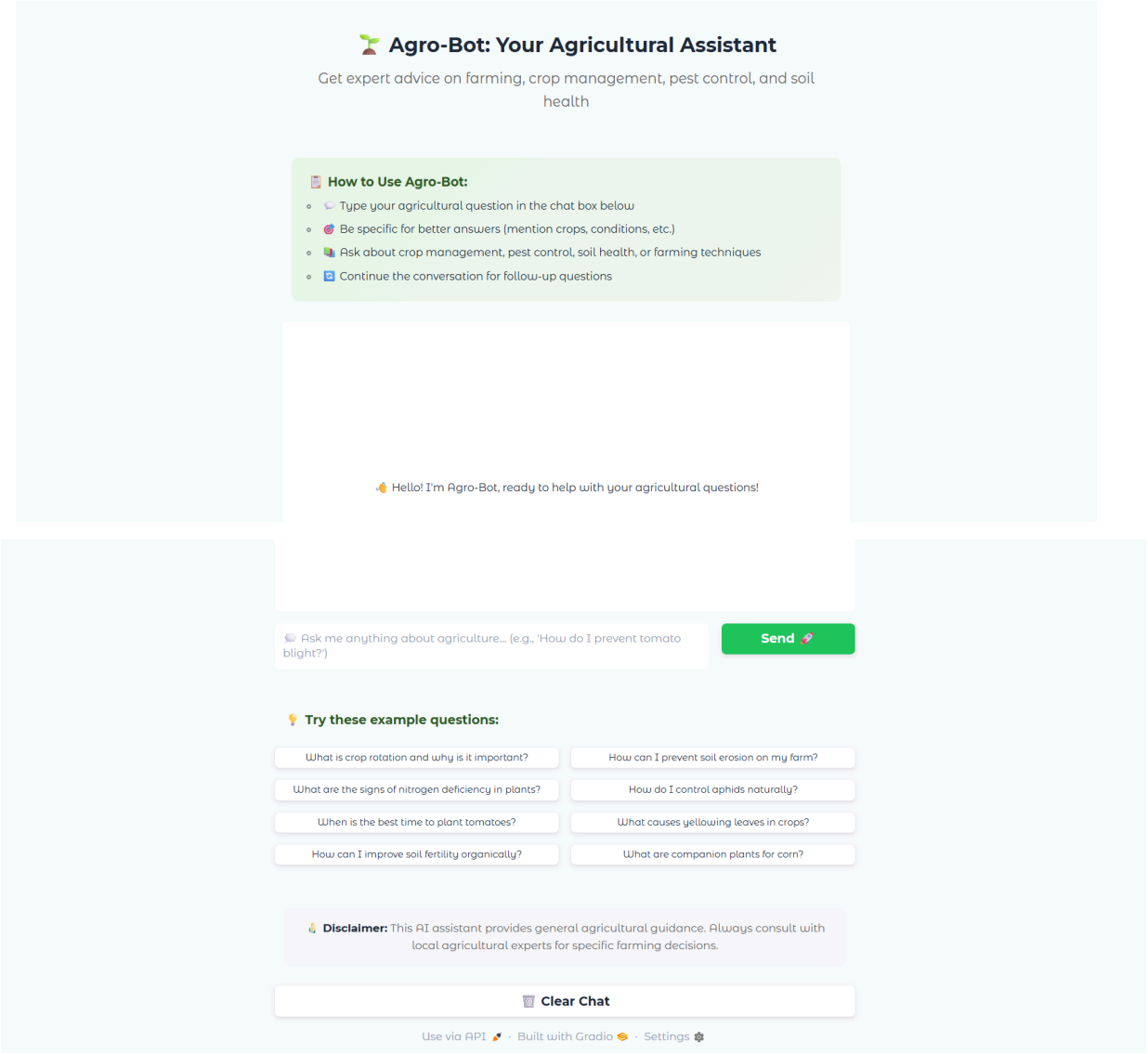
Names: Josiane Ishimwe

Date: 22-06-2025

Agricultural Chatbot using Generative QA



A screenshot of the Gradio interface

Executive Summary

This project develops an AI-powered chatbot for agriculture to assist farmers with queries on crop diseases, pest control, weather, and best practices.

Leveraging the Hugging Face T5-small model and the KisanVaani dataset, the chatbot is trained to provide users with meaningful, domain-relevant answers. The model underwent extensive preprocessing, hyperparameter tuning, qualitative testing, evaluation, UI, deployment and future improvement plans.

