Recommender Systems based on Reviews and Product Interactions

Tianxiang Chen, Phu Duong

***Abstract*—Recommender systems, a machine learning application, show the potential categories of products that customers are potentially interested in. These can be based on reviews, rating and past purchases. By focusing on the interconnected users and products, this project aims to gain insights into product popularity and consumer behavior patterns. The proposed work involves data collection, preprocessing, K core decomposition, and the application of machine learning algorithms to achieve recommendation outcomes.**

I. Data overview

We will use the 2014 Amazon review dataset released by Jianmo at UCSD. The focus will be on the 5-core dataset, where each user and item included has at least 5 reviews. This subset consists of 75.26 million data points, formatted as one-review-per-line in JSON, and is divided into 29 smaller category datasets.

A. Data Description

The dataset comprises several fields that are potentially relevant for analysis and the development of the recommendation system.

Field	Description	Type
reviewerID	Amazon generated user id	Text
asin	Unique Amazon generated product id	Identifier
overall	User rating (1-5)	Numerical
verified	Is the review verified (true, false)	Boolean
reviewTime	Time of the review (raw)	Date
style	A dictionary of the product metadata	Text
reviewerName	Name of the reviewer	Text
summary	Summary of the review	Text
reviewer Text	Text of the review	Text

Table 1: Dataset Features

B. Key Questions and limitation

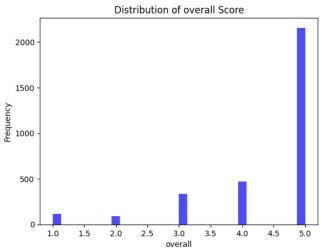
Using we hope to ask question about the effect of 1 Sentiment Analysis: which product is good 2 What is the next recommended product

Our dataset is limited to user reviews and ratings. While it includes unique Amazon product IDs, it lacks product names and descriptions.

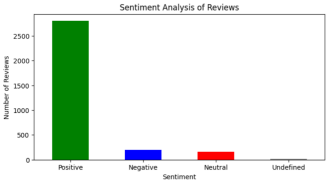
The model is limited and may be interpreted together with domain knowledge. TextBlob Sentiment Analysis: approach provides a straightforward method for sentiment classification but does not always capture the complexities and nuances of human language, such as sarcasm or contextual meaning.

II. Text Classfication

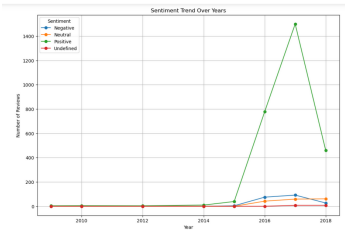
1) Distribution of Overall score: The Overall Score histogram provides information about the distribution of an 'Overall' rating score. The mean of Overall score for our data is 4.40 , which implies most ratings are very good . Most products being evaluated generally perform very well, or meet a high standard of expectation.



2) Sentiment Analysis: Next, we utilized the TextBlob to perform sentiment analysis on a dataset containing textual reviews. The reason for choosing Textblob is quite effective for broad applications. This Sentiment Analysis uses a pre-trained classifier model that applies a bag-of-words approach. It assesses the polarity and subjectivity of a text where:Polarity is a float within the range [-1.0, 1.0], where -1.0 means negative sentiment and 1.0 means positive sentiment. For visualization, a bar graph was generated to visualize the counts of each sentiment category.



The chart titled "Sentiment Analysis of Reviews" displays the distribution of sentiments categorized into four types: Positive, Negative, Neutral, and Undefined. Each column presents the number of reviews corresponding to each sentiment type. The majority of reviews are classified as Positive, with the count exceeding 2800 reviews. This can be seen as a strong positive signal for the Amazon product line. The relatively low occurrence of Negative and Neutral reviews suggests that there are fewer concerns or issues being raised by users. A significantly smaller number of Negative reviews, around 200, suggesting some criticism among a minority of reviewers. There is a very small number of Neutral reviews, appearing to be around 150, indicating that few reviews express a middle-of-the-road viewpoint



