ஜேர்மனி => ஜேர்மனி
```

Figure 3.9: Indic NLP MA library output

Another python based library known as Polyglot also supports Morphological Analysis for Tamil. It also does not produce better results but it performs very faster compared to Indic NLP MA library.

CHAPTER 4

METHODOLOGY

This chapter focuses on the methodology of the system in a descriptive manner. It includes the high level architecture of the system and explains each module as a top down approach. Each components of the system described along with its technologies, inputs, outputs, algorithms, limitations etc.

4.1 High Level Architecture

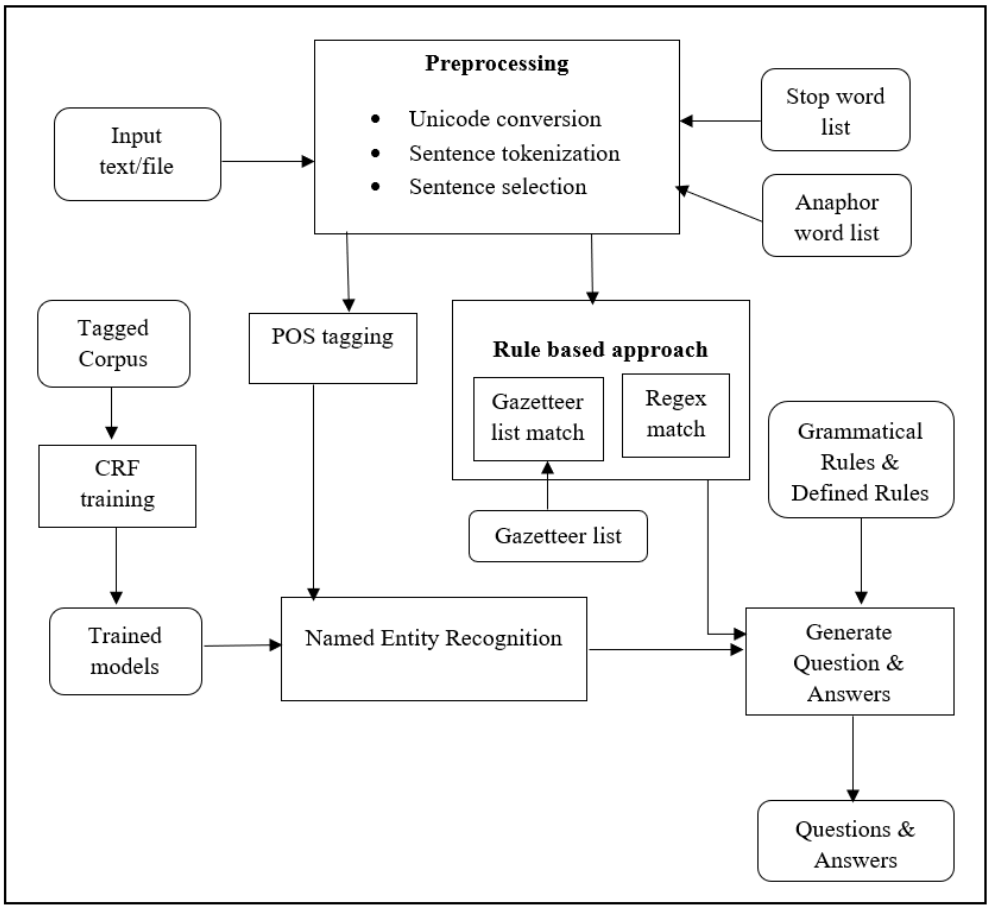


Figure 4.1: High level Architecture of the system

The system generates questions and answers on history domain in Tamil language. The user has to input text or upload file with content on the domain of interest. The system processes the input using various NLP techniques and produces questions and answers. The system also enables the user to download the generated question and answers in a text document.

4.1.1 Input

Users can paste a text or upload a file with one of the file format .doc, .txt or .pdf. The file size should be less than 10MB and users can upload one document at a time.

4.1.2 Process

1. If file is uploaded, the file size and file format is validated. If valid then it process the file. If invalid it prompts appropriate error message.
2. The text content or file content is read and tokenized into list of sentences.
3. The sentences which starts with anaphora words (e.g.: He, She, It, There, That) are ignored as not suitable for question generation.
4. Regex pattern matching and gazetteer match is done for each tokenized sentences in Rule-based module.
5. Regex match is checked to identify date, time and quantity in the sentence. The matching word is then replaced with the appropriate question word. This rule based approach generates 'how many' and 'when' type of questions.
6. Regex pattern for comma separated words of same entity, words enclosed with single quote are also checked against sentence and is replaced with blank line if match found.
7. Then words in the sentence are searched in gazetteer list. According to the named entity category it belongs to, the appropriate question word is replaced.
8. Each sentence tokenized in step 2 is tagged with POS tags. Using the trained models and POS tag of the word, the named entity of each word in each sentence in the list is predicted.
9. The appropriate question word is identified for the named entity tags predicted for that sentence using some defined rules.
10. The named entity is checked for inflections and if the word is inflected, the question word with inflected form is replaced. When forming the question previous and next words are considered into account to form a meaningful questions.

4.1.3 Output

The questions and answers generated from the rule based approach and machine learning approach are written to a file every time. Finally the questions and answers in the file is loaded in the text area in the application. The user can download the questions and answers as a text file if needed.

4.2 System Overview

The system is mainly divided into below 4 components: Preprocessing module, Rule based module, Named Entity Recognition module and Question and Answer generation module.

4.2.1 Preprocessing Module

Text pre-processing is a process of transforming text into a more digestible form to be used for further processing. It consists of following steps.

- Read file content
- Unicode conversion
- Tokenization
- Sentence selection

Read file content

If a valid file is uploaded, the content of the file is extracted for further processing. If pdf file is uploaded, then the file content is extracted using a python library called PYPDF2.

Limitations: The text in images, bullet points and headings will not be processed.

Unicode conversion

Different tamil encodings are available. OpenTamil is used to convert any encoding to Unicode. This library supports 25 different tamil encodings. (Refer chapter 3.5.1)

Limitations: If a tamil text with an encoding not supported by the library is uploaded, then the system will not generate questions and answers.

Tokenization

The text content is tokenized into list of sentences using Indic NLP library. Indic NLP is one of the python based library that can be used for tokenization in Tamil language. This library works better than split function in python when dates and common abbreviations with '.' present in the sentence. (Refer chapter 3.5.2) e.g.: கி.மு(B.C), கி.பி(A.D), 19.11.1948

Limitations: The library does not tokenize correctly when domain specific abbreviations are present in the text.

Sentence Selection

The sentences suitable for question and answer generation are selected by checking the first word of the sentence against a list of anaphora word list. Anaphora is the linguistic phenomenon for referring back to a noun or pronoun in previous sentences. The list used in the project contains around 50 anaphora words.

E.g. Adolf Hitler was born in Austria. He was the leader of Nazi party. 'He' in the second sentence refer to Adolf Hitler which is referenced from the first sentence. So it is an anaphora word.

Limitations: If an anaphora word is present in the middle of sentences, the sentence will not be ignored. Hence it can produce meaningless questions.

4.2.2 Rule based Module

Rule based approach is applied to the selected sentences to generate questions using regex patterns and gazetteers.

Regex pattern matching

Regex match is checked to identify date, time and quantity in the sentence. The matching word is then replaced with the appropriate question word to form questions of type 'when' and 'how many'. Regex pattern for comma separated words of same entity, words enclosed with single quote are also checked against sentence and is replaced with blank line if match found. Table 6.1 shows the sample regex patterns used in the system.

Example:

Sentence: வருடம் 1919 ல் மன்னர் கெய்சர் முடி துறந்த பிறகு ஜெர்மனியின் மன்னராட்சி முடிவுக்கு வந்தது. (The monarchy in Germany came to an end in 1919 when King Kaiser ended his reign)

Question: எந்த வருடம் மன்னர் கெய்சர் முடி துறந்த பிறகு ஜெர்மனியின் மன்னராட்சி முடிவுக்கு வந்தது? (Which year the German monarchy came to an end when King Kaiser ended his reign?)

Sentence: 20 ஆம் நூற்றாண்டாகும்போது சீனா, இந்தியா, இலங்கை முதலிய ஆசிய நாடுகளிலும் கைத்தொழில் புரட்சியின் செல்வாக்கு பரவியது. (By the 20th century, the influence of the Industrial Revolution had spread to Asian countries such as China, India, and Sri Lanka)

Question: 20 ஆம் நூற்றாண்டாகும்போது _____, _____, _____ முதலிய ஆசிய நாடுகளிலும் கைத்தொழில் புரட்சியின் செல்வாக்கு பரவியது. (By the 20th century, the influence of the Industrial Revolution had spread to Asian countries such as _____, _____ and _____.)

Gazetteer search

A gazetteer list for few named entities are defined. The words in the sentence is checked against this list and if a word is found in an array the corresponding key is returned. According to the returned key, the question word is replaced. The below are the named entities for which a gazetteer list is defined.

- Month - List of 12 months
- Kingdom - List of all Srilankan kingdoms
- City - List of historic cities and capital cities
- Person - List of famous kings, emperors, rulers
- Country - List of all popular and historically important countries

- Numeric word - List of numeric words under categories of units, tens, hundred, thousand scales are defined. (Figure 6.7)

Limitations: If a multi word named entity exist in gazetteer, then the whole word is expected to match in the sentence to be replaced.
E.g. ஐக்கிய அமெரிக்கா (United states of America)

Same word with different spelling will not be identified as a match.
E.g: Germany can be written as ஜேர்மனி, ஜெர்மனி.

4.2.3 Named Entity Recognition Module

Named entity recognition (NER) is used to classify named entities in text into pre-defined categories such as the names of persons, organizations, locations, date, time, quantities, etc.[15] The different approaches of Named Entity Recognition and its pros and cons are discussed in chapter 3.3.

Machine learning approach is used for the implementation of NER in the developed system. The main advantage of this approach over the rule-based approach is that it is trainable and can be adapted to different domains and languages and the maintenance cost is comparatively less. So this approach is more suitable for our application.

Conditional Random Field (CRF) classifier is used in the system. Different features suitable for the domain of interest and language are selected and are helpful in predicting the various named entity types. Corpus data is split into trained and test data where train data is used to build the model and test data is evaluated with the model. Hyper parameter optimization is applied to obtain the best model.

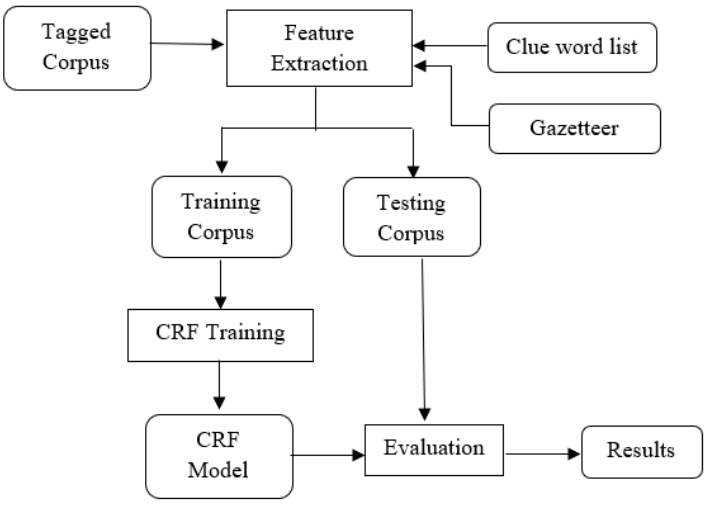


Figure 4.2: High Level diagram of NER system

Named Entity tag set

Most of the NER researches done for history domain are also focused on few popular entities like Location, Person, and Organizations from historical documents. Hence named entity tag set are defined by analyzing history textbooks used for corpus tagging. Named entity tag set used in the system are mostly domain specific.

Table 4.1: Named Entity tags with examples

NE Tag	Description	Examples
PER	Names of people like kings, leaders, etc.	1 ஆம் எலிசபெத் (Elizabeth I)
NORP	Nationalities or religious or political groups.	பிரித்தானியர் (British)
DES	Designations (Job positions) such as king, general, governor, etc.	கப்பல் மாலுமி (Ship Sailor) ஆளுநர் (Governor)
GOD	Names of god	புத்த பெருமான் (Lord Buddha) இயேசுநாதர் (Jesus Christ)
REL	Religion	இந்து சமயம் (Hinduism), இஸ்லாம் (Islam)
CONT	Names of continents in world	ஆஸ்திரேலியாக் கண்டம் (Australian continent) ஐரோப்பா (Europe)
COU	Names of countries in world	ஜேர்மனி (Germany)
TER	Territory - a division of a country like state, province, district, region etc.	தென் வேல்ஸ் (South Wales) சுமாத்ரா தீவு (Sumatra Island)
CITY	Names of cities and towns	பரிஸ் (Paris)
LAW	Includes laws, rules, policies, reforms, declarations, and agreements	கோல்புறூக் சீர்த்திருத்தங்கள் (Colebrook reforms) சீனிச் சட்டம் (Sugar Act) அமெரிக்கச் சுதந்திரப் பிரகடனம் (Unites States Declaration of Independence)
CON	Includes concepts, principles, theory, thought, etc.	நாசிசவாதம் (Nazism) நிலமானிய முறை (Feudalism)
TIME	Time period like B.C, A.D, century, decade, etc.	இருபதாம் நூற்றாண்டின் முன்னரைப் பகுதி (First half of twentieth century)
YEAR	Year in numeric or text form	1918 ஆம் ஆண்டு (Year of 1918)
MON	Names of month	செப்டெம்பர் மாதம் (Month of September)
DATE	Dates in numeric or text form	7 ஆம் திகதி (7^{th}) 1918.08.07

COM	Community – group of people, society	கிளர்ச்சியாளர்கள் (Insurgent)
WAT	Names of water bodies like ocean, river, reservoir, canal, etc.	சுயஸ் கால்வாய் (Suez canal) மகாவலி கங்கை (Mahaweli River)
TAX	Types of taxes	முத்திரை வரி (Stamp duty)
ANT	Antiquity - relics or monuments (such as coins, statues, artwork or buildings) of ancient times	புனித தந்தம் (Sacred tooth relic)
EQUIP	Equipment– weapons, machines, etc.	திசையறி கருவி (Compass)
ORG	Names of organization like company, council, department, commission, committee, etc.	உலக நாடுகள் சங்கம் (League of Nations)
TRO	Troop – Names of armed forces	எஸ்.எஸ் இராணுவம் (SS military unit)
GOV	Name of government, party, empire and federation	பாசிசவாத அரசியல் கட்சி (The National Fascist Party)
CUR	Names of currencies	டாலர் (dollar)
FAC	Facilities like road, building, motor vehicles, etc.	விமான நிலையங்கள் (Airports)
CROP	Names of crop like tea, rubber, coconut, etc.	தேயிலை (Tea), இறப்பர் (Rubber)
IND	Types of industries	இரும்புருக்குக் கைத்தொழில் (Iron Industry)
SEC	Types of industrial sectors	கல்வித் துறை (Education sector)
SKILL	Types of skills	விஞ்ஞானத் தொழினுட்ப அறிவு (Scientific Technical Knowledge)
SOU	Types of sources contains historical information	நாணயங்கள் (coins), கல்வெட்டுகள் (Inscriptions)
PRO	Products	உரம் (Fertilizer)
NUM	Numeric in digits or text form	மூன்று (three), 150
LIT	Names of Literature like Ancient fiction, books etc.	மெயின்காம்ஃப் (Main-kamph) மகாவம்சம் (Mahavamsa)
LOC	Names of location like harbor, temple, camp, parliament etc.	அநுராதபுர மகா விகாரை (Maha Viharaya, Anuradhapura)
LAN	Names of languages	டச்சு மொழி (Dutch)
EVE	Name of events like war, revolution, ritual, phenomenon, etc.	கைத்தொழில் புரட்சி (Industrial Revolution) சிலுவை யுத்தங்கள் (Crusades)
SUB	Names of subjects	மெய்யியல் (Philosophy)

A flat annotation schema, IOB2 is used which assigns B tag (B-prefix before a tag) to the beginning token of a multiple word named entity, I tag (I-prefix before a tag) to indicate that it is an inside token or ending token in a multi word named entity and O tag for other non-entity words.

E.g: Adolf (B-PER) Hitler (I-PER) was (O) a (O) German (B-NORP) politician (B-DES) who (O) was (O) the (O) dictator (B-DES) of (O) Germany (B-COU) from (O) 1933 (B-YEAR) to (O) 1945 (B-YEAR).

POS Tagging

POS tagging is a process of labelling the part of speech tag such as noun, verb, adverb, adjective, pronoun, conjunction, etc. for the words in a corpus based on its context and definition.

POS tagging is done using a python library named RippleTagger which has an accuracy of 82.08886853% for Tamil language as per the documentation of this library. This is the only publicly available python library for POS tagging for Tamil language at the time of this research implemented. This library supports Universal POS tags given below. If none of the given POS tag is identified for a given word '-' is tagged. Analysis of this library with sample code and results is discussed in chapter 3.6.

Table 4.2: POS tags supported by Ripple Tagger library

POS Tag	Description
NOUN	Noun
VERB	Verb
ADJ	Adjective
ADV	Adverb
PROPN	Proper noun
PRON	Pronoun
AUX	Auxiliary verb
PUNCT	Punctuation
PART	Part
CONJ	Coordinating Conjunction
DET	Determiner
NUM	Numerical
ADP	Ad position

Conditional Random Fields

Conditional Random Field is a discriminative model, used for predicting sequences. It increases the amount of information needed for the model to make a good prediction by using the contextual information from previous labels.

Mathematical Overview of CRF

In CRF, the input data is sequential, and previous context is considered when making predictions on a data point. Feature Functions are used to model this behavior.

Feature function: $f(X, i, l_{i-1}, l_i)$

X - Set of input vectors

i - Position of data point

l_{i-1} - Label of data point i-1 in X

l_i - Label of data point i in X

For example, $f(X, i, L\{i - 1\}, L\{i\}) = 1$, if L{i - 1} is a Noun and L{i} is a Verb, **0** otherwise. Each feature function is based on the label of the previous word and the current word.

Each feature function is assigned with a set of weights known as lambda values, which the algorithm going to learn.

$$P(y, X, \lambda) = \frac{1}{Z(X)} exp\{\sum_{i=1}^{n} \sum_j \lambda_j f_i(X, y_{i-1}, y_i)\}$$

$$\text{Where } Z(X) = \sum_{y' \epsilon y} \sum_{i=1}^{n} \sum_j \lambda_j f_i(X, y'_{i-1}, \acute{y}_i)$$

To estimate the parameters (lambda), Maximum Likelihood Estimation is used. To apply the technique, the Negative Log of the distribution should be taken first to make the partial derivative easier to calculate as below.

$$L(y, X, \lambda) = -log \prod_{k=1}^{m} P(y^k | x^k, \lambda)\}$$

$$= - \sum_{k=1}^{m} log[\frac{1}{Z(x_m)} exp\{\sum_{i=1}^{n} \sum_j \lambda_j f_j(X^m, i, y_{i-1}^k, y_i^k)]$$

To apply Maximum Likelihood on the Negative Log function, the argmin is taken as minimizing the negative will yield the maximum. To find the minimum, we need to take the partial derivative with respect to lambda.

$$\frac{\partial L(X,y,\lambda)}{\partial \lambda} = \frac{-1}{m}\sum_{k=1}^{m} F_j(y^k, x^k) + \sum_{k=1}^{m} p(y|x^k, \lambda)\, F_j(y, x^k)$$

Where: $F_j(y, x) = \sum_{i=1}^{n} f_i(X, i, y_{i-1}, y_i)$

Partial Derivative is used as a step in Gradient Descent. Gradient Descent updates parameter values iteratively until the values converge. Final Gradient Descent update equation for CRF is as below.

$$\lambda = \lambda + \alpha \left[\sum_{k=1}^{m} F_j(y^k, x^k) + \sum_{k=1}^{m} p(y|x^k, \lambda)\, F_j(y, x^k)\right]$$

Libraries and Algorithm

There are several CRF toolkits available. Among them CRF++ and CRFSuite are the most popular. CRFsuite is more robust and faster-to-train. CRF++ needs the features to be set in files, but CRFsuite can calculate features in the training. Therefore CRFsuite is used for implementation in the system. Several Python libraries provide support to CRFsuite. Sklearn-crfsuite library is used for the implemented system. For the NER task, which is basically a sequence prediction task, the chain CRF is more suitable. Therefore, lbfgs CRF (Limited-memory Broyden-Fletcher-Goldfarb-Shanno) algorithm is chosen.

Features

Feature selection is an important task in CRF implementation. Different combinations of features suitable for the domain of interest and language are selected and experimented. Chapter 7 further discusses the testing results and evaluation of the different combinations experimented. The following are the features used for the system implemented.

- Inflection Suffix
  As the tamil words are highly inflected, the inflection suffix is used as a feature.
  Inflection suffix: ஐ, ஆல், கு, இல், இன், அது, உம், மீது உடன், ஆகிய, இருந்து
  Few named entities will not be inflected with few suffix. Hence this feature can be helpful in identifying named entities.
  E.g. the suffix 'இல்', 'ஆகிய' is not possible for a person name.

- Context feature of window size
  The previous and next words can help to predict the named entity.
