NATIONAL AND KAPODISTRIAN UNIVERSITY OF ATHENS

SCHOOL OF SCIENCE

DEPARTMENT OF INFORMATICS AND TELECOMMUNICATIONS

POSTGRADUATE PROGRAM

MASTER THESIS

Experimental Evaluation of Representation Models for Content Recommendation in Microblogging Services

Efthymia Karra Taniskidou

Supervisor: **Manolis Koubarakis**, Professor UoA
Co-Supervisors: **George Papadakis**, Post-Doctoral Researcher UoA
George Giannakopoulos, Post-Doctoral Researcher IIT NCSR

ATHENS

SEPTEMBER 2016

ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Πειραματική Αξιολόγηση Μοντέλων Αναπαράστασης για Συστάσεις Περιεχομένου σε Microblogging Υπηρεσίες.

Ευθυμία Σ. Καρρά Τανισκίδου

Επιβλέπων: **Μανόλης Κουμπαράκης**, Καθηγητής ΕΚΠΑ
Συν-Επιβλέποντες: **Γιώργος Παπαδάκης**, Μεταδιδακτορικός Ερευνητής ΕΚΠΑ
Γιώργος Γιαννακόπουλος,
Μεταδιδακτορικός Ερευνητής ΙΠΤ, ΕΚΕΦΕ Δημόκριτος

ΑΘΗΝΑ

ΣΕΠΤΕΜΒΡΙΟΣ 2016

MASTER THESIS

Experimental Evaluation of Representation Models for Content Recommendation in Microblogging Services

Efthymia Karra Taniskidou

R.N.: M1390

SUPERVISOR:

Manolis Koubarakis, Professor UoA

EXAMINATION COMMITEE:

Dimitris Gounopoulos, Professor UoA

ATHENS

SEPTEMBER 2016

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Πειραματική Αξιολόγηση Μοντέλων Αναπαράστασης για Συστάσεις Περιεχομένου σε Microblogging Υπηρεσίες.

Ευθυμία Σ. Καρρά Τανισκίδου

Α.Μ.: Μ1390

ΕΠΙΒΛΕΠΩΝ:
Μανόλης Κουμπαράκης, Καθηγητής ΕΚΠΑ

ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ:
Δημήτρης Γουνόπουλος, Καθηγητής, ΕΚΠΑ

ΑΘΗΝΑ

ΣΕΠΤΕΜΒΡΙΟΣ 2016

ABSTRACT

Micro-blogging services constitute a popular means of real time communication and information sharing. Twitter is the most popular of these services with 300 million monthly active user accounts and 500 million tweets posted in a daily basis at the moment. Consequently, Twitter users suffer from an information deluge and a large number of recommendation methods have been proposed to re-rank the tweets in a user's timeline according to her interests. We focus on techniques that build a textual model for every individual user to capture her tastes and then rank the tweets she receives according to their similarity with that model.

In the literature, there is no comprehensive evaluation of these user modeling strategies as yet. To cover this gap, in this thesis we systematically examine on a real Twitter dataset, 9 state-of-the-art methods for modeling a user's preferences using exclusively textual information. Our goal is to identify the best performing user model with respect to several criteria: (i) the source of tweet information available for modeling (ii) the user type, as determined by the relation between the tweeting frequency of a user and the frequency of her received tweets, (iii) the characteristics of its functionality, as derived from a novel taxonomy, and (iv) its robustness with respect to its internal configurations, as deduced by assessing a wide range of plausible values for internal parameters. Our results can be used for fine-tuning and interpreting text user models in a recommendation scenario in microblogging services and could serve as a starting point for further enhancing the most effective user model with additional contextual information.

SUBJECT AREA: Recommendation Systems

KEYWORDS: Twitter, Microblogging, Ranking, Retweets, User model, Textual representation model, Topic model

ΠΕΡΙΛΗΨΗ

Οι microblogging υπηρεσίες αποτελούν έναν ευρέως διαδεδομένο τρόπο ανταλλαγής πληροφοριών και επικοινωνίας σε πραγματικό χρόνο. Το Twitter είναι η πιο δημοφιλής microblogging υπηρεσία, αφού επί του παρόντος συγκεντρώνει 300 εκατομμύρια ενεργούς χρήστες μηνιαίως και καταγράφει 500 εκατομμύρια tweets ημερησίως. Για να αντιμετωπιστεί ο καταιγισμός πληροφοριών των χρηστών του Twitter, έχουν προταθεί ποικίλες μέθοδοι συστάσεων για την ανακατάταξη των tweets στο χρονολόγιο ενός χρήστη, σύμφωνα με τα ενδιαφέροντά του. Στη παρούσα διπλωματική εργασία εστιάζουμε σε τεχνικές που αρχικά κατασκευάζουν ένα μοντέλο για κάθε χρήστη ξεχωριστά, με στόχο να απεικονίσουν τις προτιμήσεις του και στη συνέχεια κατατάσσουν τα tweets του χρήστη με βάση την ομοιότητά τους με το μοντέλο αυτό.

