Geoscribe

Text-based Geolocation Prediction of Social Media Users with Neural Networks

Team Bro>>Code

(Gautam Soni, Soham Thatte , Hargun Singh Chandhok)

Date: 06-02-2024

Preface

In an era dominated by social media, understanding user behavior and predicting their geolocation based on textual content is an intriguing challenge. The project "Text-based Geolocation Prediction of Social Media Users with Neural Networks" explores the fusion of natural language processing and neural network models to unravel the geographical origins of social media posts. This documentation serves as a guide, providing insights into the project's objectives, methodologies, models, and implementation details.

Introduction

The project aims to predict the geolocation of social media users based on their text data using various neural network models. The implemented models include Text CNN, Text RNN, Text RCNN, FastText, HierarchicalWithAttention, BiLSTMTextRelation, Seq2SeqAttn, and CNNWithAttn. The dataset used for training and evaluation is the CMU Geo-tagged dataset, focusing on classifying users into one of the 49 US states or predicting latitude and longitude as regression tasks.

Project Objectives

The primary goal of this project is to predict the geolocation of social media users through the analysis of their textual data. Leveraging a diverse set of neural network architectures, the project aims to address both classification and regression tasks. The models are designed to classify users into one of the 49 US states or predict latitude and longitude coordinates. The project also delves into the integration of geographical predictions with web technologies, using React for the frontend, Flask for the backend, Firebase for authentication, and the Google Maps API for visualization.

Models Overview

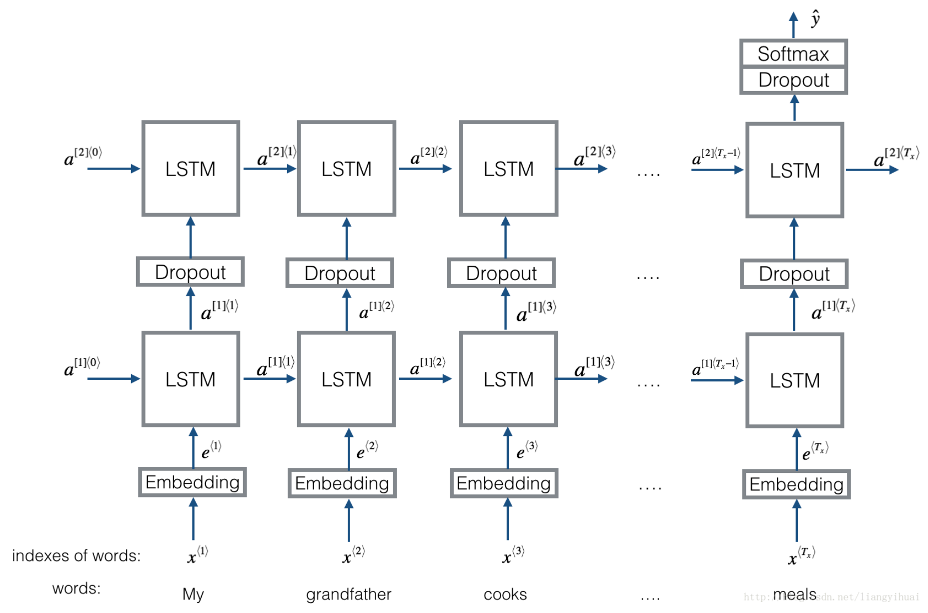
---

1. Text CNN:

- Structure: Embedding → Convolutional → Max Pooling → FC layer → Softmax
- Based on "Convolutional Neural Networks for Sentence Classification."
- Utilizes convolutional layers to capture local patterns in the text.

2. Text RNN:

- Structure: Embedding → Bi-directional LSTM → Dropout → Concat output → LSTM → Dropout → FC layer → Softmax
- Derived from the Emojifier-v2 model.
- Employs bidirectional LSTM layers to capture context information in both directions.