  E.g. In the sentence segment, 'ஜேர்மனி என்ற நாடு' (country named Germany), considering the next 2 words can be helpful.

- Numeric word
  Check if the word is in digits and a list of numeric words are defined to check if a word indicates a numeric word.

- Clue words
  Few words are observed preceding or proceeding a named entity which is helpful in identifying the named entity.
  Example: The person's name can precede or proceed with word 'King' and word 'continent' can proceed with the continent name.

- Root word
  As the tamil words are highly inflected, root words obtained is used as a feature.

- Gazetteers
  The gazetteer list is prepared for few NE tags like countries, cities, person names, months, kingdoms, continents, etc. The word is checked against a gazetteer list with and without stemming the word as the stemming library output is not very accurate.

- Prefix suffix Rule
  Named entities can occur with certain prefix or suffix.
  E.g. NORP entity which refers to nationalities has more chance to appear with suffix 'அர்'
  பிரித்தானியர், ஐரோப்பியர்

- Is previous word ending with comma
  When comma separated multiple words appear there is a high chance of named entity of the current word is the same of previous word.
  E.g. ஹிட்லர் போலந்தை ஆக்கிரமித்ததோடு பிரித்தானியா, பிரான்ஸ் போன்ற நாடுகள் ஹிட்லருக்கு எதிராக யுத்தத்தை ஆரம்பித்தன. (Hitler invaded Poland and countries like Britain, France started war against Hitler)
  The words 'பிரித்தானியா' (Britain) 'பிரான்ஸ்' (France) both are country named entity.

- IsYear Format - word is checked against a date regex pattern.

- Part-of-Speech (POS) tag
  POS tags of previous and next words of window size 5 is experimented along with the POS tag of the predicting word.

- Named Entity of previous word - dynamic feature.
- Is conjunction word
  E.g. மற்றும்', 'ஆகிய', 'போன்ற are conjunction words that appear proceeding comma separated named entities of same type.

Hyper Parameter Tuning

Hyper parameter tuning aims to find hyper parameter which provides highest performing model and lesser error rate. Two most common and simplest optimization algorithms are Random Search and Grid Search. Grid Search evaluate model with every possible combination of hyper parameters while Random Search evaluates with random combination of hyperparameters. Even though Random Search yields a high variance during computing, it is proven to yield better results comparatively and Grid Search is very computationally intensive. Hence RandomizedSearchCV is used and experimented with different folds and iterations.

4.2.4 Question and Answer Generation Module

For regex matching approach, a replacing question word defined will be replaced if a match is found. For gazetteer approach or Named Entity Recognition approach, some steps are followed to apply some defined rules to generate a meaningful question.

Steps for Question Generation

Following are the steps for question generation from gazetteer approach.

- The sentence, identified named entity, named entity type, previous word, next word are given as input for this module from Rule based module.
- The previous and next words are stemmed using Snowball Stemmer library.
- The stemmed words are checked against the clue word list.
- If previous or next words are clue words, then the appropriate question word is replaced with inflection suffix in the place of word and previous or next word which contains the clue word.

  Example:

  Sentence: 1914 முதல் 1918 வரை நடந்த முதலாவது உலக மகாயுத்தம் ஜேர்மனி நாட்டின் தலைமையிலான அச்சு நாடுகளின் தோல்வியுடன் முடிவுக்கு வந்தது.
  (The First World War, which lasted from 1914 to 1918, ended with the defeat of the Axis powers led by the country of Germany.)
  The word 'ஜேர்மனி' (Germany) is identified as a country named entity from gazetteer and it follows the word 'நாட்டின்'(of the country) which is inflected. The stem word of the following word is 'நாடு'(country) and it is present in the

clue word list of named entity ‘country’. Hence the identified named entity word with its following word is replaced with a question word.
‘ஜேர்மனி நாட்டின்’(country of Germany) is replaced with the question word ‘எந்த நாட்டின்’ (In which country)
Question: 1914 முதல் 1918 வரை நடந்த முதலாவது உலக மகாயுத்தம் **எந்த நாட்டின்** தலைமையிலான அச்சு நாடுகளின் தோல்வியுடன் முடிவுக்கு வந்தது? (The First World War which lasted from 1914 to 1918 ended with the defeat of the Axis powers led by which country?)

- If previous or next words are not clue words, then the appropriate question word is replaced with inflection suffix in the place of identified named entity word
  Example:
  Sentence: 1914 ஆகஸ்ட் முதலாம் திகதி ஜேர்மனி ரஷ்யாவுக்கெதிராக யுத்தப் பிரகடனம் செய்தது. (In 1914 August 1st, the Germany declared a war against Russia.)
  The word ‘ஜேர்மனி’ (Germany) is identified as a country named entity from gazetteer and it does not proceed or precede any clue words. Hence the named entity word is replaced with appropriate question word ‘எந்த நாடு’ (Which country).
  Question: 1914 ஆகஸ்ட் முதலாம் திகதி **எந்த நாடு** ரஷ்யாவுக்கெதிராக யுத்தப் பிரகடனம் செய்தது? (Which country declared war against Russia in 1st of August 1914?)

Following are the steps for question generation from NER approach.

- The sentence and array of named entity tags of words in sentence are given as input for this module from NER module.
- The words are checked from left to right for the named entity tags.
- If a named entity tag present with B-tag, the following I-tags are checked and the named entity word is retrieved.
- Similar to gazetteer approach, if the previous and next words are clue words, the question word is formed with the inflection suffix of the clue word and replaced in place of named entity and the previous or next word .
- If the previous or next word is not a clue word, then the inflection suffix of named entity is obtained. The question word is combined with the inflection suffix and replaced in place of the inflected named entity. If the named entity is not inflected the question word is directly replaced.

Example: There are 2 possible questions that can be generated from the system for the below sentence.

Sentence: ஆஸ்திரியா நாட்டின் இளவரசரான பிரான்சிஸ் பெர்டினாண்டும், அவருடைய மனைவியும் காரில் சென்ற போது சுட்டுக் கொல்லப்பட்டனர். (Francis Ferdinand, the prince of Austria and his wife was assassinated in Sarajevo.)

NE Tags: ['B-COU', 'O', 'O', 'O', 'B-PER', 'I-PER', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

Question 1: எந்த நாட்டின் இளவரசரான பிரான்சிஸ் பெர்டினாண்டும், அவருடைய மனைவியும் காரில் சென்ற போது சரோஜிவாவில் சுட்டுக் கொல்லப்பட்டனர்? (which country's prince and his wife was assassinated in Sarajevo?)

The word 'ஆஸ்திரியா' (Austria) is identified as country named entity and it is followed by a clue word நாட்டின் (of country) in inflected form. Hence the question word 'எந்த நாட்டின்' is replaced in the place of named entity and the next word.

Question 2: ஆஸ்திரியா நாட்டின் எந்த இளவரசனும் அவருடைய மனைவியும் காரில் சென்ற போது சரோஜிவாவில் சுட்டுக் கொல்லப்பட்டனர்? (Which prince of Austria and his wife was assassinated in Sarajevo?)

'பிரான்சிஸ் பெர்டினாண்டும்' is identified as a person entity which is itself inflected and also it proceeds a clue word 'இளவரசரான' (prince), so the named entity and the previous word is replaced with the inflected question word 'எந்த இளவரசனும்'.

Extract inflection suffix

The inflection suffix need to be extracted from a word to form meaningful question. Morphological Analysis (MA) is a technique used to split the parts of words like stem word, suffix, etc. The existing libraries for MA such as Indic NLP and Polyglot were analyzed and tested. But the results were not satisfactory (further discussed in chapter 3.6.5). Therefore an affix stripping algorithm is implemented to find the inflection suffix from the word with the help of some utils in Open Tamil library.

Limitation: When non inflected words ends with inflection suffix, the system will not be able to distinguish. E.g. The word இலங்கை is not inflected. But ends with rhyme 'ஐ'. Hence this is considered as inflected word and replaced with an inflected question word.

Question word identification

The inflected question words are maintained as a list of objects for all named entities. For each named entity all the possible inflection suffix and its corresponding question words are listed. After finding the inflection suffix of the named entity word or clue words, corresponding question word is replaced.

Rules and Assumptions

- The named entities of type Government, Kingdom, Organization, Troop, Law, Water bodies, Tax, Industry, Sector are followed by clue words most of the cases. Hence these named entities are replaced with question word if only clue words are present in the predicted entity.
- A clue word proceeding a named entity has a limited possible inflection suffix. In the system, only if the previous word containing clue word with inflection suffix either of 'ஆன' or 'ஆகிய' are considered while replacing question word. E.g. ஆசிய நாடாகிய ஜப்பான்
- If a named entity ending with comma and the following word is of same named entity type then question formation for that named entity type is ignored in NER module. But it is handled in regex pattern approach. When multi named entities occurring one after other it follows few patterns most of which is easily identifiable in regex pattern matching.

  Example: ஜேர்மன், இத்தாலி ஆகிய ஐரோப்பிய அச்சு நாடுகள் இரண்டும் 1937 இல் ஜப்பானுடன் இரகசிய ஒப்பந்தம் ஒன்றைச் செய்தன. (The European axis countries, Germany and Italy made an agreement with Japan in 1937)

  In the above example it is not possible to generate the correct question format from NER approach, as the clue word occurs does not follow immediately. So if a sequence of comma separated words precedes the words 'ஆகிய', 'போன்ற', then it is replaced with blank line in rule based module.
- If a named entity followed by a determiner words like 'என்ற', என்கின்ற' most probably it follows with the clue word of the named entity. So one word after the determiner is analyzed when replacing question word.

CHAPTER 5

TECHNOLOGY AND RESOURCES

In this chapter, technologies and libraries which is used to implement the system will be discussed in brief. The version and the access rights of libraries are also discussed in this section.

5.1 Programming Languages or Frameworks

Python is used as the main programming language for the implementation as it provides support for an extensive collection of NLP tools and libraries as well as it is object-oriented and functional programming. Since the system is a web application, an open source python based micro web framework known as Flask is used. HTML, CSS and bootstrap is used for designing web pages.

5.2 Development Tools

PyCharm is used as the IDE for the project as it supports Python and also one of the popular IDE used for python development. It has lots of features like support for debugging, integration with version control like GIT, support for python web frameworks like Flask, Django, etc. and also supports other specific template languages like JavaScript, TypeScript, HTML/CSS etc.

5.3 Version Controlling Systems

Git is used as the version controlling system as it is open source, reliable, fast and easy to use with several good features.

5.4 Libraries used

The system uses some inbuilt python libraries as well as external libraries. Table 5.1 lists down all the external libraries and other resources used for various NLP techniques along with its access rights.

Table 5.1: List of Libraries and Tools used in system

Processing Technique	**Resources**	**Description**	**Access Rights**
PDF file processing	PYPDF2	A python library built as a toolkit which is capable of processing PDF files with multiple functionality.	Free software under BSD License
Unicode conversion	Open Tamil Version: 0.97	This library provides several NLP functionalities for Tamil language. The text2unicode module is used in the system to automatically identify different encodings and convert to Unicode	Open source library licensed under MIT
Sentence Tokenization	Indic NLP	Sentence tokenization works well when common dates and abbreviations exist in the sentence. The library has frequent releases.	Free software under MIT license.
Stemming	Snowball Stemmer Release: 2.1.0	Snowball stemmer is a library which supports for stemming in Tamil language. Python 2 and Python 3 (>= 3.3) are supported.	Free software under BSD License (BSD-3-Clause)
POS tagging	Ripple Tagger Release: 2.0	Only python library supports POS tagging at the time of this system implementation. Universal POS tags are supported. No releases after 2016.	Free software
CRF implementation	Sklearn-crfsuite Version: 0.3.6	Supports CRF with different training algorithm. Allows cross validation and hyperparameter optimization.	Free software under New BSD License

CHAPTER 6

IMPLEMENTATION

This chapter discusses the implementation of the system in an informative way using code snippets, pseudocode and screenshots of the outputs. Implementation details and output of all components are discussed in the order of the flow of the system. In addition, the rules used and assumptions made, libraries used, the format of tagged data, etc. are also discussed in detail.

6.1 Web Application User Interface

The user can upload a file in either .txt, .doc, .docx and .pdf format or paste a text in the text area.

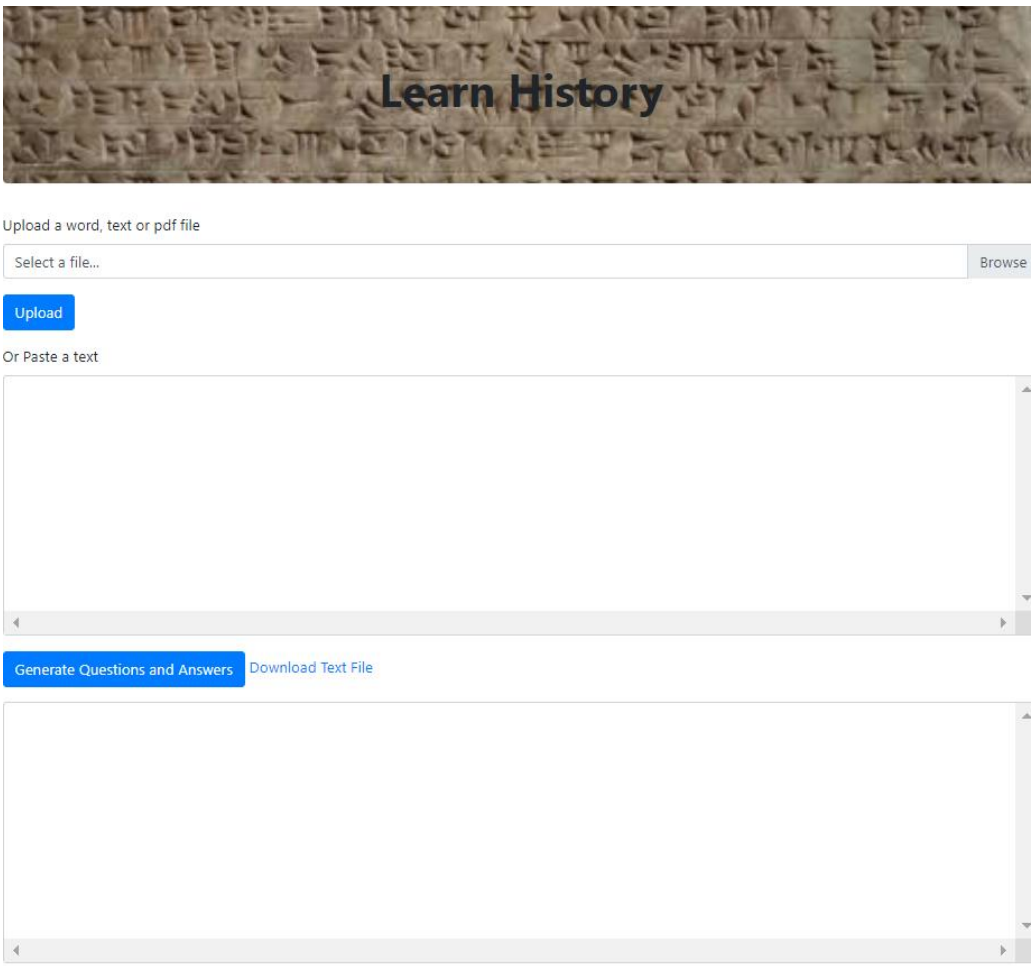


Figure 6.1: The web application UI

If the user uploads a file, the content of file will be displayed in the text area. So the user can review the content and make changes to the text if needed and click on ‘Generate Question And Answers’ button.

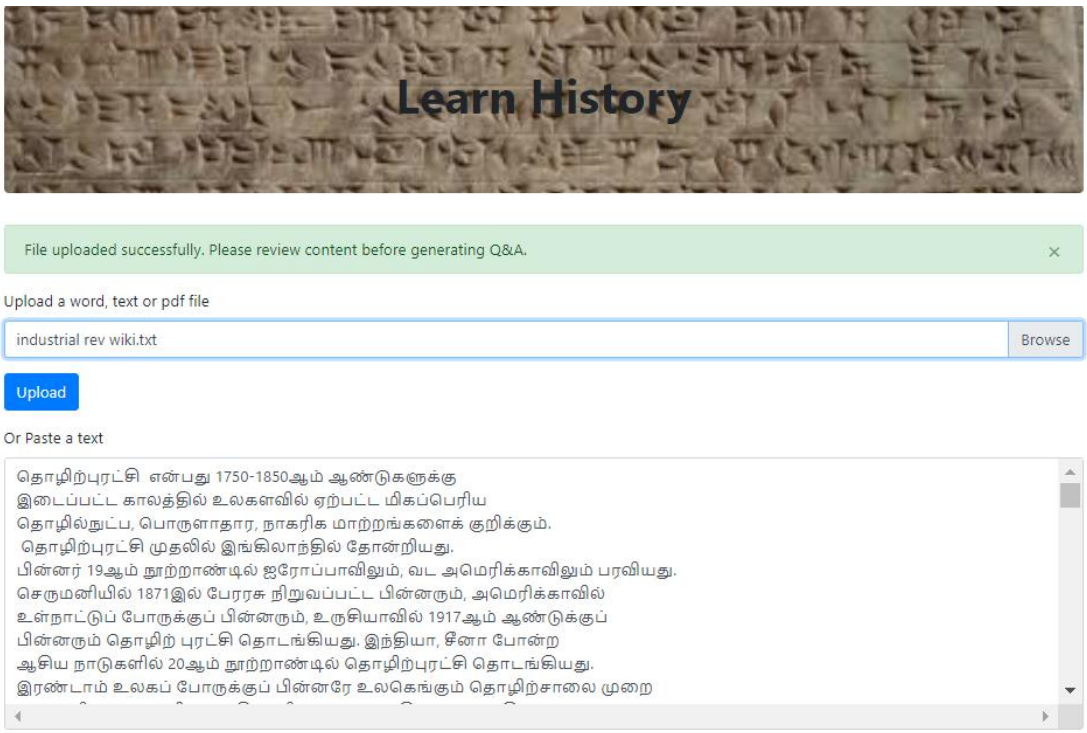


Figure 6.2: UI after valid file upload

If the user uploads a format not supported by the application, then an error message is displayed. The maximum file size which can be uploaded is 10 MB.

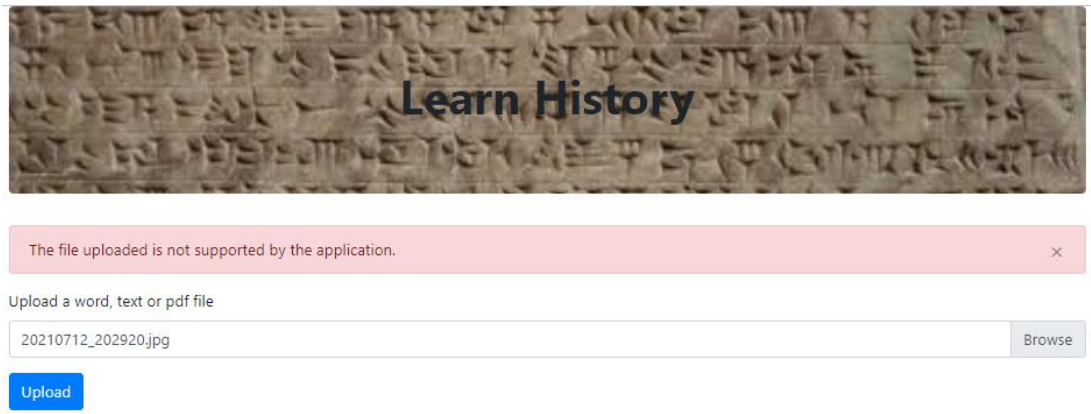


Figure 6.3: Invalid file upload Error

The questions and answers generated will be loaded into the text area. The question and answer generated can be downloaded as a text file by clicking the link ‘Download Text File’.

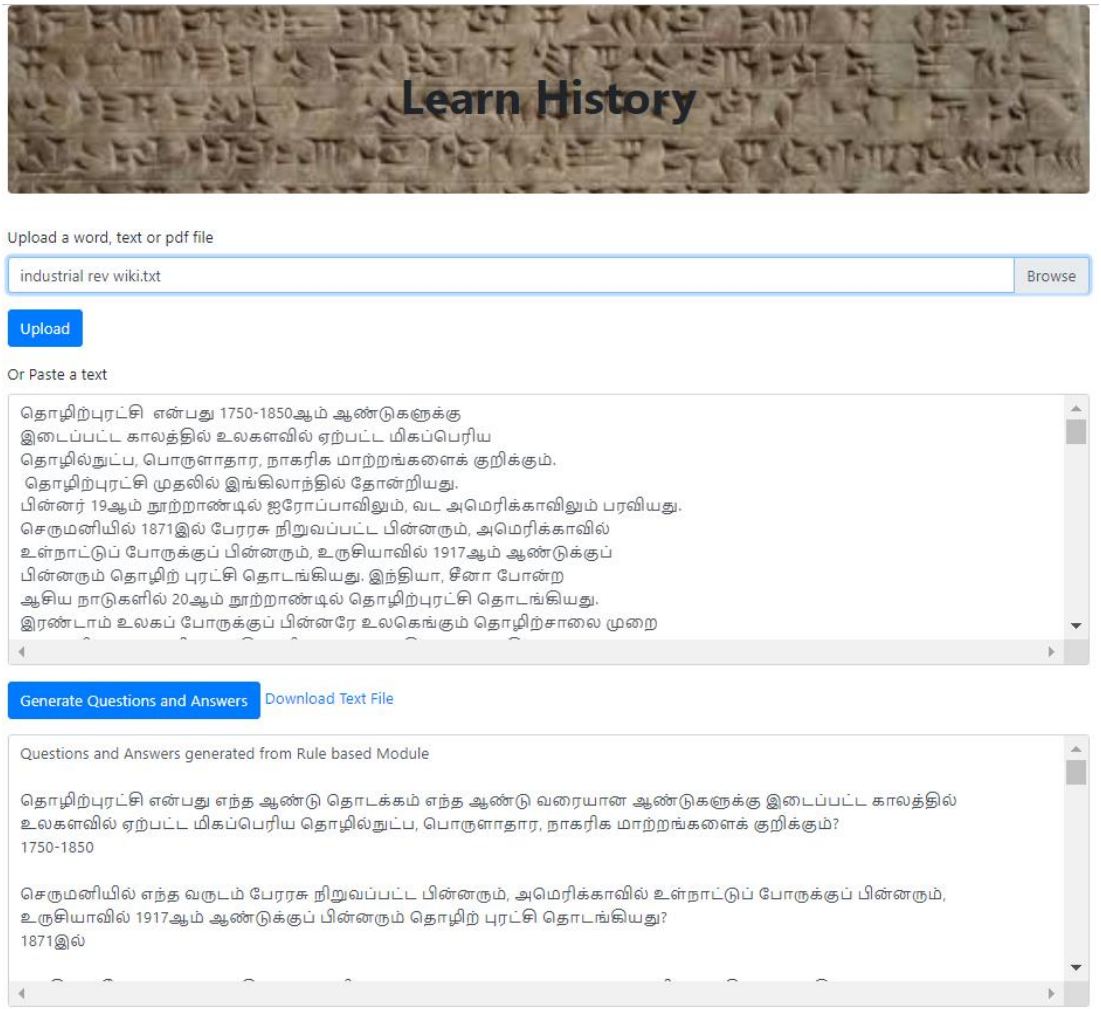


Figure 6.4: UI after questions generated

6.2 Preprocessing

When ‘Upload’ button is clicked, it validates the file format and size and read the file content. PYPDF2, a python library is used to read the pdf file content. A txt2unicode module from a library named ‘OpenTamil’ is used to convert any encoding to Unicode by auto detecting the encoding format using the method auto2unicode. Then it loads the converted text into the text area.

The input text is tokenized into sentences using the Indic nlp library. If the sentence starts with anopharic word, then it will be not processed by the system for question generation. The sentences suitable for question generation are given as input for rule based and NER module.

6.3 Rule based approach

Regex pattern matching and gazetteer check is used for the rule-based module. The preprocessed sentence is given as input for this module.

6.3.1 Regex pattern Matching

Regex match is checked to identify date, time and quantity in the sentence. Also regex pattern for comma separated words of same entity, words enclosed with single quote are checked against sentence and is replaced with blank line if match found.

Table 6.1: Regex patterns used in System

Patterns	Example	Question word
(வருடம்\s)*[1-2][0-9]{3}\s*(?:இல்\s|ல்\s)	வருடம் 1948 இல்	எந்த வருடம்
[1-2][0-9]{3}(\s)*(ஆம்\s (?:வருடம்|ஆண்டு)\s){0,1}[^\s]+\s(மாதம்\s){0,1}[0-3][0-9](\s)* (?:ஆம்|ஆந்|இல்|ல்) (\s(?:திகதி|நாள்|தேதி)){0,1}	1948 ஆம் ஆண்டு மே மாதம் 28 ஆந் திகதி	எந்த ஆண்டு எந்த மாதம் எத்தனையாம் திகதி
[1-2][0-9]{3}(\s)*(ஆம்\s (?:வருடம் | ஆண்டு | ஆண்டில் | வருடத்தில்)\s){1}	1948 ஆம் ஆண்டில்	எத்தனையாம் ஆண்டு
(0[1-9]|[12][0-9]|3[01])[- /.](0[1-9]|1[012]) [- /.](19|20)\d\d	19.11.1948 19-11-1948	எந்த ஆண்டு எந்த மாதம் எத்தனையாம் திகதி
[1-2][0-9]{3}(\s)*-(\s)*[1-2][0-9]{3}	1948-1952	எந்த ஆண்டு தொடக்கம் எந்த ஆண்டு
\d+\s*ஆம்	2 ஆம்	எத்தனையாம்
\d+\s*(?:ஆவது | வது)	2 வது	எத்தனையாவது
\d+\s*%	20%	எத்தனை வீதம்
\d+\s	28 மில்லியன்	எத்தனை
([^,\s]+,\s)+[^,]+(?:ஆகிய\s|போன்ற\s|ஆகியன\s|என்பன\s|போன்றன\s|போன்றவை\s|ஆகியவை\s|என்பவை\s|முதலியன\s|முதலிய\s|)	சீனா, இந்தியா, இலங்கை முதலிய	Replaced with blank line
"'.*'"	'நாணய விஞ்ஞானம்'	Replaced with blank line

6.3.2 Gazetteer approach

Sentence given as input to the module is processed to find the gazetteer match and if match found named entity, named entity type, previous word and next words are given to Question and Answer generation module as input. Only whole word match is considered. The below is the implementation pseudocode for gazetteer match.

Algorithm for Gazetteer check

Input: *sentence*

Process:

Word list := split sentence with space

For each word in word list

Search gazetteer list

If part word in gazetteer matches the word

Check if the next words matches

If complete match found:

Generate question

Write question and answer to file

Else: Continue loop

If whole word matches:

Get named entity word, previous and next word

Generate question

6.4 Named Entity Recognition

A NER model is trained using features suitable for language and domain of interest and saved to a file. The sentences are tagged using the trained model and POS tags identified. Based on defined rules, named entity tags are processed to form meaningful question along with some grammatical rules.

6.4.1 Corpus tagging

CONLL file format is used for corpus tagging. IOB2 annotation schema is used. Each token in the sentence is tagged in a line and each sentence is seperated by a blank line. Word token, POS tag and NER tag is seperated by a tab space in each line.

Example: கண்டி நகருக்கு மகாவலி கங்கை பாதுகாப்பு வழங்கியிருந்தது.