Στη βιβλιογραφία, μέχρι στιγμής, δεν υπάρχει περιεκτική αποτίμηση των στρατηγικών μοντελοποίησης χρηστών. Για να καλύψουμε το κενό αυτό, εξετάζουμε διεξοδικά σε ένα πραγματικό σύνολο δεδομένων του Twitter, σύγχρονες μεθόδους για τη μοντελοποίηση των προτιμήσεων ενός χρήστη, χρησιμοποιώντας αποκλειστικά πληροφορία σε μορφή κειμένου. Ο στόχος μας είναι να προσδιορίσουμε το πιο αποδοτικό μοντέλο χρήστη σε σχέση με τα ακόλουθα κριτήρια: (1) την πηγή της πληροφορίας σχετική με tweets που χρησιμοποιείται για την μοντελοποίηση, (2) το είδος του χρήστη, όπως προσδιορίζεται από τη σχέση μεταξύ της συχνότητας των tweets που ανεβάζει ο ίδιος και της συχνότητας αυτών που λαμβάνει, (3) τα χαρακτηριστικά της λειτουργικότητάς του, όπως προκύπτουν από μια πρωτότυπη ταξινόμηση, (4) την ευρωστία του σε σχέση με τις εσωτερικές του παραμέτρους. Τα αποτελέσματά μας μπορούν να αξιοποιηθούν για την ρύθμιση και ερμηνεία μοντέλων χρηστών βασισμένων σε κείμενο, με στόχο συστάσεις σε microblogging υπηρεσίες και λειτουργούν σαν σημείο εκκίνησης για την ενίσχυση του καλύτερου μοντέλου με επιπλέον συναφή εξωτερική πληροφορία.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Συστήματα Συστάσεων

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Twitter, microblogging, Κατάταξη, Retweets, μοντέλο χρήστη, μοντέλο αναπαράστασης κειμένου, μοντέλο θέματος

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω τον μεταδιδακτορικό ερευνητή Γιώργο Παπαδάκη, για την ανεκτίμητης αξίας καθοδήγησή του, την πολύτιμη κριτική του και την πρακτική αλλά και ηθική στήριξή του για να ολοκληρωθεί με επιτυχία η διπλωματική αυτή εργασία. Ευχαριστώ επίσης τον καθηγητή μου Μανόλη Κουμπαράκη, για την άψογη συνεργασία μας και τη δυνατότητα που μου έδωσε να δουλέψω με την ερευνητική του ομάδα, η οποία μου πρόσφερε μια επίσης άριστη συνεργασία και πολύτιμη βοήθειά κατά τη διάρκεια εκπόνησης της εργασίας αυτής. Ευχαριστώ και τον μεταδιδακτορικό ερευνητή Γιώργο Γιαννακόπουλο για τις χρήσιμες παρεμβάσεις του κατά την ερευνητική μελέτη της εργασίας.

Τέλος, ευχαριστώ τους φίλους μου για την ηθική συμπαράσταση τους και τους γονείς μου Δήμητρα και Σπύρο, για την υπομονή τους και κυρίως την πρακτική υποστήριξη που μου έδωσαν καθ' όλη τη διάρκεια εκπόνησης της εργασίας.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 Motivation

Throughout the last years, the emerging growth of the Internet has made available a wealth of information and new services. The advent of search engines allows for accessing resources such as books, movies and articles. More and more news papers are now accessible online to help users to keep abreast of the latest news. The abundance of easily accessible resources has led to the urgent need for dealing with the user information overload problem by providing personalized recommendations of content to facilitate and enhance the user experience in the sphere of Internet. To this end, considerable research has been conducted in the field of recommendation systems [2, 44]. A plethora of methods have been successfully applied for suggesting content such as books [36], news articles [5, 42] and web sites [4, 38]. Two are the main common characteristics of the majority of these sources: the lengthy textual content and that users are passive consumers of documents or they can barely interact by posting comments or blogs.

At the same time, the explosive growth of microblogging services boosts users interaction and provides new information sources. Young people read news from social networking sites like Facebook and from microblogs like Twitter, instead of traditional mass media[1]. Further, unlike the traditional domains, micro-blogging services such as Twitter, Weibo and Plurk [2] focus specifically on instant communication and interaction among people all over the world. Users can post short messages to their timeline in real-time through any electronic device like their mobile phones or tablets. They can also discover other people or groups of people with similar interests and explicitly connect with them to disseminate and consume information.

With the popularity of microblogs comes the daily overwhelming of timelines and online profiles with hundreds of items, that users are unable to filter in order to reach those that truly strike their attention. Thus, the need for directing user attention to posts, content or other people to follow has arisen. Recommendation systems for such platforms have been extensively studied [1, 11, 21, 48, 52]. We focus our work on recommendations on Twitter since it is the most popular micro-blogging service as of 2016. Not only does Twitter currently have over 300 million monthly active users [3] but also the number of messages

[1] http://www.digitalnewsreport.org/, http://www.bbc.com/news/uk-36528256

[2] https://twitter.com/, http://www.weibo.com/, http://www.plurk.com

[3] http://www.statista.com/statistics/282087/number-of-monthly-active-twitter-users/recorded on 21 of August 2016

everyday transmitted on Twitter has jumped from 35 million tweets per day in 2010 to over 500 million in 2016 [4] .

