**E****NHANCING CONVERSATIONAL AI MODEL PERFORMANCE AND EXPLAINABILITY FOR SINHALA-ENGLISH BILINGUAL SPEAKERS**

2022-056

Project Proposal Report

Hameed M.S.

(Dissanayake D.M.I.M., Jayasinghe D.T., Sakalasooriya S.A.H.A.)

B.Sc. (Hons) Degree in Information Technology Specialising in Software Engineering

Department of Information Technology

Sri Lanka Institute of Information Technology

Sri Lanka

January 2022

**ENHANCING CONVERSATIONAL AI MODEL PERFORMANCE AND EXPLAINABILITY FOR SINHALA-ENGLISH BILINGUAL SPEAKERS**

2022-056

Project Proposal Report

B.Sc. (Hons) Degree in Information Technology Specialising in Software Engineering

Department of Information Technology

Sri Lanka Institute of Information Technology

Sri Lanka

January 2022

# Declaration, Copyright Statement and The Statement of The Supervisors

We declare that this is our own work, and this proposal 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 our 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.

|  |  |  |
| --- | --- | --- |
| **Name** | **Student ID** | **Signature** |
| Dissanayake D.M.I.M. | IT19069432 |  |
| Hameed M.S. | IT19064932 |  |
| Jayasinghe D.T. | IT19075754 |  |
| Sakalasooriya S.A.H.A. | IT19051208 |  |

The above candidates are carrying out research for the undergraduate Dissertation under my supervision.

Name of the supervisor: Dr. Lakmini Abeywardhana

Signature of the supervisor: Date: 11/02/2022

Name of the co-supervisor: Ms. Dinuka Wijendra

Signature of the co-supervisor: Date: 11/02/2022

# Abstract

Analyzing and improving conversational AI dialogues by training the models is key to the successful implementation and maintenance of conversational AIs for better customer satisfaction. This is true even more so for low-resource languages since any newly added data to the dataset needs to be used to retrain the model so that it can handle future dialogues better. Yet, analyzing a model to identify overfitting or underfitting requires expertise, without this, it is not possible to identify the most suitable epoch to stop training at, to get the best model. This would require having to contact the conversational AI solution provider each time there is a need to improve the model, which can be a tedious task.

The solutions proposed include an efficient and code-less approach to improving model of the conversational AI assistant by eliminating the need to interact with the backend and implementing an efficient model testing and evaluation technique called “LCE: Learning Curve Explainer” with the zero-coding approach to allow non-machine-learning experts to easily evaluate the conversational AI assistant models performance by interacting with the UI which would help identify any overfitting or underfitting which will help the user choose the model that performs the best.

Keywords: Overfitting, Underfitting, Conversational AI, Codeless

# **Table of Contents**

[Declaration, Copyright Statement and The Statement of The Supervisors i](#_Toc95507872)

[Abstract ii](#_Toc95507873)

[Table of Contents iii](#_Toc95507874)

[List of Figures v](#_Toc95507875)

[List of Tables vi](#_Toc95507876)

[List of abbreviations vii](#_Toc95507877)

[1. Introduction 1](#_Toc95507878)

[1.1 Background & Literature survey 1](#_Toc95507879)

[1.2 Research Gap 3](#_Toc95507880)

[1.3 Research Problem 5](#_Toc95507881)

[2. Objectives 6](#_Toc95507882)

[2.1 Main Objectives 6](#_Toc95507883)

[2.2 Specific Objectives 6](#_Toc95507884)

[3. Methodology 7](#_Toc95507885)

[3.1 Requirements Gathering and Analysis 7](#_Toc95507886)

[3.1.1 Functional Requirements 7](#_Toc95507887)

[3.1.2 Non-functional Requirements 7](#_Toc95507888)

[3.2 Feasibility Study 8](#_Toc95507889)

[3.2.1 Technical feasibility 8](#_Toc95507890)

[3.2.2 Financial feasibility 8](#_Toc95507891)

[3.2.3 Legal feasibility 8](#_Toc95507892)

[3.2.4 Operational feasibility 9](#_Toc95507893)

[3.2.5 Scheduling feasibility 9](#_Toc95507894)

[3.3 Preparation of Datasets 9](#_Toc95507895)

[3.3.1 General dataset for machine learning model training 9](#_Toc95507896)

[3.3.2 Domain-specific dataset for conversational AI training 10](#_Toc95507897)

[3.4 Individual Component Architecture 10](#_Toc95507898)

[3.5 Codeless Model Improvement 12](#_Toc95507899)

[3.6 Model Performance Evaluation 13](#_Toc95507900)

[3.7 User Interfaces and Visualizations 13](#_Toc95507901)

[3.8 Tools and Technologies 15](#_Toc95507902)

[4. Description of Personal and Facilities 16](#_Toc95507903)

[5. Gantt Chart 18](#_Toc95507904)

[6. Commercialisation Plan 19](#_Toc95507905)

[7. Budget and Budget Justification 23](#_Toc95507906)

[References 24](#_Toc95507907)

[Appendices 26](#_Toc95507908)

[Appendix A: Survey Form 26](#_Toc95507909)

[Appendix B: Supervision Confirmation Emails 37](#_Toc95507910)

# List of Figures

[Figure 1.1: Summary of responses for whether people can identify whether a model is overfitting or underfitting by looking at the learning curve 3](#_Toc94454338)

[Figure 1.2: Summary of responses for whether people think that developers should be able to build conversational AIs without being machine learning experts 4](#_Toc94454339)

