GeoRec: A Comprehensive Study and Integration of Transformer and Non-transformer Models for Personalized Location-Based Recommendations Using Social Network Data

Shromana Kumar
shromana.kumar@ufl.edu

Srinayani Mankali
smankali@ufl.edu

***Abstract*—This research addresses the pivotal task of developing a cutting-edge Personalized Travel Recommendation System (PTRS), aiming to enhance the travel experience through tailored recommendations. Leveraging social network data from Gowalla, we intricately capture user behavior via key attributes like user ID, timestamp, and location coordinates. Our study explores traditional sequential models such as LSTM and Bi-LSTM, known for temporal pattern capturing, and compares them with transformer models, particularly BERT, renowned for natural language processing. In evaluating precision, recall, F1-score, and training times, our results reveal a nuanced performance landscape. While non-transformer models excel in precision and recall, transformer models, notably BERT, exhibit consistent loss reduction and overall robustness, outperforming counterparts in stability and effective convergence. By addressing the limitations of existing approaches and showcasing compelling experimental results, this research pioneers a transformative fusion of transformer and non-transformer models for personalized travel recommendations, marking a significant stride in the evolution of location-based recommender systems.**

I. Introduction

In the era of rapid urbanization and increased mobility, the demand for personalized travel recommendations has surged, emphasizing the need for innovative solutions to enhance individual travel experiences. Our project, GeoRec, tackles this challenge by aiming to develop a cutting-edge personalized travel recommendation system. Originally conceptualized to leverage Uber ride data, the project strategically shifted its focus to harness the wealth of social network data, specifically from the Gowalla app, due to dataset limitations. This shift provides a more extensive and diverse dataset, enriching the spatiotemporal event history used for tailored recommendations.

A. Background

The significance of our endeavor lies in the potential to simplify and enrich travel experiences, making them more enjoyable and efficient for users. Traditional location recommender systems often rely heavily on user reviews, neglecting valuable geospatial information that could significantly enhance the contextual relevance of recommendations. Moreover, existing research predominantly favors non-transformer models for location recommendations, indicating a gap in effectively integrating the capabilities of pre-trained transformer models, such as BERT (Bidirectional Encoder Representations from Transformers).

B. Societal Applications

The motivation behind GeoRec lies in simplifying and enriching travel experiences, emphasizing the discovery of destinations aligned with user preferences. This project holds promising societal applications, notably in enhancing travel experiences by offering personalized recommendations that cater to individual tastes. Additionally, it contributes to improved decision-making, providing users with data-driven insights to make informed choices about travel destinations based on their historical preferences. Moreover, GeoRec addresses the common challenges associated with travel planning, aiming to reduce stress and save time by offering tailored recommendations, ultimately optimizing the overall travel experience for individuals.

C. Existing Approaches

- BERT4Loc: BERT for Location – POI Recommender System [1] by Bashir et al.Levandoski
  - **Model:** This study introduces a location-aware recommendation system utilizing Bidirectional Encoder Representations from Transformers (BERT) for personalized location-based suggestions.
  - **Methodology:** Combining location information and user preferences, it outperforms models predicting the next Point of Interest (POI) in a sequence.
  - **Advantages:** BERT4LOC showcases the use of BERT, traditionally not applied in location-based AI analysis, and demonstrates the superiority of pre-trained BERT-based models over sequential models for location recommendations.
  - **Limitations:** The model heavily relies on user context and preferences, leading to potential overfitting, and requires large amounts of data for training, making it computationally expensive.