```
கண்டி  NOUN    B-CITY
நகருக்கு       NOUN    I-CITY
மகாவலி         NOUN    B-WAT
கங்கை PROPN    I-WAT
பாதுகாப்பு     VERB    O
வழங்கியிருந்தது.      VERB    O
```

Figure 6.5: Sample corpus annotation

6.4.2 CRF implementation

Sklearn-crfsuite library is used for the CRF implementation. The train and test data is split into 80% and 20% respectively. lbfgs CRF (Limited-memory Broyden-Fletcher-Goldfarb-Shanno) algorithm is chosen. Hyper parameter tuning is applied using the RandomizedSearchCV algorithm with 150 fold.

Features are experimented with different combinations and window size. Window size of 5 is experimented and as most of the features produce better results with window size 3, it is used for the final Question and Answer generation system.

```
crf = sklearn_crfsuite.CRF(
    algorithm='lbfgs',
    max_iterations=100,
    all_possible_transitions=True
)
params_space = {
    'c1': scipy.stats.expon(scale=0.5),
    'c2': scipy.stats.expon(scale=0.05),
}

f1_scorer = make_scorer(metrics.flat_f1_score,
                        average='weighted', labels=label_category)

rs = RandomizedSearchCV(crf, params_space,
                        cv=5,
                        verbose=1,
                        n_jobs=-1,
                        n_iter=30,
                        scoring=f1_scorer)
rs.fit(X_train, y_train)
```

Figure 6.6: CRF algorithm

Below are the implementation details of features used in training NER model.

- Inflection Suffix - Extracted by affix stripping algorithm with the help of some utils in Open Tamil library (Further discussed in chapter 6.5.2).

Inflection suffix identified by the system are ஐ, ஆல், கு, இல், இன், அது, உடன், உடைய, மீது, ஆகிய, இருந்து, உம், இலும், ஆன, இடம், ஆக, ஆம், ஆவது

- Numeric word - Check if the word is in digits using the isDigit function and a list of numeric words are defined to check if a word indicates a number. This list covers most of the possible occurrence of numeric word.

```
numbers = {
    'unit': ['ஒரு', 'இரு', 'ஒன்று', 'இரண்டு', 'மூன்று', 'நான்கு', 'ஐந்து', 'ஆறு', 'ஏழு', 'எட்டு', 'ஒன்பது', 'பத்து', 'பதினொன்று',
             'பன்னிரண்டு', 'பதின்மூன்று', 'பதினான்கு', 'பதினைந்து', 'பதினாறு', 'பதினேழு', 'பதினெட்டு', 'பத்தொன்பது'],
    'tens': ['இருபது', 'இருபத்து', 'முப்பது', 'முப்பத்து', 'நாற்பது', 'நாற்பத்து', 'ஐம்பது', 'ஐம்பத்து', 'அறுபது', 'அறுபத்து',
             'எழுபது', 'எழுபத்து', 'எண்பது', 'எண்பத்து', 'தொண்ணூறு', 'தொண்ணூற்று'],
    'hundred': ['நூறு', 'இருநூறு', 'முன்னூறு', 'நானூறு', 'ஐநூறு', 'அறுநூறு', 'எழுநூறு', 'எண்ணூறு', 'தொள்ளாயிரம்',
                'நூற்று', 'இருநூற்று', 'முன்னூற்று', 'நானூற்று', 'ஐநூற்று', 'அறுநூற்று', 'எழுநூற்று', 'எண்ணூற்று',
                'தொள்ளாயிரத்து'],
    'thousand': ['ஆயிரம்', 'ஆயிரத்து', 'இரண்டாயிரம்', 'மூலாயிரம்', 'நான்காயிரம்', 'ஐயாயிரம்', 'ஆறாயிரம்', 'ஏழாயிரம்',
                 'எண்ணாயிரம்', 'ஒன்பதாயிரம்'],
    'scales': ['இலட்சம்', 'கோடி', 'மில்லியன்', 'பில்லியன்']
}
```

Figure 6.7: List of numeric word in Tamil

- Root word - As the tamil words are highly inflected, root words obtained is used as a feature. Snowball stemmer library is used to extract root word.
- Clue words - Table 6.2 depicts some sample clue words list for the named entities predicted by the system which are observed preceding or proceeding a named entity.

Table 6.2: Clue words for Named Entities

Named Entity	Clue Words
PER	மன்னன், அரசன் (King), ஜெனரல் (General), தலைவர் (Leader), ஜனாதிபதி (President), தளபதி (Commander), இளவரசன் (Prince), அமைச்சர் (Minister)
NORP	வம்சத்தினர் (Dynasty), மக்கள் (People), வகுப்பினர் (Class), இனத்தவர் (Ethnic group) , சமுதாயத்தினர் (Society)
DES	பதவி (Position)
GOD	பெருமான், பகவான் (Lord), கடவுள், தெய்வம், குலதெய்வம் (God)
REL	சமயம் (Religion)
CONT	கண்டம் (Continent)
COU	நாடு (Country)
TER	பிரதேசம் (Region), தீவு (Island), ஊர் (Village), குடியேற்றம் (Settlement)
CITY	நகரம் (City), தலைநகரம் (Capital City), மாநகரம் (City)
LAW	சீர்த்திருத்தம் (Reform), பிரகடனம் (Declaration), சட்டம் (Act), ஒப்பந்தம் (Agreement)

CON	கொள்கை (Policy), எண்ணக்கரு (Concept), முறை (Method), சிந்தனை (Thought)
TIME	நூற்றாண்டு (Century), காலப்பகுதி (Period), தசாப்தம் (Decade), கி.பி. (A.D), கி.மு. (B.C)
YEAR	ஆண்டு, வருடம் (Year)
MON	மாதம் (Month)
DATE	திகதி (Date)
WAT	கால்வாய் (Canal), சமுத்திரம் (Ocean), ஆறு (River), கங்கை (River), நதி (River), கடல் (Sea), பெருங்கடல் (Ocean), குளம் (Pond), வாவி, நீர்த்தேக்கம் (Reservoir), நீர்நிலை (Water bodies)
ANT	ஓவியம்(Art), சித்திரம்(Art), கல்வெட்டு(Inscription),சிற்பம் (Sculpture), சிலை(Statue)
EQUIP	கருவி (Tool), இயந்திரம் (Engine), என்ஜின் (Engine), உபகரணம்(Equipment), ஆயுதம்(Weapon)
ORG	சங்கம் (Association), சபை (Council), திணைக்களம் (Department), ஆணைக்குழு (Commission), கம்பனி (Company)
TRO	இராணுவம் (Military), படை (Troop), இயக்கம் (Movement)
GOV	பேரரசு (Empire), கட்சி (Party), அரசு (Government), கூட்டரசு (Federal), ஆட்சி (Rule)
CUR	காசு (Money), நாணயம் (Coin)
FAC	நிலையம் (Station)
CROP	பயிர் (Crop)
IND	கைத்தொழில் (Industry)
SEC	துறை (Sector)
SKILL	அறிவு (Knowledge), கலை (Skill)
SOU	மூலாதாரம் (Source)
TAX	வரி (Tax), கட்டணம் (Payment)
PRO	பொருள் (Product)
NUM	எண்ணிக்கை (Count)
LIT	நூல் (Book), இலக்கியம் (Literature), காவியம், காப்பியம் (Epic Poem)
LOC	துறைமுகம் (Harbor), முகாம் (Camp), மாளிகை (Palace), விகாரை (Vihara), கோயில் (Temple), தேவாலயம் (Church), பள்ளிவாசல் (Mosque), பாராளுமன்றம் (Parliament), சிறைச்சாலை (Prison)
LAN	மொழி (Language)
EVE	புரட்சி (Revolution), யுத்தம், போர், மகாயுத்தம் (War), சடங்கு (Ritual), சம்பவம் (Incident), நிகழ்வு (Event), கிளர்ச்சி (Rebellion), புரட்சி (Revolution)
SUB	பாடம் (Subject)

- Prefix suffix Rule - Named entities can occur with certain prefix or suffix.

Table 6.3: Prefix Suffix Rule for Named Entity

NE Type	Suffix	Examples
COU	லாந்து	நெதர்லாந்து (Netherland), ஐஸ்லாந்து (Iceland)
NORP	அர்	ஜேர்மனியினர் (German), ஒல்லாந்தர் (Hollander)
CON	வாதம்	நாசிசவாதம் (Nazism), மானிடவாதம் (Humanism)
SUB	இயல்	மெய்யியல் (Philosophy), பௌதீகவியல் (Physics)
WAT	வாவி	அபய வாவி (Abeya wewa), திஸ்ஸ வாவி (Tissa wewa)

- Year Format - word is checked against a date regex pattern.
  Pattern - ^[1-2][0-9]{3}(?: இல் | ல் | ஆம் | ம்){0,1})
- POS tag - Dataset is tagged with POS tag along with NE tag. The Ripple Tagger library is used to tag the POS for all sentences.
- Conjunction words - மற்றும், ஆகிய, ஆகியன, போன்ற, போன்றன, போன்றவை, ஆகியவை, என்பன, என்பவை, ஆகியோர், போன்றோர்

6.4.2 Processing predicted tags

Input: *List of sentences*

Process:

List of list of NER tags for each sentence := Predict NER tags using saved model
FOR EACH *sentence*
    FOR EACH *named entity tags predicted for the sentence*
        IF *named entity tag starts with 'B-'*
            Check the proceeding tags if it starts with I-
            Get the whole named entity predicted
            IF *the NE ends with ',' or previous word is 'மற்றும்' or previous word ends with ','*
                Continue without processing
            ELSE IF *entity type is 'NUM' and word is not a digit*
                Get inflection suffix
                Find the question word for inflection suffix
                Replace word with question word
                Continue
            ELSE
                Get the previous and next words
                Generate question
                Continue
        If named entity tag is O
            Continue

6.5 Question and Answer Generation

The sentence, named entity identified, previous word, next word and named entity types are given as input. Steps and few rules followed to generate meaningful questions. The generated questions are written to a file which is then read and loaded in the UI once all sentences are processed.

6.5.1 Question and Answer generation algorithm

Input: *sentence, named entity (W[i]), previous word (W[i-1]), next word (W[i+1]), named entity type*

Process:

IF *named entity type is 'YEAR' or 'NUM'*
    Return
IF *part word of W[i] or stem word is clue word*
    Except the part word other tokens replaced with question word 'எந்த'
IF *W[i-1] or stem word is a clue word*
    Get the inflection suffix of W[i-1]
    IF *inflection suffix is 'ஆகிய' or 'ஆன'*
        Check previous word W[i-2]
        IF *ends with rhyme 'அ'*
            Get inflection suffix of W[i]
            Get question word for NE type and inflection of W[i]
            Replace W[i-2], W[i-1], W[i] with question word
            Write question and answer to file
            Return
        ELSE
            Get inflection suffix of W[i]
            Get question word for NE type and inflection of W[i]
            Replace W[i-1], W[i] with question word
            Write question and answer to file
            Return

ELSE IF *W[i+1] or stem word has clue word*
    Replace W[i] with question word 'எந்த'
    Write question and answer to file
    Return

ELSE IF *or W[i+1] is a word in ('என்ற', 'எனப்படும்', 'என்கின்ற') and W[i+2] is a clue word*
    Replace W[i], W[i+1] with question word 'எந்த'
    Write question and answer to file
    Return

ELSE IF *or W[i] is a PER entity and W[i+1] or stem word of W[i+1] is a word in ('என்பவர்', 'என்கின்றவர்')*
    Replace W[i], W[i+1] with question word 'யார்'
    Write question and answer to file
    Return

ELSE IF *no clue words present in previous, next or current word and named entity not in ['LAW', 'GOV', 'ORG', 'TRO', 'IND', 'SEC', 'SKILL', 'SOU', 'TAX', 'LOC']*

Get inflection suffix of W[i]
Get question word for NE type and inflection of W[i]
Replace W[i] with question word
Write question and answer to file
Return

6.5.2 Affix Stripping

Tamil characters can have 2 Unicode points for a single character. So we need to extract them in order to find the inflection suffix. The method getUnicodePoints from Open Tamil library is used to get the Unicode points of a given character.

Example: Both words ஜேர்மனியின், ரஷ்யாவின் has same inflection suffix 'இன்'. But when combining with the root word it takes a different form. Hence the Unicode points should be obtained for the second last character 'யி' and 'வி' of both words.

The method 'getUnicodePoints' returns the 2 Unicode points as a tuple as follows.

- யி => [ய், இ]
- வி => [வ், இ]

The inflection suffix which are checked in the implemented method are ஐ, ஆல், கு, இல், இன், அது, உடன், உடைய, மீது, ஆகிய, இருந்து, உம், இலும், ஆன, இடம், ஆக, ஆம், ஆவது.