[Figure 3.1: High-level architectural diagram of the individual component consisting of the LCE and training data enhancing components 11](https://mysliit-my.sharepoint.com/personal/cdap_sliit_lk/Documents/2022%20REGULAR/01%20Topic%20Evaluation/Student%20Submissions/Proposal%20Draft/TMP-22-102/IT19064932.docx#_Toc95340413)

[Figure 3.2: High-level architectural diagram of the proposed solution with all research components integrated 12](#_Toc95340414)

[Figure 3.3: Low level UI diagram of the interface where the administrator will be able to go through user conversations 14](#_Toc95340415)

[Figure 3.4: Low level UI diagram of the interface where the administrator will be able to train and evaluate models 14](#_Toc95340416)

[Figure 5.1: Gantt chart for overall project and individual component 18](https://mysliit-my.sharepoint.com/personal/cdap_sliit_lk/Documents/2022%20REGULAR/01%20Topic%20Evaluation/Student%20Submissions/Proposal%20Draft/TMP-22-102/IT19064932.docx#_Toc94454418)

[Figure 6.1: proposed commercialisation plan 22](#_Toc95499466)

# List of Tables

[Table 1.1: Comparison with existing research/tools for data improvements 4](#_Toc94454703)

[Table 1.2: Comparison with existing research/tools for model evaluation 4](#_Toc94454704)

[Table 3.1: Summary of Tools and Technologies to be used according to the tasks 15](#_Toc94454765)

[Table 6.1: Feature variations and cost difference of packages proposed by the commercialisation plan 21](#_Toc95507707)

[Table 7.1: Budget justification for the overall research project 23](#_Toc95499518)

# List of abbreviations

Abbreviation Description

AI Artificial Intelligence

API Application Programming Interface

CaaS Conversational AI-as-a-Service

CDD Conversation-Driven Development

CLI Command Line Interface

CPU Central Processing Unit

FAQ Frequently Asked Questions

GCP Google Cloud Platform

GPU Graphics Processing Unit

LCE Learning Curve Explainer

LKR Sri Lankan Rupee

NLU Natural Language Understanding

RAM Random Access Memory

SaaS Software-as-a-Service

SLIIT Sri Lanka Institute of Information Technology

TPU Tensor Processing Unit

UI User Interface

YAML Yet Another Markup Language

# Introduction

## Background & Literature survey

Conversational AIs use machine learning models that are trained on human conversation data which provides them with the capability to handle complex queries that are handed to them and has the ability to have conversations that are much more natural [1]. Training these models requires expertise. Generally, it requires having to work with the backend, knowing the CLI commands required to split the data into training and testing and then carrying out the model training. Just training alone does not suffice, it is also vital that the right model that performs the best is chosen, and this requires experimenting with the hyperparameters and then looking for any overfittings or underfittings in the learning curve which also require expertise [2].

One of the major issues in supervised machine learning is that models do not generalize from seen data to unseen data, which is called overfitting [3]. Due to overfitting, the model performs well on training data but performs poorly on testing data. This is due to the over-fitted model having difficulty in coping with the pieces of information in the testing data set, which may be different from the training data set. Also, over-fitted models are inclined to memorize all the data, including the unavoidable noise in the training data set, rather than learning the discipline behind the data set [4].

Generally, the cause for overfitting is categorized into three kinds [4]. The first is, noise learning on the training data set, this occurs either when the training data set is too small, has fewer representative data or else has too many noises. This creates a higher chance of the noises being learnt, which then acts as a basis for predictions [5]. The second category is hypothesis complexity, which refers to the trade-off in complexity, which is a trade-off between variance and bias. It refers to the equilibrium between accuracy and consistency. When the algorithm has too many inputs, the model becomes accurate on average with lower consistency [5]. This means the model can be drastically different on different data sets. The third category is multiple comparisons procedures that are ubiquitous in induction algorithms, and in other AI algorithms [6]. During these processes, multiple items are compared based on the scores from an evaluation function and the item with the maximum score is selected. Nevertheless, this process is likely to choose items that will not improve, or even reduce the classification accuracy.

Underfitting is another issue that is faced when training a model, it occurs when the model is not able to accurately capture the relationship between the input and output variables, which produces a high error rate on both the training data set and the testing data set [7]. This occurs when a model is too simple, which can be due to the model requiring more training time, less regularization, or more input features [7]. Similar to overfitting, when underfitting occurs, a model is unable to establish the dominant trend within the data, which results in training errors and the model performs poorly. Low variance and high bias are good indicators of underfitting. Due to this behaviour being visible while using the training data set, it is easier to identify underfitted models over overfitted models.

Most users who are non-machine learning experts are unable to identify when a model either overfits or underfits, by looking at the learning curve. As shown in Figure 1.1, which depicts the data retrieved by the survey conducted, it is evident that the majority are unable to identify when a model overfits or underfits by looking at the learning curve. Most of these users are also unable to use methods like callbacks, early stopping and checkpointing to help stop the model training at the right epoch which is not too high to cause overfitting or else too low to cause underfitting.

