Gated Recurrent Unit based Authorship Identification

Luca Animati **Leonardo Nicola Palma** **Walter Verrino** **Angela Yu**

Bachelor Degree of Applied Computer Science & Artificial Intelligence

Sapienza University, Rome

Abstract

In nowadays age identifying authors is becoming increasingly important in debunking AI-generated texts and identifying lost authors' manuscripts. Natural Language Processing (NLP) has proven to be a viable tool for such tasks.
Our project focuses in particular on the identification of authors through the Gated Recurrent Unit (GRU) Model. Our dataset consists of a selection of 40 authors from the Gutenberg database, 1000 paragraphs each. Our model achieves an accuracy of 80.73% at correctly classifying such paragraphs.

1. Introduction

Authorship identification is one of the most challenging and useful tasks in Natural Language Processing (NLP) nowadays.
This process involves utilising a custom dataset and a neural network model.
Our dataset is a CSV file with 2 columns: paragraph and correspondent author, with up to 1000 paragraphs per writer. The model, a Gated Recurrent Unit (GRU) algorithm, leverages writing style, grammar, vocabulary and punctuation to analyse and predict authorship.
The applicability of authorship identification includes determining the true author of an article, detecting plagiarism in books or papers and also book recommendation systems.
This project was made possible by:

- *Dataset*: Based on texts from the Gutenberg project.
- *GRU Model*: A recurrent neural network algorithm.

The paper is organised as follows:

- *[2]* - *Method*:
  We offer a walkthrough around our work, including the creation of the dataset, the word representations as well as the model.
- *[3]* - *Results and Conclusions*:
  Experiments and their related results.
- *[4]* - *Contributions*:
  Summarised sources and contributions.

1.1 Related works

Authorship identification is a well-researched area in natural language processing (NLP). Numerous universities have contributed to this field, with varying focuses. Some projects have aimed to recognize and emulate an author's style while others centred on book recommendations. Notably, Stanford University achieved the highest accuracy (89.2%) in author identification using a GRU model at the article level, tested on the Gutenberg dataset.

2. Method

2.1 Gutenberg dataset

The first task we undertook was to download paragraphs for our dataset. Our goal was to obtain 1000 paragraphs for each of 40 well-known authors. We sourced these texts from the website Gutenberg.org through the Gutendex API, a database containing thousands of free ebooks.

To compile our dataset, we identified 40 authors on the site who had authored multiple books, ensuring that we could reach the target of 1000 paragraphs per author. Additionally, we applied two filters to our selection criteria: the books had to be in English, and the file extensions needed to be .txt to facilitate the creation of the CSV table.

2.2 Pre-Processing

After downloading the text, we began the data-cleaning phase. Due to variations in indexes, prefaces, comments and footnotes, the initial step involved manually extracting the narrative text from each book. Once the files were reduced to only relevant text data, texts from the same author were combined, tokenized into words, and then arranged into paragraphs of fixed length.

Initially, we removed punctuation and stop words during the final arrangement using custom methods. However, we found it more efficient to first arrange the text into paragraphs, extract 1000 paragraphs for each author, and then perform cleaning on this reduced dataset. This included searching for and removing irrelevant words, such as “chapter” and “section”.

We created one text file per author, with each paragraph on a separate line and the file named after the author. These paragraphs were then compiled into a CSV file, with each row containing a paragraph and its corresponding author, achieved by sequentially copying lines from each text file into the CSV.

Next, we used One-Hot Encoding to represent each author. Our final dataset is a CSV file with 40 columns and 40,000 rows, plus a header, representing the encoded authors and their associated paragraphs.

2.3 Word Embedding

For word embedding, we relied on the GloVe dictionary, a pre-trained word vector representation. This dictionary enabled us to map each word in the paragraphs to vectors of 100 dimensions.

After mapping each term to its corresponding vector, we addressed the issue of words in our .txt files that were not present in the GloVe dictionary. We decided to rename every non-matching word as “<UNK>” and experimented with two approaches to fill its 100-dimensional vector. The first approach was to set all values to the average float number of the GloVe dictionary. The second approach involved randomising 100 floats, between -1 and 1 (GloVe Embedding range of values).