$\hat{y}$
Softmax
Dropout
$a^{[2]\langle 0\rangle}$
$a^{[2]\langle 1\rangle}$
$a^{[2]\langle 2\rangle}$
$a^{[2]\langle 3\rangle}$
$a^{[2]\langle T_x-1\rangle}$
$a^{[2]\langle T_x\rangle}$
LSTM
Dropout
$a^{[1]\langle 1\rangle}$
$a^{[1]\langle 2\rangle}$
$a^{[1]\langle 3\rangle}$
$a^{[1]\langle T_x\rangle}$
$a^{[1]\langle 0\rangle}$
$a^{[1]\langle T_x-1\rangle}$
$e^{\langle 1\rangle}$
$e^{\langle 2\rangle}$
$e^{\langle 3\rangle}$
$e^{\langle T_x\rangle}$
Embedding
indexes of words:
$x^{\langle 1\rangle}$
$x^{\langle 2\rangle}$
$x^{\langle 3\rangle}$
$x^{\langle T_x\rangle}$
words:
My
grandfather
cooks
....
meals

3. Text RCNN:

- Structure: Recurrent structure (convolutional layer) → Max Pooling → FC Layer → Softmax
- Implementation of "Recurrent Convolutional Neural Network for Text Classification."
- This will Learn word representations considering left and right side contexts.

4. FastText:

- Structure: Word embeddings → Averaging → Linear classifier with softmax
- Based on the "Bag of Tricks for Efficient Text Classification" paper.
- Uses n-gram features to capture partial information about word order.

5. HierarchicalWithAttention:

- Structure: Embedding → Word Encoder (bi-directional GRU) → Word Attention → Sentence Encoder (bi-directional GRU) → Sentence Attention → FC + Softmax
- Implementation of "Hierarchical Attention Networks for Document Classification."
- Utilizes attention mechanisms at both word and sentence levels for improved representation.

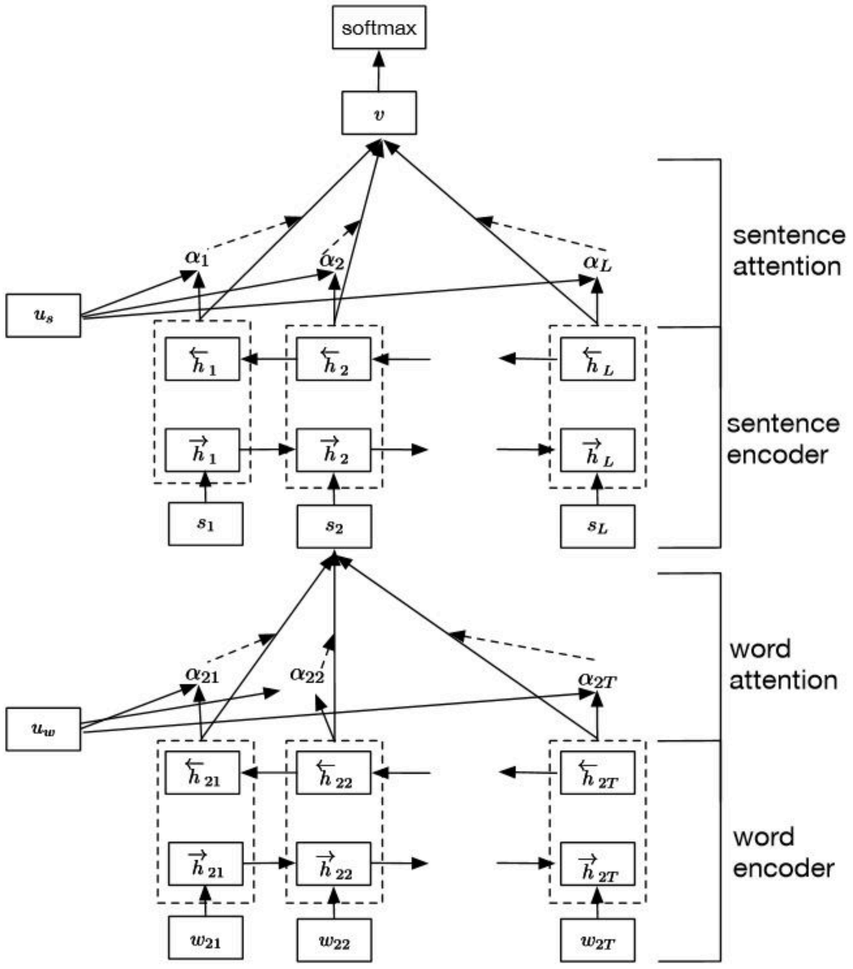


Figure 2: Hierarchical Attention Network.

