VIETNAM NATIONAL UNIVERSITY OF HOCHIMINH CITY

THE INTERNATIONAL UNIVERSITY

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

A picture containing text, scene, room, gambling house

Description automatically generated

**TOPIC MODELING-BASED RECOMMENDER SYSTEMS FOR E-COMMERCE OF COSMETIC PRODUCTS**

By

Hồ Đặng Phương Ngọc

A thesis submitted to the School of Computer Science and Engineering

In partial fulfillment of the requirements for the degree of

Bachelor of Computer Science

Ho Chi Minh City, Viet Nam

2020-2021

**TOPIC MODELING-BASED RECOMMENDER SYSTEMS FOR E-COMMERCE OF COSMETIC PRODUCTS**

APPROVED BY:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_,

Nguyễn Thị Thanh Sang, Ph.D, Supervisor

# **TABLE OF CONTENTS**

[TABLE OF CONTENTS 2](#_Toc78037090)

[ACKNOWLEDGEMENT 3](#_Toc78037091)

[LIST OF TABLES 4](#_Toc78037092)

[ABBREVIATION 6](#_Toc78037093)

[ABSTRACT 7](#_Toc78037094)

[CHAPTER 1: INTRODUCTION 8](#_Toc78037095)

[1.1. Background 8](#_Toc78037096)

[1.2. Problem statement 10](#_Toc78037097)

[1.3. Scope and Objectives 11](#_Toc78037098)

[1.4. Assumptions and Solution 12](#_Toc78037099)

[1.5. Structure of Thesis 12](#_Toc78037100)

[CHAPTER 2: LITURATURE REVIEW 13](#_Toc78037101)

[2.1. Recommendation System 13](#_Toc78037102)

[2.2. Related research on Topic Modelling Models 16](#_Toc78037103)

[2.3. Related Research on Sequential- based Models, Attention- based Models 22](#_Toc78037104)

[CHAPTER 3: METHODOLOGY AND IMPLEMENTATION 30](#_Toc78037105)

[3.1. Methodology 30](#_Toc78037106)

[3.1.1. OVERVIEW FRAMEWORK 30](#_Toc78037107)

[3.1.2. OFFLINE TRAINING OF LDA 32](#_Toc78037108)

[3.1.3. OFFLINE TRAINING OF TISASREC 36](#_Toc78037109)

[3.1.4. ONLINE RECOMMENDATION ENGINE 37](#_Toc78037110)

[3.2. Implementation 38](#_Toc78037111)

[CHAPTER 4: EXPERIMENTAL RESULTS AND EVALUATION 52](#_Toc78037112)

[4.1. Experimental Results 52](#_Toc78037113)

[4.2. Evaluation 58](#_Toc78037114)

[CHAPTER 5: CONCLUSIONS AND FUTURE WORKS 63](#_Toc78037115)

[5.1. Conclusions 63](#_Toc78037116)

[5.2. Future works 63](#_Toc78037117)

[REFERENCES 65](#_Toc78037118)

# **ACKNOWLEDGEMENT**

To begin with, I would like to express my sincere gratefulness as well as admiration to ***Dr. Nguyễn Thị Thanh Sang*** for her supervise within the Thesis process. I am very honored to be instructed by such a devoted professor who always advocates me whenever I am in need. She has provided me with many helpful methods and techniques so that I can form the initial idea of the system’s appearance as well as behaviors. From there, she has recommended me new improvements that I may apply into my system for utilization.

Secondly, I would like to express my special thankfulness to the Computer and Engineering Department of International University for having given me the golden opportunity to conduct my own Thesis which helps me to approach a considerable amount of new knowledge and expand an insightful technical view. I sincerely want to express my appreciation the members of my reading and examination committee.

Finally, I would like to deliver my wholehearted gratefulness to my parents for fulfill me with their enormous mental encouragement and unconditional love so that I can have adequate courage to complete my thesis.

# **LIST OF TABLES**

[Table 2.2. Notation and definition for LDA generative model 20](#_Toc76063859)

[Table 2.3. Notation and Description of TiSASRec formulas 26](#_Toc76063860)

[Table 3.1. Parameters and Corresponding Description for genism LDA Mallet 44](#_Toc76063861)

[Table 4.1. NDCG @10 and HR @10 of LDA- TiSASRec Model 57](#_Toc76063862)

[Table 4.2. NDCG @10 and HR @10 of LDA-TiSASRec, original TiSASRec and original SASR 60](#_Toc76063863)

**LIST OF FIGURES**

[Figure 2.1. Decomposition of document-term matrix by LSA [40]Table 2.1. Notation and definition for LSA decomposition formula 16](#_Toc76063877)

[Figure 2.2. High level view of PLSA [Alghamdi et al., 2015] 17](#_Toc76063878)

[Figure 2.3. Representation of LDA generative model [10] 19](#_Toc76063879)

[Figure 2.4. Overall Framework of TiSASRec 25](#_Toc76063880)

[Figure 3.1. The Proposed Model- Integration of LDA and TiSASRec 31](#_Toc76063881)

[Figure 3.2. Offline training for LDA 33](#_Toc76063882)

[Figure 3.3. Example of a part of Dictionary including key- location in dictionary(left) and value- word(right) 35](#_Toc76063883)

[Figure 3.4. Example of one review in bag- of- words representation 35](#_Toc76063884)

[Figure 3.5. Online Recommendation Engine of LDA- TiSASRec model 37](#_Toc76063885)

[Figure 3.6-a. Raw Data provided in Amazon Beauty Reviews (AWS) 39](#_Toc76063886)

[Figure 3.6-b. Raw Data provided in Amazon Beauty Reviews (AWS) 39](#_Toc76063887)

[Figure 3.8. Sample required data for LDA offline training process required fields. 41](#_Toc76063888)

[Figure 3.9. Sample input into TiSASRec Model 41](#_Toc76063889)

[Figure 3.10. Main Steps for pre- processing texts 42](#_Toc76063890)

[Figure 3.11. Implementation for creating corpus and dictionary 43](#_Toc76063891)

[Figure 3.13. Function to create user average topic vector list 45](#_Toc76063892)

[Figure 3.14. Function to find similar users 46](#_Toc76063893)

[Figure 3.15. Example of the similar user file. 46](#_Toc76063894)

[Figure 3.16-a. Weight computation Layers Implementation 47](#_Toc76063895)

[Figure 3.16-b. Weight computation Layers Implementation 47](#_Toc76063896)

[Figure 3.16-c. Weight computation Layers Implementation 48](#_Toc76063897)

[Figure 3.16-d. Weight computation Layers Implementation 48](#_Toc76063898)

[Figure 3.17. Output of TiSASRec Training Process 48](#_Toc76063899)

[Figure 3.18. Prediction Function for scoring item 49](#_Toc76063900)

[Figure 3.19. Loading Scoring mechanism from TiSASRec 49](#_Toc76063901)

[Figure 3.19- a. Getting ranking item list 50](#_Toc76063902)

[Figure 3.19- b. Getting ranking item list 50](#_Toc76063903)

[Figure 3.20. Scoring, ranking and outputting recommendations 51](#_Toc76063904)

[Figure 4.1. LDA Grid Search Implementation 52](#_Toc76063905)

[Figure 4.2. Grid Search results on LDA original model base 53](#_Toc76063906)

[Figure 4.3. Overall visualization of LDA Mallet performance 54](#_Toc76063907)

[Figure 4.4. Overall visualization of LDA Mallet term- topic relation for topic 1 55](#_Toc76063908)

[Figure 4.5. LDA- TiSASRec Model NDCG @10 and HR@10 on evaluate and test data 57](#_Toc76063909)

[Figure 4.6. Original TiSASRec NDCG @10 and HR@10 59](#_Toc76063910)

[Figure 4.7. Original SASR NDCG @10 and HR@10 59](#_Toc76063911)

[Figure 4.8. Comparison HR@10 for TiSASRec\_LDA, TiSASRec and SASRec 60](#_Toc76063912)

# 

# **ABBREVIATION**

|  |  |
| --- | --- |
| Notation | Description |
| LDA | Linear Dirichlet Allocation |
| TiSASRec | Time- interval Self- Attention based Sequential Recommendation system |
| MK | Markov Chain |
| CNN | Convolutional Neural Networks |
| RNN | Recurrent Neural Networks |
| SA | Self- Attention |
| SASR | Self- Attention Sequential Recommendation |
| NDCG@10 | Normalized Discounted Cumulative Gain for top 10 |
| HR@10 | Hit Rate for top 10 |

# **ABSTRACT**

The more data, the less work we must do since machines are capable of dealing with those complicated and heavy loads of data. However, when it comes to businesses, the more seems to be the less since customers are not fond of overwhelming options. Therefore, multiple attempts on recommendation systems have been delivered, yet just a few utilized and realistic ones be genuinely applied. There are two noticeable techniques in the field to be mentioned. The first candidate is Topic Modelling, particularly the Linear Dirichlet Allocation Model (LDA), which is one of the “warriors” in Natural Language Processing [[1](#ref1)]. As a recommender engine, it usually takes customer reviews as input, outputs transparent classifications/ or opaque groups of topics that customers belong to, and then recommend products of users in similar group. Nevertheless, this method is not explicitly built for recommendation but more on grouping users with same preferences. The other method is the application of neural networks as a tool for capturing users’ s preferences. A particular figure of this is Time interval awareness Self-attention- based sequential recommendation (TiSASRec) [[2](#ref2)]. The main idea of TiSASRec is to make use of time intervals between user interactions in accordance with their sequential item frames for recommendation. However, the items chosen for ranking are unqualified. As illustrated, each aforementioned model is built individually and lack of completion. This research is expected to incorporate full advantages of each model to build a complete recommendation mechanism.