```
def findinflectionsuffix(word):
    letters = tamil.utf8.get_letters(word)
    lastletter = letters[-1]
    secondlastletter = letters[-2]
    lastletterunicode = tamilutils.getUnicodePoints(lastletter)

    if lastletter == u"கு":
        return u"கு"
    elif lastletter == u"ன்":
        tuple = tamilutils.getUnicodePoints(secondlastletter)
        if tuple[1] == u"இ":
            return u"இன்"
    elif lastletter == u"ல்":
        tuple = tamilutils.getUnicodePoints(secondlastletter)
        if tuple[1] == u"ஆ":
            return u"ஆல்"
        elif tuple[1] == u"இ":
            return u"இல்"
    elif lastletter == u"து":
        tuple = tamilutils.getUnicodePoints(secondlastletter)
        if tuple[1] == u"அ":
            return u"அது"
    elif lastletterunicode[1] == u"ஐ":
        return u"ஐ"
```

Figure 6.8: Sample code segment for Inflection suffix identification

6.5.3 Question Formation

The words surrounding the identified named entity is checked for clue words. If a clue word found the corresponding question word will be replaced. If no clue words is observed common question word will be replaced. Few named entities such as LAW, CON, GOV, WAT, ANT, ORG occurs with clue words most of the time. Hence the question word will be replaced only if any clue words are identified. Table 6.4 shows the sample question word list for each named entity class.

Table 6.4: Question words for Named Entities

NE Type	Clue Words
PER	யார் (Who), எந்த மன்னன்(Which King)
NORP	எந்த வம்சத்தினர் (Which dynasty), எந்த தேசத்தினர் (Which nation)
DES	எந்த பதவி (Which Position)
GOD	எந்த கடவுள் (Which god)
REL	எந்த சமயம் (Which religion)
CONT	எந்த கண்டம் (Which continent)
COU	எந்த நாடு (Which country)
TER	எந்த பிரதேசம் (Which region), எந்த தீவு (Which island)
CITY	எந்த நகரம் (Which city), எந்த தலைநகரம் (Which capital city)
LAW	எந்த சீர்த்திருத்தம் (Which reform), எந்த பிரகடனம் (Which declaration), எந்த சட்டம் (Which act)
CON	எந்த கொள்கை (Which policy), எந்த எண்ணக்கரு (Which concept)
TIME	எந்த காலப்பகுதியில் (Which period)
YEAR	எத்தனையாம் ஆண்டு, எந்த வருடம் (Which year)
MON	எந்த மாதம் (Which month)
DATE	எத்தனையாம் திகதி (Which date)
WAT	எந்த கால்வாய் (Canal), எந்த சமுத்திரம் (Ocean), எந்த நீர்த்தேக்கம் (Which reservoir), எந்த நீர்நிலை (Which water bodies)
ANT	எந்த ஓவியம்(Which artwork), எந்த கல்வெட்டு(Which inscription), எந்த சிற்பம் (Which sculpture)
EQUIP	எந்த கருவி (Which tool), எந்த இயந்திரம் (Which engine), எந்த உபகரணம்(Which equipment), எந்த ஆயுதம்(Which weapon)
ORG	எந்த சபை (Which council), எந்த திணைக்களம் (Which department), எந்த ஆணைக்குழு (Which commission)
TRO	எந்த இராணுவம் (Which military), எந்த படை (Which troop), எந்த இயக்கம் (Which movement)
GOV	எந்த பேரரசு (Which empire), எந்த கட்சி (Which party), எந்த அரசு (Which government), எந்த கூட்டரசு (Which federal)
CUR	எந்த காசு (Which currency), எந்த நாணயம் (Which coin)
CROP	எந்த பயிர் (Which crop)

IND	எந்த கைத்தொழில் (Which industry)
SEC	எந்த துறை (Which sector)
SKILL	எந்த கலை (Which skill)
SOU	எந்த மூலாதாரம் (Which source)
TAX	எந்த வரி (Which tax)
PRO	எந்த பொருள் (Which product)
NUM	எத்தனை (How many), எவ்வளவு (How much)
LIT	எந்த நூல் (Which book), எந்த இலக்கியம் (Which literature)
LOC	எந்த துறைமுகம் (Which harbor), எந்த முகாம் (Which camp)
LAN	எந்த மொழி (Which language)
EVE	எந்த புரட்சி (Which revolution), எந்த போர்(Which war), எந்த சம்பவம் (Which incident), எந்த நிகழ்வு (Which event)
SUB	எந்த பாடம் (Which subject)

A list of question words for each named entity corresponding to the inflection suffix is defined. The inflection suffix possible for an entity only are defined in list.