6. BiLSTMTextRelation:

- Structure: Embedding → Bi-directional LSTM → Dropout → Concat output → LSTM → Dropout → FC layer → Softmax
- Based on the Dual LSTM Encoder model.
- Designed for relation classification using bidirectional LSTM layers.

7. Seq2SeqAttn:

- Structure: Embedding → Bi-directional GRU → Decoder with attention
- Derived from "NEURAL MACHINE TRANSLATION BY JOINTLY LEARNING TO ALIGN AND TRANSLATE."
- Implements a sequence-to-sequence model with attention for geolocation prediction.

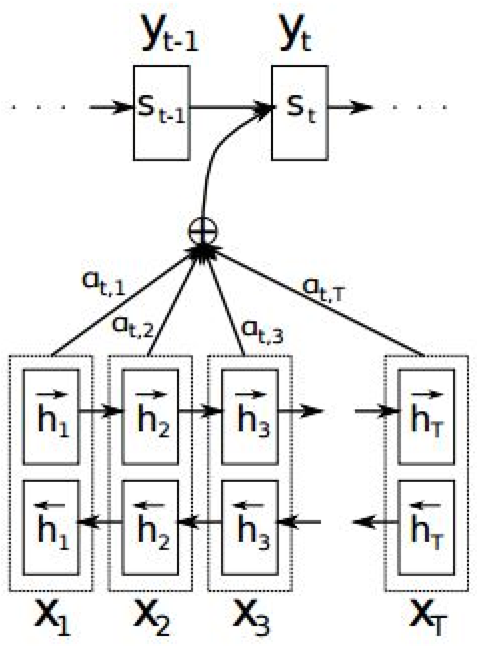


Figure 1: The graphical illustration of the proposed model trying to generate the t-th target word y_t given a source sentence $(x_1, x_2, \ldots, x_T)$.

8. CNNWithAttn:

- Structure: Based on "Neural Relation Extraction with Selective Attention over Instances."
- Combines convolutional layers with attention mechanisms for improved relation extraction.

Project Instructions

Frontend (React App):

1. Open your terminal.

2. Change the directory to the frontend React app:
   ```bash
   cd frontend
   ```

3. Install the necessary dependencies:
   ```bash
   npm install
   ```

4. After the installation is complete, start the React app:
   ```bash
   npm start
   ```

This command will launch the development server, and you should be able to access the frontend at `http://localhost:3000` in your web browser.

Backend (Flask App):

1. Open a new terminal window (keeping the frontend terminal running).
2. Change the directory to the backend:
```bash
cd backend
```
3. Install the required Python packages:
```bash
pip install -r requirements.txt
```
4. Once the installation is done, start the Flask app:
```bash
python main.py
```
This command will launch the Flask development server, and the backend should be accessible at `http://localhost:5000`.

Now, both the frontend and backend servers should be up and running. You can access the application by navigating to `http://localhost:3000` in your web browser.

Please note that these instructions assume that you have Node.js, npm, and Python installed on your machine. Additionally, make sure to check the console outputs for any error messages during the installation and startup processes.

Usage and Environment

Neural network models are provided in the "models" folder.
The project is developed in Python 3.9.7, using TensorFlow 2.15.0 and Numpy.

Frontend and Backend Implementation

- The frontend is implemented using React.
- The backend is developed using Flask.
- Firebase is employed for authentication in the tool.
- Google Maps API is utilized to map the final predicted longitudes and latitudes.

Conclusion

The project's comprehensive approach revolves around the prediction of social media users' geolocation through the application of a diverse array of neural network models.

Each of these models is carefully selected to cater to distinct aspects of text representation and context capturing, allowing for a nuanced understanding of the textual data.

The synergy of React, Flask, Firebase, and the Google Maps API plays a pivotal role in providing a well-rounded framework.

This framework serves as the backbone for crafting a user-friendly tool, characterized by robust authentication mechanisms and powerful visualization capabilities.