Voight-Kampff Testing for Reddit Users

Victor Constantinescu, Cristian Cordoș
National University of Science and Technology POLITEHNICA Bucharest, Romania
vconstantinescu2710@stud.acs.upb.ro, ioan.cordos@stud.acs.upb.ro

February 3, 2025

Abstract

In this paper we show how we built a neural network that detects whether a Reddit user is human or bot

1 Introduction

Philip K. Dick, in his influential dystopian science fiction novel "Do Androids Dream of Electric Sheep?" imagined the Voight-Kampff test as a mean of determining whether a person is human or artificial. In this paper we are attempting to create a similar device for Reddit users by looking at their meta-data and the content they post.

2 Data Acquisition

First of all we have acquired a list of recognized reddit bots, and then we scraped a very popular reddit channel - AskReddit - to obtain human usernames. This allowed us to create a balanced dataset for training and validation.

For data acquisition we are using the python library ***praw*** that abstracts the communication with the Reddit server.

Equipped with the usernames and the praw library, we have obtained the following data per user relevant for this exercise:

- Created Date & Time
- Karma
- Link Karma
- Comment Karma
- Is Reddit Admin
- Verified Email
- Comments
  - Subreddit
  - Comment
  - Score
  - Created Date & Time

- Posts
  - Title
  - Subreddit
  - Score
  - URL
  - Created Date & Time

3 Data Processing

Using the scrapped data, we created some more features that will be used in training the Voight-Kampff neural network:

- Total Posts
- Total Comments
- Average Posts Per Day
- Average Comments Per Day
- Average Post Time Gap (hrs)
- Average Comment Time Gap (hrs)
- Average Comment Length
- Most Active Subreddits (Posts)
- Most Active Subreddits (Comments)
- Upvote/Downvote Ratio (Posts)

4 Neural Network

4.1 Architecture

The proposed bot detection model employs a hybrid architecture that combines textual and numerical feature analysis. The model consists of two main components: a pre-trained BERT encoder for text processing and a custom neural network for combining text embeddings with behavioral metrics.

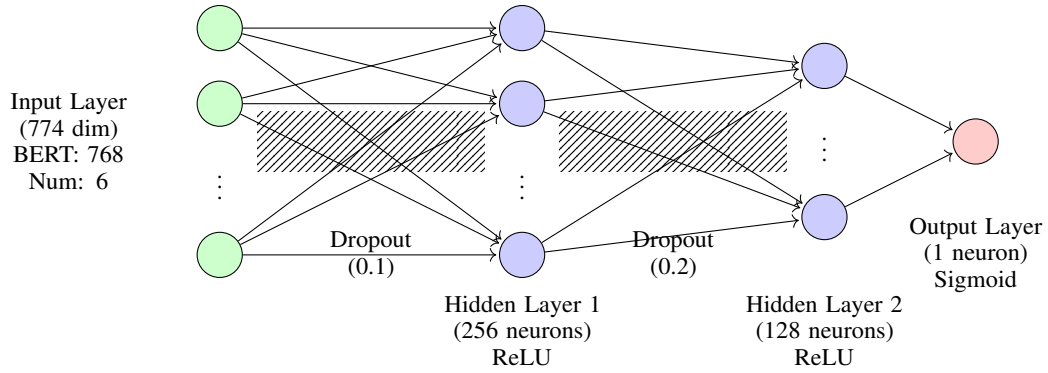
The text processing component utilizes BERT-base-uncased as the backbone, which processes user comments and post titles. To optimize computational resources, BERT's parameters are frozen during training. The model truncates input sequences to 512 tokens and employs special [SEP] tokens to separate different text entries from the same user.

The numerical features component processes six behavioral metrics, witch are mentioned in the Data Processing section of the paper.

The classifier combines BERT's 768-dimensional output with the numerical features through a three-layer neural network. The architecture employs:

- Input layer: 774 dimensions (768 from BERT + 6 numerical features)
- First hidden layer: 256 neurons with ReLU activation
- Second hidden layer: 128 neurons with ReLU activation
- Output layer: Single neuron with sigmoid activation

Dropout layers (0.1 after BERT, 0.2 between dense layers) are incorporated to prevent overfitting.



4.2 Training Process and Results

The model was trained on a balanced dataset of Reddit users, split into 80% training and 20% validation sets. Training utilized the AdamW optimizer with a learning rate of 2e-5 and binary cross-entropy loss. Early stopping was implemented with a patience of 5 epochs and a minimum improvement threshold of 0.001.