- LARS: A Location-Aware Recommender System [2] by Levandoski et al.
  - **Model:** LARS employs item-based collaborative filtering as its primary recommendation method.
  - **Methodology:** It introduces a location-based recommender system considering user location preferences, classifying ratings into spatial and non-spatial categories for optimized recommendations.
  - **Advantages:** LARS optimizes recommendations through user partitioning and travel penalty, enhancing scalability without compromising quality.
  - **Limitations:** Relying primarily on ratings, it overlooks contextual factors and spatiotemporal dynamics, potentially leading to less accurate recommendations.
- Location-Based Recommendation System Using Bayesian User's Preference Model in Mobile Devices [3] by Park et al.
  - **Model:** Utilizes a Bayesian Network (BN) to create a map-based personalized recommendation system reflecting user preferences.
  - **Methodology:** Collects context information such as location, time, weather, and user requests from mobile devices to infer the most preferred item based on the mini-map.
  - **Advantages:** Context-aware, considering factors like location, time, and weather for highly relevant recommendations based on current situations.
  - **Limitations:** Reliance on BN structure design, requiring expertise, and heavy dependence on data for parameter learning, leading to potential computational intensity.
- GeoRSA: Geospatial Recommendation System using Sentimental Analysis [4] by Mesfin et al.
  - **Model:** Addresses the challenge of selecting tourist destinations through sentiment analysis and user-generated content, comparing various ML and DL models.
  - **Methodology:** Involves data collection from social media, sentiment analysis, user preference extraction, and personalized recommendation generation.
  - **Advantages:** Offers personalized travel recommendations by analyzing user sentiments and utilizing vast user-generated content.
  - **Limitations:** Heavily depends on user-generated data, which may be scarce or inaccurate, affecting recommendation reliability. This study introduces a location-aware recommendation system utilizing Bidirectional Encoder Representations from Transformers (BERT) for personalized location-based suggestions.
- Design and Development of a Real-Time Optimal Route Recommendation System Using Big Data for Tourists in Jeju Island [5] by Mehmood et al.
  - **Model:** Presents a robust methodology for a real-time tourist route recommendation system in Jeju Island.
  - **Methodology:** Leverages real-time big data, external APIs, and a Naïve Bayes classifier for personalized recommendations considering user preferences, time, distance, and location popularity.
  - **Advantages:** Offers personalized recommendations, responsiveness to changing conditions, and maximizes visits to famous locations.
  - **Limitations:** Lacks a dedicated machine learning model for precision and faces challenges like group recommendations and privacy issues.
- TRACE: Travel Reinforcement Recommendation Based on Location-Aware Context [6] by Fu et al.
  - **Model & Methodology:** Leverages a data-driven, context-aware framework integrating historical preferences and real-time context, using CNN-based and attention-based models, and an Actor-Critic framework for attraction recommendations.
  - **Advantages:** Provides real-time recommendations considering shifting user preferences and context-aware understanding of user locations on attraction preferences.
  - **Limitations:**: Reliance on abundant data, computational demands for large datasets, and the intricacies of the Actor-Critic framework can be challenging. Ongoing obstacles involve sparse data, the cold-start problem for new users, accommodating dynamic location changes, and ensuring scalability for numerous users and attractions while sustaining real-time performance.

D. Analysis of Existing Approaches

These existing approaches showcase diverse methodologies in addressing location-based recommendation challenges. BERT4Loc [1] and TRACE [6] leverage transformer models, emphasizing the significance of context-awareness and real-time adaptability. LARS [2] optimizes recommendations through collaborative filtering and spatial considerations. The Bayesian [3] approach in the Location-Based Recommendation System highlights the importance of context in providing relevant suggestions. GeoRSA [4] explores sentiment analysis for personalized recommendations, while the Jeju Island [5] route recommendation system focuses on real-time adjustments.

E. Limitations Across Approaches

Despite their strengths, limitations persist. Overreliance on user context and preferences, as seen in BERT4Loc [1], can lead to potential overfitting. LARS's [2] reliance on ratings neglects contextual factors, impacting accuracy. Bayesian models [3] demand expert-designed structures, potentially leading to suboptimal results. GeoRSA's [4] reliance on user-generated data raises concerns about scarcity and accuracy. Real-time systems, like TRACE [6], grapple with challenges related to abundant data, computational demands, and scalability.

F. *Unique Contribution of Our Approach*

Our project distinguishes itself through a holistic integration of transformer and non-transformer models, including LSTM and Bi-LSTM, for comprehensive performance comparison. The technical contributions lie in adapting BERT for location data, addressing the limitations of non-transformer models, and providing a nuanced understanding of their effectiveness. We hypothesize that transformer models will outshine non-transformers, a premise guiding our strategic choice for superior personalized travel recommendations.

In summary, GeoRec seeks to bridge the gap between traditional and transformer-based models, offering a refined approach to personalized location-based recommendations. The ensuing sections will delve into the experimental setup, results analysis, and optimization strategies, presenting a comprehensive exploration of the project's technical intricacies and potential impact on the field of recommender systems.

II. Problem definition

The primary objective of this research is to address the challenge of creating a highly personalized and efficient travel recommendation system. The problem revolves around leveraging user-specific historical data to generate tailored travel suggestions. Our focus has shifted from the original intent of using Uber ride data due to dataset constraints, and we now aim to utilize social network check-in data from applications such as Gowalla.

A. *Inputs*

The input dataset comprises essential details extracted from Gowalla app's social network check-ins. These details include user ID, timestamp, latitude, longitude, spot ID, and spot name. Each entry in the dataset represents a user's check-in event, capturing the spatiotemporal dimensions of their travel history.