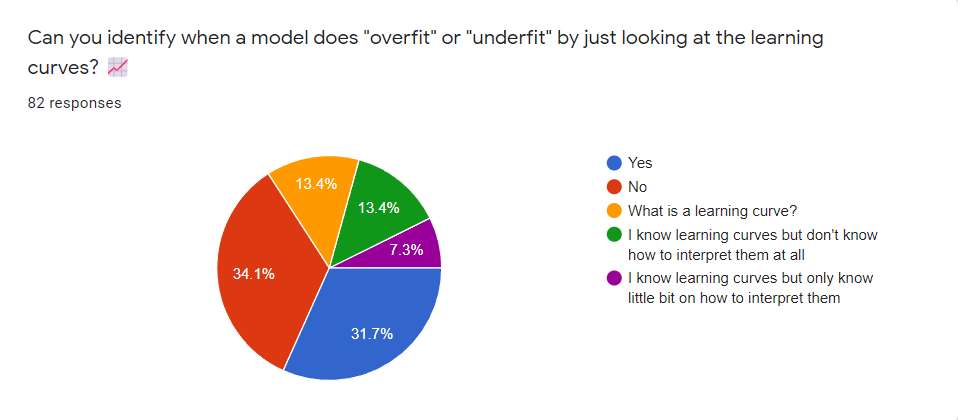


Figure 1.1: Summary of responses for whether people can identify whether a model is overfitting or underfitting by looking at the learning curve

## Research Gap

Currently analyzing a model for overfitting or underfitting requires expertise due to the steps that need to generally be followed [8]. Them being, splitting the data set into two, as training data and testing data, or else using a cross-validation scheme, where the data is repeatedly split into training and testing subsets [9]. The model is then trained and a function such as the learning\_curve() function from scikit-learn can be used to plot the learning curve which will show two curves one for the training data and the other for the testing data, which will be plotted showing the validation loss for each epoch [10]. For a non-machine learning expert these curves would make no sense, it requires a level of expertise to look at the curve and deduce whether the model is overfitting or underfitting, and what the right number of epochs would be to stop training at. As shown in Figure 1.2, which depicts the data retrieved by the survey conducted, it is evident that the majority agree that developers should be able to build AI-based chatbots without having to be machine learning experts which require training models and evaluating them.

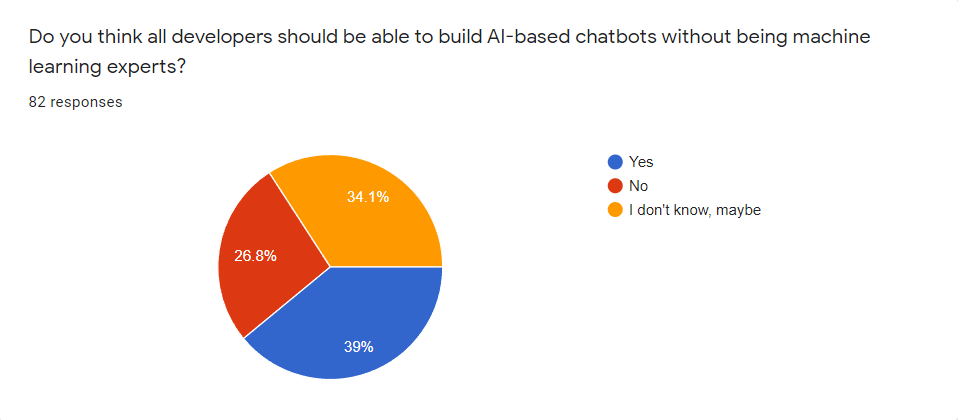


Figure 1.2: Summary of responses for whether people think that developers should be able to build conversational AIs without being machine learning experts

Few research papers and tools that exist for data improvement and model evaluation have been summarized in Table 1.1 and Table 1.2.

Table 1.1: Comparison with existing research/tools for data improvements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name/ Reference** | **Tool/ Research** | **Requires knowledge about the backend** | **Conversation Driven Development** | **Requires knowing CLI commands** |
| Rasa [11] | Tool | Yes | No | Yes |
| Rasa X [12] | Tool | Yes | Yes | No |
| This Research component | Tool | No | Yes | No |

Table 1.2: Comparison with existing research/tools for model evaluation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name/ Reference** | **Tool/ Research** | **Overfitting/ Underfitting Indicators** | **Suggest the best epoch range to train the model** | **Requires knowledge about the backend to understand the feedback** |
| Rasa [11] | Tool | No | Yes (Early stopping) but no visual indication | Yes |
| This Research component (LCE) | Tool | Yes | Yes | No |

The proposed component is a UI that will enable non-machine learning experts to improve models used in conversational AIs by adding new data and training the models with the click of a button which will also denote whether a model is overfitting or underfitting and suggest the best epoch to stop training at, making the whole process of training models a lot easier than what is currently being done.

## Research Problem

At present most businesses have integrated conversational AIs to their websites due to its convenience to customers, which allows the customers to receive the help they need faster, than having to browse the website or else contact the business to retrieve the necessary information. According to a survey done, 64% of people say they would rather message a business than call it [3]. The main reason people prefer messaging over calling is due to the number of times the calls generally do not go through which can be frustrating to a customer [13].

Integrating a conversational AI into a business's website is not a one-time thing, it requires constant maintenance and improving the conversational AI to cater to the customers without issues [14]. It can become a tedious task having to constantly contact the conversational AI service provider every time there is a need to improve the conversational AI. Hence, with the proposed component it will be possible to maintain, monitor and improve the conversational AI without requiring any machine learning expertise.

# Objectives

## Main Objectives

The main objective of implementing this component is to develop an efficient and code-less approach to improve the training data of the conversational AI assistant by eliminating the need to interact with the backend and implementing an efficient model testing/evaluating technique with the zero-coding approach to allow non-machine learning experts to improve the conversational AI assistant models.