Upon training the model with both approaches, we found that the choice between them did not significantly affect results.

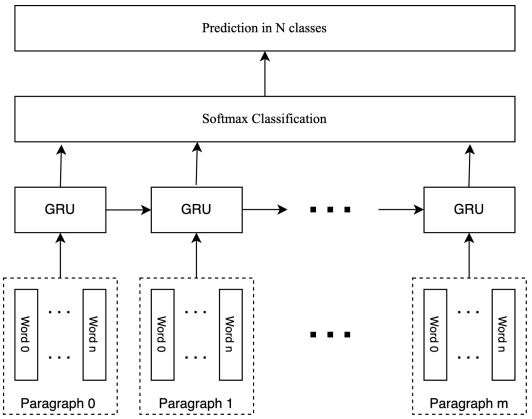
To ensure the model received properly formatted input, we standardised the length of our paragraphs. Initially, before tokenization, the average number of words per paragraph was relatively small, while after the average word count increased. For paragraphs exceeding the average number of words, we truncated them, and for shorter paragraphs, we padded them with the word “<PAD>”, represented as a zero vector of 100 dimensions.

2.4 Model

Sentence-level GRU predicts authors on continuous words, grouped in paragraphs. For each element in the input sequence, each layer computes the following function (1):

$$
\begin{aligned}
r_t &= \sigma(W_{ir}x_t + b_{ir} + W_{hr}h_{(t-1)} + b_{hr}) \\
z_t &= \sigma(W_{iz}x_t + b_{iz} + W_{hz}h_{(t-1)} + b_{hz}) \\
n_t &= \tanh(W_{in}x_t + b_{in} + r_t \odot (W_{hn}h_{(t-1)} + b_{hn})) \\
h_t &= (1 - z_t) \odot n_t + z_t \odot h_{(t-1)}
\end{aligned}
$$

Our model takes in input a paragraph as a sequence and each word as a time stamp, represented as word vectors.
The tensor passed through the GRU is computed by Eq. (1), producing a new hidden layer update and an output. It's important to note that h_t represents both the new hidden layer and the output.
As the last word is processed, the logits are calculated taking into account all the previous timestamps. The output of the GRU is finally passed to a softmax classifier, where the final prediction will be made.



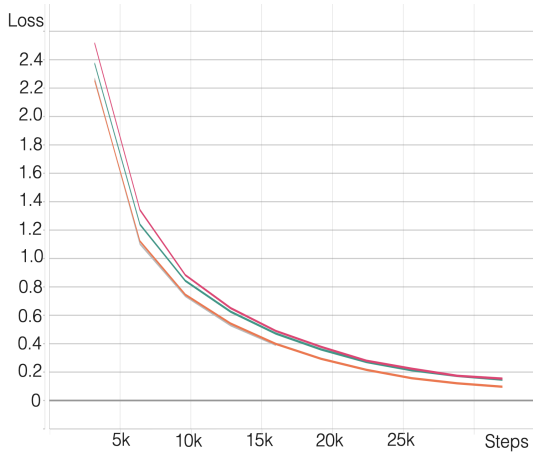
2.5 Experiment

Initially, we considered both the "<UNK>" and the "<PAD>" as 100-dimensional zero vectors.
This approach proved to be ineffective, resulting in an accuracy of just 2%.
By doing so, we actually were taking away all the weight and importance of the unknown words to the paragraph context. After this was fixed, we moved on with the tweaking of the hyperparameters, which was the most important and time-consuming task regarding the model per se.
We concluded that with a hidden size of 128 and a learning rate of 0.001 we achieved the highest test accuracy. We then added the plotting of the confusion matrix to check the predicted authors, which led to the removal of hard-to-read ones.
A validation set was also added to make sure that the model was not falling into overfitting.
By observing the loss for both the validation and the training set, we derived that per epoch and per step, they were roughly the same. This was indeed a good estimator for overfitting and we concluded that our model was learning just fine.