Several works in the literature seek to acquire a grounded understanding of Twitter structure and users' practices [8, 30]. In [25], user activities are studied and classified as information seeking, information sharing or social activities. In [11], users' interests are represented in the basis of two categories of activity: information seeking/sharing. At the core of understanding user behavior and representing user preferences, lies the modeling technique of the user.

Twitter provides rich textual information for users like their tweets and retweets which could be utilized to model their interests. Yet, the real-time nature of Twitter messages as well as the length restrictions (posts are constrained to comprise only up to 140 characters) result in careless and ungrammatical text messages. The multilingual content hinders common tokenizing and pre-processing techniques like stemming [5]. Additionally, although messages are restricted to be rather short, within this short length users have become accustomed to enclose a wealth of information by using techniques such as abbreviations and URL shortening.

The intricacies mentioned above, pose challenges to building effective user models that convert unstructured texts into a representation revealing the user characteristics. In the literature, several methods that profile users and perform recommendations by utilizing either internal Twitter data or external information have been proposed [1, 11, 21, 34, 48]. However, there is no work to systematically analyze the diverse textual sources in Twitter and to evaluate a large number of different user modeling techniques for content recommendations.

In this work, we focus on user modeling strategies so as to rank the tweets of a user's timeline and place the most interesting tweets on top. We exclusively consider content-based information found on tweets' textual content, disregarding any exogenous resources like web pages accessed by tweets' URLs or Semantic Web ontologies. There are three reasons for this choice. External sources are not always easily accessible or their exploitation is costly and time-consuming. The external content is textual in most cases and, thus, our techniques can be applied there, as well. It is also important to emphasize that many methods that incorporate external sources are orthogonal to ours and can be used in combination to ours, to enhance the most effective representation model.

[4] http://www.internetlivestats.com/twitter-statistics/ recorded on 21 of August 2016

[5] Stemming is a technique to reduce words to their root form, usually by removing suffixes.

1.2 Contributions and Research Questions

We firstly organize 12 state-of-the-art text models in a novel taxonomy. In this way, we gain a deeper insight into each model endogenous characteristics. Thus, we can better interpret each model performance and extract performance patterns based on the taxonomy. Then, we thoroughly evaluate 9 representation models by conducting experiments on a real Twitter dataset and by examining a total of 149 configurations for all models.

The questions we try to answer are the following:

1. Which is the best configuration of each representation model?
2. How robust is the performance of every representation model in terms of effectiveness?
3. Which is the most effective model for recommending short texts in micro-blogging systems?
4. Which tweet information source for building user models is the best one with respect to effectiveness?

Our contributions are summarized as follows:

- We perform the first systematic study that considers 9 different representation models, involving both n-gram graphs, bag models and 5 state-of-the-art topic models. To the best of our knowledge, there is no other equally comprehensive analysis in the literature including both n-gram graphs, bag and topic models.
- We introduce a taxonomy to elucidate our experimental results. We demonstrate that each category yields different results, with the fully order-preserving models outperforming the others.
- We determine the best configuration for each representation model, the most effective model and the most robust one across all configurations. In that way, we direct practitioners to select the most effective or robust representation strategy or to fine-tune a model they already use for recommendations, not only on Twitter, but also on other micro-blogging services with similar characteristics.
- We compare 5 different sources and 8 combinations of them to specify the most effective and the less informative one for representing Twitter users in a recommendation context.

- We examine three distinctive user categories to specify if there is a model that is most effective for different kinds of user behavior.

1.3 Thesis outline

Section 2 describes the main notions of recommendation systems. It also defines formally the problem we tackle in this thesis. Section 3 goes through the work related to ours. Section 4 introduces a novel taxonomy of text representation models and a detailed analysis of them. Section 5 contains our experimental setup and Section 6 presents the outcomes of our study. The conclusions of our study and future work are discussed in Section 7.

2 BACKGROUND AND PROBLEM DEFINITION

2.1 Recommendation Systems

Recommendation is the process of suggesting items to users that are likely to match their interests. As previously mentioned, it has been applied to diverse domains such as movies, news articles, music and books.

Two are the major ways to determine users' preferences for items: explicit and implicit feedback. In the former case, ratings or comments are directly provided by users. In many situations, this method is intractable since users are reluctant to provide ratings or the identification of positive or negative comments is difficult and time consuming. Therefore, preference should be indirectly inferred by behavior such as purchasing history or movie viewing. Also, there are situations when direct feedback is not available at all. For example, in microblogging services people do not directly rate other posts, users or URLs. In such fields however, users are remarkably active and their actions are informative enough about their tastes. For instance, in microblogs, users approve or disapprove posts of others by like or dislike them, re-post messages to disseminate information that interest them and consciously select others to follow. In this thesis, we explore a dataset from Twitter and we rely only on implicit user feedback and, more specifically, on the re-posting of followees' tweets.