The graph titled "Sentiment Trend Over Years" tracks the number of reviews classified by sentiment from 2010 to 2018, with sentiments labeled as "Negative," "Neutral," "Positive," and "Undefined." Notably, the "Positive" sentiment peaks sharply in 2017, suggesting a highly favorable response to a product or event, before dramatically declining by 2018. "Neutral" sentiment shows a steady increase over the years, while "Negative" and "Undefined" sentiments remain consistently low throughout the period. This trend indicates a predominantly positive reception with few fluctuations in extreme negative or undefined responses. Understanding the factors behind the 2017 peak could provide insights into customer preferences or successful marketing strategies.

In conclusion, the review sentiment and overall rating score result provide consistent results. For most products, customers generally express satisfaction, as evidenced by the overwhelming dominance of positive sentiments, particularly around 2017. This suggests a successful period likely driven by specific factors that resonated well with customers. From a business perspective, they can maintain the high standards that lead to positive reviews and investigate the negative reviews to identify specific areas for improvement. Decide which product line is the best Overall, the sentiment analysis results provide useful insights into the perceptions of the reviewers. From there, the business department can highlight the strong product line and reveal the negative product. As a result, these pieces of information will lead to understanding of user concerns and having the right business strategies.

III. Recommendation System

A. Sequence preparation

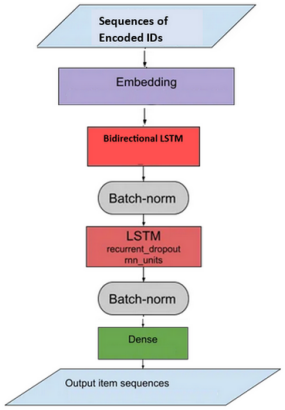
First, we performed tokenization and encoding of asin and reviewerID, then we transformed asin and reviewerID from String into Numeric.Next, we grouped the dataset by $reviewerID_encoded$ and created the next sub-sequence and corresponding labels.

B. Create Sequence and Labels

It uses a sliding window of size window size (default is 3) to create input sequences and corresponding labels. Each sequence consists of window size items, and the label for each sequence is the item that immediately follows it. This setup is typical for prediction tasks where the goal is to predict the next item in a sequence.

C. Build Recommendation System

The diagram you provided represents a neural network architecture, for sequence processing, likely used for tasks like language modeling, sequence prediction.



Recommendation System Architecture

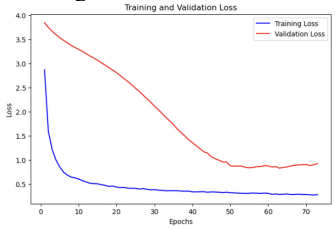
Here's a breakdown of each component in the architecture:

- Embedding Layer
  - Function: Converts integer representations of items (like word indices) into dense vectors of a fixed size.
  - Purpose: To create meaningful representations of each item that capture more information than the integer encoding, enabling the model to understand and process items in a more nuanced way.
- Bidirectional LSTM Layer
  - Function: Applies an LSTM (Long Short-Term Memory) layer in both forward and backward directions on the input sequence.
  - Purpose: To capture dependencies in both directions (past and future context) of the input sequence, enhancing the model's ability to learn from the entire sequence for more accurate predictions.
- Dropout Layer
  - Function: Randomly sets a fraction of the input units to 0 at each update during training, which is specified by the dropout rate.
  - Purpose: To prevent overfitting by reducing the model's reliance on any particular set of features.
- Batch Normalization Layer
  - Function: Normalizes the activations of the previous layer, adjusting and scaling the outputs.
  - Purpose: To stabilize and accelerate the learning process by maintaining the mean output close to 0 and the output standard deviation close to 1, which can also lead to using higher learning rates effectively.
- LSTM Layer
  - Function: Processes the sequence data using LSTM units, but only in one direction (typically forward).
  - Purpose: To capture temporal dependencies within the data, leveraging the sequence's forward context to make predictions.
- Dropout Layer (second instance)
  - Function: Similar to the first Dropout layer, it helps in preventing overfitting by randomly omitting features during training.
  - Purpose: To ensure the model does not overfit especially after learning additional complexities from the second LSTM layer.