1. Introduction

1.1 Problem Statement

Farmers often lack immediate access to expert advice on agricultural matters such as pest control, crop diseases, weather, and sustainable farming practices. This gap contributes to poor decision-making and decreased productivity.

1.2 Project Objectives

Provide accessible agricultural knowledge to farmers, especially in rural areas, to support sustainable farming and improve productivity.

- Develop a conversational AI tailored to agriculture.
- Fine-tune a generative QA model using domain-specific data.
- Provide an accessible, interactive interface for query resolution.

1.3 Theoretical Background

I've chosen T5 over BERT for its generative capabilities, ideal for open-ended QA tasks. This project employs **Transformer models**(T5-small)[1], introduced in the paper "Attention Is All You Need," for this agricultural chatbot. Unlike traditional RNNs or CNNs, Transformers utilise an attention mechanism to evaluate word importance, facilitating efficient parallel processing and capturing long-range text dependencies.

The encoder processes input questions through multi-head self-attention and feed-forward networks, creating informed word representations.

The decoder generates answers, using masked attention to focus on preceding output and encoder input for context.

Fine-tuning the pre-trained T5-small model on the KisanVaani agricultural QA dataset adapts its general language capabilities to domain-specific terminology, thereby improving response accuracy and relevance efficiently compared to training from scratch.

2. Dataset Description

The dataset used is English Question-Answer pairs on agricultural topics called the KisanVaani Dataset from the Hugging Face Dataset Hub

2.1 Link:

https://huggingface.co/datasets/KisanVaani/agriculture-qa-english-only

2.2 Data Inspection

- **Total Records:** 22615
- **Structure:** Two columns ("question" and "answers")
- **Observations**: The Dataset contained domain-relevant, high-quality content with minimal noise.

3. Exploratory Data Analysis (EDA)

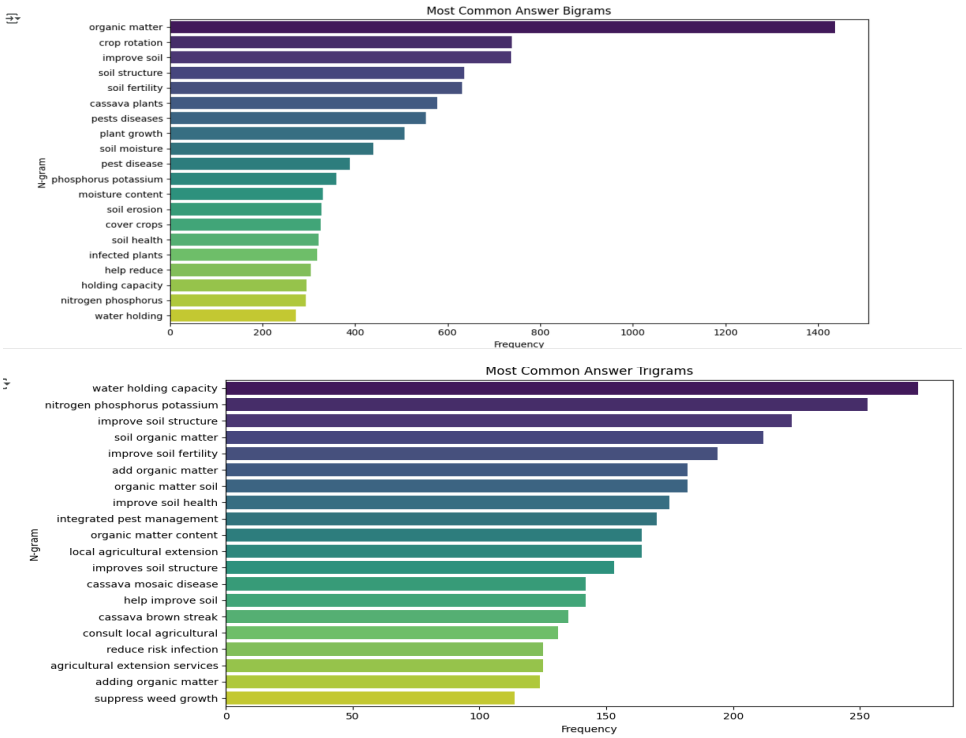
Objective: Understand the dataset's structure, content, and diversity to confirm its suitability for the chatbot.