## Specific Objectives

To reach the main objective, the specific objectives that need to be achieved are as follows:

1. Finding the best possible approach to allow training data improvements to be made without any coding knowledge or manually interacting with the backend.
2. Providing a way to efficiently re-train and deploy new machine learning models, of the conversational AI assistant for non-technical users.
3. Evaluating machine learning models and designing an algorithm to automatically identify any overfitting or underfitting scenarios in evaluation reports generated by RASA and indicating them in the front-end.

# Methodology

## Requirements Gathering and Analysis

The requirement gathering and analysis phase focuses on studying how learning curves are interpreted by machine learning experts to identify overfittings and underfittings and looking into frameworks like Rasa X which use Conversation-Driven Development (CDD) to improve the conversational AI assistant [12].

The requirements of the proposed solution are as follows,

1. Should be able to plot the learning curve and identify any overfitting and underfitting in the curve and notify the user regarding it in the UI.
2. Should be able to suggest the best range of epochs to use to train the model, avoiding any overfitting or underfitting.
3. Should be able to manually type in questions and answers and then train the model with them using the UI.
4. Should be able to carry out CDD, where questions asked by users can be added to the dataset along with the relevant answer and then retrain the model

### Functional Requirements

The functional requirements of this research component are as follows.

1. LCE should denote any overfittings or underfittings in the trained model.
2. LCE should suggest the appropriate range of epochs to use to train the model.
3. Should be able to use conversations users have had with the conversational AI, to improve the machine learning model (CDD).

### Non-functional Requirements

The non-functional requirements of this research component are as follows.

1. Model evaluations done by LCE should be reliable.
2. Model evaluations done by LCE should be consistent.
3. LCE should not hinder the model training time.

## Feasibility Study

A feasibility study was conducted to determine the research component's technical, financial, legal, operational, and scheduling feasibility.

### Technical feasibility

The research component necessitates a thorough understanding of data science, Rasa machine learning models, Rasa framework, and software development. To train Rasa machine learning models, there need to be adequate computing resources. Computing resources such as TPUs, GPUs, or High CPU/RAM instances are not required to train machine learning models in Rasa, though they are preferable. The Rasa framework may require a significant amount of disk space to cache model files; therefore, hard drive space needs to be a top priority. For the overall deployment of the final product, there should be a properly configured cloud infrastructure.

### Financial feasibility

Cloud infrastructure for training and deploying machine learning models should not be prohibitively expensive. Minor fees for purchasing a domain name and cloud infrastructure resource usage can be applied, yet these are acceptable because the final product has a market value that covers the costs.

### Legal feasibility

The datasets utilized in the research components need to be publicly accessible. Scraped data from private websites should have the permission of the original owners, as well as written consent. Without the written approval of the original authors, the research component should not extract content from previous work. Any of the used Python packages or software must either be legally purchased or open-source, with credit given to the original authors if required.

### Operational feasibility

This research component should not conflict with other research components in the same or other research projects. In contrast to previous work, the research component needs to address the gap in data improvements and model performance evaluation and maintain novelty. The research component must be feasible to meet the scope of the research project requirements.

### Scheduling feasibility

The research component's tasks should cohere to the Gantt chart in section 5 of this proposal report as well as the pre-determined research project milestones.

## Preparation of Datasets

Two datasets have been prepared, one being a general dataset, that will be used to train the machine learning model. The other is a domain-specific dataset, that will be used to train the conversational AI. The difference between the two datasets will be explained in subsections 3.3.1 and 3.3.2.

### General dataset for machine learning model training

The general dataset that will be used to train the machine learning model consists of articles related to SLIIT (Sri Lanka Institute of Information Technology), which have been scraped from publicly available articles and the SLIIT website. All the information that is in English will be converted to Sinhala using Google translate, which will then be rewritten in informal, spoken Sinhalese, which will also contain English words where necessary, by the four members of the research team. This is a data augmentation approach to increase the volume of the available data to overcome the low resource nature of the gathered Sinhala text. Using informal, spoken Sinhalese to train the model will help the conversational AI perform better since most users use informal, spoken Sinhalese when messaging.

### Domain-specific dataset for conversational AI training

The domain-specific dataset will contain FAQs and answers that will be written in Sinhala-English code-switched textual data, that will be used to train the conversational AI. The data will be written in different permutations and spellings to cover all the possible ways of asking the question. And the corresponding answers to these questions will be written as well. The data will contain more than 75 intents (categories) under which the questions will fall, and each intent will contain a minimum of 10 questions.

## Individual Component Architecture

Figure 3.1 illustrates the individual research component and Figure 3.2 illustrates the overall system architecture after integrating all the research components. The proposed solution in this research paper will make use of the component named “Tracker store database to store conversations” which can be seen in Figure 3.2, which as the name suggest will save the conversations that end users have with the conversational AI. This will give the administrator the option to read through the conversations to get a better idea regarding the requirements of the end-users. Also, the questions asked from the conversational AI to which the conversational AI is unable to answer will be displayed in a different area where the administrator will be able to review them and decide on whether to ignore a particular question or else to assign the question to an existing intent (category), else create a new intent to assign it to along with the answer to the question, which will then be used to retrain the model under the component named as “Training Data Enhancing Module”, and the component named as “Learning Curve Explainer (LCE)” would also analyse for any overfittings or underfittings in the trained model and suggest the best epoch range to use which will improve the model. Both these components can be seen in Figure 3.1.