```
"CITY": {
    "ஐ": "எந்த நகரத்தை",
    "ஆல்": "எந்த நகரத்தால்",
    "கு": "எந்த நகரத்துக்கு",
    "இல்": "எந்த நகரத்தில்",
    "இன்": "எந்த நகரத்தின்",
    "அது": "எந்த நகரத்தினது",
    "உம்": "எந்த நகரமும்",
    "மீது": "எந்த நகரத்தின்மீது",
    "உடன்": "எந்த நகரத்துடன்",
    "இலும்": "எந்த நகரத்திலும்",
    "லிருந்து": "எந்த நகரத்திலிருந்து",
    "plural": "எந்த நகரங்கள்",
    "mei": "எந்த நகர",
    "None": "எந்த நகரம்"
},
```

Figure 6.9: Sample Question word list for inflected named entity

CHAPTER 7

EXPERIMENTS AND EVALUATION

In this section, the test carried out on the Question and Answer generation system developed and the test results and evaluation are discussed. The test is also carried out on Named Entity Recognition (NER) module which is one of the main module of the system. The test results are shared with the parameters most suitable for the evaluation of the system and discussed by having the idea of emphasizing the applicability of the approach. Based on the evaluation results, more useful insights were extracted about the system and the effectiveness of the system is evaluated.

7.1 NER Module Evaluation

CRF model is trained with features and different feature combination is experimented to identify the best feature combination. The final NER system with best feature combination is hyper parameter tuned to find the best model.

7.1.1 Corpus Tagging

The dataset is manually tagged from the content in Tamil medium Grade 10 & Grade 11 History textbook from Srilankan syllabus. The dataset contains 23405 words from 1800 sentences picked from chapters covering popular topics in history and it contains 6925 NE tags.

Chapters in the books were divided among 2 annotators, where one annotated around 10k tokens and other around 13k tokens. Kappa score which is used for the measure of inter-annotator agreement when annotating the corpus, was 0.92 which shows a very good agreement among the two annotators.

NE tag set of 38 tag types related to history domain is introduced for this system and the distribution of each NE tag in the dataset is depicted in Table 7.1. A flat annotation schema, IOB2 is used which assigns B tag to the beginning of NE, I tag for the remaining consecutive chunks of the NE and O tag for other non-entity words.

7.1.2 Performance Evaluation Metrics

- Precision (P): Precision is the fraction of the correct tags generated by the NER to the total number of tags generated.

  Precision (P) =Correct tags generated by NER/Total generated tags by NER

- Recall(R): Recall is the fraction of the correct tags generated by the NER to the total number of actual tags.

Recall (R) = Correct tags generated by NER / Total number of actual tags.

- F-Score: F-score is the weighted harmonic mean of precision and recall.

F-Measure = (β 2 +1) PR/ (β 2R+P) where β is the weighting between precision and recall and typically β=1.

- Accuracy – Total number of correct prediction/Total number of predictions

Table 7.1: Tag distribution in the dataset

NE Tag Type	Percentage (%)	NE Tag Type	Percentage (%)
PER	6.569	EQUIP	0.461
NORP	3.664	ORG	2.212
DES	3.314	TRO	0.785
GOD	0.042	GOV	0.307
REL	0.042	CUR	0.034
CONT	1.127	CROP	0.316
COU	6.765	IND	0.256
TER	1.016	SEC	1.016
CITY	1.546	SKILL	0.256
LAW	0.452	SOU	0.316
CON	0.734	TAX	0.640
TIME	2.434	PRO	1.102
YEAR	3.263	NUM	5.484
MON	0.751	LIT	0.529
DATE	0.538	LOC	1.033
WAT	0.136	LAN	0.298
ANT	0.333	EVE	3.135
COM	1.272	SUB	0.153
FAC	1.119	O	46.266

7.1.3 Experiments

CRF model is trained with features in section 4.2.3 and different feature combinations are experimented to find the best feature combination. Table 7.2 shows the features and position for basic feature combination used.

Experiments conducted by removing features POS tag, Stem word, Clue word, Gazetteer, previous NE tag and Inflection suffix for different context positions. Table 7.3 shows the F1-score of different experiment results conducted.

Table 7.2: Basic feature combination

Features	Positions
Word, POS tag, clue word, stem word, gazetteers, numeric word, year format	n-2, n-1, n, n+1, n+2
Previous tag label, word ending with comma	n-1, n-2
Conjunction word	n+1, n+2
Inflection suffix, prefix suffix rule	n

Table 7.3: Experiment Results for Different Feature Combination

Experiment	Positions Excluded	F1-Score
All features		0.717
All other features - POS	n-2, n-1, n, n+1, n+2	0.672
	n-2, n-1, n+1, n+2	0.683
	n-2, n+2	0.725
All other features - Gazetteers	n-2, n-1, n, n+1, n+2	0.694
	n-2, n-1, n+1, n+2	0.711
	n-2, n+2	0.717
All other Features - Stem word	n-2, n-1, n, n+1, n+2	0.557
	n-2, n-1, n+1, n+2	0.664
	n-2, n+2	0.718
All other features - Clue word	n-2, n-1, n, n+1, n+2	0.698
	n-2, n-1, n+1, n+2	0.680
	n-2, n+2	0.730
All other feature - Previous NE tag	n-2, n-1	0.530
	n-2	0.706
All other features - Inflection suffix	n	0.708

According to the Table 7.3 results, most features give better results with window size 3. Excluding POS tag, Gazetteers, Stem word, Clue word, previous NE tag and Inflection suffix reduces the F1-score. Removing stem word and previous NE tag makes a huge impact on performance. Training data set did not have a balanced number of tagged data for all NE classes. Hence, Hyper parameter tuning with RandomizedSearchCV algorithm applied to find the best hyperparameters for the model. Model is fitted with 5 cross validation and 30 iterations totaling 150 folds. Best CV score obtained is 0.752.

Table 7.5 shows the features and position for best feature combination obtained from experiments. The system achieved a fair results with micro averaged Precision, Recall and F1-Score of **87.9%, 67.1%** and **76.1%** respectively. The system accuracy for the best feature combination is **89.6%.** Table 7.4 shows the Precision, Recall, F1-Score and Support for each NE class of the best feature combination experiment.

Table 7.4: Performance metrics for all Named Entity tags

NE Type	Precision	Recall	F1-Score	Support
ANT	0.666	0.416	0.450	3
CITY	0.630	0.538	0.577	33
COM	0.375	0.214	0.272	21
CON	1.000	0.336	0.500	11
CONT	1.000	0.629	0.765	29
COU	0.936	0.502	0.585	153
CROP	0.000	0.000	0.000	2
DATE	1.000	1.000	1.000	10
DES	0.589	0.241	0.342	71
EQUIP	1.000	0.375	0.533	4
EVE	0.984	0.812	0.890	52
FAC	0.800	0.361	0.477	17
GOV	1.000	0.875	0.928	4
IND	1.000	1.000	1.000	7
KIN	1.000	0.903	0.946	31
LAN	1.000	1.000	1.000	3
LAW	1.000	0.690	0.816	7
LIT	1.000	0.397	0.553	12
LOC	0.954	0.769	0.851	13
MON	0.937	0.902	0.919	14
NORP	0.862	0.421	0.545	79
NUM	0.783	0.562	0.655	32
ORG	0.969	0.984	0.976	31
PER	0.824	0.621	0.701	87
PRO	0.416	0.185	0.256	27
REL	0.833	0.750	0.733	6
SEC	1.000	0.825	0.901	18
SKILL	1.000	1.000	1.000	2
SOU	0.000	0.000	0.000	1
SUB	0.000	0.000	0.000	0
TAX	1.000	1.000	1.000	1
TER	0.778	0.390	0.516	21
TIME	0.980	0.973	0.976	24
TRO	0.928	0.701	0.976	11
WAT	1.000	0.750	0.857	4
YEAR	0.939	0.979	0.959	48
Micro Average	0.879	0.671	0.761	1262
Macro Average	0.790	0.620	0.668	1262
Weighted Average	0.860	0.671	0.735	1262

Table 7.5: Best feature combination

Features	Positions
Word, POS tag, Clue word, Stem word, Gazetteers, Numeric word, Year format	n-1, n, n+1
Previous tag label, word ending with comma	n-1
Conjunction word	n+1
Inflection suffix, prefix suffix rule	n

According to the Table 7.4 results, the tags like ANT, CROP, SUB, SOU obtained a very low results due to insufficient training data for these tags. Even though the tags like GOV, IND, LAN, LAW, REL, TAX, SKILL, WAT occurs less in dataset, it obtained a very good results due to the clue word feature. Most of the time these NE tags are proceeded with clue words defined.

7.2 Q&A Generation Module Evaluation

The system is evaluated manually and error analysis is done on the evaluation results. The system is improved with some post processing rules and a final evaluation is done on the improved system. Questions are evaluated against the syntactic and semantic correctness.

7.2.1 Initial Evaluation

Initial experiments were carried out from few history documents. The generated questions are analyzed on question format and question appropriateness. Based on the analysis, post processing rules were applied to reduce the error rate.

7.2.2 Error Analysis

Experiments were done on few history documents and error analysis is done. Some of the errors observed occurred due to limitations of the system and the few errors could be eliminated by doing modification to the system. So the possible solutions are also discussed and post processing is done to improve the system accordingly.

Table 7.6: Error Analysis

Sentence	Question generated	Analysis
60 மில்லியன் ஐரோப்பியர்களை உள்ளடக்கிய சுமார் 70 மில்லியன் போர் வீரர்கள் சண்டையில் ஈடுபட்டிருந்தனர். (About 70 million warriors, including 60 million	எத்தனை மில்லியன் ஐரோப்பியர்களை உள்ளடக்கிய சுமார் 70 மில்லியன் போர்வீரர்கள் சண்டையில் ஈடுபட்டிருந்தனர்? (How many millions of Europeans out of 70	The sentence is replaced with the question word 'How many' in place of the digit. Even though it does not have any anaphora words, the context is missing in the sentence. It

Europeans, were involved in the war)	million warriors were involved in the war?)	does not say which war the question indicates.
மேற்கு ஐரோப்பாவின் பகுதிகளான நெதர்லாந்து, பெல்ஜியம் ஆகியவற்றை கைப்பற்றியது. (Western European parts such as Netherland, Belgium are captured)	மேற்கு ஐரோப்பாவின் பகுதிகளான நெதர்லாந்து, எந்த நாடு ஆகியவற்றை கைப்பற்றியது? (Western European parts such as Netherland, which country are captured?)	'பெல்ஜியம்'(Belgium) has been identified as a country named entity. 'நெதர்லாந்து'(Netherland) is not identified. Hence only Belgium is replaced as question word and the question format is not correct. Also this question is meaningless because the subject is missing. The system is not capable of identifying if the subject is missing from the sentence. This is a limitation of the system. **Solution**: Comma separated words can be replaced with blank line instead of question word.
ஓட்டோமான் பேரரசு 1914 அக்டோபரில் இக் கூட்டணியில் இணைந்தது. (Ottoman Empire joined this alliance in 1914)	எந்த பேரரசு 1914 அக்டோபரில் இக் கூட்டணியில் இணைந்தது? (Which Empire joined this alliance in 1914?)	'ஓட்டோமான் பேரரசு' is identified as Government named entity. Hence it is replaced with the word 'எந்த பேரரசு' (Which empire). The question format is correct. 'இக் கூட்டணியில்'(this alliance) referenced from previous sentence. Hence this question is meaningless. If the anaphoric word is in the middle of sentence, the sentence is not ignored. This is a limitation of the system.

ஹிட்லர் வேந்தராக பதவியேற்ற சில நாட்களிலே ஜெர்மன் பாராளுமன்றம் சிலரால் தீக்கிரையானது. (German parliament was set to fire by someone within few days of Hitler's inauguration)	ஹிட்லர் வேந்தராக பதவியேற்ற சில நாட்களிலே எந்த நாடு பாராளுமன்றம் சிலரால் தீக்கிரையானது? (Which country parliament was set to fire by someone within few days of Hitler's inauguration?)	ஜெர்மன் is identified as country. But it indicates a place German parliament. So the whole word should have been identified as a location entity and appropriate question word should have been replaced. Still the question generated can be meaningful if question word is 'எந்த நாட்டு' (Which country's) instead of 'எந்த நாடு' (which country)
இலங்கை கறுவா உட்பட வாசனைத் திரவியங்களுக்குப் புகழ்பெற்ற நாடாக விளங்கியது. (Sri Lanka is famous for its spices, including cinnamon)	எந்த நாட்டை கறுவா உட்பட வாசனைத் திரவியங்களுக்குப் புகழ்பெற்ற நாடாக விளங்கியது? (Which country became famous for its spices, including cinnamon?)	இலங்கை (SriLanka) ends with Unicode point 'ஐ' which is considered as an inflection suffix and the question word formed with the inflection suffix which makes the question format wrong.
19 ஆம் நூற்றாண்டின் ஆரம்பத்தில் ஜேர்மன் மொழி பேசிய மக்கள் வாழ்ந்த 350 சிறிய நாடுகள் இருந்தும் ஜேர்மனி என அழைக்கக்கூடிய ஒரு நாடு இருக்கவில்லை. (Even though 350 small countries inhabited by German speaking people existed in the early 19th century, there was not a single country that could be called Germany.)	19 ஆம் நூற்றாண்டின் ஆரம்பத்தில் ஜேர்மன் மொழி பேசிய மக்கள் வாழ்ந்த 350 சிறிய நாடுகள் இருந்தும் எந்த நாடு என அழைக்கக்கூடிய ஒரு நாடு இருக்கவில்லை.	Germany is identified as country named entity. The system process only previous word and next word when forming question. So the clue word 'country' in sentence is not identified by the system. Hence the entity is replaced by the question word 'எந்த நாடு'(which country). **Solution:** This could be handled by checking if the entity word surrounded by words such as called, named, etc.

பிரித்தானியா மற்றும் ஏனைய ஐரோப்பிய நாடுகளில் ஏற்பட்டதைப் போன்ற கைத்தொழில் புரட்சி 19 ஆம் நூற்றாண்டில் இலங்கையில் ஏற்படாவிட்டாலும் அதன் செல்வாக்கு இலங்கையிலும் ஏற்பட்டது. (Although the Industrial Revolution did not occur in Sri Lanka in the 19th century as it did in Britain and other European countries, its influence did have in Sri Lanka as well)	பிரித்தானியா மற்றும் ஏனைய ஐரோப்பிய நாடுகளில் ஏற்பட்டதைப் போன்ற கைத்தொழில் புரட்சி 19 ஆம் நூற்றாண்டில் இலங்கையில் ஏற்படாவிட்டாலும் அதன் செல்வாக்கு எந்த நாட்டிலும் ஏற்பட்டது? (Although the Industrial Revolution did not take place in Sri Lanka in the 19th century as it did in Britain and other European countries, did it have any influence in any country?)	Sri Lanka is identified as a country named entity. It occurs in 2 places. But only one word is replaced with question word. Therefore the question become meaningless as the answer contained in the question itself.
பல நூறு மைல்கள் நீளமான பாதைகள் பிரித்தானியாவிலும் ஸ்கொட்லாந்திலும் மிகத் துரிதமாக அமைக்கப்பட்டன. (Several hundred miles of trails were laid very rapidly in Britain and Scotland)	பல எத்தனை மைல்கள் நீளமான பாதைகள் பிரித்தானியாவிலும் ஸ்கொட்லாந்திலும் மிகத் துரிதமாக அமைக்கப்பட்டன? (How many miles long trails were built the fastest in Britain and Scotland?)	Hundred is identified as a numeric entity. Hence it is replaced with question word ‘எத்தனை’ (How many). But the sentence says several hundred miles. So the question is not meaningful. **Solution:** This type of questions can be ignored by checking if words like several, few, many, etc. are preceding the numeric word when generating question.
கைத்தொழிற் துறையிலும் போக்குவரத்துத் துறையிலும் ஏற்பட்ட துரித வளர்ச்சி காரணமாகக் கடிதப் போக்குவரத்தை விட இரண்டு இடங்களுக்கிடையே விரைவான தொடர்பாடல் முறையொன்றை	கைத்தொழிற் துறையிலும் போக்குவரத்துத் துறையிலும் ஏற்பட்ட துரித வளர்ச்சி காரணமாகக் கடிதப் போக்குவரத்தை விட எத்தனை இடங்களுக்கிடையே விரைவான தொடர்பாடல்	இரண்டு(Two) identified as numeric word. Hence it is replaced by the question word ‘எத்தனை’ (How many). But this question is not meaningful.

மேற்கொள்ள வேண்டிய தேவை ஏற்பட்டது. (The rapid growth of industry and transportation necessitated a faster communication system between the two destinations than correspondence.)	முறையொன்றை மேற்கொள்ள வேண்டிய தேவை ஏற்பட்டது? (The rapid growth of industry and transportation necessitated a faster communication system between how many destinations than correspondence?)	
வரலாற்று தகவல்களைப் பெற்றுக் கொள்ளக்கூடிய மூலாதாரங்கள் இலக்கிய மூலாதாரங்கள், தொல்பொருள் மூலாதாரங்கள் என இரண்டு வகைகளாகப் பிரிக்கலாம். (The sources from which historical information can be obtained can be divided into two categories as literary sources and archaeological sources)	வரலாற்று தகவல்களைப் பெற்றுக் கொள்ளக்கூடிய மூலாதாரங்கள் இலக்கிய மூலாதாரங்கள், தொல்பொருள் மூலாதாரங்கள் என எத்தனை வகைகளாகப் பிரிக்கலாம்? (The sources from which historical information can be obtained can be divided into how many categories as literary sources and archaeological sources)	இரண்டு(Two) identified as numeric word. Hence it is replaced by the question word 'எத்தனை' (How many). But this question become meaningful as the question itself contains the answer. **Solution:** This can be processed by checking if it indicates the number of categories following with the list.

According to the error analysis, the below is the summary of limitations of the system.

- The system cannot identify if subject is missing in the sentence.
- The system is not capable of identifying if context is missing in the sentence.
- When non inflected words ends with any inflection suffix, the system will not be able to distinguish. Hence inflected question word will be replaced and system generates grammatically incorrect questions.
- If anopharic words occur in middle of sentence, the sentence is not ignored for question generation. Hence it produces meaningless question.
- After a sentence replaced with question word, it can contain the answer or clue for an answer in the question itself which can produce inappropriate questions.

7.2.3 Post Processing Rules

Post processing rules were applied based on the error analysis done to enhance the performance of Question and Answer generation module. Below are the rules added.

- If identified named entity ends with comma and does not have an inflection suffix then gap fill question is formed be replacing named entity with blank line.
- If named entity proceeded by word 'என', 'என்று' the question formation is ignored.
- If the numeric words are preceded by words like பல (several) and சில (few), then the question formation is ignored.
- If a numeric entity is followed by list of comma seperated words, the question formation can be ignored as it may indicate the number of categories which can be predicted from the question itself.

7.2.4 Final Evaluation

A history text about Industrial Revolution is taken from Wikipedia Tamil. The system cannot handle bullet points, headings and diagrams. Therefore only the text paragraphs are taken from the Wikipedia. System generated 44 questions and answers. The evaluation is done from 16 native Tamil speakers from various categories like undergraduates, experts (History teachers) and others (graduates and employees). History is a compulsory subject in Srilankan syllabus in ordinary level examinations. Hence the system evaluation is done from non-expert people as well.

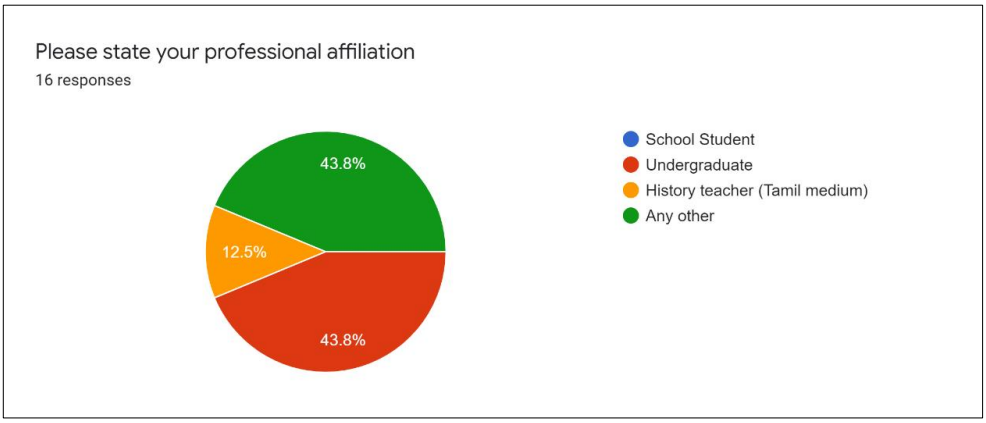


Figure 7.1: Pie chart for profession info of evaluators

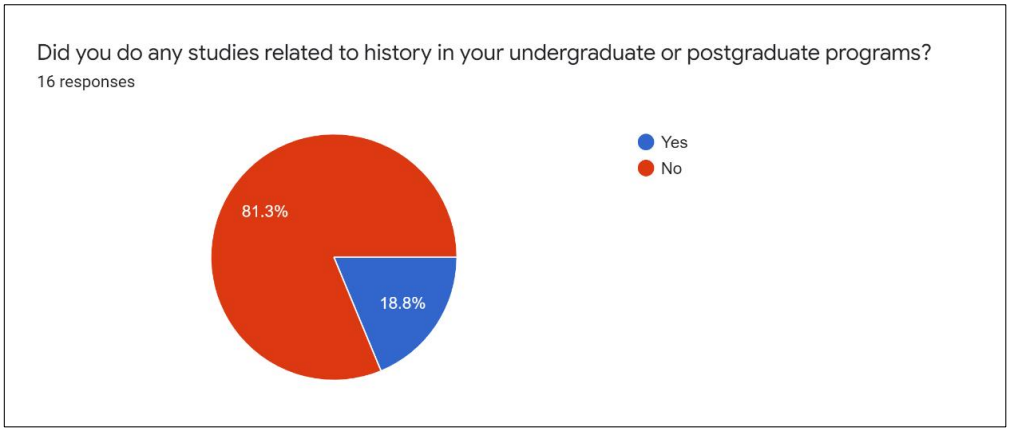


Figure 7.2: Pie chart for higher studies info of evaluators

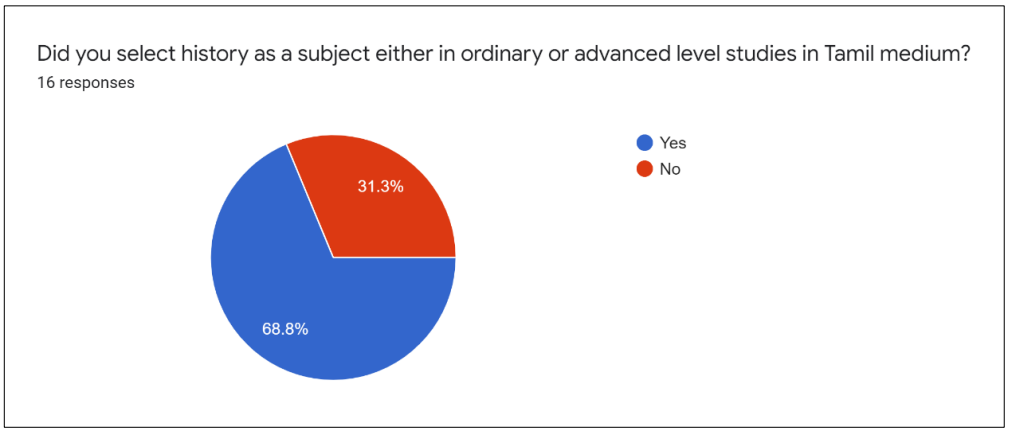


Figure 7.3: Pie chart for secondary school studies info of evaluators

The questions are evaluated on the correctness of question format (i.e. correct question word and inflection suffix of the question word) and the appropriateness of questions. (i.e. if context of question is present and question is meaningful). Below 4 options are given to evaluate each of 44 questions and answers generated by the system.

Option 1: கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது. (Question format is wrong and question is meaningless)

Option 2: கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது. (Question format is correct, but question is not appropriate)

Option 3: கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது. (Question format is wrong, but still the question is meaningful)

Option 4: கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது. (Question format is correct and question is appropriate)

Table 7.7: Summary of Evaluation responses

	Profession	**History studied in O/L or A/L in Tamil Medium**	**Done history related studies in UG or PG studies**	**No. of option 1**	**No. of option 2**	**No. of option 3**	**No. of option 4**
1	Other	No	No	3	1	11	29
2	Other	Yes	No	7	1	4	32
3	Other	Yes	Yes	4	4	7	29
4	Undergraduate	Yes	No	0	3	3	38
5	Undergraduate	Yes	No	5	11	9	19
6	Undergraduate	Yes	No	11	8	16	9
7	Other	Yes	No	3	4	4	33
8	Undergraduate	Yes	No	0	3	11	30
9	Undergraduate	Yes	No	2	1	14	27
10	History teacher	Yes	Yes	8	2	7	27
11	Other	No	No	10	9	6	19
12	Other	No	No	4	2	6	32
13	Undergraduate	No	No	9	1	16	18
14	Undergraduate	No	No	6	0	8	30
15	Other	Yes	No	1	2	7	34
16	History teacher	Yes	Yes	3	4	5	32
Total				76	36	134	438
Average				4.75	3.5	8.375	27.375

Table 7.8: Performance metrics for QA generation module evaluation

Questions generated	**Percentage to the total questions generated (Average of 16 respondents)**
Meaningful questions	81.25%
Grammatically correct questions	67.33%
Both grammatically correct and Meaningful questions	62.22%

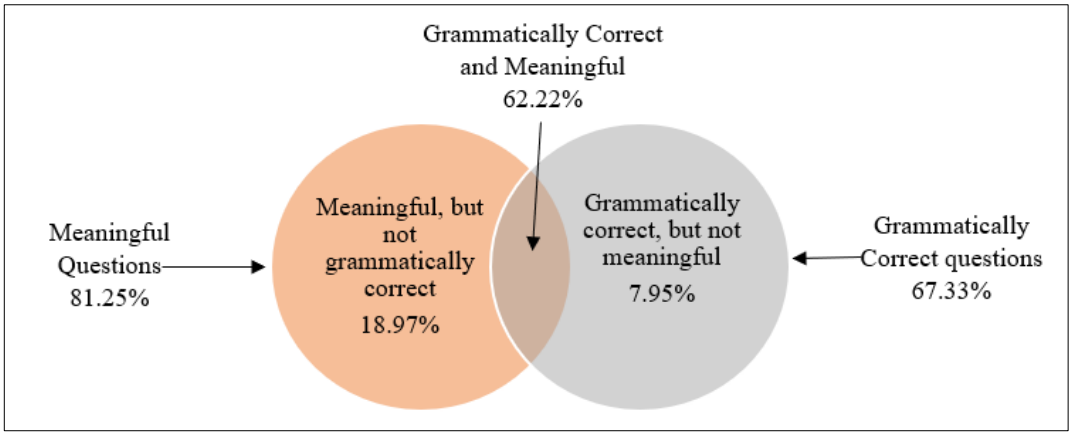


Figure 7.4: Venn diagram for overall Question Evaluation Results

There are 3 respondents who have done History related studies in their undergraduate or postgraduate studies and out of them 2 are Tamil medium History teachers. Hence these 3 respondents are considered as experts and performance metrics are obtained separately on the average of their responses.

Table 7.9: Performance metrics for QA generation module evaluation from experts

Questions generated	**Percentage to the total questions generated (Average of 16 respondents)**