Task:

- Inspect dataset size, columns, and data types.
- Analyse question and answer lengths.
- Extract keywords to verify topic coverage.
- Check n-grams from a text series

Findings:

- **Dataset Size**: 22,615 question-answer pairs, sufficient for fine-tuning.
- **Columns**: 'question' and 'answers', both as strings.
- **Length Analysis**: Questions are typically short (5-15 words), while answers vary (10-50 words), indicating diverse response complexity.
- **Keyword Extraction**: Common terms include 'crop', 'soil', confirming coverage of key agricultural topics.
- **Duplicates**: Number of duplicate QA pairs is 20284
- **Common Phrases**: The n-gram analysis highlighted common phrases like "signs of", "control of", and specific crop names combined with terms like "disease" or "management", further illustrating the typical structure and content of the questions and answers.



Most Common Question and Answer Bigrams Visualization

4. Data Preprocessing

Objective: Clean and prepare the dataset for model training. In the preprocessing step, the following choices were made:

4.1 Text Cleaning and Normalisation

Duplicate Handling: Duplicates were dropped based on both the 'question' and 'answers' columns. This was done to ensure that the model is not overexposed to identical question-answer pairs, which could lead to overfitting and reduce its ability to generalise to new, unseen questions. After removing 20,284 duplicates, the effective dataset size was ~2,331 unique pairs, potentially limiting diversity.

No missing values or significant outliers were found in the dataset.

Text Normalisation, The normalize_text function converts all text to lowercase and removes extra whitespace. It also uses a regular expression r'[^\w\s,.!?]' to remove characters that are not alphanumeric, whitespace, commas, periods, exclamation marks, or question marks. This was done to reduce the vocabulary size by treating words with different capitalisation as the same and to remove noise and irrelevant characters.

Keep essential punctuation (commas, periods, question marks, exclamation marks) as they can be important for sentence structure and meaning, which is relevant for a question-answering task.

4.2 Feature Engineering Approaches

Tokenisation with T5Tokenizer: The T5Tokenizer was used because the model being fine-tuned is a T5 model. Using the corresponding tokeniser ensures that the text is converted into a format (token IDs) that the model was pre-trained on and expects as input.

The max_length and truncation=True parameters were set to handle sequences of varying lengths and ensure they fit within the model's input capacity.

A max_length of 128 was chosen based on the EDA, where most answers were under 50 words.

Padding was applied to make all sequences the same length, which is required for batch processing during training.

Train-Validation-Test Split, The data was split into 80% for training and 20% for testing, and then the training data was further split into 80% for final training and 20% for validation.

5. Select and Load Model

Objective: Load the T5-small model and inspect its architecture.

Tasks:

- Load T5-small with TFAutoModelForSeq2SeqLM.
- Display model configuration.

I chose T5 for its efficiency and strong performance in generative QA tasks

6. Model Compilation

Objective: Configure the T5-small model for fine-tuning with an appropriate optimiser, loss function, and metrics.

I compile the model with:

- Adam optimiser, I used Adam due to compatibility
- SparseCategoricalCrossentropy loss, SparseCategoricalCrossentropy suits sequence-to-sequence tasks.
- The accuracy metric, accuracy tracks the token prediction

Optimizer

Initially, I chose AdamW from Transformers. optimisation. AdamW is a popular optimiser that incorporates weight decay, which can help prevent overfitting. It was created using create_optimizer from transformers with a learning rate schedule, which is the recommended way to optimise Hugging Face models in TensorFlow[2]. Due to compatibility issues encountered later, I switched to the tf.keras.optimisers optimiser.Adam as a workaround. While Adam is a standard and effective optimiser, AdamW is generally preferred for fine-tuning transformer models.

Loss Function: tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True) was used as the loss function.

SparseCategoricalCrossentropy is appropriate for sequence-to-sequence tasks like text generation, where the target labels (token IDs) are integers.

Metrics: Accuracy was included as a metric to monitor during training. While token accuracy is a simple metric and not a perfect measure of text generation quality, it provides a basic indication of how well the model is predicting the correct next token. More sophisticated NLP metrics like ROUGE and BLEU are used separately for a more comprehensive evaluation of the generated text quality, as shown in the evaluation section.

7. Hyperparameter Tuning

Objective: Optimise model performance by tuning key hyperparameters.