# **CHAPTER 1: INTRODUCTION**

***This session main goal is to deliver an overview introduction of research process, deliverables as well as a transparent explanation of each session.***

## **Background**

* + 1. **E- Commerce Recommendation**

The development of the Internet has offered human beings with significant improvements in life quality. Thanks to modern machine and technologies, daily tasks can be performed effortlessly. However, human beings are asking more and more of the modernity and automaticity. It is inevitable but it is not impossible.

Shopping is one of the of the essence needs at the current time. Nevertheless, we manage to spare time for shopping due to their out- of- stock condition causing time- consuming direct visits, or lack of product information overwhelming customers to place a try. Therefore, online shopping comes as a solution to mitigate the problems quite effectively.

In the meantime, online transaction is intensively competitive since there are excessive calls for new products and more products. Companies oppose vigorously to win customers’ s loyalty as well as to break their total revenue records. Recommendation system has been known as an optimal tool mitigating the situation. Thanks to those systems, customers can achieve their desirables with ease despite their little intention or unawareness of products.

* + 1. **Topic Modelling**

The process of collecting and evaluating text used to cope with many difficulties since it was performed manually, which costs a lot of time, money, and computation errors. Therefore, researchers alter their methodology to extract information from by manually by humans to automatically by computers.

The main idea of all related models in this field is to extract relation between documents (via words, contexts, etc.) for later- on classification/ clustering activities. Most models developed concentrating on explicitly or implicitly extract the topic within each input document, which has revealed interesting information. There is a dominant phrase when it comes to topic modelling which is topic extraction – the activity in which documents are found to be related in some semantic scenarios (or topics); then those topics are denoted and used for a post- model purpose such as clustering for clustering documents or to recommend related documents [[3](#ref3)]. Several topic modelling methods have been proposed for Paper and News Recommendations, which concentrates less attention to preferences but more to documents’ s content relation.

However, topic modelling itself when being applied into e- commerce cannot solve completely the Recommendation question: “*Which item is likely to be the next item?”* It can work as a tool for grouping users – since users have various tastes in different types of products or even their requirements when purchasing an item such as price, product categories, concerns, etc. In this research, it is applied as a grouping tool for extracting similar users to incorporates with another model to output a complete recommendation list.

* + 1. **Sequence- based Models and Attention Mechanism**

Not only reviews but also customers ‘s behaviors illustrate users’ s preferences. The main idea of sequence- based models is capturing user’ s behaviors through their interaction histories (clicks, views, likes, purchased items, reviews, etc.) and seeking for item(s) which may be the next item(s) user would like to interact with. Attention mechanism is an improvement of the aforementioned method since it can integrate the pros from many ancestral sequential methods and can work well in various data density. According to Wang-Cheng Kang et al. [[4](#ref4)], this does not only capture the “context” from recent behaviors of users to find suitable items but also takes in the “relevance” of the item to their interaction list, which means that this method exploits more in the acquaintance of new and old items supporting for recommendations.

The mentioned TiSASRec is a more advanced version of this method since it also takes in sequence of user’ s interaction records as input and output a suitable recommendation. It provides additional information as known as “time interval” between every two interactions of any user who is being considered. This time difference matters due to the difference time span for a particular sequence of item interactions of two different users. For example, both of users may have bought the item Olay Sunscreen and Tatcha Exfoliator; however, the time interval between those two records of the first user is a week, and that of the other one is a month. Although the positions of the two items are the same for both users, the time intervals are so different that they would get different recommendations.

Although the idea is adequate, TiSASRec is not absolutely efficient. There is another informative factor that should be taken into consideration in order to aid this recommendation engine which is review sent by online customers. The developed version up until now still input random products for rating and scoring based on user history records and deliver the product with the highest score as recommendation. To be specific, the product may be irrelevant and reduce the ability to deliver a high- scored product. Therefore, there must be a way to re- control the query at the prediction layer of this model raising the ability to predict a relevant product with as the maximum score possible.

## **Problem statement**

Linear Dirichlet Allocation (LDA) is more a grouping tool than a complete recommendation methodology, and Time interval Self- attention- based Sequential Recommendation (TiSASRec)’ s Recommendation mechanism has not been built completely; therefore, this model can be completed thanks to the integration of LDA results into TiSASRec at the prediction layer of TiSASRec which adds relevant items of similar users for score calculation and return final recommendations. As items adding into TiSASRec recommendation engine have strong relation to the current user, the prediction score can be raised considerably, and a proper output can be released. In this research, LDA is proved to support available recommendation engines in terms of targeting sequence of items within groups of users, not only that of the user himself or that resulted from randomized items.

## **Scope and Objectives**

This is an experimental research to complete the recommendation engine of the available Attention- based integrated with Sequential and Attention based model TiSASRec [[2](#ref2)] by the integration of Linear Dirichlet Allocation (LDA). This improved approach is expected to enhance the predicted score of the recommended item by the model due to the relevant relation of the item to the user preferences captured via LDA.

There are ***two goals*** to attain in the research:

* The input item bucket at the prediction layer can be optimized by relevant items extracted from LDA model, which means that output of the LDA model is the input aiding TiSASRec.
* The predicted score of the output item(s) (the number of items depends on the number of required recommendations from the model) can be upgraded to a higher level – the score(s) is expected to be higher which shows that the offer item(s) has more chance to reach the customer.

The two- model integration is conducted on

* Macbook Pro 2017 - 2.3 GHz Dual-Core Intel Core i5 with 8 GB RAM Memory.
* Google’s Colaboratory GPU runtime

## **Assumptions and Solution**

* In order for the two models to be integrated, input of each model is assumed to have been gone through corresponding pre- process steps to obtain correct forms before input process into each individual model.
* Each model has different input type and the results after training process of LDA and TiSASRec are combined in the part of Online Recommendation Engine to output final recommendation list delivered to the considering user.

## **Structure of Thesis**

This is a brief introduction of the report outline. The report consists of five transparent sections:

*Introduction* which shows a brief introduction of topic introduction, motivation, aim of the project purpose of the report and the project outline;

*Literature Review* which provides acknowledgement of recommendation system, topic modelling (with some developed method in the field), and sequential- based and attention- based mechanism (with some related methods in the field);

*Methodology and Implementation* which illustrates the architecture and of the new model which is the integration of LDA into TiSASRec (methodology), and then applies the aforementioned model into practice (implementation);

*Experimental results and Evaluation* which shows the performance of the proposed model from Implementation;

and *Discussion, Conclusion and Future Work* which summarizes and provides viewpoints on the obtained results from the previous section as well as the whole research processes and denotes some enhancement for an improved version in the future.

# **CHAPTER 2: LITURATURE REVIEW**

***This session main goal is to deliver an overview definition of the thesis research criterion, which is the Recommendation System in E- commerce. From there, illustrations of some ubiquitous and renowned systems relating to Topic Modelling models and Sequential- based and Attention- based Models/ Research are given to explain main mechanisms of the research techniques. A detailed summary of TiSASRec [19] is also presented as a related criterion to the proposed model.***

## **Recommendation System**

* + 1. **Introduction**

Imagine yourself surfing YouTube to find a want- to- watch video, or scrolling down Spotify to find your favorite songs, … That sounds a little bit of artificial and effort- consuming. Recommendation System has come into life to mitigate the problem. This system offers a list of new items based on your history of interactions when you use that service such as *likes*, *shares, search keywords, favorite items, views,* or specifically your *records of orders* if the service mentioned here is online shopping on some specific websites.

According to Resnick and Varian [1997] [[5](#ref5)], recommendation engine is capable of learning user’s preferences and use the information to suggest new items to that user, or Herlocker et al. [2000] defined recommender system as prediction provider for a next feasible item to be interacted with by a given user according to his/ her taste.