Meaningful questions	81.06%
Grammatically correct questions	74.25%
Both grammatically correct and Meaningful questions	66.66%

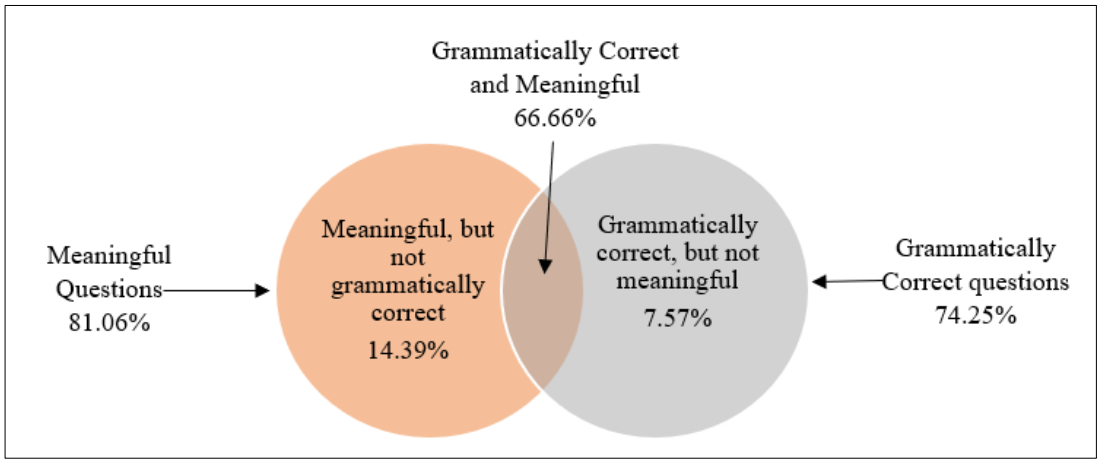


Figure 7.5: Venn diagram for Question Evaluation Results by experts

Out of 44 questions generated, 27 are from rule based module and the rest from NER module. Table 7.10 shows the percentage of meaningful and grammatically correct questions generated from Rule based and NER module separately.

Table 7.10: Performance metrics for QA generation from different approaches

Questions generated	**NER Module**	**Rule Based Module**	
		Regex Match	**Gazetteers**
Total questions generated	38.63%	50%	11.36%
Meaningful questions	71.27%	88.1%	86.3%
Grammatically correct questions	69.65%	76.82%	66.28%
Both grammatically correct and Meaningful questions	55.9%	69.74%	62.52%

Half of the questions generated are 'when' type from regex matching approach in rule based module. More than 85% of questions generated by rule based module are meaningful. While around 71% of questions generated by NER module are meaningful as it identifies the named entity wrong or identifies partially in some cases. Most of the questions generated by NER module is 'which country' type questions as NER module identifies country tag correctly in most of the cases due to the fact that dataset has more country tags. Around 65-70% of the questions generated by the NER module and gazetteer approach are grammatically correct, as in some cases the inflection suffix is identified by affix stripping algorithm when the word is not actually inflected.

Inter Rater Reliability

Krippendorff's Alpha (α) is the Inter rater reliability coefficient used to measure the agreement among the 16 raters using the SPSS Statistics tool. Calculated for both grammatical & semantical correctness evaluation separately and it achieved an alpha value of 0.5750 & 0.4426 respectively which shows a moderate agreement between raters.

Table 7.11: Inter-rater Reliability estimate for QA evaluation

Metrics		**Results for Grammatical Correctness Evaluation**	**Results for Semantical Correctness Evaluation**
Alpha		**0.5750**	**0.4426**
LL95%CI		0.4599	0.3033
UL95%CI		0.6812	0.5720
Units		44.0000	44.0000
Observers		16.0000	16.0000
Pairs		5280.0000	5280.0000
Number of bootstrap samples		10000	10000
Probability of failure to achieve an alpha of at least alphamin	.9000	1.0000	1.0000
	.8000	1.0000	1.0000
	.7000	0.9921	1.0000
	.6700	0.9587	0.9996
	.6000	0.6530	0.9889
	.5000	0.0939	0.7706

CHAPTER 8

DISCUSSION

In this section, the system is analyzed with the evaluation results of the previous chapter. Named Entity Recognition module and Question and Answer generation system is evaluated and further discussed on how well the system performs and the limitation of the system.

8.1 Named Entity Recognition Module

The NER module in the implemented system is the first attempt for NER for history domain in Tamil language with novel entity tags specific to history domain used. Most of the tamil NER systems and NER for history domain developed previously are generic limited to few popular tags such as Person, Location, Organization and Miscellaneous. Researches on NER on historical domain for other languages achieved F-scores from 60-70% on average for rule-based and traditional ML systems and the best for neural system is around 80%. [17].

The system has few limitations due to the challenges related to domain and language discussed in section 3.1. A word can be written in different ways in Tamil, so it is not possible to include all word forms in gazetteers. Hence it does not account the complete benefits of using gazetteers. Also the words are highly inflected in Tamil. Therefore, experiment was conducted with stem word checking against gazetteer list, but that did not improve the performance due to the less accuracy of stemmer library which was used to find the stem word. Clue words helped to identify the named entities when the dataset has insufficient data for few tag categories. Inflection suffix feature did not provide a noticeable improvement due to the context drift in history domain.

NE tag of previous word, which is a dynamic feature improves the results upto a great extent. Features such as stem word, POS tag, clue words also contributes more for the performance.

Despite the lack of resource and challenges of the historical domain and Language, Our NER module could achieve a better micro averaged F1-Score of 76.1% by selecting the features suitable for the domain of interest and language and with hyper parameter optimization.

8.2 Question and Answer generation Module

There are very few researches done on QA generation in Tamil and there is no research done so far for QA generation in history domain in Tamil. Tamil is a low resource languages with limited dataset, NLP libraries and tools that are publicly available.

The designed system generates questions and answers from rule based and NER module. In rule based module, the regex match approach generates question of type ‘When’ and ‘How many’ in the correct format most of the time. In NER approach, the ‘country’ named entity is identified correctly in most of the sentences as more tagged data available for this entity in the dataset . Hence it generates more questions from it.

In both modules, before forming questions the root word of next and previous word is checked to ensure any clue words exist. But when the root word is not identified correctly then it produces a question with incorrect format. The sentence that has anaphora words in the middle are not identified during sentence selection. Also the sentence without subject or object is possible in Tamil language. These types of sentences are not ignored. Hence it can produce meaningless questions even if the question formed is grammatically correct. In few case, if there is a multi-word named entity present part of the word is correctly identified with the named entity. So when replacing with the question word this will generate wrong format question. When extracting inflection suffix using affix stripping, there is a possibility a word without inflection get stripped in few cases. Hence this could also produce wrong format questions.

The Rule-based module produced more meaningful & grammatically correct questions compared to NER module. The system performs well in producing more meaningful questions compared to grammatically correct questions according to the evaluation results.

CHAPTER 9

CONCLUSION

A web-based Question and Answer generator capable of generating questions from a given history related textual content in Tamil is developed using a rule-based approach. The system generates WH and gap fill questions from a Rule-based module & Named Entity Recognition (NER) module. Regex patterns and gazetteers list are used in Rule-based module and Named Entity Recognizer is built using machine learning techniques. A dataset of 23k token was tagged from Grade 10 & 11 Tamil medium history textbook with novel entity tag set specific to history domain. Machine learning approach is used for NER due to the limited amount of dataset. CRF classifier is used as it is more suitable for domain specific systems due to the fact that feature design is flexible. Features suitable for the domain of interest and language are selected and experimented. POS tag, stem word, clue word, gazetteers, previous NE tag label are the features observed to contribute more for the performance. Clue word list is defined for each named entity class with words that are observed to proceed or precede a named entity tag and it helped to improve the performance for tag category with insufficient training data in the dataset. Hyper parameter optimization is applied to find the best model as the dataset is not well balanced. Random search algorithm is used for hyper parameter tuning with 5 cross validation and 30 iteration totaling 150 folds. The NER module in system achieved a fair results with micro averaged Precision, Recall and F1-Score of **87.9%, 67.1%** and **76.1%** respectively for the best feature combination. The named entities identified from gazetteer or NER approach, is replaced with a suitable question word using defined rules. Morphological Analysis(MA) is an important technique to find the inflection suffix. An affix stripping algorithm was implemented as the existing libraries for MA did not produce a good results for the system. The system was manually evaluated and then post processing rules were applied from the error analysis. The generated questions and answers from the final improved system is evaluated by 16 native Tamil speakers from different category including 3 experts. According to the evaluation results, **62.22%** of total generated questions are both grammatically and semantically correct when considering the average of responses. Krippendorff's Alpha is used to measure the inter-rater reliability score and it achieved a moderate agreement with values 0.5750 & 0.4426 respectively for grammatical & semantical correctness ratings.

CHAPTER 10

FUTURE WORK

Tagging a sufficiently large and balanced corpus would improve the performance. Deep learning techniques could be used with a large, annotated corpus.

There are only few libraries or tools exist for the NLP techniques such as POS tagging and Morphological Analysis which supports Tamil language. The existing libraries does not produce a very good results for the domain selected. Hence the accuracy is highly affected. Building a better tool for POS tagging and Morphological Analysis would improve the results.

Semantic based approach can be used that operate on deeper level to identify if context, subject or object is missing in a sentence. Anaphoric resolution to handle any word reference from previous sentence could be implemented to produce more and meaningful questions.

Text summarization before question generation would produce more meaningful questions.

The system can be enhanced to support different question types & domains.

REFERENCES

[1] R. Vijayakrishna and L. Sobha, “Domain Focused Named Entity Recognizer for Tamil Using Conditional Random Fields,” in *Proceedings of the IJCNLP-08 Workshop on NER for South and South East Asian Languages*, 2008, pp. 59–66.

[2] Malarkodi C.S, Pattabhi, R.K & Sobha L 2012, ‘Tamil NER–Coping with Real Time Challenges’, Proceedings of the Workshop on Machine Translation and Parsing in Indian Languages(MTPIL-2012), COLING, pp. 23-38.

[3] Pillai, A. Sivathanu and L. Sobha. “Named Entity Recognition for Indian Languages: A Survey.” (2013).

[4] J. Dahanayaka and A. Weerasinghe, “Named entity recognition for sinhala language,” in *2014 14th International* Conference on Advances in ICT for Emerging *Regions (ICTer)*. IEEE, 2014, pp. 215–220.

[5] Theivendiram, Pranavan & Uthayakumar, Megala & Nadarasamoorthy, Nilusija & Thayaparan, Mokanarangan & Jayasena, Sanath & Dias, Gihan & Ranathunga, Surangika. (2018). Named-Entity-Recognition (NER) for Tamil Language Using Margin-Infused Relaxed Algorithm (MIRA). 10.1007/978-3-319-75477-2_33.

[6] Nathan, Malarkodi & devi, Lalitha. (2020). Fine-Grained Named Entity Recognizer for Tamil.Tamil Internet Conference (TIC 2019)

[7] Rajendran, Srinivasan & Cn, Subalalitha. (2019). Automated Named Entity Recognition from Tamil Documents. 1-5. 10.1109/ICESIP46348.2019.8938383.

[8] R. Azeez and S. Ranathunga, "Fine-Grained Named Entity Recognition for Sinhala," *2020 Moratuwa Engineering Research Conference (MERCon)*, 2020, pp. 295-300, doi: 10.1109/MERCon50084.2020.9185296.

[9] Malarkodi C S and Sobha Lalitha Devi. 2020. A Deeper Study on Features for Named Entity Recognition. In Proceedings of the WILDRE5– 5th Workshop on Indian Language Data: Resources and Evaluation, pages 66–72, Marseille, France. European Language Resources Association (ELRA).

[10] Maithilee L. Patawar and M. A. Potey2. (2016).Named Entity Recognition from Indian tweets using Conditional Random Fields based Approach. International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 5, Issue 5, May 2016

[11] Thenmalar, S. J. Balaji, and T.V. Geetha, “Semi-supervised Bootstrapping approach for Named Entity Recognition, “International Journal on Natural Language Computing., Vol. 4, no. 5, pp. 01-14, October 2015.

[12] Patil, Nita & Patil, Ajay & Pawar, B.V.. (2020). Named Entity Recognition using Conditional Random Fields. Procedia Computer Science. 167. 1181-1188. 10.1016/j.procs.2020.03.431.

[13] Ekbal, A., Haque, R., & Bandyopadhyay, S. (2008). Named Entity Recognition in Bengali: A Conditional Random Field Approach. IJCNLP.

[14] Sobhana, N.V & Pabitra, Mitra & Ghosh, Soumya. (2010). Conditional Random Field Based Named Entity Recognition in Geological text. International Journal of Computer Applications. 1. 10.5120/72-166.

[15] Prakash Hiremath & Shambhavi B. R. A Survey on Named Entity Recognition for South Indian Languages International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-3 Issue-6, August 2014

[16] J. Li, A. Sun, J. Han, and C. Li, "A survey on deep learning for named entity recognition," arXiv preprint arXiv:1812.09449, 2018.

[17] Maud Ehrmann, Ahmed Hamdi, Elvys Linhares Pontes, Matteo Romanello, and Antoine Doucet. Named entity recognition and classification on historical documents: A survey. arXiv preprint arXiv:2109.11406, 2021

[18] Nissim, Malvina & Matheson, Colin & Reid, James. (2004). Recognising Geographical Entities in Scottish Historical Documents.

[19] Lucia C. Passaro and Alessandro Lenci. 2014. "il piave mormorava...": Recognizing locations and other named entities in italian texts on the great war. In Proceedings of the First Italian Conference on Computational Linguistics CLiC-it 2014 & and of the Fourth International Workshop EVALITA 2014, pages 286–290, Pisa (Italy).

[20] Neudecker, C., Wilms, L., Faber, W. J. und Veen, T. van (2014). Large-scale refinement of digital historic newspapers with named entity recognition. In: Proceedings of the IFLA 2014 Newspaper Section Satellite Meeting.

[21] Teemu Ruokolainen and Kimmo Kettunen. 2018. À La Recherche Du Nom Perdu–Searching for Named Entities with Stanford NER in a Finnish Historical Newspaper and Journal Collection. In 13th IAPR International Workshop on Document Analysis Systems. IEEE Computer Society, Vienna, Austria, 1–2.

[22] Sunghwan Mac Kim and Steve Cassidy. 2015. Finding Names in Trove: Named Entity Recognition for Australian Historical Newspapers. In Proc. of the Australasian Language Technology Association Workshop 2015. ACL, Parramatta, Australia, 57–65

[23] Blouin Baptiste, Benoit Favre, Jeremy Auguste, Christian Henriot. Transferring Modern Named Entity Recognition to the Historical Domain: How to Take the Step?. *Workshop on Natural Language Processing for Digital Humanities (NLP4DH)*, Dec 2021, Silchar (Online), India.

[24] Stefan Schweter and Johannes Baiter. 2019. Towards Robust Named Entity Recognition for Historic German. In Proc. of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019). ACL, Florence, Italy, 96–103.

[25] Vignesh N and S.Sowmya. “Automatic Question Generator in Tamil.” International Journal of Engineering Research & Technology (IJERT) Vol. 2 Issue 10, October - 2013 IJERTIJERT ISSN: 2278- 0181

[26] Hariharan, V., Kumar, M. A., & Soman, K. P. (2019). Named Entity Recognition in Tamil Language Using Recurrent Based Sequence Model. In Innovations in Computer Science and Engineering (pp. 91-99). Springer, Singapore.

[27] Pannerselvam, Kathiravan & Rajiakodi, Saranya. (2021). Named Entity Recognition (NER) for Social Media Tamil Posts Using Deep Learning with Singular Value Decomposition.

[28] Lovenia, Holy & Limanta, Felix & Gunawan, Agus. (2018). Automatic Question-Answer Pairs Generation from Text. 10.13140/RG.2.2.33776.92162.

[29] P. Sharma, U. Sharma and J. Kalita, "Named entity recognition in Assamese using CRFS and rules," *2014 International Conference on Asian Language Processing (IALP)*, Kuching, 2014, pp. 15-18, doi: 10.1109/IALP.2014.6973498.

[30] Dissanayake, Teshani & Hettige, Budditha. (2021). Thematic Relations Based QA Generator for Sinhala.

[31] Deepali K. Gaikwad , C. Namrata Mahender. 2018.”Question Generation System for Marathi Text” International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT).(2017).Vol.3(ISSN : 2456-3307)

[32] Dhaval Swali, Jay Palan, Ishita Shah. “Automatic Question Generation from Paragraph”, International Journal of Advanced Engineering and Research Development.

[33] Priti Gumaste, Shreya Joshi, Srushtee Khadpekar, Shubhangi Mali.(2020). AUTOMATED QUESTION GENERATOR SYSTEM USING NLP LIBRARIES. International Research Journal of Engineering and Technology (IRJET) Volume 7, Issue 6, June 2020

[34] Asif Ekbal, Rejwanul Haque, and Sivaji Bandyopadhyay. 2008. Named entity recognition in Bengali: A conditional random field approach. In Proceedings of IJCNLP, pages 589594.

[35] Nasser Alshammari, Saad Alanazi. "The impact of using different annotation schemes on named entity recognition", Egyptian Informatics Journal, 2020

[36] Mazidi, K., & Nielsen, R. D. (2015). Leveraging multiple views of text for automatic question generation. In Conati, C., Heffernan, N., Mitrovic, A., Verdejo,

M.F. (Eds.) Artificial intelligence in education (pp. 257–266). Cham: Springer International Publishing.

[37] Huang, Y., & He, L. (2016). Automatic generation of short answer questions for reading comprehension assessment. Natural Language Engineering, 22(3), 457–489.

[38] Jouault, C., Seta, K., Hayashi, Y. (2017). SOLS: An LOD based semantically enhanced open learning space supporting self-directed learning of history. IEICE Transactions on Information and Systems, 100(10), 2556–2566

[39] Mazidi, K., & Tarau, P. (2016b). Infusing NLU into automatic question generation. In: the 9th International Natural Language Generation conference, pp. 51–60.

[40] Zhang, L., & VanLehn, K. (2016). How do machine-generated questions compare to human-generated questions? Research and Practice in Technology Enhanced Learning, 11(7).

[41] Blstˇ ak, M., & Rozinajov ´ a, V. (2017). Machine learning approach to the process of question generation. ´ In Blstˇ ak, M., & Rozinajov ´ a, V. (Eds.) ´ Text, speech, and dialogue (pp. 102–110). Cham: Springer International Publishing

[42] Kaur, A., & Singh, S. (2017). Automatic question generation system for Punjabi. In: The international conference on recent innovations in science, Agriculture, Engineering and Management

[43] Liu, M., Rus, V., Liu, L. (2017). Automatic Chinese factual question generation. IEEE Transactions on Learning Technologies, 10(2), 194–204.

[44] Blstˇ ak, M., & Rozinajov ´ a, V. (2018). Building an agent for factual question generation task. In ´ 2018 World symposium on digital intelligence for systems and machines (DISA) (pp. 143–150).

[45] Flor, M., & Riordan, B. (2018). A semantic role-based approach to open-domain automatic question generation. In: the 13th Workshop on Innovative Use of NLP for Building Educational Applications, pp. 254–263.

[46] Kumar, V., Boorla, K., Meena, Y., Ramakrishnan, G., Li, Y. F. (2018). Automating reading comprehension by generating question and answer pairs. In Phung, D., Tseng, V.S., Webb, G.I., Ho, B., Ganji, M., Rashidi, L. (Eds.) Advances in knowledge discovery and data mining (pp. 335–348).

[47] Kusuma, S. F., & Alhamri, R. Z. (2018). Generating Indonesian question automatically based on Bloom's taxonomy using template based method. KINETIK:

Game Technology, Information System, Computer Network, Computing, Electronics, and Control, 3(2), 145–152.

[48] Lee, C.H., Chen, T.Y., Chen, L.P., Yang, P.C., Tsai, R.TH. (2018). Automatic question generation from children's stories for companion chatbot. In: 2018 IEEE International Conference on Information Reuse and Integration (IRI), pp. 491–494.

[49] Marrese-Taylor, E., Nakajima, A., Matsuo, Y., Yuichi, O. (2018). Learning to automatically generate fill-in-the-blank quizzes. In: the 5th workshop on natural language processing techniques for educational applications

[50] Wang, Z., Lan, A.S., Nie, W., Waters, A.E., Grimaldi, P.J., Baraniuk, R.G. (2018). QG-net: a data-driven question generation model for educational content. In: the 5th Annual ACM Conference on Learning at Scale, pp. 15–25.

[51] Zhang, T., Quan, P., et al. (2018). Domain specific automatic Chinese multiple-type question generation. In 2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), IEEE, pp. 1967– 1971.

[52] Tamura, Y., Takase, Y., Hayashi, Y., Nakano, Y. I. (2015). Generating quizzes for history learning based on Wikipedia articles. In Zaphiris, P., & Ioannou, A. (Eds.) Learning and collaboration technologies (pp. 337–346).