The training dynamics, visualized in Figures 1 and 2, reveal several key findings:

1. Convergence: The model achieved stable performance within 8 epochs, with validation accuracy reaching approximately 80%.
2. Learning Behavior:
   - Initial rapid improvement in validation accuracy from 60% to 79% in the first two epochs
   - Gradual training accuracy improvement from 52% to 79% over eight epochs
   - Stable validation accuracy hovering around 80% after epoch 3
3. Loss Patterns:
   - Training loss showed a significant spike at epoch 4 before stabilizing
   - Validation loss maintained relative stability after epoch 2, indicating good generalization
   - Final validation loss converged to approximately 1.4

The model demonstrates robust performance with minimal signs of overfitting, as evidenced by the parallel trajectories of training and validation metrics in later epochs. The final model achieves 80% validation accuracy, suggesting effective discrimination between human and bot accounts based on both textual content and behavioral patterns.

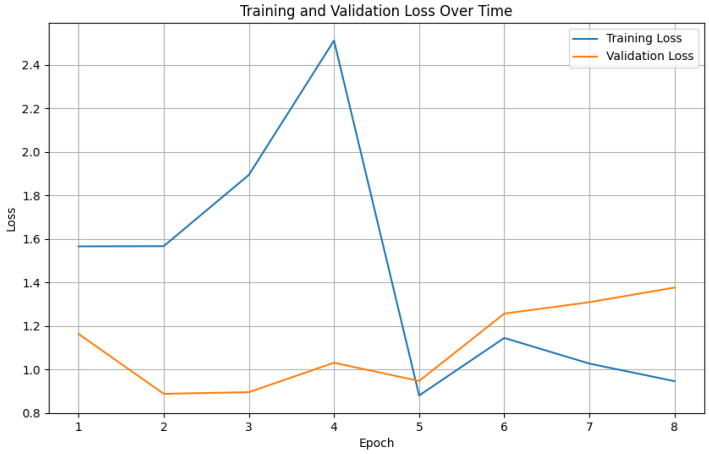


Figure 1: Training and validation loss over epochs. The training loss shows initial instability before converging, while validation loss maintains relative stability after epoch 2.

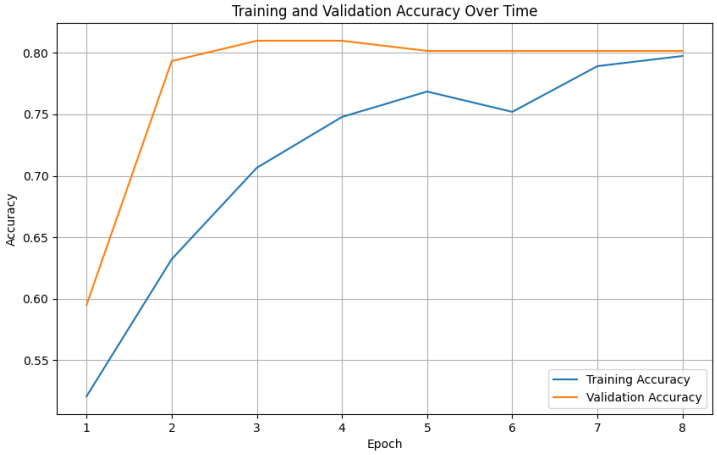


Figure 2: Training and validation accuracy over epochs. The validation accuracy shows rapid initial improvement followed by stable performance around 80%.

The model's performance was further validated on a separate test set. The evaluation metrics reveal robust classification capabilities, with notable results shown in the confusion matrix (Figure 3) and ROC curve (Figure 4).

The confusion matrix demonstrates:

- 13 true negatives (humans correctly identified)
- 21 true positives (bots correctly identified)
- 11 false positives (humans misclassified as bots)
- 3 false negatives (bots misclassified as humans)

These results translate to a precision of 65.6% and a recall of 87.5% for bot detection. The model shows a stronger ability to identify bots (21 correct identifications) compared to humans (13 correct identifications), suggesting a slight bias toward bot classification. This bias might be advantageous in applications where missing a bot (false negative) is considered more costly than misclassifying a human (false positive).

The ROC curve analysis (Figure 4) shows an Area Under the Curve (AUC) of 0.75, indicating good discriminative ability. The curve's shape suggests particularly strong performance in the high-specificity region, with the model achieving a true positive rate of approximately 0.8 while maintaining a relatively low false positive rate of 0.2.