* + 1. **Goals of recommendation systems**

In the time of well- developed technologies, recommendation system has expanded their capability to learn users’ s behaviors as well as their interaction, and to suggest the same item (list of items) that may in common with the user’ s interactive list of items. From that, it can raise the number of customer interactions to e- commerce websites and contribute significantly to the growth of total product providers’ s revenue.

In a nutshell, recommendation has upgraded the user experiences as well as companies ’s revenues decently. It manages to free the gap between customers and product providers by delivering insightful information of customers ’s behaviors.

* + 1. **Real- world applications of recommendation system in E- commerce**

1. ***Amazon Recommendation System***

Not until the 1990s did Amazon break the common trend of recommendations at the current time, which was the user- based algorithm taking advantage of people with similar preferences for recommendations; instead, the leading e- commerce site devised its own algorithm of item- based [2], which specifies next feasible items through the user’s history interactions and pairs them to similar items, using metrics and composing a list of recommendations.

1. ***Shopee Recommendation System***

Shopee Recommendation system is a perfect hybrid of three approaches: popularity-based – which recommend based on best seller products, content filtering – which recommend based on same product characteristics matching the user’ s search and collaborative filtering – which recommend products of the same users. This hybrid engine combines user profile, item features and user- item interactions to recommend unique products to each of its customer.

1. ***Sephora or Ulta Recommendation System***

Sephora and Ulta are two famous online websites for beauty products transactions. The sites comprise of thousands of products from more than over a hundred beauty brands worldwide. Recommendation System of both of them are popularity- based, which means that they mainly recommend to customers best seller or new arrival products and hardly make use of user information to optimize the engine.

* + 1. **Categories of Recommendation System**

There are many technologies and methods for recommendation system. All of them can be classified into two large groups: Content- Based Filtering and Collaborative Filtering. In Collaborative Filtering, there are two child methods know as User- User Collaborative Filtering and Item- Item Collaborative Filtering.

* ***Content- Based Filtering***: ***recommendations are given by matching the items’ s description with one particular user’ s profile of interests.***

The system decides the set of recommendations by focusing on the content, the characteristics, or the properties of considering item and suggesting the similar items. In particular, the system considers one (or more) aspect(s) to compute the distance from the current item to the items in the database with the same characteristics and recommends the ones with the lowest distance length.

* **Collaborative Filtering**: ***recommendations are given based on users’ s behaviors (clicks, likes, dislikes, views, purchases, history of orders, etc.).***

Collaborative Filtering concentrates more on the relation between user- user or between item- item. The system extracts behaviors such as ratings for grouping users or grouping items into groups of same components/ behaviors by computing similarity and then recommend product of user from another similar user’ s records or another similar product of the considered one.

*Note: Similar users or items are users in a same group or items in a same group.*

## **Related research on Topic Modelling Models**

* + 1. **Topic Modeling Models**

Topic Modeling is an “unsupervised” machine learning technique of which target is to discover patterns of words and to group documents that share same patterns [[1](#ref1)]. After scanning through a large number of documents, topic model is required to denote latent topic structure of a collection of documents. This technique can help motivate inquire, provide insights of texts, and support organizing documents. There are some major applications in this field are texts processing image processing, biological data processing and survey information processing [[3](#ref3)].

* + 1. **Latent Semantic Analysis (LSA)**

***Latent Semantic Analysis (LSA)*** is a high- dimensional linear associative model [[6](#ref6)]. The main target of the model is to represent texts as vector to make semantic content so that similarity between texts can be computed and effective related words can be revealed [[1](#ref1)]. Since that LSA needs to “breaks down” the enormous number of terms in relation to documents into topics over documents and terms over topics, it must take advantage of ***Singular Value Decomposition (SVD)*** to decompose massive document- term matrix into constituent parts including document- topic matrix, diagonal matrix of singular values, and topic- word matrix.

**A picture containing shape

Description automatically generated**

Figure 2.1. Decomposition of document-term matrix by LSA [7]Table 2.1. Notation and definition for LSA decomposition formula

|  |  |
| --- | --- |
| **Notation** | **Definition** |
| A’ | Document- term matrix |
| U ∈ ℝ^(m ⨉ t) | Document-topic matrix (rows = each topic vector in terms of documents, columns = each document in d documents) |
| V ∈ ℝ^(n ⨉ t) | Term-topic matrix (rows = each term vector in terms of topics, columns = each topic in t topics) |
| t | Number of topics |
| σ1…σ2 | Singular values of document-term matrix |

From document vectors and term vectors by LSA, we can calculate and evaluate the similarity of different documents, of different words, or of terms (or queries) in terms and documents when we look for the most relevant documents to our search query.

* + 1. **Probabilistic Latent Semantic Analysis (PLSA)**

This model was introduced in [[8, 9](#ref8)] as an improvement to its antecedent which is LSA as described above. The core idea here is to find out a probabilistic model with latent topics that can generate data observed in the document- term matrix. PLSA helps reduce words with multiple meanings and cluster group of words of similar contexts.

Diagram

Description automatically generated

Figure 2.2. High level view of PLSA [[Alghamdi et al., 2015](#ref4)]

As illustrated in the above figure:

* Given a document di with provided probability P(di), topic zk presents in that document with probability P(zk |di)
* Given a topic zk, word wj is drawn from zk with probability P(wj|zk)

Formally, the joint probability of seeing a document and a word co-occurrence is shown as below:

**Formula 2.1. Probability of a document and a word co-occurrence**

where P(D), P(Z|D), P(W|D) is set of parameters; P(D)- probability of document D can be directly determined from the given corpus, P(Z|D), P(W|D) – probability of topic Z given document D and probability of word W given document D respectively- are computed by expectation-maximization (EM) algorithm, which finds the likeliest parameter estimates for the model relying on the unobserved latent variables- topics.

* + 1. **Latent Semantic Allocation (LDA)**

Linear Dirichlet Allocation (LDA) is a generative model built based on Bayesian models [[4]](#ref4). It takes advantages of Dirichlet priors of document- topic and word- topic distributions which aims better generalization*.* An approachable definition of Dirichlet distributions is “distribution over distributions” [[7](#ref7)], which means the distribution that we can draw from provided we are given a type of distribution.

For example, if we are given that there are three topics exist in our corpus, there are three distribution mixtures that can be drawn out such as (80% topic A, 10% topic B, 10% topic C), (10% topic A, 80% topic B, 10% topic C) or (10% topic A, 10% topic B, 80% topic C).

If we draw a random probability of Dirichlet distribution, we can obtain distribution that strongly sample a specific topic: Z, Y or Z; it is unlikely for us to obtain such a equal distribution of 33% for each topic. **The idea of Dirichlet Distribution is that it provides the distribution that most models a specific type which implies the main topics in each document within the corpus.**

There are an enormous number of applications for Linear Dirichlet Allocation [[3](#ref3)]: Emotion topic by Pairwise- Link- LDA model extracting link structure between content and citations [[10](#ref10)], role discovery by Author Recipient Topic (AUT) model by analyzing messages from senders to receivers [[11](#ref11)], news topic classification [[12](#ref12)], automatic text grading through information retrieval [[13](#ref13)], reviews sentiment analysis [[14](#ref14)], etc.

For its flexible ability to exchange of both words and documents, LDA has been more interested in than LSA and PLSA. There is nothing better than convert lengthy sentences and words into probability distribution of topic for a document, we can not only grasp the dominant topic of a document but also be capable of grouping the documents of similar topics into one place for later- on process.

Diagram

Description automatically generated

Figure 2.3. Representation of LDA generative model [[10](#ref10)]

**According to** [***Figure 3.9***](#fig3_9)**, LDA generative process for one document in the whole corpus can be described as below:**

* Determine the number of words in document.
* Choose a topic mixture for the document over a fixed set of topics.
* Generate a word in the document by:
* First, pick a ***topic*** based on the document’ s multinomial distribution above.
* Next, pick a ***word*** based on the topic’ s multinomial distribution**.**

Table 2.2. Notation and definition for LDA generative model

|  |  |
| --- | --- |
| **Notion** | **Definition** |
|  | Parameter vector of the Dirichlet prior for per- document topic distributions |
| θ | Topic distributions of a document |
|  | Parameter vector of the Dirichlet prior for per- topic word distributions |
| φ | Word distributions of a topic |
| K | Total latent topics in the corpus |
| Z | A topic in K (Z∈ K) and that topic is represented in document N |
| W | A word represented in document N |
| N | A document in the corpus |
| M | The corpus (all documents) |
|  | Number of documents comprise topic k |
|  | Number of word w in topic k |

Formally, the whole process is represented through this ***pseudo algorithm***:

**Algorithm 2.1. LDA Algorithm (Gibbs Sampling Method)**

Randomly assign θi ~ Dir (α) (i = 1, … M; θi ∈ Delta K)

θi, k is the probability document i (i ∈ {1, ... M}) has topic k (k∈ {1, ... K})

Randomly assign i ~ Dir (β) (i = 1, ... K; i ∈ Delta W)

is the probability topic i (k∈ {1, ...K}) has word w (w ∈ {1, ...W})

For each iteration i:

For each document di (i = 1, …M:

For each word ( ) currently assigned to topic = k (k∈ {1, …K}):

Decrement ,

Sample = k with probability proportion to

Increment ,

Update θi, i

**Formula 2.2. LDA Mallet Topic Assignment for Each Word Formula**

where P () is the probability word n is assigned to topic k, are other word except for the considering word n (within one document), is the distribution of topic k in document d, is total topic distributions of every document in the corpus that consist of word n, is word n distribution in topic k and current document, is total word n distribution of every topic in the document.