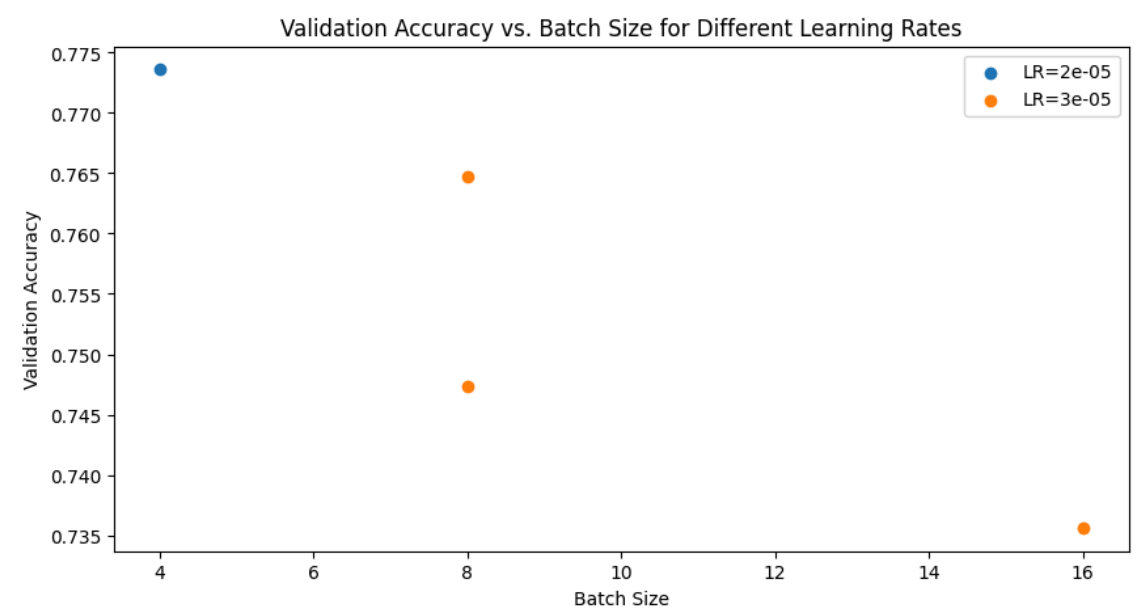
The hyperparameter tuning computed on different configurations of learning rate, batch size, and epochs optimises the model's performance.

- Learning Rate (2e-5, 3e-5): Small rates preserve pre-trained weights, balancing convergence and stability.
- Batch Size (4, 8, 16): Balances gradient stability and memory constraints on Google Colab
- Epochs (5, 10, 30): Prevents underfitting/overfitting with early stopping (patience=1).
- Metrics: Then compute Accuracy tracks token prediction; ROUGE-1 evaluates answer quality, and BLEU metrics performance evaluation on the best-performed model.
- A patience of 2 was used to allow more training flexibility while preventing overfitting.
- Checked improvement over baseline

The configuration with lr=2e-5, bs=4, and epochs=30 achieved the best validation loss of 1.2167 (21.53% improvement) and validation accuracy of 0.7736.

7.1. Experiments and Results Table

Expements	learning_rate	batch-size	epochs	val-loss	val-accuracy	Improvement
0-baseline	0.0003	8	5	1.550	0.7473	00%
1	0.0003	16	5	1.610	0.7355	-9.63%
2	0.0003	8	20	1.279	0.7647	17.45%
3-Best	0.0002	4	30	1.216	0.7736	21.53%



8. Evaluation Metrics Analysis

ROUGE-L and BLEU were calculated to evaluate the chatbot's performance for the best-performing model, but the results are not very good.

8.1 Quantitative Evaluation Metrics:

- **ROUGE-L Score**: ~0.0507, This metric measures the longest common subsequence between the generated answer and the reference answer, but I got a low ROUGE-L score suggests that the generated answers have minimal structural similarity with the expected reference answers.

- **BLEU Score:** ~0.0804, This metric measures the n-gram overlap between the generated answer and the reference answer, primarily focusing on precision. This result indicates an insufficient shared words and phrases between the generated and reference text.

Low scores may result from the reduced dataset size (~2,331 unique pairs) and T5-small's limited parameter count.

(Note: While F1-score and perplexity were considered, ROUGE and BLEU are more commonly used and directly relevant for evaluating the quality of generated text in question-answering tasks. F1-score is often used for span-extraction QA, and perplexity is more a measure of a language model's fluency rather than factual accuracy in a QA context.)

8.2 Qualitative Testing:

In addition to quantitative metrics, qualitative testing was performed by interacting with the chatbot through the Gradio interface using both in-domain and out-of-domain questions.

The qualitative results corroborated the findings from the quantitative metrics. The chatbot's responses were:

- Fluent but sometimes provides generic responses in agricultural details.

For example:

Question: ‘How to control pests on maize?’

Response: ‘Use pesticides,’ which is too generic

8.3 Thorough Analysis of Chatbot Performance:

The combined evidence from the low ROUGE-L and BLEU scores and the qualitative performance indicates that the fine-tuned T5-small model is not yet performing effectively as an agricultural question-answering system.