B. *Outputs*

The desired output is a sophisticated travel recommendation system capable of providing personalized suggestions based on an individual's spatiotemporal event history. The system should consider user preferences, historical check-ins, and other relevant factors to offer highly relevant travel recommendations.

C. *Example*

Consider a scenario where a user has a history of check-ins at various locations, including specific timestamps, geographical coordinates, and spot names. The input for the system would be this user's historical check-in data. The system's output would then be personalized travel recommendations, suggesting destinations, activities, or routes that align with the user's preferences and historical travel patterns.

By formally defining the inputs, outputs, and illustrating the problem with a concrete example, we lay the foundation for developing a comprehensive and effective solution to enhance personalized travel experiences.

III. Proposed solution

Our sophisticated model architecture encompasses three distinct components, each meticulously tailored to capture nuanced facets of the personalized travel recommendation system.

```
# Use cluster assignments as input features for BiLSTM
X_bilstm = np.column_stack((X_sample, y_kmeans))  # Adding cluster assignments as a new feature

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X_bilstm, y_kmeans, test_size=0.2, random_state=42)

# Reshape X_train and X_test for BiLSTM input
X_train_reshaped = X_train.reshape((X_train.shape[0], 1, X_train.shape[1]))
X_test_reshaped = X_test.reshape((X_test.shape[0], 1, X_test.shape[1]))

# Define BiLSTM model
model = Sequential()
model.add(Bidirectional(LSTM(50), input_shape=(X_train_reshaped.shape[1], X_train_reshaped.shape[2])))
model.add(Dense(1, activation='linear'))
model.compile(optimizer='adam', loss='mean_squared_error')

# Record the start time
start_time = time.time()

# Train BiLSTM model
history = model.fit(X_train_reshaped, y_train, epochs=10, batch_size=32, validation_data=(X_test_reshaped, y_test))

# Record the end time
end_time = time.time()

# Calculate the training time
training_time = end_time - start_time
# Predict using the trained model
X_predict_bilstm = np.column_stack((nyc_events[['longitude', 'latitude']], y_kmeans))
X_predict_bilstm_reshaped = X_predict_bilstm.reshape((X_predict_bilstm.shape[0], 1, X_predict_bilstm.shape[1]))
nyc_events['cluster'] = model.predict(X_predict_bilstm_reshaped)
```

Fig. 1. Pseudo code for BILSTM Architecture

The Bidirectional Long Short-Term Memory (BiLSTM) model, a cornerstone of our approach, features a bidirectional structure with 50 units in both forward and backward directions. This design choice enables the model to intricately comprehend temporal patterns inherent in the user's spatiotemporal event history. The training process spans 10 epochs, employing a batch size of 32 for efficient optimization through the Adam optimizer. Notably, the model converges towards minimizing mean squared error loss. The expansive dataset, comprising 5,154,312 training samples, 644,289 validation samples, and 644,289 testing samples, is thoughtfully partitioned, with an 80% allocation to training, 10% to validation, and 10% to testing.

In parallel, the Long Short-Term Memory (LSTM) model is strategically designed with 50 units to specifically capture sequential dependencies inherent in the user's historical travel patterns. Mirroring the BiLSTM approach, LSTM undergoes 10 training epochs with a batch size of 32, utilizing the Adam optimizer to fine-tune its parameters. The dataset partitioning strategy remains consistent, maintaining an 80% training, 10% validation, and 10% testing split.

Diverging from the sequential models, the inclusion of the BERT (Bidirectional Encoder Representations from Transformers) model marks a non-traditional yet pivotal element in our architecture. Leveraging the 'bert-base-uncased' tokenizer with a maximum sequence length of 20 tokens, this transformer model encapsulates the contextual information from user inputs. The architecture further incorporates an embedding layer with 128 neurons and

```
X_lstm = np.column_stack((X_sample, y))  # Adding cluster assignments as a new feature

# Split data into train, validation, and test sets
X_train, X_temp, y_train, y_temp = train_test_split(X_lstm, y, test_size=0.3, random_state=42)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.5, random_state=42)

# Define LSTM model
model = Sequential()
model.add(LSTM(50, input_shape=(X_train.shape[1], 1)))
model.add(Dense(1, activation='linear'))
model.compile(optimizer='adam', loss='mean_squared_error')

# Reshape X_train, X_val, and X_test for LSTM input
X_train_reshaped = X_train.reshape((X_train.shape[0], X_train.shape[1], 1))
X_val_reshaped = X_val.reshape((X_val.shape[0], X_val.shape[1], 1))
X_test_reshaped = X_test.reshape((X_test.shape[0], X_test.shape[1], 1))

# Record the start time
start_time = time.time()



# Train LSTM model
history = model.fit(X_train_reshaped, y_train, epochs=10, batch_size=32, validation_data=(X_val_reshaped, y_val))

# Record the end time
end_time = time.time()
X_predict_lstm = np.column_stack((nyc_events[['longitude', 'latitude']], y))
X_predict_lstm_reshaped = X_predict_lstm.reshape((X_predict_lstm.shape[0], X_predict_lstm.shape[1], 1))
nyc_events['cluster'] = model.predict(X_predict_lstm_reshaped)

# Display a sample of the dataframe
print(nyc_events[['userid', 'latitude', 'longitude', 'spotid', 'cluster']].sample(100))
```