Another factor that differentiates recommendations is the kind of predictions the process target to. Traditional systems predict the absolute rating values for items yet unknown to the user [4, 38]. However, rating prediction is not directly applicable to domains such as microblogging services. Much research has also be done for predicting the relative preference order of items rather than their individual rating values [12, 13, 27]. Our approach focuses on producing a ranking of the tweets of a user's timeline with respect to how much likely is for the user to like each of the tweets.

Recommendations are classified into three main groups according to how they are performed:

1. **Content-based systems** recommend items to users based on similar items they liked in the past.
2. **Collaborative filtering** recommend items to users based on the items that the most similar users have preferred in the past.
3. **Hybrid approaches** combine content-based and collaborative filtering schemes.

In the next subsections, we provide an overview of the first two basic categories of recommendation systems and describe which method we follow in our experiments. Our method is best classified as content-based, but in some sense we have also consider the collaborative factor when representing users by their followees' or followers' tweets (more details in Section 2.2)

2.1.1 Content-Based Systems

In the majority of Content-based recommender systems, the recommendation process consists of the following three main steps: Firstly, a model is built for each item using a set of its features. Secondly, a model is constructed for each user to capture his interests and tastes, by combining models of items he has interacted with before. Then, for matching the user model against candidate documents' models, a similarity metric is most often used and those items with the highest values are selected for recommendation. This procedure, also known as topic relevance, has its roots in information retrieval and it has been used in several papers [11, 4, 10, 36].

The core component of the content-based approach is the item model. Its ultimate goal is to precisely reflect the properties of the items in question. After determining a representation of items, the user model is naturally derived by the individual models of items for which the user has expressed like or dislike. There are items that entail apparent features that are easily obtainable and reveal their characteristics with accuracy; for example movies are characterized by actors, directors, release date and genres or electronic products have descriptions available with sizes, colors, etc. On the other hand, there are classes of items without readily available attributes, like images and text documents. For example, we want to suggest news articles to users about themes they like, but how can we identify the ideas discussed in the unstructured format of text articles? Is it possible to distinguish those words that summarize an article's topics? Likewise, is it possible to characterize the user-generated noisy content of microblogs?

In this last question lies the epicentre of this work. We focus on modeling texts in the Twitter service. The most popular text-based models in the literature are the bag models which come in two variants: the *token n-grams* and the *character n-grams* models. Both models assign a vector to each document or user, but the former considers individual words or sequences of adjacent words as dimensions while the latter considers sequences of adjacent characters. The bag-of-tokens model is popular in recommendation systems [4, 5, 36, 38] whereas the bag-of-characters has been successfully applied to domains such as text categorization [9], spam filtering [28] and authorship attribution [14]. The

determinant factor for the performance of the above models is the value of n which defines whether the model makes the bag-of-words assumption (for n=1), or preserves the relative ordering between word or character tokens (for n $>$ 1).

Despite their popularity, the bag models suffer from high dimensional feature spaces whose size increases with the increase of n. Also, they cannot distinguish between different semantic meanings of the same word, without techniques such as Lemmatization and Stemming which are yet language-dependent and not applicable to multilingual environments like microblogs. *Topic models* [6, 45] ameliorate the former problem by modeling textual content into a topic-space of fixed low dimensionality. In essence, they discover latent topics in a collection of documents by counting word co-occurrence patterns. Documents can be represented as a distribution over the uncovered topics and in turn, topics are taken as distributions over words.

Topic models also capture polysemy through uncertainty over topics [45], namely the same word can be assigned high probability for more than one topics (for example, the word ”book” can have a high weight in both a topic about reading and a holidays topic). They have been effectively utilized for lengthy texts like news articles and papers' abstracts. On the contrary, micro-blogging short messages like tweets challenge the application of topic models due to their noisy nature and the scarcity of word co-occurrence patterns. For this reason, aggregation strategies have been employed to form lengthy pseudo-documents by merging tweets that adhere to some commonalities. In our experiments, we test several configurations of topic models' internal parameters as well as we examine different aggregation strategies for tweets.

Another text model effectively used for text summarization and text classification is the *n-gram graphs*, which represents documents as undirected graphs [15]. This is a language-agnostic model that allocates one node per token or character n-gram and it goes beyond the bag-of-words assumption by connecting two nodes with one weighted edge denoting the frequency of co-occurrence of the corresponding n-grams. In that way, it adds rich contextual information to the model and copes with spelling and grammatical mistakes. Further, this approach enables the n-gram graphs to be less sensitive to polysemy, as they associate every word with its context, which helps to disambiguate its meaning.

2.1.2 Collaborative Filtering

The goal of collaborative filtering is to suggest items to users based on the preferences of other like-minded users. Instead of relying on items features, they represent users in terms of their ratings. Collaborative filtering algorithms are divided into two main categories: Memory-based and Model-based.

Memory-based techniques compare users in terms of their previous ratings or their indications of preference when only implicit feedback is available. In that way, they determine a set of users usually known as neighbors, that share similar tastes with a target user. Then, the rating for a user and an item is calculated by aggregating the ratings of the most similar other users for this specific item. The simplest aggregation formula is the average of the ratings. A popular one is a weighted sum of the ratings where the weights are the similarity values between users. However, this approach does not consider the tendency of some people to assign either very high or very low scores. To address this issue, the average rating value for each user, calculated across all values of her rating history, is very often subtracted from the individual ratings. In this case, the weighted sum is computed on the deviations of each user's ratings from her average value. Typical similarity functions are the cosine similarity where each user is treated as a vector of his ratings and the Pearson correlation coefficient [6].