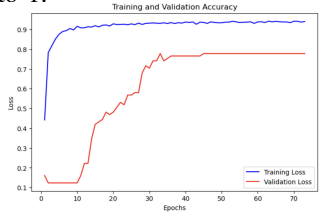
- Batch Normalization Layer
  - Function: Continues to normalize the output from the second LSTM layer.
  - Purpose: To maintain effective learning conditions, ensuring the activations do not diverge too much during training, which helps in maintaining overall network performance.
- TimeDistributed Dense Layer
  - Function: Applies a Dense layer to every temporal slice of the input data. It uses the softmax activation function to produce a probability distribution over the output classes for each time step.
  - Purpose: To make predictions at each time step, allowing the model to output a sequence of predictions that correspond in size and time alignment to the input sequence.
- Hyper-Parameter set up Loss function: sparse categorical crossentropy optimizer: Adam with learning rate=0.001 dropout rate = 0.3 embedding dim = 64 (The size of the vector space in which words will be embedded) rnn units = 128 (The number of units in each LSTM layer.) batch size = 128 num items = The max value of asin encoded +1(The maximum number of unique items that will be encoded using the embedding layer.)

IV. Evaluation

Implements a K-Fold Cross-Validation approach to train a machine learning model,specifically designed for sequence data. The dataset is divided into 5 folds, and the model is trained on each fold separately to validate its performance. To evaluate the model we plot the loss and Accuracy for both Training and Validation data.



As you can see both Training and Validation loss converge where Training converges to 0.5 and Validation loss converges to 1.



Same as the Accuracy plot, both Training and Validation dataset converge where Training converges to 0.9 and Validation loss converges to 0.8. And based on the Accuracy plot I think my model might have a small underfit problem since there is a 0.1 gap between training and validation.

V. Prediction

Once we fit our model,we can start to ask it to generate the Recommend item based on the user's purchase history. Given the user history, provided the asin from the recommender system. For instance, input sequence is ["B000K2PJ4K", "B005AGO4LU", "B016XAJLVO", "B009MA34NY", "B001LNSY2Q"]. Then the output will include the 3 Recommend items, the average overall score of those Recommend items. The result shows below:

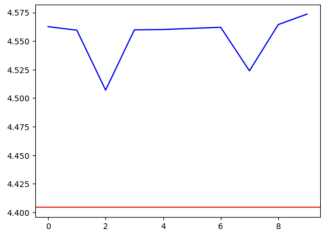
```
[ 0 15 29 17 12]
1/1 [==============================] - 0s 11ms/step
Recommended ASINs: ['B010RRWKT4' 'B0092UF54A' 'B014IBJKNO']
4.5625625625625625
```

Since the purchase behavior pattern has randomness in it, the result from the algorithm is hard to say whether it is correct or not. So how to evaluate the Recommended items from the algorithm? Then we decided to compare the average overall score of Recommend items with the average overall score of all dataset which is 4.40.

Assume We have 100 user's history purchase records, then compare the Average Overall Score of Recommended ASINs with the mean of Overall score for our data. The result shows below:

x-axis: user id

y-axis: average of overall score



The blue line represents the average overall score of Recommend items for those 100 users. Red line represents the average overall score for our data. And as you can see that each point in the blue line is above the red line. That means our algorithm can generate the good Recommend items for our customer.

VI. References

Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C. (2003). A Neural Probabilistic Language Model. Journal of Machine Learning Research, 3, 1137-1155.

Yang, X., Guo, Y., Liu, Y., Steck, H. (2019). A Graph-based Recommendation across Heterogeneous Domains. In Proceedings of the 13th ACM Conference on Recommender Systems (RecSys '19). Association for Computing Machinery, New York, NY, USA, 299–307.

Jianmo Ni, Jiacheng Li, Julian McAuley Empirical Methods in Natural Language Processing (EMNLP), 2019