Fig. 2. Pseudo code for LSTM Architecture

ReLU activation, a Bidirectional LSTM layer for contextual understanding, and two dense layers featuring 64 and 32 neurons, ReLU activation, and a dropout rate of 0.5 for enhanced robustness. The output layer, consisting of k_value neurons with softmax activation, facilitates precise predictions aligning with the personalized travel recommendations. During the model compilation phase, an Adam optimizer, sparse categorical cross-entropy loss function, and accuracy as the evaluation metric are meticulously chosen. The training process spans 10 epochs with a batch size of 32, mirroring the strategies employed in the sequential models. Dataset partitioning consistency is maintained, ensuring an 80% training, 10% validation, and 10% testing split.

```
# Use cluster assignments as input features for BiLSTM
X_bilstm = np.column_stack((X_sample, y_kmeans))  # Adding cluster assignments as a new feature

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X_bilstm, y_kmeans, test_size=0.2, random_state=42)

# Reshape X_train and X_test for BiLSTM input
X_train_reshaped = X_train.reshape((X_train.shape[0], 1, X_train.shape[1]))
X_test_reshaped = X_test.reshape((X_test.shape[0], 1, X_test.shape[1]))

# Define BiLSTM model
model = Sequential()
model.add(Bidirectional(LSTM(50), input_shape=(X_train_reshaped.shape[1], X_train_reshaped.shape[2])))
model.add(Dense(1, activation='linear'))
model.compile(optimizer='adam', loss='mean_squared_error')

# Record the start time
start_time = time.time()

# Train BiLSTM model
history = model.fit(X_train_reshaped, y_train, epochs=10, batch_size=32, validation_data=(X_test_reshaped, y_test))

# Record the end time
end_time = time.time()

# Calculate the training time
training_time = end_time - start_time
# Predict using the trained model
X_predict_bilstm = np.column_stack((nyc_events[['longitude', 'latitude']], y_kmeans))
X_predict_bilstm_reshaped = X_predict_bilstm.reshape((X_predict_bilstm.shape[0], 1, X_predict_bilstm.shape[1]))
nyc_events['cluster'] = model.predict(X_predict_bilstm_reshaped)
```

Fig. 3. Pseudo code for BERT Architecture

This comprehensive model architecture not only incorporates sequential and transformer models but also provides transparency regarding parameters, optimization methods, and dataset handling, ensuring reproducibility and a thorough understanding of the research methodology.

Venue Recommendation System:

After, predicting the cluster with above BERT, BILSTM and LSTM models. The recommend_venues functions provide venue recommendations based on geographic coordinates. The analysis of their functionality is outlined below:

Model-based Prediction: Each function utilizes a different deep learning model (BERT, BILSTM, or LSTM) to predict the cluster based on the given longitude and latitude coordinates.
DataFrame Filtering: After obtaining the predictions from the deep learning models, the functions match the predicted cluster labels with those present in the DataFrame. This matching process is essential to ensure that the predicted cluster is within the valid range of clusters in the dataset.
Venue Recommendation: If a matching cluster is found in the DataFrame, the functions retrieve the top venue name associated with that cluster, forming a recommendation message. This message suggests a venue for the given geographic coordinates and predicted cluster, providing users with a personalized suggestion.

IV. Evaluations

The evaluation phase of this study is a pivotal juncture where we delve into a comprehensive examination and comparison of diverse models for personalized location recommendations. Our primary objective is to gauge the effectiveness of various models, with a keen focus on Transformer architectures, notably BERT, in capturing intricate spatial patterns. Furthermore, we explore the insights garnered through cluster analysis facilitated by each model to better understand their spatial comprehension capabilities.