Model-based algorithms function offline by utilizing the users' whole rating history to build a model. This model, in turn, is used to predict user-item rating values. The main difference between memory-based and model-based algorithms is that the former harnesses some heuristics to predict ratings while the latter, confronts to statistical and machine learning techniques to learn a model from the underlying data collection.

2.2 Recommendation Systems in Twitter

Messages on Twitter are called *tweets* or *statuses* and are limited to contain up to 140 characters; we refer to posts made by a user as his outgoing tweets. Users choose to receive statuses' updates of people by *following* them; we call the tweets received by a user's followees, his incoming tweets. Users can also retweet their followees' statuses to share information with followers. An excessive number of tweets is transmitted everyday causing an information overload problem for users. Nonetheless, Twitter has set up only

[6] The Pearson correlation coefficient measures the linear relationship between two samples. Given two samples $\{x_1, \ldots, x_n\}$ and $\{y_1, \ldots, y_n\}$ it is defined as: $r = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i-\bar{y})^2}}$

	T	R	F
T	-	-	-
R	TR	-	-
E	TE	RE	EF
F	TF	RF	-
C	TC	RC	-

Table 1: Abbreviations for combinations of Twitter information sources

a few inadequate tweets filtering mechanisms.

For example, users track the updates only of those people they follow, but the statuses are presented in chronological order as the newness of a tweet is considered more significant than its relatedness with the user's interests. In order for a user to reach those posts that truly cater for his concerns, he may have to skip several statuses, a situation which is rather intractable. Additionally, Twitter offers a list of the trending topics all over the world or the possibility to attend regional news. However, the same tweets are presented to all users irrespective of personal interest. From the above insufficiencies of Twitter itself, the need for personalized recommendation of Twitter content stems; to this end, several methods that utilize either internal Twitter data or external information have been proposed from the research community [11, 13, 17, 21, 48].

2.2.1 Representation Sources in Twitter

Five are the distinctive sources of tweets we consider for modeling users' preferences: tweets (T) and retweets (R) of the user in question, tweets and retweets of his followees (E), his followers (F) and the reciprocally connected users (C). We also examine combinations of them presented on Table 1.

To introduce notation, consider a Twitter user u with followees $e(u) = \{e_1, \ldots, e_k\}$, followers $f(u) = \{f_1, \ldots, f_m\}$ and reciprocally connected users $c(u) = \{c_1, \ldots, c_l\} = f(u) \cap e(u)$. Let $T(u) = \{t_1, \ldots, t_n\}$ be the tweet history of a user and $R(u) = \{r_1, \ldots, r_l\}$ the retweet history. As outgoing posts of u we define the $T(u) \cap R(u)$. Subsequently, the followees' and followers' posts are given by the formulas $E(u) = \bigcup_{e_i \in e(u)} (T(e_i) \cup R(e_i))$ and $F(u) = \bigcup_{f_i \in f(u)} (T(f_i) \cup R(f_i))$ respectively. Their combination is defined as $EF(u) = E(u) \cup F(u)$. The followees' tweets are called incoming user tweets throughout this thesis. The history coming from the reciprocally connected users is: $C(u) = \bigcup_{g_i \in c(u)} (T(g_i) \cup R(g_i))$ The rest combinations are constructed with the same way.

The choice of tweets and retweets as potentially effective representation resources for reflecting preference is straightforward. Users' messages enclose the themes and ideas about which they are interested in chatting or in informing their followers. Retweets comprise the subjects that have captured their attention so intensely, that they decided to reproduce them in order to propagate the information.

However, published tweets do not capture all aspects of a user's preferences. For example, a politician may mostly publish political or national issues, but at the same time he could be a football fun. By exclusively representing this politician with his posting history, his interest about football is lost. Therefore, as shown in prior works [11, 12], followees' statuses can be utilized to model a different kind of interest; that of a user as an information seeker.

Concerning the followers' tweets, we could hypothesize that people follow others to attend topics of common preference. However, a follower's statuses can introduce noise since a user exerts little control on his followers. For example, consider that tge politician of the previous example, follows a player of his favorite team, who is yet totally indifferent to politics. Now, if we model the player with respect to his followers' posts, the politician will introduce noise, especially if he is an active Twitter user. Note that the same argument does not hold in the case of followees, whom the user actively chooses so as to have some interests in common. It remains to be experimentally confirmed the truth of the above intuition.

Furthermore, the *following* relationship in Twitter does not imply friendship as it does in other social media like Facebook. For example, Johnny Depp may have hundreds of followers who are interested in the famous actor's latest news about his filmography. Undoubtedly, Johnny Depp does not really know the majority of his followers, let alone to consider them his friends. To the contrary, in sites like Facebook users connect with fewer, however more intimate, people. In some sense, the reciprocal connections can be considered as closer relationships between Twitter users. From this intuition stems the idea to take the tweets of reciprocally connected users as a source of representation with potential.