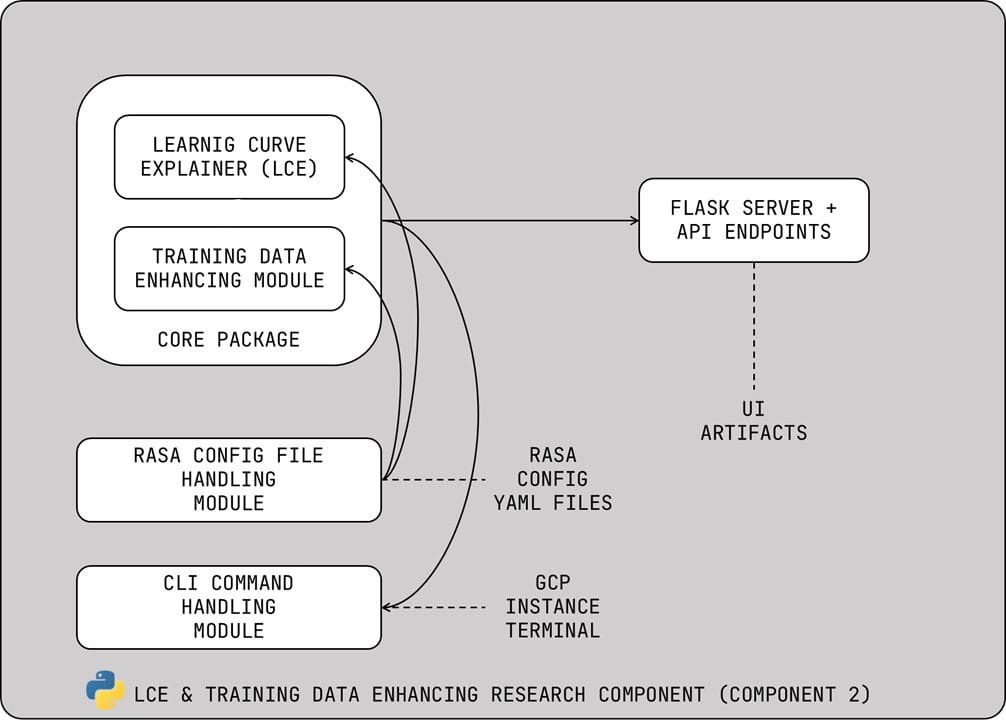


Figure 3.1: High-level architectural diagram of the individual component consisting of the LCE and training data enhancing components

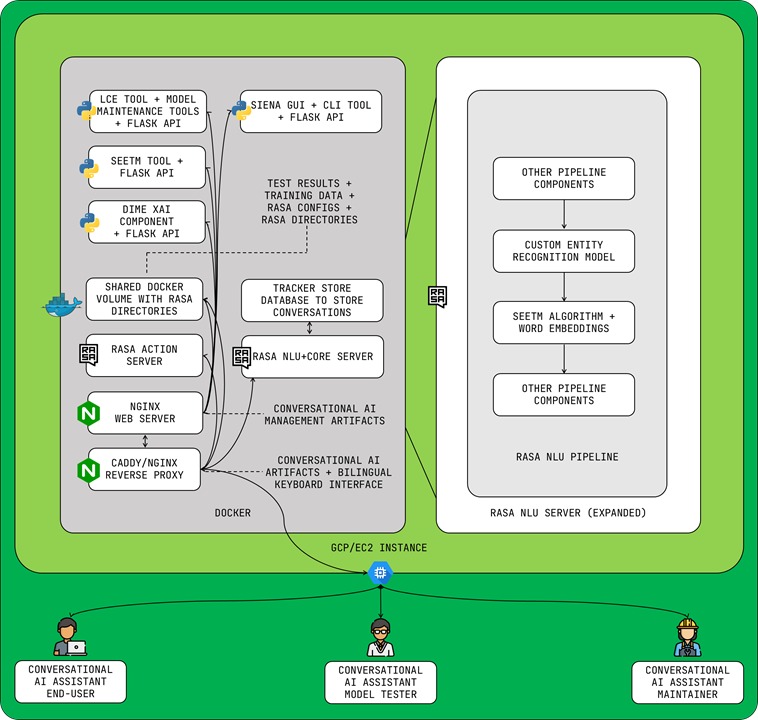


Figure 3.2: High-level architectural diagram of the proposed solution with all research components integrated

## Codeless Model Improvement

Training models require machine learning expertise. Currently using a tool like Rasa requires knowing the Rasa framework well. Training a model using Rasa requires knowing the backend well and knowing the relevant CLI commands required to train a model. With the proposed solution it will be possible to train models without requiring any knowledge regarding the Rasa backend or any CLI commands. It will be possible to train the model with just the click of a button, all the necessary commands will be invoked using bash command calls to the backend.

## Model Performance Evaluation

Training a model with too many epochs may lead to overfitting, while too few can lead to underfitting. Methods like early stopping can be used to overcome this, where a large number of epochs can be given and the model will stop training when the model’s performance stops improving against a hold out testing dataset. Overfitting can also be overcome using methods like model checkpoint callbacks. These can be done using an API like Keras [15], yet they require having to work with the backend and having considerable knowledge about the Keras API. With the proposed solution the user does not require any knowledge since the learning curve will be plotted and any overfittings and underfittings will be denoted along with the recommended number of epochs to use to train the model.

## User Interfaces and Visualizations

The proposed solutions will mainly contain two UIs. Figure 3.3 illustrates a rough sketch of one of the UIs that will be implemented, which is where the administrator will be able to read conversations end-users have had with the conversational AI. The questions that the conversational AI was not able to answer will be highlighted which the administrator can add to the dataset by choosing the relevant intent which the question falls under from the existing list, or else creating a new one and providing the relevant answer to the question.