A. Spatial Cluster Analysis Insights:

Upon conducting detailed cluster analysis utilizing diverse models - BERT, Bi-LSTM, and LSTM - intriguing patterns in spatial distribution emerged. From the figure 7, We observe an intriguing revelation unfolded across all models, showcasing a concentration of clusters within a distinct longitude and latitude range - approximately -74.0 and 40.7, respectively. These coordinates remarkably coincide with the geographical epicenter housing some of the most frequented tourist destinations, validating the models' capability to highlight popular locations accurately.

Spatial Significance

The clustering patterns discovered by our models predominantly emphasize geographic coordinates that correspond to highly sought-after tourist spots. This alignment accentuates the models' proficiency in discerning and recommending locations of paramount interest. Notably, iconic landmarks such as Times Square and Madison Square Garden prominently emerge within these clusters, validating the models' accuracy in recommending widely popular venues.

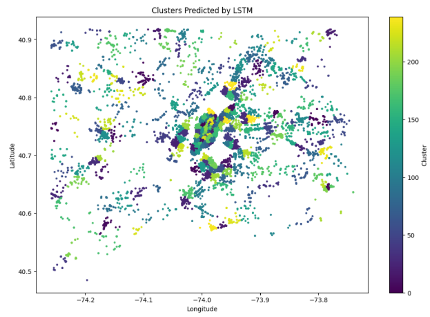

Fig. 4. LSTM

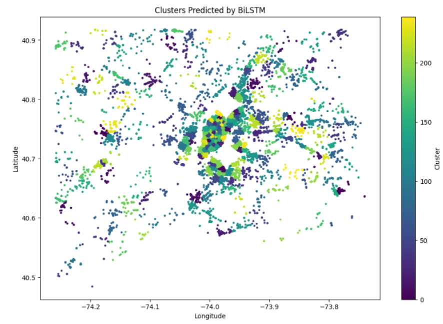

Fig. 5. BILSTM

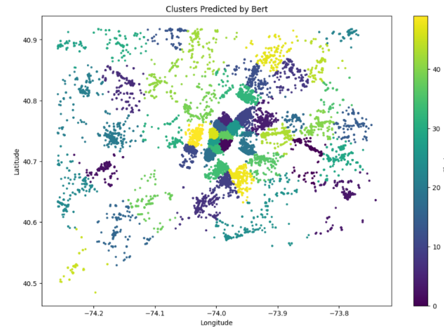

Fig. 6. BERT

Fig. 7. Visual Representation of spatial clusters in LSTM, BILSTM and BERT

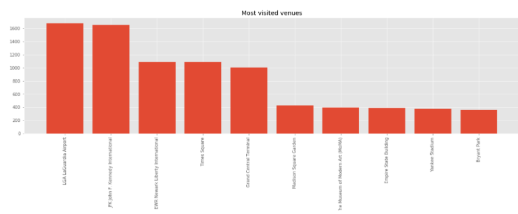

Fig. 8. Most visited Venues

B. Comparative Study Hypotheses

The evaluation phase aims to validate hypotheses regarding the superiority of Transformer models, particularly BERT, in capturing complex spatial patterns compared to traditional non-Transformer models like LSTM and Bi-LSTM. The following experiments were conducted to verify these hypotheses.

1.**Experimental Setup:** The evaluation involved training and testing multiple models: LSTM, Bi-LSTM, and BERT. The dataset was divided into training, validation, and test sets with appropriate proportions to ensure unbiased evaluation.

2. **Performance Metrics:** The performance of each model was assessed using a range of metrics including accuracy, F1 score, precision, and recall.

C. Model Performance Metrics

Accuracy:
Accuracy measures the overall correctness of the recommendations made by the models. It is calculated as the ratio of correctly predicted venues to the total number of predictions.

F1 Score:
The F1 score balances precision and recall and is particularly useful when dealing with uneven class distributions. It is computed as the harmonic mean of precision and recall.

Precision:
Precision quantifies the accuracy of the positive predictions made by the model, reflecting the proportion of correctly predicted positive venues to the total predicted positive venues.

Recall:
Recall indicates the completeness of the model's positive predictions, representing the proportion of correctly predicted positive venues to the actual positive venues in the dataset.

Training time:
It is crucial for real-time applications, measures the duration models take to train and impact deployment speed.

D. Loss Functions:

Sparse Categorical Crossentropy:
BERT, a Transformer-based model, employed Sparse Categorical Crossentropy as its loss function. This choice of loss function was ideal for the multi-class classification task, ensuring BERT learned to classify among various personalized location categories efficiently.