2.2.2 Twitter Challenges

Despite the rich textual content that is present in Twitter, tweets have some special characteristics that distinguish them from other conventional domains. These characteristics pose challenges to many information retrieval tasks, including user modeling for recom-

mendation.

1. **Sparsity (C1).** Tweets are too short, since they contain only a maximum of 140 characters. Any error can be corrected in a subsequent tweet. Consequently, the available content to precisely model a user or a tweet is very limited.

2. **Noise (C2).** The real-time nature of Twitter forces users to post quickly, without taking into account spelling errors in their writing. As a result, misspellings are very common in tweets, making in fact similar words to look rather different.

3. **Multilinguality (C3).** The global popularity of Twitter has led to a high diversity in tweet content languages. Common pre-processing techniques such as stemming and lemmatization , employed for analyzing text more effectively, rely on the language characteristics to reduce words to their common root. Thus, they are language-specific and do not directly apply to multilingual content. Moreover, the same rules for tokenizing English documents do not apply to languages such as Japanese, in which words are typically not separated by space or punctuation.

4. **Non-standard language (C4).** Tweets are unstructured, ungrammatical and usually written in a slang language, since they constitute a way of everyday informal communication. Due to the length constraint, words are replaced with a shortened form being dissimilar with the original one. For example, the word “goodnight” is often replaced with the abbreviation “gn”. Also, special symbols are used such as emoticons and hashtags, or words are modified to put emphasis. For example the word ”yes” is often replaced by ”yeeeees”.

It would be of high interest to examine not only which representation model successfully deals with the above challenges, but also which is the property of such a model that makes it overcome Twitter’s short texts difficulties. While describing the representation models of our study and interpreting the results of our experimental evaluation, we refer to the above challenges and elaborate on why each model copes or not with them. It is worth recalling here that almost all microblogging platforms share the same distinctive features. Thus, our exploratory analysis on models for Twitter also applies to modeling for microblogs in general.

2.3 Problem Definition

In this section, we formally present the base recommendation algorithm we use across all of our experiments.

As test set for evaluating user models, we take a subset of the incoming tweets of the user. Positive examples are the retweeted incoming tweets, since the retweeting action is considered as an indirect indication of a user's judgment of usefulness. Intuitively, a user chooses to retweet a post after carefully reading it and for spreading information that strikes his attention [12]. Due to the excessive number of negative examples, we perform sampling when selecting the test set.

In this context, we can formally define the recommendation task in Twitter as follows:

Definition 1 *Given a representation model M, a user u, an information source s and a set of training tweets $T_{train} \subset s$, a user model $M(u, s)$ is constructed. Then, for unseen test tweets $T_{test} = \{t_1, \ldots, t_N\}$ the models $\{M(t_1), \ldots, M(t_N)\}$ are built. Given a similarity function sim, the recommendation algorithm ranks the candidate tweets according to their similarity: $sim(M(u, s), M(t_i))$. The aim is to place the retweets higher than the non-retweets at the ranking.*

2.4 Summary

In this section, we presented the basic background on recommendation systems. We described the two main categories of recommendations, i.e., Content-based systems and Collaborative filtering. Then, we focused on recommendations in Twitter as well as we discussed the most prevalent information sources available in Twitter, i.e., a user's tweets and retweets, her followees', her followers' and the reciprocally connected users' tweets and retweets. Twitter's special characteristics were also mentioned, i.e., sparsity, noise, multilinguality and non-standard language. Finally, the recommendation problem we try to solve in this thesis was formally defined.

3 RELATED WORK

In this section, we briefly describe the related work from two different perspectives: user and document bag models employed in a recommendation setting and topic models for representing texts.

Pazzarani et al. [38] deal with the task of suggesting links from an index page to follow or constructing queries for finding interesting arbitrary pages in the web. They utilize user ratings on explored pages and build bag-of-words user models with boolean weights indicating the presence or absence of a word. Billsus et al. [5] build two separate profiles with TF-IDF weights, to reflect the short-term and long-term user preferences about news articles. They classify news stories as relevant or not with the user model, using cosine similarity.

In Social media and Twitter, several recommendation works rely on user models from textual content available on Twitter. Chen et al. [10] compare with three other algorithms, a content matching approach for suggesting new followees that is based on comparing bag-of-words user models with TF-IDF, based on cosine similarity. Chen et al. [11], in order to recommend URLs, consider both the user's tweets and her followees' tweets, construct two different TF-IDF bag-of words user models and compare user and URL models with cosine similarity. Kywe et al. [32] perform hashtag recommendation by modeling tweets and users with TF-IDF bag-of-words and bag-of-hashtags vectors respectively. They compute cosine similarities between a target user-tweet pair and other user-tweet pairs to select the most relevant hashtags.

Two studies are similar to our work in the sense of comparing different user modeling strategies in a Twitter recommendation context. The Twittomender system in [21], examines models from a user's tweets, a user's followees' and followers' tweets and combinations of them. Abel et al. [1] compare hashtag-based, entity-based and topic-based user models. Both papers build bag-of-words models and use cosine similarity but the former utilizes TF-IDF weights while the latter Term Frequency weights. Yet, we consider a different recommendation task, namely tweet recommendation, and we go beyond these study in the following ways: (i) Our work compares several weighting schemes and similarity metrics in combination with the different Twitter information sources. (ii) Besides the bag-of-words model, our analysis involves X other bag and topic models (iii) We elucidate our results in the basis of a taxonomy of the various models.