* + 1. ***Correlated Topic Model (CTM)***

This is a descendance of Linear Dirichlet Allocation and dependent on LDA algorithm [[1](#ref1)]. Although LDA model Dirichlet distribution mechanism can support to find a dominant topic over the topic distribution within a document effectively, it fails to find correlation between topics for a document. For example, one news about politics may also related to environment and science. By applying the logistic normal distribution [[12](#ref12)], CTM is a better fit than LDA when looking for sets of topics in the field of OCRed.

## **Related Research on Sequential- based Models, Attention- based Models**

The implementation of deep learning into building recommendation has been popular in recent years. There have been many renown systems introduced and some of them claimed to receive significant achievements. There are two of deep learning methods to be profoundly taken into consideration in this research are Sequential- based and Attention- based methods.

* + 1. **Sequential- based Models Introduction**

Every interaction of user worth considering: a click, a view, a like, an add- to- cart action, etc. Usually, customers do not do the only- one surf on website but sequential ones in periodical/ random timestamps. The website usually records customer actions as sequences and this kind of system apply this behavior for recommendation. From the saved sequences, the system exploits the order of each customer interactions and predicts the next behavior due to the ***“context”*** of what they have done recently [[15, 16](#ref15)].

Multiple attempts have been devoted on this field. Some of the proliferated renown systems can be mentioned such as Markov Chains (MCs), Convolutional- based models (CNN), Recurrent Neural Network- based models (RNNs), etc.

1. **Markov Chain Models’ s mechanism**

MC models are classic examples of sequential- based method of which core idea is to capture the previous (or previous few) interaction [[16, 17, 18](#ref16)]. Based on the L- order of the model, it uses L- previous items as input sequence and output a proper recommendation based on the probability calculated from the input sequence by applying the stochastic/ random mechanism. MCs are expected to capture the latest behavior(s) of users for extracting a suitable context of user behavior. This is an effective personalized method since it specifically cares for the users’ s preference in recent times. This approach is claimed to achieve considerable results in short- term item transitions [[15, 16, 17](#ref15)]. However, for longer- time- span or more complicated semantic item transitions, the efficiency is reduced compared to its short- sequence version.

1. **Convolutional Neural Networks Models’ s mechanism**

CNN models as sequential- based models are not very popular in Recommendation Field; however there still exists research into the area. CNN offer a back- and- forth mechanism which we can look into any point of the sequences; nevertheless, the available memory is usually limited, and this means CNN works the most effectively in the short- term behaviors model than the longer ones. Usually, when CNN is applied, it cooperates with attention- based mechanism to fulfill a complete model. For example, [[15, 19](#ref15)] introduced a recommendation system in which considers items as “images” and mine user’ s dynamics of interactions by a CNN at a union- level. SHAN [[20](#ref20)] represents of a number of previous items and a long record of user historical interactions as a two – layer attention model to obtain both short and long behavior of the user.

1. **Recurrent Neural Networks Models’ s mechanism**

RNN models are more popular than CNN models as sequential recommendation engines. RNN pros over the aforementioned methods is its ability to capture user’s long sequence of interactions, which means that the number of interactions within one sequence can be a finite one and can be considerably larger than that of MC models’ s ability to handle. As illustrated in [[22, 23, 24, 25](#ref22)], RNN manage to capture all of user’ s previous actions and this kind of model offer the best efficiency in dense dataset, which consists of as many as possible of interactions per one user. However, the performance is reduced significantly in a sparse dataset.

* + 1. **Attention- based Models Introduction**

Attention- based mechanism is one improvement for CNN’ s or RNN’ s logic in specific and for sequential recommendation system in general. In addition to the sequential idea of exploiting one (or multiple) user’ s previous actions, attention takes into account the ***“relevance”*** of those actions. Instead of the input of user’ s item transitions only, this mechanism adds a weight score calculated for each pair of input with high weight for relevant items, low weight for irrelevant ones. By this mechanism, it easily to form a context faster and even more accurate than the original encoder/ decoder sequential- based models. Due to its high efficiency in offering relevant output, it has been applied widely in recommendation recently [[26, 27](#ref26)]. One of the popular applications of attention mechanism is the ***Transformer*** of which the key algorithm is the scaled dot product attention [[28](#ref28)].

Thinking of ***Attention*** as a soft dictionary, we have ***Queries, Keys*** and ***Values*** which are all vectors; in which every key matches the query to some extend as shown in the dot- product weight and a mixture of all values is returned with their softmax- normalized dot- products as mixture weights.

The idea is upgraded into Self- Attention of which mechanism consisting of ***Queries, Keys*** and ***Values*** of the same set (or same object). Self- attention is sequence- to- sequence layer with parallel computation capability and perfect long- term memory, which is difference from the original version Attention of set- to- set layer and not considering the sequence structure of input.

In order for self- attention models to capture the relative positions of user’ s dynamics, we can apply the positional encodings or relative position embeddings before the self- attention layer which affects the output weights significantly [[2, 28](#ref2)].

* + 1. **Time Interval Self- Attention Sequential Recommendation**

This is an open- source project [[2](#ref2)]. The expected output from the author is a ranked item list as suggestions from the model as recommendations to users. These scores are computed due to the weights learned from customer interaction records (item that they have bought)and the time interval between each two of the items in the item sequences.

TiSASRec absorbs the elite from the self- attention mechanism introduced above. However, the drawback of this method is its shortage in considering the time intervals between two interactions per one user. As it has been claimed by ***Jiacheng Li et al***. [[2](#ref2)], although two users may have the same sequence of interactions with the homogenous positions of each item in the dynamic. However, user 1 may have a- week interaction interval between two different products, and that of the second user is a month. Intuitively, item which is more recent popular seems to catch higher potential to approach customers. This shows the popularity of products in user’ s preference from time over time. Therefore, the two users are expected two get two different recommendations instead of getting a same one. This infers significantly the personalized mechanism for each user, which shows care and detailed concentration on each customer and may gain higher trust from customers. The offered products have more capability of being interacted and raise sales as well.

That is the idea of TiSASRec, which is not only cares for absolute positions between items by the self- attention mechanism but also takes the relative time intervals between each two items in the sequence history for final weight computation.

Diagram

Description automatically generated

Figure 2.4. Overall Framework of TiSASRec

Table 2.3. Notation and Description of TiSASRec formulas

|  |  |
| --- | --- |
| **Notation** | **Description** |
| U, I | User set, Item set |
| Su | Interactive item sequence of user U |
| s | Interactive item extracted from Su |
| Tu | Corresponding time sequence to interactive item sequence of user U |
| t | Corresponding time sequence to extracted item sequence s of user U |
| Ru | Time interval matrix between any two items |
| n | Maximum sequence length |
| d | Latent vector dimension |
| Mu | Time interval matrix |
| MI | Item embedding matrix |
| MKP, MVP | Embedding matrix of position for key and value |
| MKT, MVT | Embedding matrix of time intervals for key and value |
| Zt | Output of model at time step t |

Before the model can be used for predicting, there are five- layer types to pass for the training process of TiSASRec model (not including input and output):

* **Input**:

The model takes in ***sequences of item sequences*** (, , , … )  ***and corresponding time sequences of those sequences (one sequence per user)*** (, , , … ), define an ***n*** number of interactions to keep in sequence (for larger sequence than n: keep only the most recent n interactions, for smaller sequence than n: pad more items to fit n items) in forms of matrices including item sequence, position sequence and time- interval relation matrix (after doing the clipped procedure to personalize the relative time interval per user).

=

From Mu, ***Jiacheng Li et al***. [[2](#ref2)]insisted on the hypothesis that maximum relative time interval between two items are inefficient beyond a threshold; therefore he applied **= clip(Mu)** where the clip operation is applied to every interval  **= min(k, ).**

* **Embedding Layer**:

This layer does the sequences and matrices retrieval rom input and transform them into three types of embeddings: item embedding, position embedding, and time- interval embeddings (for weight computation in the next layers).