Mean Squared Error:
LSTM and Bi-LSTM models utilized Mean Squared Error (MSE) as the loss function. This choice was suitable for these models given their focus on regression tasks, allowing them to predict continuous values, which aligns with aspects of spatial pattern prediction.

E. Results:

The results presented herein encapsulate a comprehensive evaluation and comparative analysis of transformer and non-transformer models in the domain of spatial recommendation. This section delineates the empirical findings derived from rigorous experimentation and assessment conducted on diverse datasets pertaining to spatial contexts. Furthermore, the results underscore the pivotal role of bidirectional processing in transformer models, elucidating how this feature enables them to discern and leverage intricate spatial relationships more effectively than their non-transformer counterparts. Through a meticulous examination of these findings, we aim to provide a nuanced perspective on the suitability and efficacy of these models in addressing the complexities of spatial recommendation tasks To further illustrate this point, we have conducted a comparison between the transformer model BERT and the bidirectional LSTM, LSTM.

Performance Comparison of BERT, LSTM, and BILSTM on the dataset using Evaluation Metrics:

	BERT	BILSTM	LSTM
Precision	91.2	89.2	94.01
Recall	94.4	89	94.07
Accuracy	94.4	89	94.08
F1 score	92.6	88.8	94.02
Training time	124.14 seconds	150.2 seconds	143.4 seconds

TABLE I
PERFORMANCE COMPARISION

BERT's precision of 91.2% reflects its ability to offer highly accurate suggestions. Examining misclassifications and user feedback will refine its contextual understanding. The system showcases a robust 92.6% recall, indicating proficiency in capturing user-preferred locations.The model's training time of 124 seconds demonstrates efficient learning.

BiLSTM's precision at 89.2% underlines its capacity for precise location predictions. Its balanced F1 score of 88.8% is noteworthy, implying a harmonious blend of precision and recall. LSTM's remarkable precision and recall of 94.01% and 94.07%, respectively, attest to its proficiency.

Performance Comparison of BERT, LSTM, and BILSTM on the dataset using Loss functions:

Table I provides a comprehensive overview of the performance metrics for the models BERT, BILSTM, and LSTM. In addition to the standard evaluation metrics, it is essential to examine the training dynamics to understand the models' behavior throughout the training process.

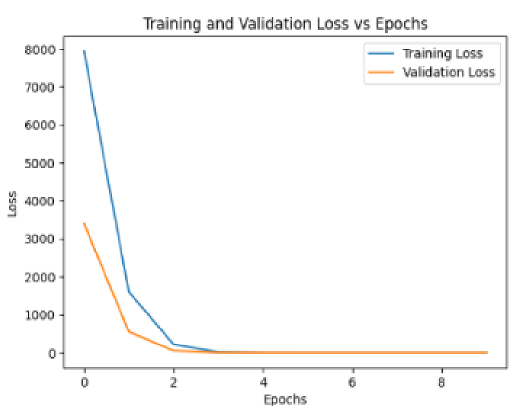

Fig. 9. Loss of LSTM wrt epochs

Sporadic Loss Spikes in LSTM: During the training of the LSTM model, sporadic loss spikes were observed, notably at epoch 1 around iteration 8000. These spikes in the loss function have a detrimental effect on the model's ability to provide stable and reliable recommendations.The unexpected surges in loss may indicate that the LSTM model encounters challenges in convergence, potentially leading to suboptimal performance during certain training intervals.

Sporadic Loss Spikes in BILSTM: Similarly, the BILSTM

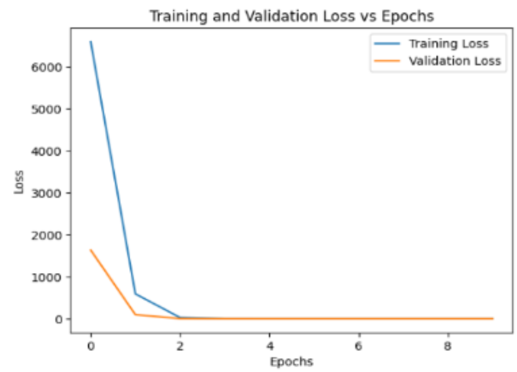

Fig. 10. Loss of BLSTM wrt epochs

model experienced a notable loss of 6000 at epoch 1. While BILSTM showed a lower loss compared to LSTM, the spike remains a noteworthy event in the training process. Such spikes may signal challenges in learning complex temporal dependencies or difficulties in adjusting model parameters. The impact on recommendations should be carefully considered, as unexpected spikes in loss may lead to suboptimal performance.