[53] Helena Hubková, Pavel Kral, and Eva Pettersson. 2020. Czech Historical Named Entity Corpus v 1.0. In Proc. of the 12th Language Resources and Evaluation Conference. ELRA, Marseille, France, 4458–4465.

[54] Tanti Kristanti and Laurent Romary. 2020. DeLFT and Entity-Fishing: Tools for CLEF HIPE 2020 Shared Task. In Working Notes of CLEF 2020 - Conference and Labs of the Evaluation Forum, Linda Cappellato, Carsten Eickhoff, Nicola Ferro, and Aurélie Névéol (Eds.), Vol. 2696. CEUR-WS, Thessaloniki, Greece, 1–10.

[55] Vera Provatorova, Svitlana Vakulenko, Evangelos Kanoulas, Koen Dercksen, and Johannes M van Hulst. 2020. Named Entity Recognition and Linking on Historical Newspapers: UvA.ILPS & REL at CLEF HIPE 2020. In Working Notes of CLEF 2020 - Conference and Labs of the Evaluation Forum, Linda Cappellato, Carsten Eickhoff, Nicola Ferro, and Aurélie Névéol (Eds.), Vol. 2696. CEUR-WS, Thessaloniki, Greece, 8.

[56] Stefan Schweter and Luisa März. 2020. Triple E - Effective Ensembling of Embeddings and Language Models for NER of Historical German. In Working Notes of CLEF 2020 - Conference and Labs of the Evaluation Forum, Linda Cappellato, Carsten Eickhoff, Nicola Ferro, and Aurélie Névéol (Eds.), Vol. 2696. CEUR-WS, Thessaloniki, Greece, 1–13.

[57] Kurdi, Ghader & Leo, Jared & Parsia, Bijan & Sattler, Uli & Al-Emari, Salam. (2019). A Systematic Review of Automatic Question Generation for Educational Purposes. International Journal of Artificial Intelligence in Education. 30. 10.1007/s40593-019-00186-y.

[58] Xinya Du and Claire Cardie. 2017. Identifying where to focus in reading comprehension for neural question generation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2067–2073.

[59] Sun, Yuan & Chen, Chaofan & Chen, Andong & Zhao, Xiaobing. (2021). Tibetan Question Generation Based on Sequence to Sequence Model. Computers, Materials & Continua. 68. 3203-3213. 10.32604/cmc.2021.016517.

[60] Yao Zhao, XiaochuanNi, YuanyuanDing, and Qifa Ke. 2018. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3901–3910.

[61] Yanghoon Kim, Hwanhee Lee, Joongbo Shin, and Kyomin Jung. 2019. Improving neural question generation using answer separation. In *AAAI Conference on Artificial Intelligence (AAAI)*.

[62] Alexander Fabbri, Patrick Ng, Zhiguo Wang, Ramesh Nallapati, and Bing Xiang. 2020. Template-Based Question Generation from Retrieved Sentences for Improved Unsupervised Question Answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4508–4513, Online. Association for Computational Linguistics.

[63] Iulian Vlad Serban, Alberto García-Durán, Caglar Gulcehre, Sungjin Ahn, Sarath Chandar, Aaron Courville, and Yoshua Bengio. 2016. Generating Factoid Questions With Recurrent Neural Networks: The 30M Factoid Question-Answer Corpus. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 588–598, Berlin, Germany. Association for Computational Linguistics.

[64] Reddy, Sathish & Raghu, Dinesh & Khapra, Mitesh & Joshi, Sachindra. (2017). Generating Natural Language Question-Answer Pairs from a Knowledge Graph Using a RNN Based Question Generation Model. 376-385. 10.18653/v1/E17-1036.

[65] Rakshit, Geetanjali & Flanigan, Jeffrey. (2021). ASQ: Automatically Generating Question-Answer Pairs using AMRs.

[66] Preksha Nema and Mitesh M. Khapra. 2018. Towards a Better Metric for Evaluating Question Generation Systems. In *Proceedings of the 2018 Conference on*

Empirical Methods in Natural Language Processing, pages 3950–3959, Brussels, Belgium. Association for Computational Linguistics.

[67] J. Li, A. Sun, J. Han and C. Li, "A Survey on Deep Learning for Named Entity Recognition," in *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 1, pp. 50-70, 1 Jan. 2022, doi: 10.1109/TKDE.2020.2981314.

[68] A.Salini and U.Jeyapriya, "Named Entity Recognition Using Machine Learning Approaches", in International Journal of Innovative Research in Science, Engineering and Technology, Vol. 6, Special Issue 11, Sep 2017.

[69] Pan, Liangming & Lei, Wenqiang & Chua, Tat-Seng & Kan, Min-Yen. (2019). Recent Advances in Neural Question Generation.

[70] N. Alsaaran and M. Alrabiah, "Classical Arabic Named Entity Recognition Using Variant Deep Neural Network Architectures and BERT," in *IEEE Access*, vol. 9, pp. 91537-91547, 2021, doi: 10.1109/ACCESS.2021.3092261.

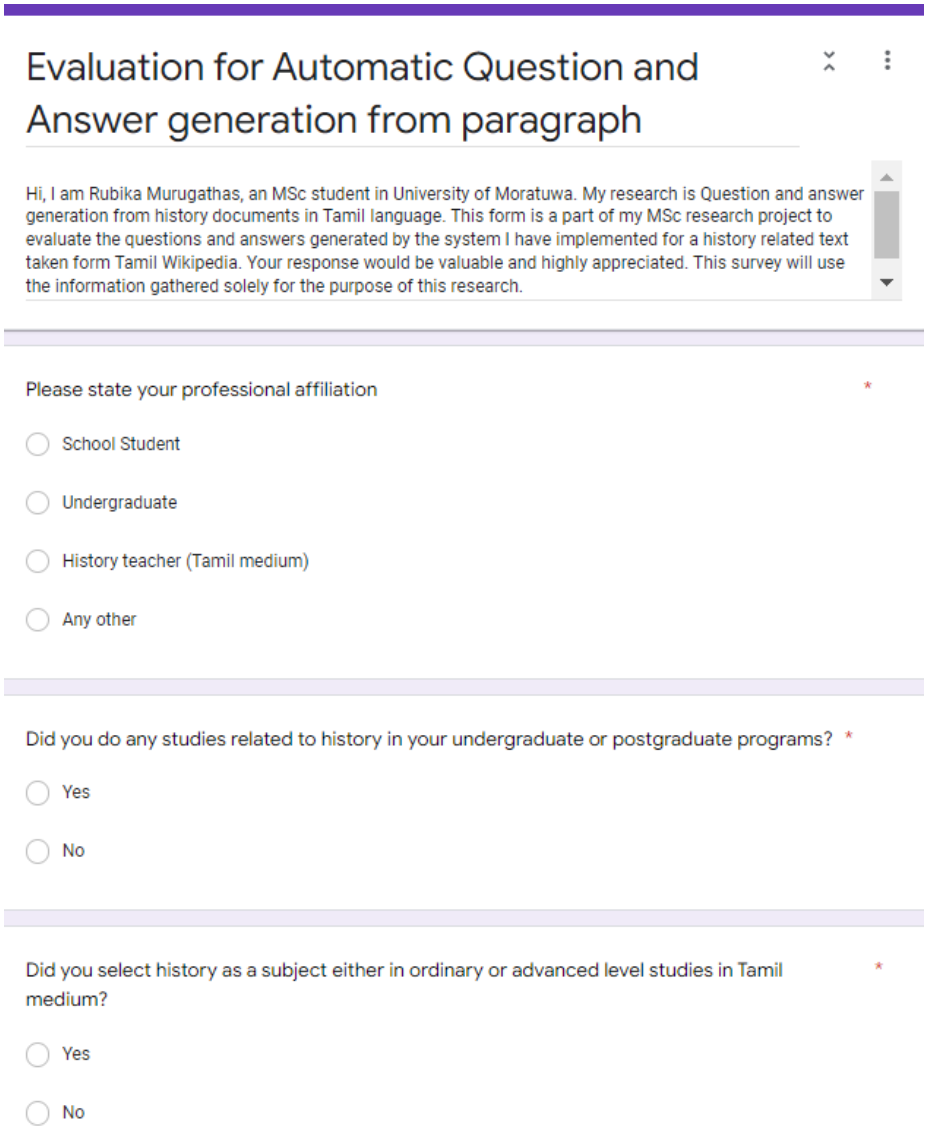
[71] Goyal A, Gupta V, Kumar M (2021) A deep learning-based bilingual hindi and punjabi named entity recognition system using enhanced word embeddings. Knowl Based Syst, 107601

[72] Eligüzel, N., Çetinkaya, C. & Dereli, T. Application of named entity recognition on tweets during earthquake disaster: a deep learning-based approach. Soft Comput 26, 395–421 (2022).

[73] T. Al-Moslmi, M. Gallofré Ocaña, A. L. Opdahl and C. Veres, "Named Entity Extraction for Knowledge Graphs: A Literature Overview," in *IEEE Access*, vol. 8, pp. 32862-32881, 2020, doi: 10.1109/ACCESS.2020.2973928.

APPENDIX – A: Question Evaluation Form

Question and Answer Evaluation Form



Evaluation for Automatic Question and Answer generation from paragraph

Hi, I am Rubika Murugathas, an MSc student in University of Moratuwa. My research is Question and answer generation from history documents in Tamil language. This form is a part of my MSc research project to evaluate the questions and answers generated by the system I have implemented for a history related text taken form Tamil Wikipedia. Your response would be valuable and highly appreciated. This survey will use the information gathered solely for the purpose of this research.

Please state your professional affiliation *

- School Student
- Undergraduate
- History teacher (Tamil medium)
- Any other

Did you do any studies related to history in your undergraduate or postgraduate programs? *

- Yes
- No

Did you select history as a subject either in ordinary or advanced level studies in Tamil medium? *

- Yes
- No

Evaluation form

Questions and Answers generated by the system

தொழில் புரட்சி பற்றிய விக்கிபீடியாவில் இருந்து பெறப்பட்ட உள்ளடக்கம் கீழே உள்ளது. 44 கேள்விகள் மற்றும் பதில்கள் பந்தியில் இருந்து கணினியால் உருவாக்கப்பட்டுள்ளன. கேள்வி வடிவம் மற்றும் பொருளின் அடிப்படையில் கேள்வியின் சரியான தன்மையை மதிப்பிடுவதற்கு 4 தேர்வுகள் கொடுக்கப்பட்டுள்ளன. மிகவும் பொருத்தமான ஒன்றைத் தேர்ந்தெடுக்கவும்.

தொழிற்புரட்சி என்பது 1750-1850ஆம் ஆண்டுகளுக்கு இடைப்பட்ட காலத்தில் உலகளவில் ஏற்பட்ட மிகப்பெரிய தொழில்நுட்ப, பொருளாதார, நாகரிக மாற்றங்களைக் குறிக்கும். தொழிற்புரட்சி முதலில் இங்கிலாந்தில் தோன்றியது. பின்னர் 19ஆம் நூற்றாண்டில் ஐரோப்பாவிலும், வட அமெரிக்காவிலும் பரவியது. செருமனியில் 1871இல் பேரரசு நிறுவப்பட்ட பின்னரும், அமெரிக்காவில் உள்நாட்டுப் போருக்குப் பின்னரும், உருசியாவில் 1917ஆம் ஆண்டுக்குப் பின்னரும் தொழிற் புரட்சி தொடங்கியது. இந்தியா, சீனா போன்ற ஆசிய நாடுகளில் 20ஆம் நூற்றாண்டில் தொழிற்புரட்சி தொடங்கியது. இரண்டாம் உலகப் போருக்குப் பின்னரே உலகெங்கும் தொழிற்சாலை முறை தோன்றியது. தொழிற்புரட்சி மனித சமுதாயத்தின் பெரும் திருப்புமுனையாக அமைந்ததுடன், மனிதர்களின் அன்றாட வாழ்க்கையின் எல்லா கூறுகளிலும் ஏதாவது ஒரு வகையில் தாக்கத்தை ஏற்படுத்தியது எனலாம். தொழிற்புரட்சியின் காரணமாக ஒரு நாட்டின் பொருளாதாரம் வேளாண்மையை மட்டுமே சார்ந்து இருந்த நிலை மாறி, தொழிலகப் படைப்புகளின் பங்களிப்பும் கூடத் தொடங்கியது. தொழிற்புரட்சியின் விளைவாகத் தொழில்நுட்பக் கல்வி விரிவடைந்தது. பஞ்சு நூற்பாலைகளில் தொடங்கி மாந்தர்கள் கைகளால் செய்த பற்பல பணிகளை இயந்திரமயமாக்கி, பெரும் எண்ணிக்கையிலும், மலிவாகவும் பொருள்களைப் படைக்கப் புதுமுறைகள் உருவாக்கினார்கள். உற்பத்தித் துறை மட்டுமல்லாது அச்சுத்தொழில், வெகு மக்கள் தொடர்பு ஊடகங்கள், போக்குவரத்து, கல்வி, மருத்துவம் முதலான பல துறைகள் பெருகி சேவை என்பது தொழில் என்ற நிலைக்கு மாறியுள்ளது. இங்கிலாந்தில் தோன்றிய தொழிற்புரட்சி மேற்கத்திய நாடுகளில் அரசியல் பொருளாதாரம் வாழ்வியல் மற்றும் தொழிற்சாலை, வர்த்தகம் ஆகியவற்றில் அடிப்படை மாற்றத்தை ஏற்படுத்தியது. தொழிற்புரட்சி என்ற சொல் விவசாயம் சார்ந்த கைவினைத் தொழில் சார்ந்த உழைப்பாளிகளை மையப் படுத்திய பொருளாதார அமைப்பிலிருந்து இயந்திர உற்பத்தி, தொழிற்சாலைகள், மூலதனம் பரிவர்த்தனை போன்றவற்றை மையப்படுத்திய முறைக்கு மாறுவதைக் குறிப்பதாகும்.

முதலீட்டாளர்கள் கச்சாப்பொருளைச் சேகரித்து அதனைச் செய்பொருட்களாக்க கைவினைஞர்கள் மற்றும் கலைஞர்களுக்கு விநியோகம் செய்தனர். வேறுபட்ட பல்வேறு இடங்களில் புதிய கண்டுபிடிப்புகள் தோன்றியதால் இயந்திரங்கள் உருவாயின.

தொழிற்புரட்சி என்ற சொல் புதிய அறிவியல் தொழில்நுட்ப மாற்றங்களை ஏற்றுக் கொண்டு செயல்பட்டு ஆலைகளில் பொருட்கள் பேரளவில் உற்பத்தி செய்ததை விளக்குவதற்குப் பயன்பட்டது. இயந்திரங்கள் உற்பத்தி முறையை முழுவதுமாக மாற்றியமைத்தன. இப்புரட்சி முழுவதுமாகப் பொருள் உற்பத்தியை அடிப்படையாகக் கொண்டிருந்தது. வன்முறையின்றி, இரத்தமின்றி அமைதியான முறையில் தொழில் உற்பத்தியில் மாற்றத்தை ஏற்படுத்தியது. தொழிற்புரட்சி என்ற சொல்லை முதன் முதலில் பிளாங்கி என்ற பிரஞ்சு எழுத்தாளர் உருவாக்கினார்.

18ஆம் நூற்றாண்டின் இடைக்காலம் வரை மக்கள் தங்கள் வீடுகளிலும் பட்டறைகளிலும் தங்கள் உள்ளூர் தேவைக்கேற்ற முறையில் பொருள்களை உற்பத்தி செய்தனர். இங்கிலாந்தில் 18-ஆம் நூற்றாண்டில் தொழிற்புரட்சிக்குச் சாதகமான சூழ்நிலை நிலவியது. இங்கிலாந்து கடல் போக்குவரத்திலும், காலனியாதிக்கத்திலும் முதன்மையான உலகநாடாக விளங்கியது. அதனுடைய கடல் வலிமையும், குடியேற்ற ஆதிக்கமும் அந்நாட்டின் தொழில் மற்றும் தொழிற்சாலைகளின் தீவிர வளர்ச்சிக்குத் தேவையான கச்சாப்பொருள்களையும், புதிய சந்தைகளையும் பெற்றுத் தந்தன. மேலும் கனிம வளம் போன்ற பல இயற்கைவளங்களைப் பிரித்தானியா கொண்டிருந்தது. அதனுடைய கடற்கரையமைப்பு மற்றும் பருவநிலை ஆகியவை தொழிற்சாலைகளுக்குச் சாதகமாக இருந்தன.

இங்கிலாந்தின் பொருளாதார முன்னேற்றத்திற்கு தனியார்களது பெரும்பங்கும் தனிச்சிறப்பளித்தது. தொழிற்புரட்சியின்போது இங்கிலாந்தில் செய்யப்பட்ட இயந்திரங்கள் தரமானதாகவும், வலிமை வாய்ந்ததாகவும் இருந்தன. சமயக் கொடுமையின் காரணமாக ஸ்பெயின் மற்றும் பிரான்சிலிருந்து வெளியேறிய புராட்டஸ்டண்டு கைவினைஞர்கள் இங்கிலாந்திற்குக் குடியேறினர். இங்கிலாந்து அரசாங்கம் அவர்களுக்கு அடைக்கலமும் பண உதவியும் தந்து அதற்குப் பதிலாக அவர்களின் திறமையை முழுமையாகப் பயன்படுத்திக் கொண்டது. இதனால் இங்கிலாந்தில் தொழிற்சாலைகளுக்குப் புத்துணர்ச்சி ஏற்பட்டது. தேவைப் பெருக்கத்தை ஈடுகட்ட உற்பத்தியின் வேகத்தைப் பெருக்கக் கூடிய வழிவகைகள் நாடப்பட்டன. எனவே பிரித்தானியாவில் பல்வேறு இடங்களில் பெருமளவில் தொழிற்சாலைகள் ஏற்பட்டு வளர்ச்சியடைந்தன.

1700களில் அகண்ட பிரித்தானியாவின் ஒரு பகுதியில் தொழிலாளர்களைச் சார்ந்திருந்த பொருளாதாரமானது இயந்திரங்களைச் சார்ந்த உற்பத்தி முறைக்கு மாறியது.இது துணி உற்பத்தித் தொழிலின் இயந்திரமயமாக்கம், இரும்பு உற்பத்தித் தொழில்நுட்பத்தின் வளர்ச்சி, தூய்மைப்படுத்திய நிலக்கரியின் கூடிய பயன்பாடு ஆகியவற்றுடன் தொடங்கியது. கால்வாய்கள், சாலைகள், தொடர்வண்டிப் பாதைகள் என்பன அமைக்கப்பட்டதும் வணிக விரிவாக்கத்துக்கு வழிகோலியது.

இரயில் வண்டிகள் இயக்கவும், இரயில் பாதைகளை உருவாக்கவும் ஜார்ஜ் ஸ்டீபென்சன் எடுத்த முயற்சிகள் தொழிற்புரட்சியின் முக்கியக் கூறுகளில் ஒன்றாக அமைந்தது. உற்பத்திக்குத் தேவையான பொருட்களைப் பெறவும், உற்பத்தி செய்யப்பட்ட பொருட்களைப் பெறவும், சந்தைகளில் விற்கவும், தேவையான இடங்களுக்கு விரைந்து இடையூறின்றி பொருட்களை அனுப்பவும் இவருடைய கண்டுபிடிப்புகள் பெரிதும் உதவின. நீராவிப் பொறியின் கண்டுபிடிப்பினாலும் அதன் ஆற்றல் முதலில் நெசவு இயந்திரங்களிலும் பின்னர் இரும்பு தயாரித்தல் போன்றவற்றிலும் பயன்படுத்தப் பட்டதன் மூலமும் உற்பத்தி வெகுவாகப் பெருகியது.

முதல் தொழிற்புரட்சி 18ஆம் நூற்றாண்டில் தொடங்கி 1850ஆம் ஆண்டில் இரண்டாம் தொழிற்புரட்சியுடன் இணைந்தது. 1850ஆம் ஆண்டளவில், நீராவிக் கப்பல், நீராவித் தொடர்வண்டிகள் பின்னர் உள்ளெரி பொறிகள், மின் உற்பத்தி என்பவற்றின் அறிமுகத்தோடு தொழில்நுட்பமும், பொருளாதாரமும் வீறு கொண்டு வளர்ந்தன.

தொழிற்புரட்சியின் கண்டுபிடிப்புகளில் முதன்மையானது நீராவி இயந்திரமாகும். நியூகாமன் என்பவரின் நீராவி இயந்திரக் கண்டுபிடிப்பைப் பற்றிப் படித்த ஜேம்ஸ் வாட் என்பவர் புதிய நீராவி இயந்திரத்தை 1769ல் உருவாக்கினார். இதனால் நெசவுத்தொழிற்சாலைகளில் குதிரை மற்றும் நீர் ஆற்றலுக்குப் பதிலாக நீராவி ஆற்றல் பயன்படுத்தப்பட்டது. ஜார்ஜ் ஸ்டீபென்சன் நீராவி இரயில் இயந்திரத்தை 1825ல் கண்டுபிடித்தார். 1830ல் முதல் பயணியர் தொடர்வண்டி மான்செஸ்டருக்கும் லிவர்பூலுக்கும் இடையே விடப்பட்டது. 1814ல் கண்டுபிடிக்கப்பட்ட நீராவி அச்சு இயந்திரத்தால் அச்சுப்பொருட்களின் விலை குறைந்தது. இதற்குப் பின்னர் மைக்கேல் பாரடே டைனமோவைக் கண்டுபிடித்தார். ஆபிரகாம் டெர்பி என்பவர் இரும்புத்தாது உருகுவதற்கு கரிக்குப் மாற்றாக நிலக்கரியைப் பயன்படுத்தும் ஆய்வுகளை மேற்கொண்டார்.
1760இல் ஜான் ஸ்மீட்டன் என்பவர் டெர்பியின் ஆய்வுடன் நீராற்றலை இணைத்து அதனை மேம்படுத்தினார். ஹம்பிரி டேவி கண்டுபிடித்த பாதுகாப்பு விளக்கினால் சுரங்க வேலை செய்வோர் பாதுகாப்புடன் பணிபுரிந்தனர். 1784இல் ஹென்றி கார்ட் இரும்பைத் துண்டாக்க உருளையைப் பயன்படுத்தும் புதிய முறையை அறிமுகப்படுத்தினார். 1856ல் பெர்ஸ்மர் இரும்பு எஃகு உற்பத்தி செய்யும் புதிய முறையைக் கண்டுபிடித்தார். இக்காலம் முதல் நிலக்கரியும் இரும்பும் நீராவியுடன் இணைந்து செயல்பட்டதால் தொழில்மயமாதலுக்கு அடித்தளமாகியது.

கேள்வி: தொழிற்புரட்சி என்பது எந்த ஆண்டு தொடக்கம் எந்த ஆண்டு வரையான ஆண்டுகளுக்கு இடைப்பட்ட காலத்தில் உலகளவில் ஏற்பட்ட மிகப்பெரிய தொழில்நுட்ப, பொருளாதார, நாகரிக மாற்றங்களைக் குறிக்கும்? *
விடை: 1750-1850

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: தொழிற்புரட்சி முதலில் எந்த நாட்டில் தோன்றியது? விடை: இங்கிலாந்தில் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: செருமனியில் எந்த வருடம் பேரரசு நிறுவப்பட்ட பின்னரும், அமெரிக்காவில் உள்நாட்டுப் போருக்குப் பின்னரும், உருசியாவில் 1917ஆம் ஆண்டுக்குப் பின்னரும் தொழிற் புரட்சி தொடங்கியது? *
விடை: 1871இல்

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: செருமனியில் 1871இல் பேரரசு நிறுவப்பட்ட பின்னரும், எந்த நாட்டில் உள்நாட்டுப் போருக்குப் பின்னரும், உருசியாவில் 1917ஆம் ஆண்டுக்குப் பின்னரும் தொழிற் புரட்சி தொடங்கியது? விடை: அமெரிக்காவில் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: இந்தியா, சீனா போன்ற ஆசிய நாடுகளில் எத்தனையாம் நூற்றாண்டில் தொழிற்புரட்சி தொடங்கியது? விடை: 20ஆம் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: இந்தியா, சீனா போன்ற எந்த கண்ட நாடுகளில் 20ஆம் நூற்றாண்டில் தொழிற்புரட்சி தொடங்கியது? விடை: ஆசிய *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: எத்தனையாம் உலகப் போருக்குப் பின்னரே உலகெங்கும் தொழிற்சாலை முறை தோன்றியது. விடை: இரண்டாம் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: எந்த துறை மட்டுமல்லாது அச்சுத்தொழில், வெகு மக்கள் தொடர்பு ஊடகங்கள், போக்குவரத்து, கல்வி, மருத்துவம் முதலான பல துறைகள் பெருகி சேவை என்பது தொழில் என்ற நிலைக்கு மாறியுள்ளது. விடை: உற்பத்தித் துறை *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: எந்த நாட்டில் தோன்றிய தொழிற்புரட்சி மேற்கத்திய நாடுகளில் அரசியல் பொருளாதாரம் வாழ்வியல் மற்றும் தொழிற்சாலை, வர்த்தகம் ஆகியவற்றில் அடிப்படை மாற்றத்தை ஏற்படுத்தியது? விடை: இங்கிலாந்தில் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: எத்தனையாம் நூற்றாண்டின் இடைக்காலம் வரை மக்கள் தங்கள் வீடுகளிலும் பட்டறைகளிலும் தங்கள் உள்ளூர் தேவைக்கேற்ற முறையில் பொருள்களை உற்பத்தி செய்தனர்? விடை: 18ஆம் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: எந்த நாட்டில் 18-ஆம் நூற்றாண்டில் தொழிற்புரட்சிக்குச் சாதகமான சூழ்நிலை நிலவியது? விடை: இங்கிலாந்தில் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: எந்த நாடு கடல் போக்குவரத்திலும், காலனியாதிக்கத்திலும் முதன்மையான உலகநாடாக விளங்கியது? விடை: இங்கிலாந்து *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: எந்த நாட்டின் பொருளாதார முன்னேற்றத்திற்கு தனியார்களது பெரும்பங்கும் தனிச்சிறப்பளித்தது? விடை: இங்கிலாந்தின் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: எந்த நாடு அரசாங்கம் அவர்களுக்கு அடைக்கலமும் பண உதவியும் தந்து அதற்குப் பதிலாக அவர்களின் திறமையை முழுமையாகப் பயன்படுத்திக் கொண்டது? விடை: இங்கிலாந்து *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: 1700களில் அகண்ட எந்த நாட்டின் ஒரு பகுதியில் தொழிலாளர்களைச் சார்ந்திருந்த பொருளாதாரமானது இயந்திரங்களைச் சார்ந்த உற்பத்தி முறைக்கு மாறியது? விடை: பிரித்தானியாவின் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: எத்தனையாம் ஆண்டு ஆண்டளவில், நீராவிக் கப்பல், நீராவித் தொடர்வண்டிகள் பின்னர் உள்ளெரி பொறிகள், மின் உற்பத்தி என்பவற்றின் அறிமுகத்தோடு தொழில்நுட்பமும், பொருளாதாரமும் வீறு கொண்டு வளர்ந்தன? விடை: 1850ஆம் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: யார் என்பவரின் நீராவி இயந்திரக் கண்டுபிடிப்பைப் பற்றிப் படித்த ஜேம்ஸ் வாட் என்பவர் புதிய நீராவி இயந்திரத்தை 1769ல் உருவாக்கினார்? விடை: நியூகாமன் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: ஜார்ஜ் ஸ்டீபென்சன் நீராவி இரயில் இயந்திரத்தை எத்தனையாம் ஆண்டு கண்டுபிடித்தார்? விடை:1825ல் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: எத்தனையாம் ஆண்டு முதல் பயணியர் தொடர்வண்டி மான்செஸ்டருக்கும் லிவர்பூலுக்கும் இடையே விடப்பட்டது? விடை: 1830ல் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: ஆபிரகாம் டெர்பி என்பவர் இரும்புத்தாது உருகுவதற்கு கரிக்குப் மாற்றாக எந்த பொருளைப் பயன்படுத்தும் ஆய்வுகளை மேற்கொண்டார்? விடை: நிலக்கரியைப் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: எத்தனையாம் ஆண்டு பெர்ஸ்மர் இரும்பு எஃகு உற்பத்தி செய்யும் புதிய முறையைக் கண்ட்டுபிடித்தார்? விடை: 1856ல் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: தொழிற்புரட்சி என்ற சொல்லை முதன் முதலில் பிளாங்கி என்ற எந்த நாடு எழுத்தாளர் உருவாக்கினார்? விடை: பிரஞ்சு *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: எத்தனையாம் நூற்றாண்டின் இடைக்காலம் வரை மக்கள் தங்கள் வீடுகளிலும் பட்டறைகளிலும் தங்கள் உள்ளூர் தேவைக்கேற்ற முறையில் பொருள்களை உற்பத்தி செய்தனர்? விடை: 18ஆம் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: சமயக் கொடுமையின் காரணமாக எந்த நாட்டின் மற்றும் பிரான்சிலிருந்து வெளியேறிய புராட்டஸ்டண்டு கைவினைஞர்கள் இங்கிலாந்திற்குக் குடியேறினர்? விடை: ஸ்பெயின் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: _____, _____, _____ _____ என்பன அமைக்கப்பட்டதும் வணிக விரிவாக்கத்துக்கு வழிகோலியது. விடை: கால்வாய்கள்,சாலைகள்,தொடர்வண்டிப் பாதைகள் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: இரயில் வண்டிகள் இயக்கவும், இரயில் பாதைகளை உருவாக்கவும் யார் எடுத்த முயற்சிகள் தொழிற்புரட்சியின் முக்கியக் கூறுகளில் ஒன்றாக அமைந்தது? விடை: ஜார்ஜ் ஸ்டீபென்சன் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: முதல் தொழிற்புரட்சி 18ஆம் நூற்றாண்டில் தொடங்கி எத்தனையாம் ஆண்டு இரண்டாம் தொழிற்புரட்சியுடன் இணைந்தது? விடை: 1850ஆம் ஆண்டில் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: முதல் தொழிற்புரட்சி எத்தனையாம் நூற்றாண்டில் தொடங்கி 1850ஆம் ஆண்டில் இரண்டாம் தொழிற்புரட்சியுடன் இணைந்தது? விடை: 18ஆம் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: நியூகாமன் என்பவரின் நீராவி இயந்திரக் கண்டுபிடிப்பைப் பற்றிப் படித்த ஜேம்ஸ் வாட் என்பவர் புதிய நீராவி இயந்திரத்தை எந்த வருடம் உருவாக்கினார்? விடை: 1769ல் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: நியூகாமன் என்பவரின் நீராவி இயந்திரக் கண்டுபிடிப்பைப் பற்றிப் படித்த யார் புதிய நீராவி இயந்திரத்தை 1769ல் உருவாக்கினார்? விடை: ஜேம்ஸ் வாட் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: ஜார்ஜ் ஸ்டீபென்சன் நீராவி இரயில் இயந்திரத்தை எந்த வருடம் கண்டுபிடித்தார்? விடை: 1825ல் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: ஜார்ஜ் ஸ்டீபென்சன் நீராவி எந்த இயந்திரத்தை 1825ல் கண்டுபிடித்தார். விடை: இரயில் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: எந்த வருடம் முதல் பயணியர் தொடர்வண்டி மான்செஸ்டருக்கும் லிவர்பூலுக்கும் இடையே விடப்பட்டது? விடை: 1830ல் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: எந்த வருடம் கண்டுபிடிக்கப்பட்ட நீராவி அச்சு இயந்திரத்தால் அச்சுப்பொருட்களின் விலை குறைந்தது? விடை: 1814ல் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: எந்த வருடம் ஜான் ஸ்மீட்டன் என்பவர் டெர்பியின் ஆய்வுடன் நீராற்றலை இணைத்து அதனை மேம்படுத்தினார்? விடை: 1760இல் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: 1760இல் எந்த நாடு ஸ்மீட்டன் என்பவர் டெர்பியின் ஆய்வுடன் நீராற்றலை இணைத்து அதனை மேம்படுத்தினார்? விடை: ஜான் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: எந்த வருடம் ஹென்றி கார்ட் இரும்பைத் துண்டாக்க உருளையைப் பயன்படுத்தும் புதிய முறையை அறிமுகப்படுத்தினார்? விடை: 1784இல் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: 1784இல் யார் இரும்பைத் துண்டாக்க உருளையைப் பயன்படுத்தும் புதிய முறையை அறிமுகப்படுத்தினார்? விடை: ஹென்றி கார்ட் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: எந்த வருடம் ஜான் ஸ்மீட்டன் என்பவர் டெர்பியின் ஆய்வுடன் நீராற்றலை இணைத்து அதனை மேம்படுத்தினார்? விடை: 1760இல் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: 1760இல் எந்த நாடு ஸ்மீட்டன் என்பவர் டெர்பியின் ஆய்வுடன் நீராற்றலை இணைத்து அதனை மேம்படுத்தினார்? விடை: ஜான் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: எந்த வருடம் ஹென்றி கார்ட் இரும்பைத் துண்டாக்க உருளையைப் பயன்படுத்தும் புதிய முறையை அறிமுகப்படுத்தினார்? விடை: 1784இல் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: 1784இல் யார் இரும்பைத் துண்டாக்க உருளையைப் பயன்படுத்தும் புதிய முறையை அறிமுகப்படுத்தினார்? விடை: ஹென்றி கார்ட் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: எந்த வருடம் பெர்ஸ்மர் இரும்பு எஃகு உற்பத்தி செய்யும் புதிய முறையைக் கண்டுபிடித்தார்? விடை: 1856ல் *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

கேள்வி: இதற்குப் பின்னர் யார் டைனமோவைக் கண்டுபிடித்தார்? விடை: மைக்கேல் பாரடே *

- கேள்வி வடிவம் தவறானது. எனினும் கேள்வி அர்த்தமுள்ளது.
- கேள்வி வடிவம் சரியானது, ஆனால் கேள்வி பொருத்தமற்றது.
- கேள்வி வடிவம் சரியானது, மற்றும் கேள்வி பொருத்தமானது.
- கேள்வி வடிவம் தவறானது மற்றும் கேள்வி அர்த்தமற்றது.

APPENDIX – B: Implementation Code

CRF implementation