Figure 3.4 illustrates a rough sketch of the UI that will be implemented in which the administrator will be able to configure and train models using the data newly added using the UI in Figure 3.3 to which the learning curve will be plotted indicating any overfitting or underfitting in the model and suggesting the best range of epochs to use to train the model.

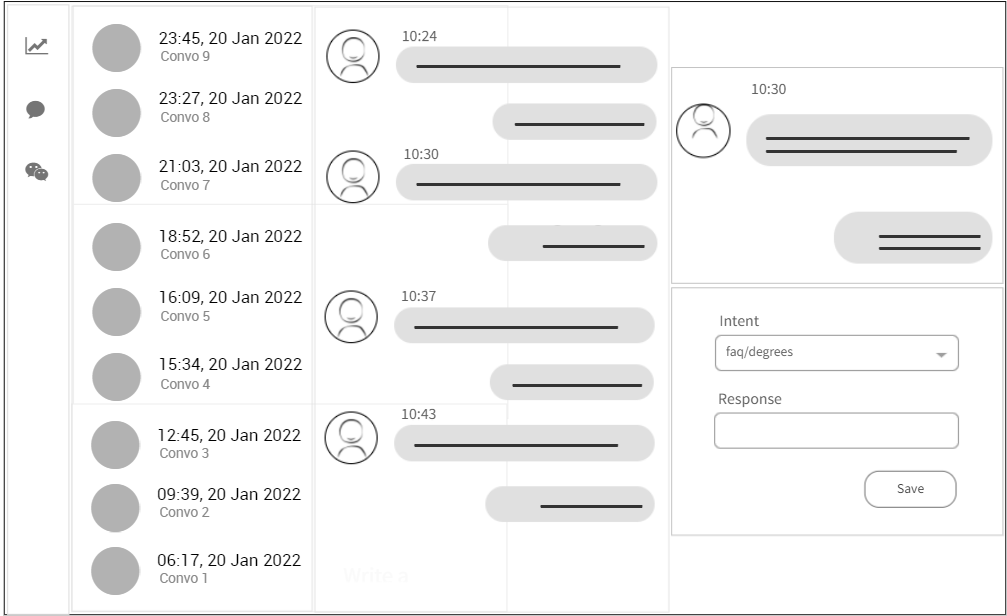


Figure 3.3: Low level UI diagram of the interface where the administrator will be able to go through user conversations

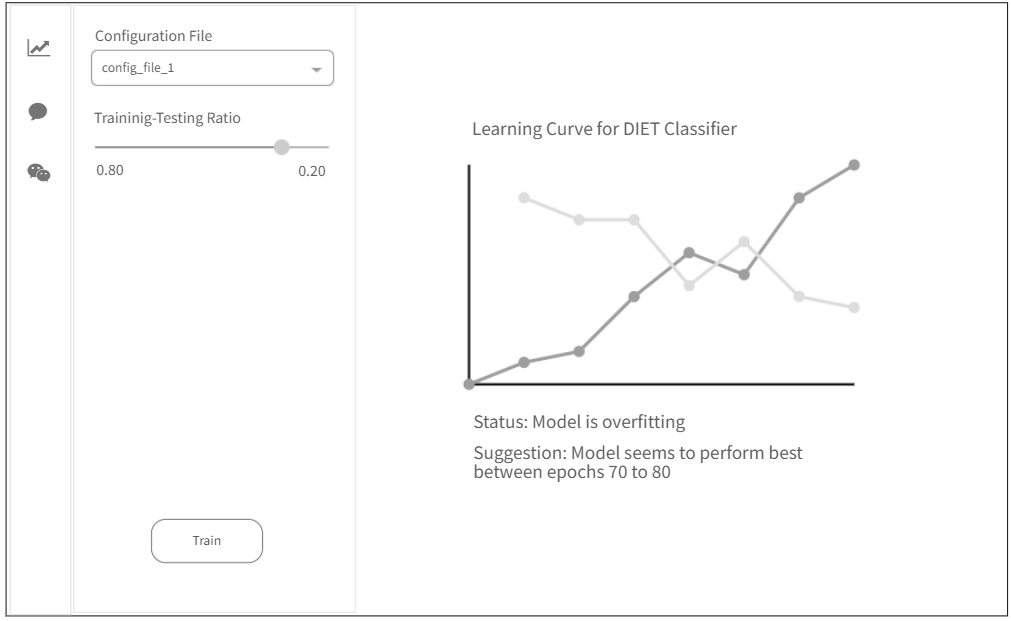


Figure 3.4: Low level UI diagram of the interface where the administrator will be able to train and evaluate models

## Tools and Technologies

The list of tools and technologies used in this research component and the overall research has been stated in Table 3.1.