**Item Embedding Matrix representation**

**, ,**

**Positional Embedding Matrices for keys and values respectively representation**

**= , =**

**Relative Time Interval Embedding Matrices for keys and values representation**

* **Time Interval Self- Attention Sequential Layers**

As ***Jiacheng Li et al***., [[2](#ref2)] claimed, interaction sequence per one user may consist of many instances within the same timestamp leading to the model’ s unawareness condition of position or relation information. Therefore, the position must be captured via the below algorithm performed in the time interval aware self- attention layer

For each input sequence EI = (ms1, ms2, …msn) where msi Rd, compute a new sequence Z = (z1, z2, … zn) where zi Rd. Each output zi is computed as weighted sum of linearly transformed input elements and the position embeddings.

**Formula 2.2. Weight coefficient :**

where is the weight coefficient for each item in the interaction sequence.

**Formula 2.3. Weightedsum of linearly transformed input elements and the position embeddingsformula:**

where Wd Rd x d is the input projection for value, From the obtained results, each weight coefficient is computed using a soft- max function.

**Formula 2.4. Compatibility function for inputs, time intervals and positions [19]:**

where WK  Rdxd, WV Rdxd are input projection for a query and key respectively, is a factor to avoid large values of the inner product, especially in the case of multi dimensions, is the item embedding vector of item j, is the time interval embedding vector between item i and item j, is the position embedding vector of item j.

* **Feed Forward Layers**

After each time- interval self- attention layer, two linear transformations applying ReLU activation in between to transform linear form of model- in terms of the compatibility of the items, time intervals and positions- into non-linear representation [[2](#ref2)].

* **Normalize and Drop- out rate Layers**

According to ***Jiacheng Li et al****.* [[2](#ref2)] in order to solve overfitting, unstable training process such as gradient vanishing, extended unexpected training time, etc., the author adds in this layer for stabilize the model.

Normalization layer does the normalization on inputs across features so that can accelerate the neural networks.

* **Prediction Layers**

The model ends up with prediction layer provided the adequate knowledge of items, time- intervals and positions weight computation.

**Formula 3.6. Preference Score for item i formula (predicted score):**

where MIi Rd is the embedding of item i and Zt is the representation of given the first t items (i.e., s1, s2, … sn) and their corresponding time intervals (i.e., ru1(t+1), ru2(t+1), … rut(t+1)) between the t and (t+1) item.

* **Output**

Expected output is a sequence ***o = (o1, o2, … on)*** which is a ranked item list. The highest ranked items will be the recommended items at time.

# **CHAPTER 3: METHODOLOGY AND IMPLEMENTATION**

***This chapter goal is to illustrate a detailed view of the proposal model which is the combination of Linear Dirichlet Allocation (LDA) and Time Interval Self- Attention Sequential Recommendation (TiSASRec) as an online recommendation engine.***

## **Methodology**

### **Overview Framework**

Below is the overview framework of the integration LDA- TiSASRec recommendation model:

Diagram

Description automatically generated

Figure 3.1. The Proposed Model- Integration of LDA and TiSASRec

The model has two separate procedures which are ***online and offline***:

* **Online mechanism** is for real- time recommendation when user sign in on the website, the system automatically loads the recommendations in terms of their records of interactions and reviews that they have posted on the website.
* **Offline mechanism**is for each model to train and learn the patterns in combination with weighting computation, and the results are combined to release recommendations in customer online session.

### **Offline training of LDA**

Input of this process are reviews of all customers stored in the system. The system collects all the reviews, passes them through processing procedures, extracts vital words and phrases, and establishes bag- of- words representation for all reviews in the system called corpus, which is the input into LDA Model for training process.

After LDA training process, expected output are the number of topics exist in the corpus plus their related words *(****Output 1****)* and topic distributions per each review ***(Output 2)*** in terms of input corpus. Each user may belong to a group of similar users, and they share same thoughts about types of products, same preferences for items or purchasing process. Capturing this idea, we can group reviews per one user and calculate each user average topic distributions, then calculate distance between each two users exist in the system for top- n similar users.

It is not only the items in the customer bucket tell their preferences, but the reviews posted on the internet also enable the product providers to acknowledge opinions/ or thoughts of customers. Review data is a valuable resource to exploit, however, experts have not spent profound interests in this field decently to have this source as a support factor in their recommendation system.

**Diagram

Description automatically generated**

Figure 3.2. Offline training for LDA

1. **Preprocess Reviews in Text**

The reviews will pass through these procedures to be considered as ***clean reviews:***

* ***Expand contractions***

Contractions are shortened version of words or syllables by removing specific letters and sounds. In case of English, contractions are usually vowel removal from the word. Ideally, by applying the mapping contractions and their corresponding expansions, all the contractions in reviews are expanded.

* ***Remove special characters***

These may be special symbols or even punctuation occurrences in sentences having little significance for extracting features or information based on NLP and ML.

* ***Remove stop- words***

Stop words such as the, a, me, etc. denoting no useful information are usually removed in order to retain words having maximum significance and context.

* ***Tokenize words***

Tokenization means breaking down or splitting textual data into smaller meaningful components called tokens. Sentence tokenization/ segmentation is the process of splitting a text corpus into sentences acting as the first level of tokens which the corpus comprising.

* ***Lemmatizing***

Lemmatization purpose is to remove word affixes to get to a base form of the word. The base form is also known as the root word presented in the dictionary.

1. **Transform Reviews in Text into Bag- Of- Words Representation**

From clean reviews, there are two event occurrences:

* High frequency plus low frequency words reviews are filtered out to extract a dictionary of words exist in corpus

Text

Description automatically generated

Figure 3.3. Example of a part of Dictionary including key- location in dictionary(left) and value- word(right)

* Each review is transformed into bag- of- words representation until every review has been converted forming a complete corpus- which is the input for LDA Model.
* *Bag- of- word representation = List of tuples, each tuple comprises (position in dictionary, frequency of that word in current tuple)*
* *Corpus: List of bag- of- word representation for every review*

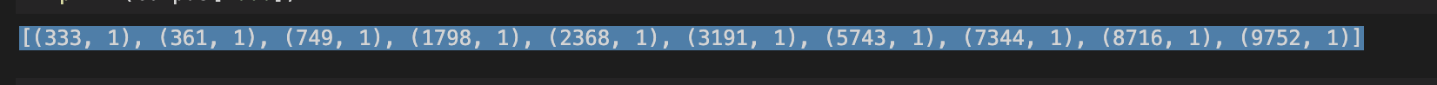


Figure 3.4. Example of one review in bag- of- words representation

1. **Training LDA Model on the corpus**

LDA algorithm is applied on the cleaned corpus for the training process. After applying the model, the expected outputs are all distribution of topic over words and topic distribution of each document over hidden topics (all distributions are presented in vector representation).

1. **Calculating and exporting *similar user file***

After attaining the topic distributions of every review, we group topic distributions per one user (every review of one user which has been transformed into vector representation by LDA). Then, applying the mean calculation from Numpy library to get the mean vector for topic distribution per one user.

Each user has one mean topic distribution vector. By applying ***Pearson Correlated Coefficient***, we can calculate the similarity between every two users within the users available in the system. Pearson Correlated Coefficient is a similarity metric ranging from -1 to 1 to show the correlation between two vectors (in this case is the topic distribution vectors): the nearer the value to 1, the higher the similarity between two vectors, and vice versa.

**Formula 3.1. Pearson Correlation:**

where x, y refers to two average topic distribution vectors of two different user in the system, N is the number of element (topic) that each vector comprises.

This extraction reduces significant amount of time for reviews to be cleaned, transformed, extracted into topic distributions representation and for each user’ s average topic distributions to be computed when doing online recommendation since these processes consumes much time to conduct.

### **Offline training of TiSASRec**

The mechanism of this model has been well explained in ***Chapter 2, part 3.1.1***.

The idea of applying TiSASRec here is to get the scoring mechanism after the model learns the context and behaviors of each user in the system.

The output of this offline training process is the checkpoint (the post model) contains the scoring mechanism for each user in the system (based on their input sequence of interactions).

### **Online Recommendation Engine**

**Diagram

Description automatically generated**

Figure 3.5. Online Recommendation Engine of LDA- TiSASRec model

**Assuming that the customer has already signed in and the system has his/ her ID (identification)**

According to ***Figure 3.5***, we can see that the expected recommendation engine incorporates the results from LDA and TiSASRec for its recommendation engine as description below:

* From ***LDA results***, a file of similar users is delivered. The new system accesses this file, selects IDs of similar users of the current user by accessing the aforementioned file.
* From the group of similar users, the model extracts interacted items of them by accessing the **historical interactions file** and getting interactions of provided user IDs as the ***item list (\*)*** which serves as input into **learned TiSASRec Model** (scoring mechanism) for predicting and calculating scores for these items.
* After offline TiSASRec training process, each sequence of interactions is learned, and each user preference is acknowledged by the model. The new model takes advantage of these patterns and scoring mechanism and inputs the aforementioned ***item list (\*)*** to predict and calculate the ranking score for these items.
* Then, when the rank list is formed, the new model extracts top ten products which have the highest scores to output as recommendations for the customer who is in current session.