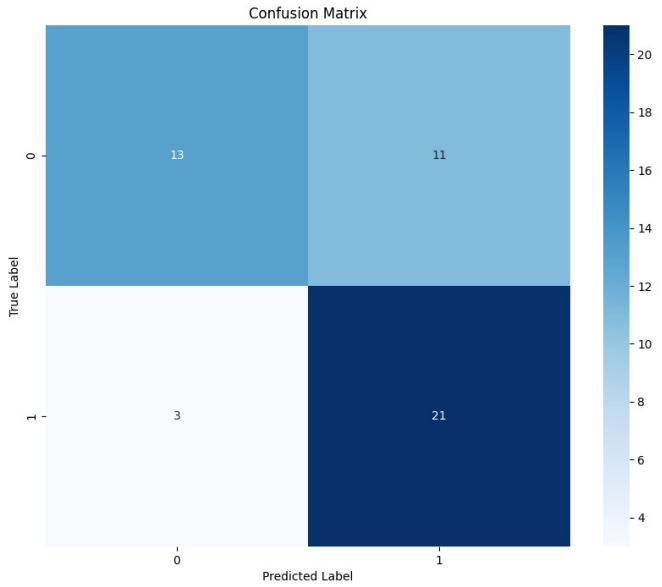


Figure 3: Confusion matrix showing the model's classification performance. The matrix reveals strong performance in bot detection (21 true positives) with moderate success in human identification (13 true negatives).

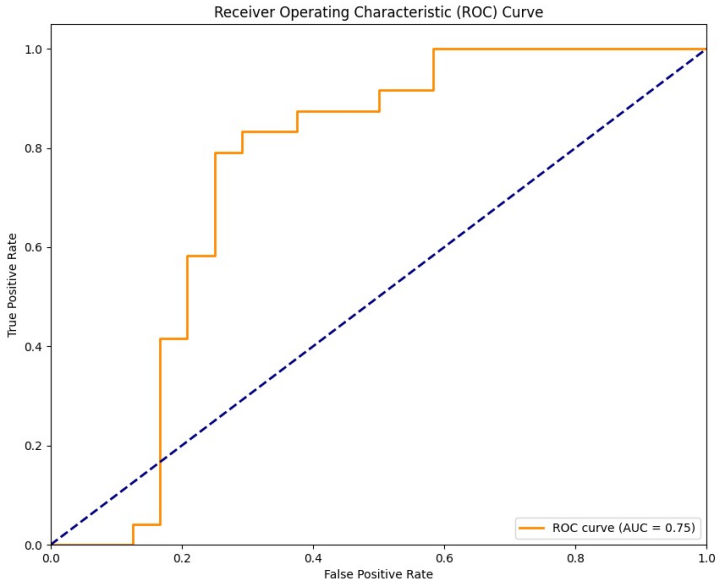
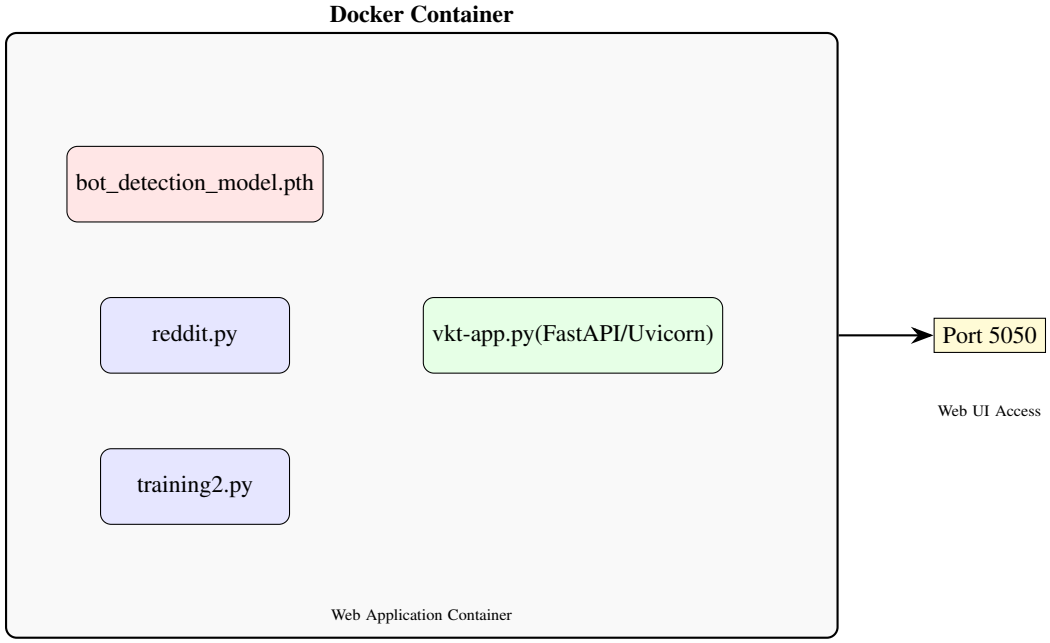


Figure 4: ROC curve displaying the model's classification performance across different threshold settings. The AUC of 0.75 indicates good discriminative ability, particularly in high-specificity regions.

5 Operationalization and Decision Thresholds

Using uvicorn we built a very simple web application that takes a Reddit username and after the user presses the "Test" button it will show the probability of the user being classified either as Human or as Bot.