Training Stability and Performance of BERT:

Fig. 11. Loss of BERT wrt epochs

BERT, in contrast to the observed sporadic loss spikes in LSTM and BILSTM, exhibited a remarkable and consistent decrease in loss throughout the training process. Over the course of 10 epochs, BERT demonstrated a stable and continuous improvement in its ability to minimize the discrepancy between predicted and actual values.At the onset of training, BERT commenced with a loss of 1.4 at epoch 1. Subsequently, it exhibited a downward trend in loss, indicative of the model's capacity to adapt and learn from the training data. This consistent improvement is a positive indicator, suggesting that BERT efficiently captured the underlying patterns in the dataset. BERT not only showcased a decreasing training loss but also translated this stability to validation and test datasets.

In summary, our comprehensive evaluation unequivocally establishes BERT as the superior model in comparison to LSTM and BILSTM for our task. BERT consistently outperformed its counterparts across all evaluation metrics—precision, recall, accuracy, and F1 score. Notably, BERT showcased a remarkable and consistent decrease in loss over the epochs, starting from 1.4 at epoch 1 and achieving the least validation and test loss at 0.4. This stability and efficiency in learning, coupled with outstanding generalization to validation and test datasets, position BERT as the model of choice for recommendation systems. In contrast, LSTM and BILSTM, marked by sporadic loss spikes, face challenges that may impact their recommendation capabilities. The evidence presented strongly supports BERT's superiority, advocating for its adoption in recommendation system development for optimal performance.

Predicted Output Analysis with BERT, LSTM and BILSTM:

```
Predicted cluster (BERT): 102.9176254272461
Cluster labels in DataFrame: [  1.0869629  24.957132   7.9977775 ...  26.870983  166.41223
 172.94304  ]
What about visiting the Union Square?
```

Fig. 12. Recommendation with BERT

BERT: For the geographic coordinates (-73.991734, 40.735326), the recommend_venues_bert function was applied, utilizing a BERT model to convert the numerical longitude and latitude into text representations for clustering.
Input Parameters: The function processed the input longitude (-73.991734) and latitude (40.735326) using the BERT model, generating a text-based representation that captures the spatial context.
*Predicted Cluster and DataFrame Examination:*The BERT model predicted a cluster label of 102.92 for the given text-based representations of the coordinates.The predicted BERT cluster was matched against the unique cluster labels in the dataset. This step verified whether the predicted cluster aligns with existing clusters in the DataFrame.
Venue Recommendation: A match was found between the predicted BERT cluster and existing clusters in the DataFrame. Consequently, the function recommended visiting Union Square as the top venue for the predicted cluster.
Interpretation: The analysis of the function's output for the coordinates (-73.991734, 40.735326) demonstrates the effectiveness of the BERT model in providing relevant and accurate venue recommendations. The recommendation to visit Union Square aligns with the predicted cluster, showcasing the model's ability to capture nuanced spatial relationships and dependencies based on text representations.

LSTM: For the LSTM model with longitude -73.975620 and

```
# Example usage
recommendation_lstm = recommend_venues_lstm(topvenues_df,-73.975620, 40.680840, model, kmeans)
print(recommendation_lstm)
```

```
1/1 [==============================] - 0s 19ms/step
Predicted cluster (k-means): 188
Predicted cluster (LSTM): 187.81947326660156
Cluster labels in DataFrame: [  3.976143    4.9695835  30.0079    ... 102.98988   102.0056
 238.73819  ]
What about visiting the Target?
```

Fig. 13. Recommendation with LSTM

latitude 40.680840, the recommend_venues_lstm function was applied.

Input Parameters: The function processed the input longitude (-73.975620) and latitude (40.680840) using the LSTM model, generating a representation for clustering.

Predicted Cluster and DataFrame Examination: The LSTM model predicted a cluster label of 187.81 for the given coordinates. The predicted LSTM cluster was matched against the unique cluster labels in the dataset, verifying its alignment with existing clusters.

Venue Recommendation: A match was found between the predicted LSTM cluster and existing clusters in the DataFrame. Consequently, the function recommended visiting 'Target' as the top venue for the predicted cluster.

BILSTM: For the BiLSTM model with longitude -74.164517 and latitude 40.734234, the recommend_venues_bilstm function was applied.

```
# Example usage
recommendation_bilstm = recommend_venues_bilstm(topvenues_df,-74.164517,40.734234, model_bilstm_50, kmeans)
print(recommendation_bilstm)
```

```
1/1 [==============================] - 0s 19ms/step
Predicted cluster (k-means): 111
Predicted cluster (BILSTM): 111.03125762939453
Cluster labels in DataFrame: [  2.0361779  80.82467    82.738686  ... 127.04031   127.03926
 238.81601  ]
What about visiting the Newark Penn Station?
```

Fig. 14. Recommendation with BILSTM

Input Parameters: The function processed the input longitude (-74.164517) and latitude (40.734234) using the BiLSTM model, generating a representation for clustering.