Table 3.1: Summary of Tools and Technologies to be used according to the tasks

|  |  |
| --- | --- |
| **Task** | **Tools to be used** |
| Algorithm Implementation and tool development | Python 3.8, NumPy, Pandas, sklearn, Rasa 2.8.12, PyCharm, Visual Studio Code, Google CoLab |
| Server development as a standalone frontend for the UI | Flask, JavaScript, Bootstrap, React JS (optional) |
| NoSQL Database development (for the overall research solution) | MongoDB Atlas, MongoDB Compass |
| Cloud Infrastructure management (for the overall research solution) | One out of Google Cloud Platform or Amazon Web Services |
| Conversational AI development by Integrating all research components (Backend) | Rasa 2.8.12, MongoDB Atlas, Gensim, spaCy, Docker |
| Conversational AI frontend development | React JS, Bootstrap, socket.io, JavaScript |
| Overall system deployment | Caddy 2, NGINX, Git, Docker, docker-compose |

# Description of Personal and Facilities

1. Data preparation
   1. Data collection
   2. Data augmentation
   3. Data deduplication
   4. Data preprocessing
2. Building Conversational AI
   1. Add training data
   2. Build NLU pipeline
   3. Decide the number of validation data points
   4. Enable TensorBoard
   5. Train a model
   6. Test NLU model
3. Algorithm development
   1. Read TensorBoard results
   2. Develop Learning Curve Explainer (LCE) algorithm to interpret TensorBoard results
   3. Build the LCE package
   4. Build a locally executable server
4. Implementation
   1. Integrate LCE with Rasa
   2. Build the frontend
   3. Implement codeless maintenance
   4. Build Docker and Docker Compose files
   5. Build Caddy and NGINX servers
   6. Integrate component UI with main UI
5. Deployment
   1. Update the Git repository
   2. Set up GCP instance
   3. Deploy Docker containers
   4. Test LCE and codeless maintenance after deployment

# Gantt Chart

Figure 5.1: Gantt chart for overall project and individual component

# Commercialisation Plan

Each research component of the overall research project provides applications for various Natural Language Processing and Machine Learning model evaluation-related tasks. Although the outputs of each research component can help design many products and services, they were all integrated to build a conversational AI that fully supports Sinhala-English code-switching. The main reason for developing a conversational AI are as follows.

1. Conversational AIs are a trending topic, and there is an increasing demand for Sinhala-based conversational AIs. Many businesses are looking to increase their customer reach by using conversational AIs for a vast range of business tasks, including providing technical assistance, automating manual tasks such as booking tickets, and providing the information requested by the customers. Although that is the case, it is hard to find Sinhala-based conversational AIs, especially in Sri Lanka. Thus, developing Sinhala-based conversational AIs was identified as a potential business opportunity.
2. Although there are many conversational AI development frameworks, most of them lack evaluation tools to debug machine learning models. In frameworks like Rasa, model evaluations are highly technical, and the average developers without machine learning knowledge fail to identify problems and debug the machine learning models. Solving this issue by allowing non-technical users to maintain and evaluate machine learning models and conversational AIs is another business opportunity where evaluation tools can be built and released as add-on features.
3. Businesses are moving towards cloud-based solutions, especially SaaS products from traditional standalone applications, web applications, and on-premises solutions, where the maintenance cost is high. A cloud-based highly configurable conversational AI would be an appropriate solution for many businesses where the effort for maintenance is considerably low.

The end product of the overall research concentrates on designing a solution for the above potential business ideas and opportunities. The conversational AI developed as the research end-product is mainly a SaaS or simply a CaaS (conversational AI-as-a-service) product that a business can purchase with a set of add-on features, as illustrated in Figure 6.1. In addition to the CaaS packages, on-premises and demo cloud-based conversational AI packages are available for affordability and convenience. The end product of the overall research has the following archivable user benefits.

1. Businesses can purchase a CaaS package and eliminate conversational AI maintenance efforts.
2. Developers can use code-less maintenance tools to maintain conversational AIs they purchase without having in-depth knowledge about the backend deployment and the conversational AI development framework.
3. Businesses can purchase evaluation tools as add-ons and generate model explanations and evaluate machine learning models themselves to avoid extra maintenance costs. Here, the generated evaluations are easy to understand by non-technical users, which is not a feature of any existing chatbot framework.
4. Users of conversational AI can easily use the Sinhala-English code-switchable keyboard interface, and businesses can attract more users from having this feature as it eliminates the need to use third-party Sinhala typing services.
5. Businesses have the freedom to purchase either the CaaS packages or on-premises packages as per their need, while anyone can test the demo conversational AI for a period of 1 month before deciding to purchase any of the other packages that have a cost assigned.
6. Businesses can reach a vast customer range through Sinhala-English code-switching-based conversational AIs, especially within the Sri Lankan market, and it is possible to eliminate the need for having a dedicated customer care staff. It will dramatically lower the expenses of the business.

The proposed commercialisation plan of the end product of the overall research component contains a set of convenient packages and the offered features of each package differ based on the cost attached to it. The distribution of the features and the package cost were carefully planned and designed by analyzing the existing purchasable packages of similar SaaS products and conversational AIs. Table 6.1 clearly illustrates the feature variation and the cost difference of the packages offered by the proposed commercialisation plan.

Table 6.1: Feature variations and cost difference of packages proposed by the commercialisation plan

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Feature** | **Packages** | | | | |
| **Demo** | **On-Prem** | **CaaS** | | |
| **Starter** | **Pro** | **Genius** |
| Intents | 10 | Unlimited | 20 | 180 | 400 |
| API Integrations | 2 | Unlimited | 2 | 110 | 200 |
| Bot Analytics | Icon  Description automatically generated | Icon  Description automatically generated | - | Icon  Description automatically generated | Icon  Description automatically generated |
| CDD | Icon  Description automatically generated | Icon  Description automatically generated | Icon  Description automatically generated | Icon  Description automatically generated | Icon  Description automatically generated |
| Sinhala Entity Annotating | Icon  Description automatically generated | - | - | Icon  Description automatically generated | Icon  Description automatically generated |
| ML Evaluation Tools | - | - | - | - | Icon  Description automatically generated |
| Maintenance Fee | - | 2 Free + $9.99 per additional call | - | - | - |
| Trial Duration | 1 Month | - | - | - | - |
| Package Price | Free | $199.99 | $9.99 | $34.99 | $49.99 |

A picture containing text, outdoor

Description automatically generated

Figure 6.1: proposed commercialisation plan

# Budget and Budget Justification

The budget justification for all four research components has been stated in Table 6.1.