## **Implementation**

* + 1. **Dataset Information**

Data applied in this research is open- source Beauty Amazon Review Dataset downloaded via AWS Command Line Interface from AWS and can be found via [***https://s3.amazonaws.com/amazon-reviews-pds/readme.html***](https://s3.amazonaws.com/amazon-reviews-pds/readme.html)*.* The data is available in TSV files in the amazon-reviews-pds S3 bucket in AWS US East Region.

Information including in the dataset is: (data separator is [tab] or “\t”, not [newline] or “\n” as below representation, this representation is just for better comprehension)

marketplace(string)[tab]

customer\_id(string)[tab]

review\_id(string)[tab]

product\_id(string)[tab]

product\_parent(string)[tab]

product\_title(string)[tab]

star\_rating(int)[tab]

helpful\_votes(int)[tab]

total\_votes(int)[tab]

vine(string)[tab]

verified\_purchase(string)[tab]

review\_headline(string)[tab]

review\_body(string)[tab]

review\_date(bigint)[tab]

year(int)[tab]

***A screenshot of a computer

Description automatically generated with medium confidence***

Figure 3.6-a. Raw Data provided in Amazon Beauty Reviews (AWS)

A screenshot of a computer

Description automatically generated with medium confidence

Figure 3.6-b. Raw Data provided in Amazon Beauty Reviews (AWS)

However, for each model in the data we just need specific criteria of data as description below:

* ***For LDA***: customer\_id, review\_text (concatenate from review\_headline and review\_body)
* ***For TiSASRec***: customer\_id. product\_id, review\_date
* In addition, ***for a simple web demo***, I have designed ***simple database*** including three entities: **customer, product, and review.**
* Customer: customer\_id, customer\_name
* Product: product\_id, product\_title, product\_parent
* Review: review\_id, star\_rating, review\_date, review\_headline, review\_body, review\_date, unix\_review\_time (compute from review\_date)

Graphical user interface, application, table

Description automatically generatedFigure 3.7. Collections of database beauty\_amazon

* **Overview of dataset processing steps:**
* unix\_review\_time is calculated from review\_date by the formula applying Pandas library
* Since reviews and unix\_review\_time are two main factors of LDA and TiSASRec, so I first omit any record of user with less than 10 reviews (***assuming reviews are the signs of customer’ s interactions with the system)***
* reviews\_body and reviews\_headlines are concatenated into review\_text to be added into LDA only (to reduce the number of null review\_body or null review\_headline)
* Null records are dropped.
* Convert customer\_id and product\_id into integer representation by Hassh library in Python (customer still hold some string value and all product\_id are string values, which cause difficulty for customer\_id and product\_id to serve as input into TiSASRec).
* **Total Remaining Records up until this time is 617, 368 records** (including some highlight information are customer identities, their reviews, and products that those reviews are about).
* From this stage, we can add the containing field into Database for web demo version.
* The current dataset is represented in two formats for two different models– since this is a research and there is no update in real- time update, the current data file is used for offline training of LDA and TiSASRec instead of direct connection to database to inquire data:
* **Dataset for LDA*:***

Graphical user interface, text, application

Description automatically generated

Figure 3.8. Sample required data for LDA offline training process required fields.

* **customer\_id: for grouping topic distributions by customer\_id after review processing by LDA model**
* **reviews: main input into LDA model to obtaining topics.**

Input into LDA model is a ***corpus*** including ***reviews under bag- of- words representation***. This process of pre- processing and transforming text- form reviews to the LDA Model acceptance is clearly illustrated in [***Chapter 4- Implementation Session***](#_CHAPTER_4:_IMPLEMENTATION)**.**

* ***Dataset for TiSASRec:***

Graphical user interface, text, application

Description automatically generated

Figure 3.9. Sample input into TiSASRec Model

**Required fields:**

* **customer\_id: main input into TiSASRec Model**
* **product\_id: main input into TiSASRec Model**
* **unix\_review\_time: main input into TiSASRec Model**

Input into TiSASRec is a .txt file including three columns corresponding to customer\_id, product\_id, unix\_review\_time; each row is customer- interaction records followed by the order: customer\_id [\t] product\_id [\t] unix\_review\_time

Implementation of LDA offline training process

The main input into LDA model are reviews. Since that these provided from Amazon reviews are collected from their web based, reviews may contain unexpected characters and cause difficulty for the transformation phase. Therefore, these reviews must be pre- processed first before any further implementation.

* + 1. **Implementation for offline training of LDA**

1. **Pre- processing reviews**

**Text

Description automatically generated**

Figure 3.10. Main Steps for pre- processing texts

1. **Transforming reviews into corpus**

Reviews are input into LDA model for topic patterns extraction. However, the model acceptance of input are reviews under bag- of- words representation. Therefore, the text must be initially transformed into the correct format – which is the bag- of- words format.

After pre- processing, when all the reviews are represented in list of each review’ s usable words, we start to form the overall corpus and dictionary by applying ***genism.corpora library***:

Text

Description automatically generated

Figure 3.11. Implementation for creating corpus and dictionary

1. **Training LDA Model**

***c.1. LDA from genism.wrappers.ldamallet library overview***

Mallet is a Java- based package for statistical natural processing, classification, clustering, topic modeling, and other machine learning applications. This package and genism provide alike functions supports the text representation into vector form for usable information by computers [[35]](#ref35).

* ***Sample lda model generation from genism wrappers LDA Mallet model:***

*lda = genism.models.wrappers.ldamallet.LdaMallet(mallet\_path= mallet\_path, corpus = corpus, dictionary = dictionary, num\_topics = 10, iterations = 200, random\_seed = 42, alpha = 0.01)*

Table 3.1. Parameters and Corresponding Description for genism LDA Mallet

|  |  |
| --- | --- |
| **Hyperparameter** | **Description** |
| corpus | List of reviews in bag- of- words representation |
| dictionary | Dictionary of positions and words appear in the corpus |
| num\_topics | Number of expected different topic appearances in the corpus |
| iterations | Maximum iteration through the corpus when inferring topic distributions of the corpus |
| random\_seed | Like random\_state of LDA |
| alpha – (int/ default) | This is a prior- belief for each topic exist in the corpus.   1. Case int (i.e. 0.01): alpha = 0.01 2. Case default: alpha = 5.0 / num\_topics |

***c.2. Training LDA Model by Mallet Method***

***Text

Description automatically generated***Figure 3.12. Implementation of LDA Model applying genism.models.wrappers.ldamallet

1. **Get similar users**

Similar users are computed from the user\_dict which contains average topic distribution vector per user. This list is formed from loading the Model, group reviews by one user and computed the mean vector applying Numpy mean function for 2-D array. Then, the list is stored into a separate file:

Text

Description automatically generated

Figure 3.13. Function to create user average topic vector list

From the list, we can calculate correlation between the current user and each other user to get the most similar users. The acceptance similar score here is 0.85:

Text

Description automatically generated

Figure 3.14. Function to find similar users

***Text

Description automatically generated***

Figure 3.15. Example of the similar user file.

Note: Each row of the lda.csv file- which is the similar user file is the similar user record of 1 user; the first item is the current user, and all the following items are his/ her similar users.

* + 1. **Implementation of TiSASRec offline training process**

Since the original version TiSASRec [[2](#ref2)] functions have been well explained in his paper, I revise only his main layer implementation – which is the Time interval Self- Attention Sequential Layers of the Model Class to get a view of what and how his model builds.

Text

Description automatically generated

Figure 3.16-a. Weight computation Layers Implementation

Graphical user interface, text

Description automatically generated

Figure 3.16-b. Weight computation Layers Implementation

Text

Description automatically generated

Figure 3.16-c. Weight computation Layers Implementation

Text

Description automatically generated

Figure 3.16-d. Weight computation Layers Implementation

After the training of TiSASRec, we get the following files which stores the scoring mechanism of the user behaviors learned by the model.