Regarding topic models, Ramage et al. [39] characterize content on Twitter and apply recommendation of tweets and users by harnessing a supervised generative model,

Labeled LDA. Godin et al. [17] apply LDA to suggest hashtags for tweets. Many authors leverage pooling techniques to find more coherent topics in short texts. A pooling scheme consists of the aggregation of short texts that share similar content or express similar ideas, into lengthy pseudo-documents. Hong et al. [23] experimentally compare the effectiveness of a standard LDA model and the Author Topic Model (ATM), trained on individual messages and on aggregated messages by user and by hashtag, in finding popular Twitter messages and in classifying Twitter users and messages. Our analysis differs from theirs in that we examine combinations of internal parameters and aggregation methods for X topic models instead of two. Also we comparatively evaluate not only topic, but also bag models in a recommendation context. Mehrotra et al. [35] present five pooling techniques for tweets and an algorithm for automatically assigning hashtags to unlabeled tweets in order to improve the performance of LDA. Alvarez-Melis and Saveski [3], introduce a tweet pooling strategy by conversation and show that it outperforms other pooling schemes, by evaluating LDA and ATM in a relevant tweet retrieval task. We decided to implement for our experiments the three most common techniques, i.e unpooled messages, pooling by user and pooling by hashtag.

Besides pooling techniques, several extensions to existing topic models can be found to the literature for dealing with short documents. Yan et al. [49] propose a generative topic model for short texts which copes with the sparsity problem by using biterms to explicitly model the corpus-level word co-occurrense patterns. Zhao et al. [51] introduce the Twitter-LDA model which assumes that each tweet comprise only one topic. Yin et al. [50] suggest a collapsed Gibbs Sampling algorithm for the Dirichlet Multinomial Mixture Model to alleviate the issues of sparsity and high-dimensionality in short text clustering.

Relevant to our work are also systems that incorporate external resources for augmenting short texts representation and improving recommendation performance. IJntema et al. [24] build user profiles from concepts identified in articles the user has read and propose semantic-based techniques for news recommendation. Ramanathan et al. [41] utilize Wikipedia concepts to construct hierarchical user profiles. Kapanipathi et al. [29] perform personalized Twitter stream filtering, representing tweets as RDF triples constructed from information such as author, location and time and building user profiles by fetching data from different Social Media and by using ontologies. Lu et al. [34] boost tweets personalized ranking by extracting concepts from Wikipedia to model tweets. User profiling is further enriched by a random walk on Wikipedia concept graph to detect more relevant concepts. Jin et al. [26] introduces Dual Latent Dirichlet Allocation, which learns topics from both short texts and auxiliary documents to improve short text representation for classification. In our analysis, we exclusively consider user modeling strategies that are

entirely based on the textual content of tweets; thus, any other exogenous information is complementary and can be combined with our models to increase their performance.

