Predicting Cancer Reoccurance With AI

Team sdmay24-10: Chris Tague, Eric Schmitt, Mark Hanson, Thriambak Giriprakash, Bishal Ghataney, Norfinn Norius
Faculty Advisor / Client: Ashraf Gaffar

Introduction

This is a discovery project to see if there is a correlation with the pathology of cancer cells and cancer reocurrence. We built a simple AI model and trained it on cancer cell spectrum data with the goal of predicting the reocurrence of cancer in order to improve cancer treatment.

Context

This project is intended to be used by patients who have had cancer and their doctors. It is used by uploading CSV files containing data spectrums of cancer cells to a secure website where a trained AI model is hosted. The AI model will use the uploaded data to return the predicted amount of time until cancer reoccurance. The website requires users to make an account in order to login and use the model.

Design Requirements

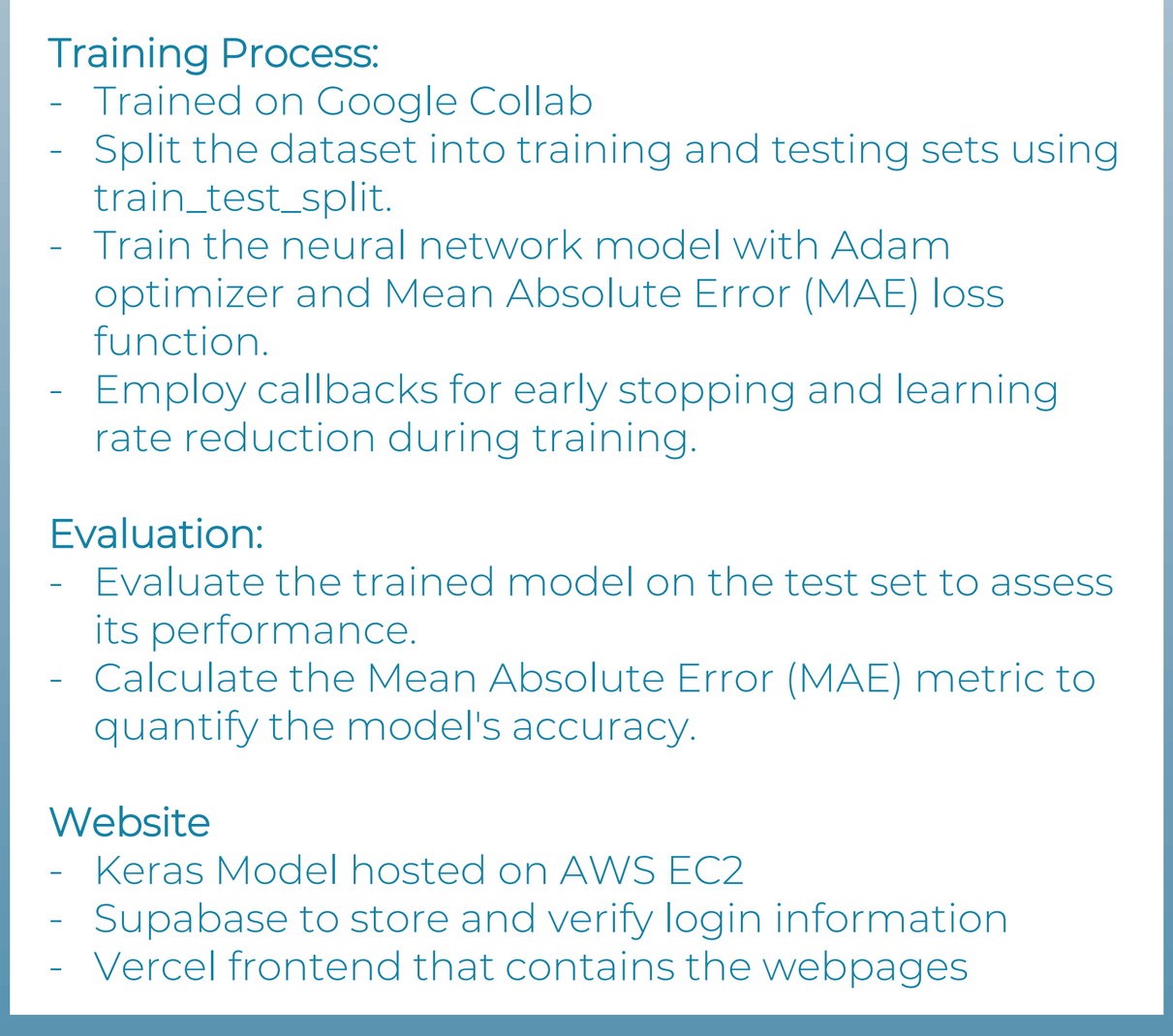
Data Preparation:
- Read data from an Excel file containing samples and their corresponding survival data.
- Load spectral data for each sample from CSV files.
- Preprocess the data by filtering out missing samples and normalizing the input features using StandardScaler.