A picture containing graphical user interface

Description automatically generated

Figure 3.17. Output of TiSASRec Training Process

There is one function of the model to pay attention to is the predict function:

**This is the main functions called to get the final recommendations for the system.**

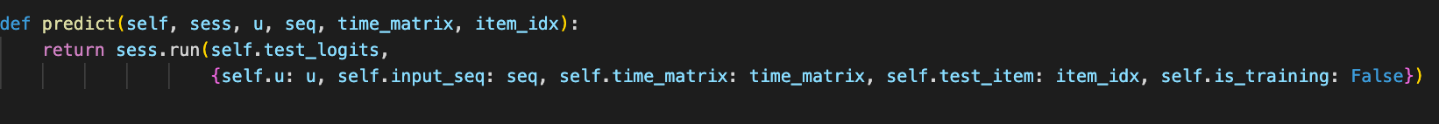
******

Figure 3.18. Prediction Function for scoring item

* + 1. **Implementation of online recommendation**

**Assuming the considering user has already signed in and the system has acknowledged the user id.**

To obtain the ranked list as final recommendations, we modify the waiting list to be ranked that is added together with the user sequence.

Recap: After finishing the offline training process of LDA, we have obtained a file containing similar users of user in current session. From the ids from this file,

Reload Scoring Mechanism:

Text, chat or text message

Description automatically generated

Figure 3.19. Loading Scoring mechanism from TiSASRec

Text, chat or text message

Description automatically generated

Figure 3.19- a. Getting ranking item list

From ***Figure 3.18***, we load the checkpoint stored after training the TiSASRec Model. This checkpoint stores the scoring mechanism for each user according to each user preference. Then, for each item set getting from

Text

Description automatically generated

Figure 3.19- b. Getting ranking item list

Text

Description automatically generated

Figure 3.20. Scoring, ranking and outputting recommendations

Applying the lda.csv, we can get similar user ids and then accessing the interaction file, we can get the item set belonging to those user ids as preference item set; and then applying this function to calculate and extract a ranking item list. Finally, top 10 highest items will be recommended to current customer.

**CHAPTER 4: EXPERIMENTAL RESULTS AND EVALUATION**

***This session target is to illustrate the results obtained from experimenting the proposal model. Besides, some evaluation from the above results is pointed out in the part as well.***

## **Experimental Results**

* + 1. **Find the best number of topics for dataset**

After the pre- processing and transformation steps, we must find an optimal number of topics for this corpus. Therefore, I ran a grid- search on the whole corpus with num\_topics in range (15, 26).

The reason for this variation of number of topics is human judgement due to results from base run of num\_topics = 15, I don’t see the transparency between topics, so I choose that number to be the starting point.

**Text

Description automatically generated**

Figure 4.1. LDA Grid Search Implementation

**The accuracy metric for considering the best number of topics is Coherence Score Measure**, which is the most popular metric for topic modeling [[30](#ref30), [31](#ref31), [32](#ref32)].

Coherence metric used here is ***c\_v*** by [[30](#ref30)] attains the cooccurrence counts for a given word using a sliding window and then calculates the Normalize Pointwise Mutual Information (NPMI) of every top word to other top words and returns a set of vectors. Critically, each vector is calculated by cosine similarity and the coherence is the mean of these similarities.

**Formula 4.1. Normalize Pointwise Mutual Information Formula:**

, p () is the probability that two random variables i and j co- occur, p () is the probability of variable i’ s occurrence, p () is the probability of variable j’ s occurrence.

**Formula 4.2. Mutual Information Vector between two random variables:**

where is set of random variables, is set of variables to calculate mutual dependence, is the normalize pointwise information between two variables i and j.

Chart, line chart

Description automatically generated

Figure 4.2. Grid Search results on LDA original model base

**From the above figure, we can claim that the best number of topics in this dataset is 20 since 20 gives the highest coherence score in range 15- 25 topics.**

* + 1. **Visualization LDA Mallet Results (visualization of 20 topics)**

The tool we use here is pyLDAvis. The key feature of this tool is the ability to show the topic- term relationship using relevance in an interactive and compact way. Critically, this tool provided users with the topic representation (on the left) and the most relevant terms given one topic on the left selected (bar- charts on the right) by applying the relevance formula [[33](#ref33)].

**Formula 4.3. Relevance formula:**

Text

Description automatically generated with low confidence

where kw denotes the probability of w {1, ..., V}, pw is the empirical distribution of the corpus, is the weight given to the probability of term w under topic k relative to its lift (measuring on log scale)

* + 1. **LDA Mallet Training Process**

Chart, bubble chart

Description automatically generated

Figure 4.3. Overall visualization of LDA Mallet performance

Chart

Description automatically generated

Figure 4.4. Overall visualization of LDA Mallet term- topic relation for topic 1

* + 1. **TiSASRec with LDA integration Performance (LDA- TISASRec Model)**

**The metric applying here is NDCG@10 and HR@10, which are the same as the two original versions for easier comparison.**

***Normalize Distributed Cumulative Gain (NDCG)*** is the measure of ranking quality of which ranse varies in (0,1). **NDCG@10** is the measure of position of top 10 highest ranked items. To calculate this measure, we must first calculate the cumulative gain of the recommended order (DCG) and cumulative gain of the ideal recommended order- ideal order (DCGi) is the order ranked from high to low with respect to decreasing order).

**Formula 4.4. DDG Formula:**

where DCG is distributive cumulative gain of recommended order, is item’ s rank at position i- corresponding to index i in ranked item list.

**Formula 4.5. DDGi Formula:**

where is distributive cumulative gain of recommended ideal order, is item’ s rank at position i- corresponding to index i in ranked item list.

**Formula 4.6. NDCG Formula:**

where NDCG- normalized distributive cumulative gain is ranking quality given DCG and of a ranked item list, DCG is the cumulative gain of recommended order of the ranked item list, is cumulative gain of recommended ideal order of the ranked item list.

**Hit Rate** is the proportion of counts the ground- truth item in top n highest ranked items over the total times of ranking actions. HR@10 is hit rate considered on if ground- truth item in top 10 products.

**Formula 4.7. HR Formula:**

**HR =**

where HR is the hit score evaluated from the ranked item list, number of hits is the number of times the ranked list satisfies a given condition, number of miss is the number of times the ranked list dissatisfies a given condition

**After running the TiSASRec with the item chosen from LDA similar users’ s item basket, we collect the result as followed:**

***Chart, line chart

Description automatically generated***

Figure 4.5. LDA- TiSASRec Model NDCG @10 and HR@10 on evaluate and test data

Table 4.1. NDCG @10 and HR @10 of LDA- TiSASRec Model

|  |  |  |  |
| --- | --- | --- | --- |
| **Dataset** |  | **NDCG @ 10** | **HR @ 10** |
| Amazon Beauty | Validate Data | 0.3645 | 0.5332 |
| Test Data | 0.3494 | 0.5119 |

## **Evaluation**

* + 1. **Evaluation of LDA results**

Looking at the representation via pyLDAvis tool, we can see that topics extraction by ***LDA Mallet*** results in very little overlapped circles, which means that the topics are well separated. Therefore, this is an effective way for distinguish among groups of users.

For a visualization of 1 topic 1 only: Mallet manages to show high relevance of words within 1 topic. As it is illustrated in the [***Figure 5.9***](#fig5_9), words like *head, shave, cut, charge, power, wax, etc.* may be words relating to male shaving or waxing tools. These group may belong to male who interested in hair- care products.

**Therefore, the LDA Mallet’ s results can be integrated into TiSASRec for completing the LDA- TiSASRec model.**

* + 1. **Evaluation of the online recommendation by integrating LDA into TiSASRec Table**

Shape

Description automatically generated

Figure 4.6. Original TiSASRec NDCG @10 and HR@10

Chart, line chart

Description automatically generated

Figure 4.7. Original SASR NDCG @10 and HR@10

Table 4.2. NDCG @10 and HR @10 of LDA-TiSASRec, original TiSASRec and original SASR

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** |  | **NDCG @ 10** | **HR @ 10** |
| LDA- TiSASRec | Validate Data | **0.3645** | **0.5332** |
| Test Data | **0.3494** | **0.5119** |
| TiSASRec | Validate Data | 0.3268 | 0.5048 |
| Test Data | 0.3161 | 0.4985 |
| SASR | Validate Data | 0.3445 | 0.5132 |
| Test Data | 0.3194 | 0.4819 |

Chart, line chart

Description automatically generated

Figure 4.8. Comparison HR@10 for LDA- TiSASRec, TiSASRec and SASRec

The obtaining results denotes that TiSASRec + LDA model returns the better performance than the original SASR and TiSASRec versions. This is due to the reduction in random factors contributing to the process of evaluation, which has slightly lower rating scores of the two other models.

Therefore, we can see that the process of choosing products which are candidates for model scoring mechanism is very important since that the relevant item list results in higher score, which means the potential of these high ranked products is increased. By applying the combination of TiSASRec for capturing context of user interaction records and LDA for capturing the user preferences due to his/ her similar groups can positively affect the recommendation.