```
def word2features(sent, i):
    word = sent[i][0]
    postag = sent[i][1]

    features = {
        'bias': 1.0,
        'suffix': findinflectionsuffix(word),
        'isnumber': check_if_word_is_number(word),
        'postag': postag,
        'clueword': check_clue_word_list(word),
        'gazetteerword': gazetteer_check(word),
        'isordinal': check_if_ordinal_numeric_word(word),
        'isyearformat': is_year_format(word),
        'stemword': get_stem_word(word)
    }

    if i > 0:
        pword1 = sent[i - 1][0]
        postag1 = sent[i - 1][1]
        plabel1 = sent[i - 1][2]
        features.update({
            '-1:word': pword1,
            '-1:postag': postag1,
            '-1:clueword': check_clue_word_list(pword1),
            '-1:isnumber': check_if_word_is_number(pword1),
            '-1:gazetteerword': gazetteer_check(pword1),
            '-1:previouslabel': plabel1,
            '-1:isordinal': check_if_ordinal_numeric_word(pword1),
            '-1:isyearformat': is_year_format(pword1),
            '-1:isconjunction': word_is_conjunction(pword1),
            '-1:endwithcomma': check_if_word_ends_with_comma(pword1),
            '-1:stemword': get_stem_word(pword1),
        })
    else:
        features['BOS'] = True

       if i < len(sent) - 1:
            nword1 = sent[i + 1][0]
            postag1 = sent[i + 1][1]
            features.update({
                '+1:word': nword1,
                '+1:postag': postag1,
                '+1:clueword': check_clue_word_list(nword1),
                '+1:gazetteerword': gazetteer_check(nword1),
                '+1:isnumber': check_if_word_is_number(nword1),
                '+1:isconjunction': word_is_conjunction(nword1),
                '+1:endwithcomma': check_if_word_ends_with_comma(nword1),
                '+1:stemword': get_stem_word(nword1)
            })
       else:
            features['EOS'] = True

       return features
```

Training and Testing NER model

```
def train_and_test():
    sentences = read_conll_as_list()
    X = [sent2features(sentence) for sentence in sentences]
    y = [sent2labels(sentence) for sentence in sentences]

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)

    crf = sklearn_crfsuite.CRF(
        algorithm='lbfgs',
        max_iterations=100,
        all_possible_transitions=True
    )
    params_space = {
        'c1': scipy.stats.expon(scale=0.5),
        'c2': scipy.stats.expon(scale=0.05),
    }

    f1_scorer = make_scorer(metrics.flat_f1_score,
                            average='weighted', labels=label_category)

    rs = RandomizedSearchCV(crf, params_space,
                            cv=5,
                            verbose=1,
                            n_jobs=-1,
                            n_iter=30,
                            scoring=f1_scorer)
    rs.fit(X_train, y_train)

    print('best params:', rs.best_params_)
    print('best CV score:', rs.best_score_)
    print('model size: {:0.2f}M'.format(rs.best_estimator_.size_ / 1000000))

    filename = 'crf_model.sav'
    pickle.dump(rs, open(filename, 'wb'))

    labels = list(rs.classes_)
    labels.remove('O')

    y_pred_test = rs.predict(X_test)
    print('F1 score on the test set = {}\n'.format(flat_f1_score(y_test, y_pred_test,
                                                                 average='weighted', labels=labels)))
    print('Accuracy on the test set = {}\n'.format(flat_accuracy_score(y_test, y_pred_test)))

    sorted_labels = sorted(labels, key=lambda name: (name[1:], name[0]))
    print('Test set classification report: \n\n{}'.format(flat_classification_report(y_test, y_pred_test,
                                                                                     labels=sorted_labels, digits=3)))
```

APPENDIX – C: SPSS Tool Output

```
Run MATRIX procedure:

Krippendorff's Alpha Reliability Estimate


            Alpha    LL95%CI    UL95%CI      Units   Observrs      Pairs
Nominal      .5750      .4599      .6812    44.0000    16.0000   5280.0000

Probability (q) of failure to achieve an alpha of at least alphamin:
   alphamin          q
      .9000     1.0000
      .8000     1.0000
      .7000      .9921
      .6700      .9587
      .6000      .6530
      .5000      .0939

Number of bootstrap samples:
  10000

Judges used in these computations:
Columns   1 -  11
 Rater1   Rater2   Rater3   Rater4   Rater5   Rater6   Rater7   Rater8   Rater9   Rater10  Rater11
Columns 12 - 16 Rater12 Rater13 Rater14 Rater15 Rater16

Examine output for SPSS errors and do not interpret if any are found

------ END MATRIX -----
```

```
Run MATRIX procedure:

Krippendorff's Alpha Reliability Estimate


            Alpha    LL95%CI    UL95%CI      Units   Observrs      Pairs
Nominal      .4426      .3033      .5720    44.0000    16.0000   5280.0000

Probability (q) of failure to achieve an alpha of at least alphamin:
   alphamin          q
      .9000     1.0000
      .8000     1.0000
      .7000     1.0000
      .6700      .9996
      .6000      .9889
      .5000      .7706

Number of bootstrap samples:
  10000

Judges used in these computations:
 Columns   1 -  11
 Rater1   Rater2   Rater3   Rater4   Rater5   Rater6   Rater7   Rater8   Rater9   Rater10  Rater11
Columns 12 - 16
 Rater12 Rater13 Rater14 Rater15 Rater16

Examine output for SPSS errors and do not interpret if any are found

------ END MATRIX -----
```