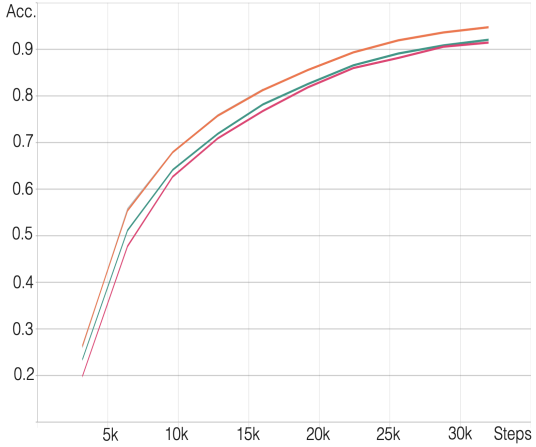
3. Results and Conclusions

We achieved a maximum accuracy of 94% during the training phase and 80.73% in the testing phase, demonstrating the substantial potential of GRUs in capturing the stylistic and linguistic features essential for distinguishing between different authors. To further illustrate these results, in the following page we present several graphs that provide a clearer understanding of the model's performance and the underlying patterns it has learned.

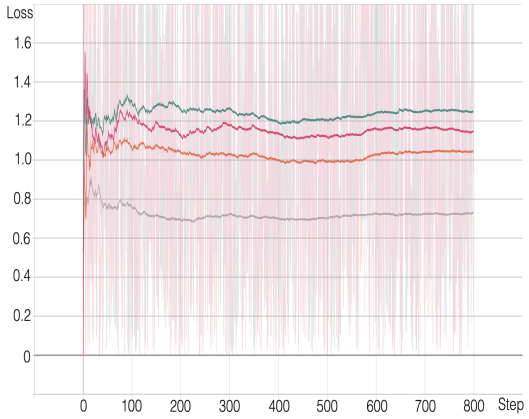
3.1 Results



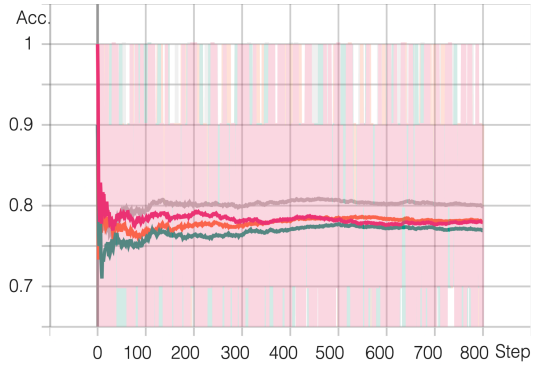
(a) Train loss vs steps



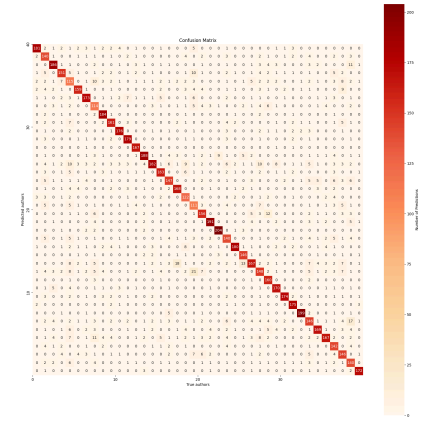
(b) Train accuracy vs steps



(c) Test loss vs steps



(d) Test accuracy vs steps



(e) Confusion Matrix

3.2 Conclusions

By smoothing the accuracy and loss results during the testing phase, it becomes evident that the model performs consistently across most authors, with only a few outliers. Despite these encouraging results, there are still numerous opportunities for further research and enhancement. Future studies could examine the effects of different preprocessing techniques, explore alternative model architectures, and undertake extensive hyperparameter tuning to further improve performance. Additionally, expanding the dataset to include a larger number of authors and a broader spectrum of writing styles would allow for a more comprehensive assessment of the model's generalizability and robustness.

4. Contributions

[1] Chen Qian, Tianchang He, Rao Zhang. (2021) *Deep Learning based Authorship Identification,* Stanford University.

[2] Simeon Kostadinov. (2017) *Understanding GRU Networks,* towardsdatascience.com.

[3] Josh Starmer. (2023) *Recurrent Neural Networks (RNNs), Clearly Explained!!!,* StatQuest with Josh Starmer (YT).