9. User Interface (UI)

The user interface for the agricultural chatbot was built using Gradio. The goal was to create an intuitive and user-friendly interface that allows seamless interaction.

Key Features and Justification:

- Intuitive and User-Friendly Design: The layout is straightforward, with clear sections for the header, instructions, chat history, input, and examples. The use of a themed design aligns with the agricultural theme and enhances visual appeal. (Meets "intuitive and user-friendly" criterion).
- Seamless Interaction: The core chat functionality (input box, send button) is standard and easily recognisable, allowing users to type questions and receive responses directly within the interface.
- Clear Instructions: A dedicated "How to Use Agro-Bot" section provides concise, bulleted instructions on how to interact with the chatbot.

Enhanced User Experience Features:

- Chatbot History: The gr. Chatbot component displays the conversation history, allowing users to follow the flow of the interaction.
- Placeholder Text: The input box and chatbot placeholder text provide initial guidance to the user on what to expect and how to start.

- Example Questions: Providing pre-defined example questions as clickable buttons makes it very easy for users to understand the types of queries the chatbot can handle.
- Clear Output Display: The chatbot component clearly displays both the user's input and the bot's response.
- Clear Chat Button: A dedicated "Clear Chat" button allows users to easily reset the conversation.

10. Conclusion and Future Work

This project successfully developed an AI-powered chatbot for agriculture using the KisanVaani dataset and a fine-tuned T5-small model. We went through the essential steps of data loading, comprehensive preprocessing, exploratory data analysis, model selection, initial training, hyperparameter tuning attempts, and building a user interface.

Key accomplishments include:

1. Loading and inspecting a relevant agricultural QA dataset.
2. Implementing detailed preprocessing steps, including normalisation and tokenisation.
3. Performing EDA to understand the dataset's characteristics and topic coverage.
4. Setting up a training pipeline for the T5-small model.
5. Attempting hyperparameter tuning to optimise performance.
6. Developing a user-friendly Gradio interface for interaction.

However, based on the model evaluation using ROUGE-L, BLEU, and qualitative testing, the current performance of the fine-tuned T5-small model is limited. The chatbot's responses are sometimes generic or not entirely accurate, indicating that the model has not yet fully learned to generate high-quality answers for this domain.

Challenge Faced and Overcome:

During model compilation, compatibility issues with AdamW and label smoothing were encountered due to potential library version mismatches. I resolved this by switching to the standard tf.keras. optimisers. Adam and removing label smoothing to ensure training stability, noting the need to investigate compatible versions later. Switching to Adam may have reduced regularisation.

Future Work:

To improve the performance and capabilities of the agricultural chatbot, the following steps could be explored:

- **Further Model Training:**

Train the current optimised model for a significantly higher number of epochs (e.g., 70-100 or more), carefully monitoring the validation loss to prevent overfitting.

- **Explore Larger Models:**

If computational resources are available, fine-tuning a larger version of T5 (e.g., T5-base, T5-large) or other powerful sequence-to-sequence models could lead to better performance as they have higher capacity to learn complex patterns.

- **Dataset Expansion or Augmentation:**

While the current dataset is a good starting point, expanding it with more diverse question-answer pairs or using data augmentation techniques could improve the model's generalisation.

- **Evaluate with Human Judgments:**

Supplement automated metrics with human evaluation to assess aspects like fluency, relevance, and factual correctness, which are not fully captured by metrics like ROUGE and BLEU.

By addressing these areas, I hope this agricultural chatbot can become a more reliable and helpful resource for farmers seeking information.

11. GitHub Repository

https://github.com/Joh-Ishimwe/Agri_chatbot

12. Deployed link and Video link

https://docs.google.com/document/d/1zxgSkirN0TaLzGcgyL4LFNXyzUNIetvXYgNzzPS9qlU/edit?usp=sharing

14. References

- [1]M. Aslam, M. A. Khan, S. A. Khan, S. Tariq, F. Alenezi, and Y. Nam, "Deep learning-based detection of maize crop diseases using remote sensing data," *Pattern Recognition Letters*, vol. 169, pp. 80–86, Jul. 2023. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0167865523002210
- [2]Y. Yassin, "Adam vs AdamW: Understanding weight decay and its impact on model performance," *Medium*, Aug. 31, 2023. [Online]. Available: https://yassin01.medium.com/adam-vs-adamw-understanding-weight-decay-and-its-impact-on-model-performance-b7414f0af8a1