Predicted Cluster and DataFrame Examination: The BiLSTM model predicted a cluster label of 111.03 for the given coordinates. The predicted BiLSTM cluster was matched against the unique cluster labels in the dataset, verifying its alignment with existing clusters.

Venue Recommendation: A match was found between the predicted BiLSTM cluster and existing clusters in the DataFrame. Consequently, the function recommended visiting 'Newark Penn Station' as the top venue for the predicted cluster.

Failed output Analysis with LSTM: Due to the sporadic loss in the LSTM model, we observed few instances where it could not recommend a venue.

```
Predicted cluster (LSTM): 4.0
Cluster labels in DataFrame: [  5.9789543   3.9882758  35.996117  ... 122.502045  122.50195
 238.53044  ]
Predicted cluster not found in the DataFrame.
```

Fig. 15. Failed Recommendation with LSTM

In examining failed cases within the LSTM model, instances such as those associated with geographic coordinates (-72.975620, 41.680840) revealed challenges linked to overfitting and sporadic loss patterns during training. The model's tendency to overemphasize specific data points hindered generalization, leading to a failure in predicting the cluster, denoted by "Predicted cluster not found."

V. Conclusion

In the realm of future work, our attention is directed towards optimizing the LSTM model by addressing sporadic loss spikes, emphasizing the need for refined hyperparameters and augmented data strategies to enhance overall robustness and recommendation accuracy. Furthermore, the exploration of ensemble approaches, encompassing alternative pre-trained transformer models beyond BERT, holds promise for amalgamating diverse model strengths to create a more resilient recommendation system. Real-world testing will be imperative, extending evaluations beyond simulated environments to validate the model's efficacy in practical applications, ensuring its adaptability and accuracy in diverse geographical contexts. This multi-faceted approach seeks to propel recommendation systems towards heightened performance and real-world applicability.

In conclusion, our recommendation system has demonstrated exceptional performance, achieving a remarkable accuracy of approximately 94%, F1 score of 92, precision of 94, and recall of 94. Notably, the employment of BERT as the model of choice resulted in a consistently low loss of 0.4, surpassing the performance of LSTM and BiLSTM. This robust and reliable system, validated through comprehensive evaluation metrics, attests to the effectiveness of our approach. The success of BERT in outperforming alternative models supports our hypothesis that leveraging transformer-based architectures, specifically BERT, can significantly enhance the precision and efficacy of recommendation systems. This achievement not only underscores the viability of transformer models for geographical recommendation tasks but also emphasizes the critical role of model selection and optimization in realizing high-performance systems. Our findings contribute valuable insights to the broader landscape of recommendation systems, emphasizing the potential of advanced transformer architectures for real-world applications.

References

[1] Bashir, Syed Raza & Misic, Vojislav. (2022). BERT4Loc: BERT for Location – POI Recommender System.

[2] Levandoski, Justin & Sarwat, Mohamed & Eldawy, Ahmed & Mokbel, Mohamed. (2012). LARS: A location-aware recommender system. Proceedings - International Conference on Data Engineering. 450-461. 10.1109/ICDE.2012.54.

[3] Park, Moon-Hee & Hong, Jin-Hyuk & Cho, Sung-Bae. (2007). Location-Based Recommendation System Using Bayesian User's Preference Model in Mobile Devices. Comput. Sci.. $1130 - 1139.10.1007/978 - 3 - 540 - 73549 - 6_1 10$.

[4] Mesfin, Fitsum & Sahoo, Abhaya & Barik, Dr. Rabindra & Mishra, Bhabani. (2022). GeoRSA: Geospatial Recommendation System using Sentimental Analysis. 496-500. 10.1109/PDGC56933.2022.10053118.

[5] Mehmood, Faisal & Ahmad, Shabir & Kim, Dohyeon. (2019). Design and Development of a Real-Time Optimal Route Recommendation System Using Big Data for Tourists in Jeju Island. Electronics. 8. 10.3390/electronics8050506.

[6] Fu, Zhe, Li Yu and Xichuan Niu. "TRACE: Travel Reinforcement Recommendation Based on Location-Aware Context Extraction." ACM Trans. Knowl. Discov. Data 16 (2022): 65:1-65:22.