Table 7.1: Budget justification for the overall research project

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Component Name** | **Individual Item Price (LKR)** | **Number of Items** | **Duration** | **Total Item Price (LKR)** |
| Domain Name | 2148.43/year | 1 | 1 year | 2148.43 |
| GCP Instance | 10683.84/month | 1 | 6 months | 64103.06 |
| Reference Book: Basaka Mahima by J.B. Dissanayake | 1250.00 | 1 | - | 1250.00 |
| Research Paper Publication | 25000.00 | 1 | - | 25000.00 |
| Grand Total | - | - | - | 92,501.49 |

# References

1. G. Caldarini, S. Jaf, K. McGarry, “A Literature Survey of Recent Advances in Chatbots,” 2022. [Online]. Available: <https://arxiv.org/abs/2201.06657>
2. Z. Luvsandorj, “Introduction to NLP - part 4: Supervised text classification model in python,” *Medium*, 19-Dec-2021. [Online]. Available: https://towardsdatascience.com/introduction-to-nlp-part-4-supervised-text-classification-model-in-python-96e9709b4267. [Accessed: 06-Feb-2022].
3. “Conversational AI: What it is, why you should care and how to do it right,” *Social Media Marketing & Management Dashboard*, 15-Dec-2021. [Online]. Available: <https://blog.hootsuite.com/conversational-ai/>. [Accessed: 28-Jan-2022].
4. X. Ying, “An overview of overfitting and its solutions”, in *Journal of Physics: Conference Series*, 2019, vol 1168, bl 022022.
5. G. Paris, D. Robilliard, en C. Fonlupt, “Exploring overfitting in genetic programming”, in *Lecture Notes in Computer Science*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, bll 267–277.
6. D. D. Jensen en P. R. Cohen, “Multiple comparisons in induction algorithms”, *Machine Learning*, vol 38, no 3, bll 309–338, 2000.
7. By: IBM Cloud Education, “What is underfitting?,” *IBM*. [Online]. Available: <https://www.ibm.com/cloud/learn/underfitting>. [Accessed: 28-Jan-2022].
8. M. A. Babyak, “What you see may not be what you get: a brief, nontechnical introduction to overfitting in regression-type models”, *Psychosomatic medicine*, vol 66, no 3, bll 411–421, 2004.
9. G. Varoquaux, L. Buitinck, G. Louppe, O. Grisel, F. Pedregosa, en A. Mueller, “Scikit-learn: Machine learning without learning the machinery”, *GetMobile: Mobile Computing and Communications*, vol 19, no 1, bll 29–33, 2015.
10. F. Pedregosa *et al.*, “Scikit-learn: Machine learning in Python. the Journal of machine Learning research 12 ”, 2011.
11. T. Bocklisch, J. Faulkner, N. Pawlowski, en A. Nichol, “Rasa: Open source language understanding and dialogue management”, *arXiv preprint arXiv:1712. 05181*, 2017.
12. “Introduction to rasa X,” *Open source conversational AI*, 10-Dec-2021. [Online]. Available: <https://rasa.com/docs/rasa-x/>. [Accessed: 22-Jan-2022].
13. J. Lee, “Do virtual reference librarians dream of digital reference questions?: A qualitative and quantitative analysis of email and chat reference”, *Australian Academic & Research Libraries*, vol 35, no 2, bll 95–110, 2004.
14. M. A. Nezami en R. Rukham, “Crowdsourced NLP Retraining Engine in Chatbots”, in *International Conference on Emerging Technologies and Intelligent Systems*, 2021, bll 311–320.
15. K. Team, “Simple. flexible. powerful.,” *Keras*. [Online]. Available: https://keras.io/. [Accessed: 08-Feb-2022].

# Appendices

## Appendix A: Survey Form

Figure A.1: Complete survey form questions and responses – part 1

Graphical user interface, application

Description automatically generated

Chart, pie chart

Description automatically generated

Figure A.2: Complete survey form questions and responses – part 2

Chart, pie chart

Description automatically generated

Figure A.3: Complete survey form questions and responses – part 3

Chart, application, pie chart

Description automatically generated

Figure A.4: Complete survey form questions and responses – part 4

Chart, pie chart

Description automatically generated

Figure A. 5: Complete survey form questions and responses – part 5

Chart, pie chart

Description automatically generated

Figure A.6: Complete survey form questions and responses – part 6

Chart, pie chart

Description automatically generated

Figure A.7: Complete survey form questions and responses – part 7

Graphical user interface, chart, application, pie chart

Description automatically generated

Figure A.8: Complete survey form questions and responses – part 8

Chart, application, pie chart

Description automatically generated

Figure A.9: Complete survey form questions and responses – part 9

Chart, pie chart

Description automatically generated

Figure A.10: Complete survey form questions and responses – part 10

Graphical user interface, application

Description automatically generated

Figure A.11: Complete survey form questions and responses – part 11

## Appendix B: Supervision Confirmation Emails

Graphical user interface, text, application, email

Description automatically generated

Figure B.1: Research project supervision confirmation email

Graphical user interface, text, application, email

Description automatically generated

Figure B. 2: Research project co-supervision confirmation email