**Project Name:** Classify Nationalities

**Github Link:** https://github.com/projectsforstudents2022/Classify\_Nationalities.git

**Why was this project created?**

To learn more about how characters are organized for naming, consider using personal names. Compared to other proper nouns like corporate names or acronyms, human names have far more traditional structures. When the names are restricted to a single ethnicity, these patterns become more obvious. Thus, a neural network-based model that predicts a personal name's country entirely based on its characters, without the use of outside sources or manually created features.

**What problem is it solving?**

We provide a methodology based on recurrent neural networks that accurately predicts the nations of each name. Our model's performance is on par with that of earlier feature-based models without mentioning any custom features or naming conventions. We suggest estimating a person's country simply based on the characters in their name, as opposed to earlier models that mostly relied on predicting the ethnicity of a name. The performance is significantly enhanced by character level embeddings, demonstrating the efficiency of autonomous feature extraction.

**Entire explanation of project**

* **PROPOSED APPROACH**

We must first import the necessary libraries before loading Dataset, which is a text file containing the person's name and nationality, separated by a comma. More than 20k names and 18 different nations, including Portuguese, Irish, and Spanish, are included in the collection. Given the size of the data, we will divide it into training and testing groups at a ratio of 70 to 30. Since the dataset in this classification issue is unbalanced, we shall employ a stratified sampling approach. Instead of using the raw text data, the sequence model we will create will be fed character encodings. Therefore, we must characterise the input and encode it. To obtain the encodings for the entire word, we must concatenate the character-level encodings that have already been created. We just locate the index of the occurrence of that specific nationality in our list of nationalities to encode nationality. After that, encode that index.

Make conclusions from the data prior to beginning the model training process. Next, a general training setup will be developed that can be used for different networks, such as LSTM and GRU. We must provide the loss function and optimization strategy before we can train our network. In this instance, we'll utilize NLLLoss to determine the network's loss and the SGD optimizer to locate the global minima. We go through the data loader for each epoch in our training loop. Obtain the labels and input data, Before computing the gradient for the following batch, reset any prior gradient that may have been present in the optimizer. Get the output by doing the forward pass. The progress messages are printed at the conclusion of each period.

Algorithm for creating next word prediction model :

**Step 1:** Import Libraries & Load Dataset

**Step 2:** Encoding Names and Nationalities

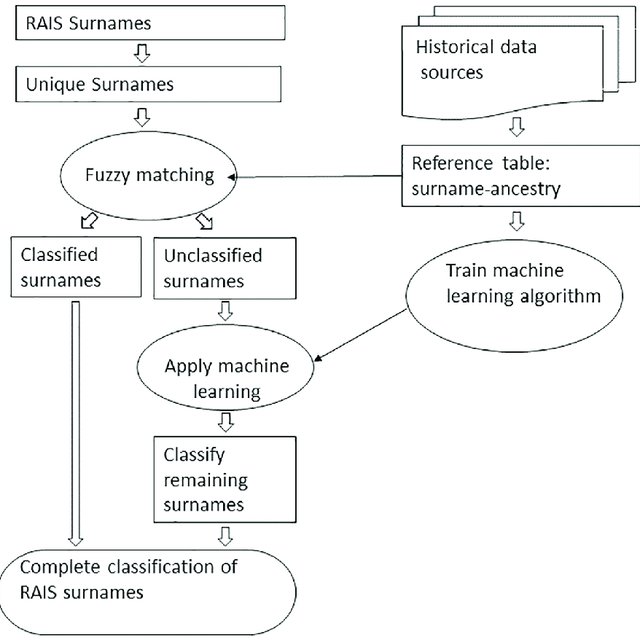
**Step 3:** Build RNN Model

**Step 4:** Inference on RNN Model

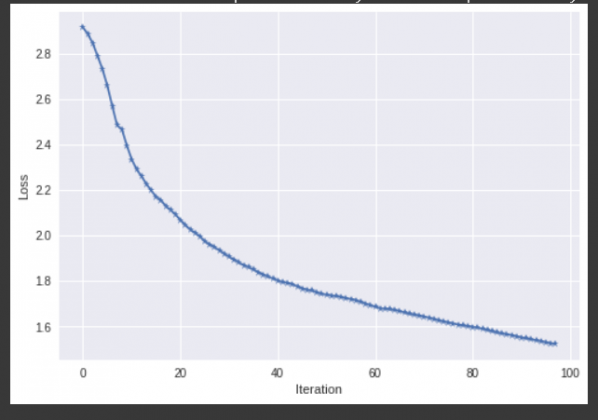
**Step 5:** Train Model

**Step 6:** Testing

* **DATA FLOW DIAGRAM**



* **RESULT**



* **CONCLUSION**

We spoke about the necessity to identify a person's country based on their name in this post. After that, we learned how to import our unique dataset into the format needed to train our model. Before training the model, we then spoke about how to encode the names and nationalities. Finally, we have seen examples of RNN and LSTM Model implementations utilized for data training.