# **CHAPTER 5: CONCLUSIONS AND FUTURE WORKS**

***This session target is to summary my general viewpoint of the whole process from doing research until model evaluating as well as some future work to enhance this model.***

## **5.1. Conclusions**

Through the whole process of conducting this research, I acknowledge that there is a strong connection between group of user preferences and their history of interactions. These are considered as valuable resources for exploiting and developing for recommendation system. Most Recommendation Systems at the current time can capture short- term behavior of users, but not long- term and sequentially with high concentration in item popularity time span for each user. In the case of cosmetics, there are some products that need to be re- purchased occasionally due to the weather/ climate periodical changes. This is the reason why I have done profound research in this TiSASRec incorporating with user review topics for stronger preference links. If these kinds of recommendation system can be put into real- world production, I believe recommendation products have higher chance of approaching customers and revenues can be raised significantly.

## **5.2. Future works**

However, as I have mentioned, these systems have not been exported into real- world scenarios and remain research for a long time. In addition, the rank scores for item have not been as high as expected (under 0.5% for NDCG and under 0.8% for HR) and the increase rate of my research is just slightly, not significantly (just about 0.05% for NDCG and 0.03% for HR). I did not manage to connect the LDA into TiSASRec completely, now I just inherit the results from separate file to input as a connection of two models. Therefore, there is some enhancement I would like to make as my future work to raise the efficiency of this model:

* Connect LDA and TiSASRec to connect straight into database.
* Integrate LDA straight into TiSASRec Model, which means incorporate the two models into one united one (I am thinking of making a user preference embedding to raise the ranking score of a specific item if the considering item is in the same preference group learned via reviews).
* Make offline training become online training (synchronize all the processes, not separately like in this research)
* Offer products for new customer as well- not only old customers like the current mechanism of this model.

# **REFERENCES**

[1] Rubayyi Alghamdi and Khalid Alfalqi. A survey of Topic Modeling in Text Mining. In *(IJACSA) International Journal of Advanced Computer Science and Applications,* Vol. 6, No. 1, 2015.

[2] Jiacheng Li, Yujie Wang, and Julian McAuley. 2020. Time Interval Aware Self-Attention for Sequential Recommendation. In *Proceedings of the 13th International Conference on Web Search and Data Mining (WSDM '20)*. Association for Computing Machinery, New York, NY, USA, 322–330.

[3] Blei David and Lafferty John. 2006. Dynamic Topic Models. In *Proceedings of the 23rd International Conference on Machine Learning (ICML’06),* 2006, 113-120.

[4] Xuerui Wang and Andrew McCallum. 2006. Topics over time: a non-Markov continuous-time model of topical trends. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '06)*. Association for Computing Machinery, New York*,* NY, USA, 424–433.

[5] Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich. 2012. Recommender System: An Introduction***,*** First Edition. International Journal of Human–Computer Interaction, 28:1, 72-73.

[6] Thomas K. Landauer, Peter W. Foltz and Darrell Laham. 1998. An introduction to latent semantic analysis. In *Discourse Processes*, 25:2-3, 259-284.

[7] Joyce Xu. 2018. Topic Modeling with LSA, PLSA, LDA & lda2Vec. In *series for Nanonets*, [info@nanonets.com](mailto:info@nanonets.com). Accessed on March 15th, 2021. <https://www.kdnuggets.com/2018/08/topic-modeling-lsa-plsa-lda-lda2vec.html>.

[8] Thomas Hofmann. 1999. Probabilistic latent semantic indexing*.* In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '99)Association for Computing Machinery, New York, NY, USA, 50–57.*

[9]N. Bassiou, C. Kotropoulos. RPLSA. 2011. A novel updating scheme for Probabilistic Latent Semantic Analysis. Computer Speech & Language. Vol 25, 741-760.

[10] Ramesh M. Nallapati, Amr Ahmed, Eric P. Xing, and William W. Cohen. 2008. Joint latent topic models for text and citations. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '08)*. Association for Computing Machinery, New York, NY, USA, 542–550.

[11] Xuerui Wang and Andrew McCallum. 2006. Topics over time: a non-Markov continuous-time model of topical trends. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '06)*. Association for Computing Machinery, New York*,* NY, USA, 424–433.

[12] Xuan-Hieu Phan, Le-Minh Nguyen, and Susumu Horiguchi. 2008. Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In *Proceedings of the 17th international conference on World Wide Web (WWW '08)*. Association for Computing Machinery, New York, NY, USA, 91–100.

[13] Tuomo Kakkonen, Niko Myller, and Erkki Sutinen. 2006. Applying latent dirichlet allocation to automatic essay grading. In *Proceedings of the 5th international conference on Advances in Natural Language Processing (FinTAL'06)*. Springer-Verlag, Berlin, Heidelberg, 110–120.

[14] Priyantina, R.A., & Sarno, R. (2019). Sentiment Analysis of Hotel Reviews Using Latent Dirichlet Allocation, Semantic Similarity and LSTM. International Journal of Intelligent Engineering and Systems*, 12*, 142-155.

[15] W. Kang and J. McAuley. Self-Attentive Sequential Recommendation. In *IEEE International Conference on Data Mining (ICDM’18)*, 2018, 197-206. DOI: 10.1109/ICDM.2018.00035.

[16] Ruining He, Wang-Cheng Kang, and Julian McAuley. 2017. Translation-based Recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems* (RecSys '17). Association for Computing Machinery, New York, NY, USA, 161–169.

[17] He, R., & McAuley, J. (2016). Fusing Similarity Models with Markov Chains for Sparse Sequential Recommendation.  2016*.* In *IEEE 16th International Conference on Data Mining (ICDM)*, 191-200.

[18] Rendle, S., Freudenthaler, C., & Schmidt-Thieme, L. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *WWW '10*.

[19] Jiaxi Tang and Ke Wang. 2018. Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. *In Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18).* Association for Computing Machinery, New York, NY, USA, 565–573.

[20] Trinh Xuan Tuan and Tu Minh Phuong. 2017. 3D Convolutional Networks for Session-based Recommendation with Content Features. In *Proceedings of the Eleventh ACM Conference on Recommender Systems (RecSys '17).* Association for Computing Machinery, New York, NY, USA, 138–146.

[21] Haochao Ying, Fuzhen Zhuang, Fuzheng Zhang, Yanchi Liu, Guandong Xu, Xing Xie, Hui Xiong, and Jian Wu. 2018. Sequential recommender system based on hierarchical attention network. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI'18)*. AAAI Press, 3926–3932.

[22] Hidasi, B., & Karatzoglou, A. (2018). Recurrent Neural Networks with Top-k Gains for Session-based Recommendations. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM ’18)*, 2018, 843-852.

[23] Alex Beutel, Paul Covington, Sagar Jain, Can Xu, Jia Li, Vince Gatto, and Ed H. Chi. 2018. Latent Cross: Making Use of Context in Recurrent Recommender Systems. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18).* Association for Computing Machinery, New York, NY, USA, 46–54.

[24] How Jing and Alexander J. Smola. 2017. Neural Survival Recommender. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining (WSDM '17)*. Association for Computing Machinery, New York, NY, USA, 515–524.

[25] Lakshmanan Rakkappan and Vaibhav Rajan. 2019. Context-Aware Sequential Recommendations withStacked Recurrent Neural Networks. In *The World Wide Web Conference (WWW '19)*. Association for Computing Machinery, New York, NY, USA, 3172–3178.

[26] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. Attentive Collaborative Filtering: Multimedia Recommendation with Item- and Component-Level Attention. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '17)*. Association for Computing Machinery, New York, NY, USA, 335–344.

[27] Wang, S., Hu, L., Cao, L., Huang, X., Lian, D., and Liu, W. 2018. Attention-Based Transactional Context Embedding for Next-Item Recommendation. In *AAAI*.

[28] Rehurek, R., and Sojka, P. (2011). Gensim–python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech*

[29] McCallum, Andrew Kachites. 2002. "MALLET: A Machine Learning for Language Toolkit." 2002. <http://mallet.cs.umass.edu>.

[30] Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the Space of Topic Coherence Measures. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining (WSDM '15)*. Association for Computing Machinery, New York, NY, USA, 399–408.

[31] Qiang, J., Zhenyu, Q., Yun, L., Yunhao, Y., and Xindong, W. (2019). Short Text Topic Modeling Techniques, Applications, and Performance: A Survey. Retrieved from *arXiv: Information Retrieval*. DOI: https://arxiv.org/abs/1904.07695

[32] Lisena Pasquale, Harrando,Ismail, Kandakji Oussama and Troncy Raphaël. 2020. TOMODAPI: A Topic Modeling API to Train, Use and Compare Topic Models. 132-140.

[33] Sievert, C., & Shirley, K.E. 2014. LDAvis: A method for visualizing and interpreting topics.