5.1 Code Layout



The application is packaged in a docker container and contains the following artifacts:

- **bot_detection_model.pth** - the pretrained model weights
- **reddit.py** - reddit download functions
- **training2.py** - model classes
- **vkt-app.py** - the application itself, launched with fastapi under uvicorn

The application exposes a web UI over the 5050 port.

5.2 Decision Thresholds

The model's deployment implements a multi-threshold classification system that considers both prediction probability and confidence metrics. The system produces three possible classifications:

- Bot: Assigned when bot probability exceeds 50% AND activity score exceeds 30%
- Human: Assigned when bot probability is below 50% AND activity score exceeds 30%
- Inconclusive: Assigned when activity score falls below 30%, regardless of bot probability

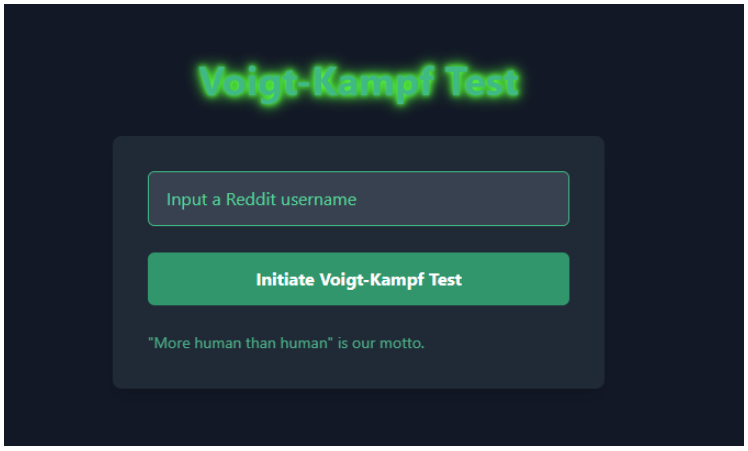


Figure 5: Voigt-Kampf Test interface for Reddit user analysis. The system provides a simple input field for username entry and references the classic "More human than human" motto.

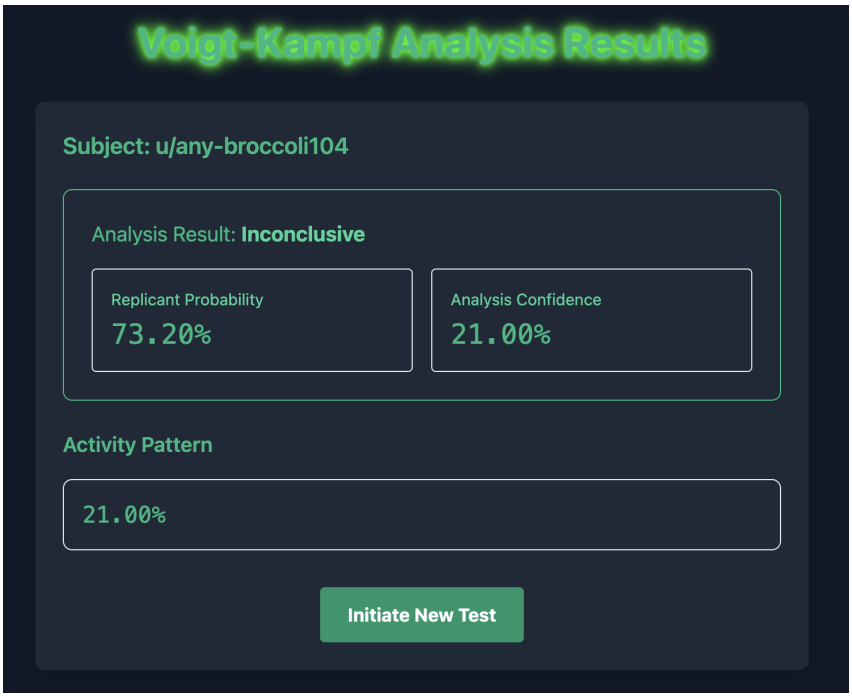


Figure 6: Analysis results interface showing the multi-threshold classification system. The display includes the replicant probability score, analysis confidence

The overall analysis confidence is primarily determined by the activity pattern score, which requires a minimum of 30% for conclusive classification. This prevents false classifications of inactive or new accounts while maintaining high precision for active users. For example, a user might show high bot probability (74.88%) but receive an "Inconclusive" classification due to low activity (14%), as demonstrated in the deployment interface.

6 Conclusions and Future Steps

In order to test the model we scraped the SubSimGPT2Interactive for bots. This subreddit has the propriety that bots are marked explicitly through their flair.
The 24 bots we tested were all properly detected as bots.
The model performs poorly - falsely detecting them as bots - with human users with low activity or

recent accounts.
This suggests that obtaining more training data is required in order to fine tune the model to better cover corner cases.