Model Architecture:
- Utilize a neural network model consisting of
  - Input layer with BatchNormalization.
  - Two dense hidden layers with ReLU activation and Dropout regularization.
  - Output layer for regression.
- Advanced architecture:
  - Conv1D
  - MaxPooling1D
  - Flatten

Training Process:
- Trained on Google Collab
- Split the dataset into training and testing sets using train_test_split.
- Train the neural network model with Adam optimizer and Mean Absolute Error (MAE) loss function.
- Employ callbacks for early stopping and learning rate reduction during training.

Evaluation:
- Evaluate the trained model on the test set to assess its performance.
- Calculate the Mean Absolute Error (MAE) metric to quantify the model's accuracy.

Website
- Keras Model hosted on AWS EC2
- Supabase to store and verify login information
- Vercel frontend that contains the webpages



Technical Implementation & Testing

Initial Model Architecture:
- Started with a simple neural network architecture comprising BatchNormalization as the input layer and two dense layers.
- Encountered high Mean Absolute Error (MAE), indicating poor model accuracy.

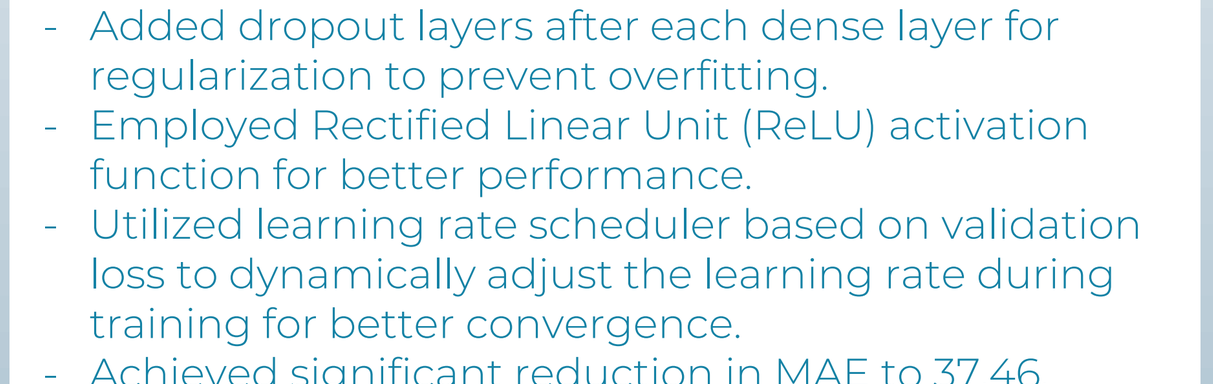
Exploration of Advanced Architectures:
- Experimented with more complex architectures including Conv1D, MaxPooling1D, and Flatten layers.
- Observed worsening of MAE, leading to abandonment of advanced architectures.

Attempted Ensemble Bagging Approach:
- Tried ensemble bagging and boosting approach with the original architecture to improve MAE.
- Minimal improvement observed in MAE, prompting abandonment of ensemble approach.

Adoption of Early Stopping and Optimizer Change:
- Implemented early stopping mechanism to prevent overfitting during model training.
- Switched from Stochastic Gradient Descent (SGD) optimizer to Adam optimizer for better convergence.
- Noticed gradual decrease in MAE from 61 to 45.

Introduction of Dropout, ReLU Activation and Learning Rate Scheduler:
- Added dropout layers after each dense layer for regularization to prevent overfitting.
- Employed Rectified Linear Unit (ReLU) activation function for better performance.
- Utilized learning rate scheduler based on validation loss to dynamically adjust the learning rate during training for better convergence.
- Achieved significant reduction in MAE to 37.46.

Exploration of Transfer Learning:
- Explored transfer learning by using pre-trained models to enhance model accuracy.
- Encountered system crashes and errors during implementation, leading to discontinuation of this approach.



Results & Limitations

- Most approaches tried failed due to high MAE with the best MAE achieved being 37.4
- We applied two statistical analyses to the data
  - **Pearson correlation**: coefficients measure linear correlations
  - **Spearman's Rank correlation**: measures can be used to measure any sort of correlation using monotonic measurements
  - **Results**: We found a very slight negative correlation between the input fields (spectrum and reoccurrence).
- Based on **Chart 2** the model appears to guess between about 125 to 175 months for all data entries, suggesting the model only learned to guess the average of its training data.
- Through months of rigorous testing we have concluded that the data which we have been tasked with analyzing does not show any strong correlation between the spectra and reoccurance.