4 REPRESENTATION MODELS

4.1 Taxonomy

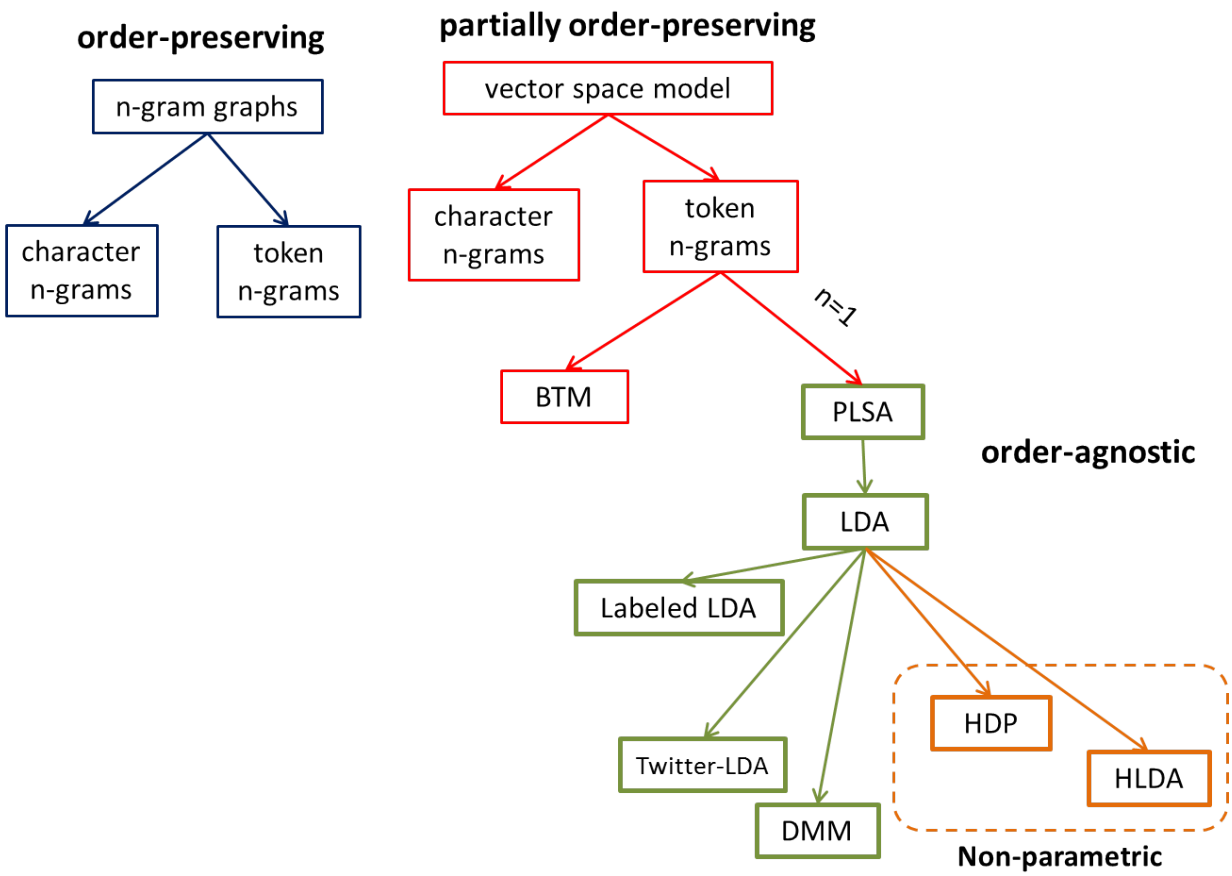

Figure 1: Taxonomy of the text-based models.

We now present a taxonomy of 12 text-based representation models to shed light on their behavior. The taxonomy is based on the structure of the representations, which determines how much information is enclosed in them. It comprises three main categories for models:

1. **Order-agnostic** models represent a document without taking into account the n-grams ordering inside the text, thus not capturing multiword expressions.
2. **Partially order-preserving** models are built based on word (or character) sequences, accounting for the relative order between words (or characters). However, they lose contextual information as they disregard the ordering of the sequences themselves.

3. **Fully order-preserving** models improve on the disadvantages of the other two categories, since they can incorporate not only the relative word (or character) ordering, but also the absolute ordering between sequences of words (or characters).

All order agnostic and partially order-preserving models we consider fall into the class of vector space models. These models represent a tweet t_i as a vector of weights $M(t_i) = (w_{i1}, \ldots, w_{im})$. The same representation may apply to users by aggregating the weights of a user's tweets. In this way, we end up with a user model $M(u) = (v(w_{i1}), ..., v(w_{im}))$ where $v(w_{ij})$ denotes the importance of the dimension j for user u. Three are the strategies we examine in this study for building the user model from the individual tweets: (i) a simple summing of the weights where $v(w_{ij}) = \sum_{i=1}^{N_T} w_{ij}$ and N_T is the total number of tweets (ii) the centroid of the tweets after normalizing the weight vectors to have unit length, where $v(w_{ij}) = \frac{\sum_{i=1}^{N_T} \frac{w_{ij}}{\|M(t_i)\|\|}}{N_T}$ (iii) the Rocchio algorithm[7] where $v(w_{ij}) = \frac{b}{N_p} \sum_{t_i \in N_p} \frac{\vec{M}(t_i)}{\|M(t_i)\|\|} - \frac{c}{N_n} \sum_{t_i \in N_n} \frac{\vec{M}(t_i)}{\|M(t_i)\|\|}$; N_p is the number of positive tweets, N_n the number of negative tweets and α and β are parameters that control the relative importance of all positive and negative examples.

Additionally, 8 out of 12 models belong to the class of topic models. *Topic models* uncover the latent semantic structure of texts by determining the topics they talk about. The topics and how the documents exhibit them are considered as the hidden structure of a topic model which can be discovered by exclusively analyzing the observed variables, i.e., the words of the original text. In general, they assume that each document is a mixture of multiple topics which is a natural assumption to make, considering the heterogeneity of themes and ideas in most documents. Each topic is in turn considered as a set of words that tend to co-occur. *Probabilistic topic models* represent each document as a distribution over topics, where each topic is itself modeled as a distribution of words.

To specify notation, let a document collection of D documents from a vocabulary of size V. We denote the number of topics with Z, a single topic with z, a document with d and a word with w. θ_d refers to the distribution of document d across topics and ϕ_z to the distribution of topic z across words. In the recommendation scenario described in section 2.3, we use topic models as follows: each tweet is assumed as a text document and is represented as the tweet's distribution over topics, that is $M(t) = \theta_d$. To model a user u,

[7]The Rocchio algorithm was initially proposed for relevance feedback in the vector-space model and it was adapted to text classification [43]. In this context, it represents each class, of a set of classes C, as a prototype vector, constructed by aggregating the positive and negative examples for that specific class. To classify an unseen document, it computes the similarity between all class vectors and the document vector and selects the most similar class. Likewise, a user model can be built by combining positive and negative training data and then the testing tweets are compared with that model.

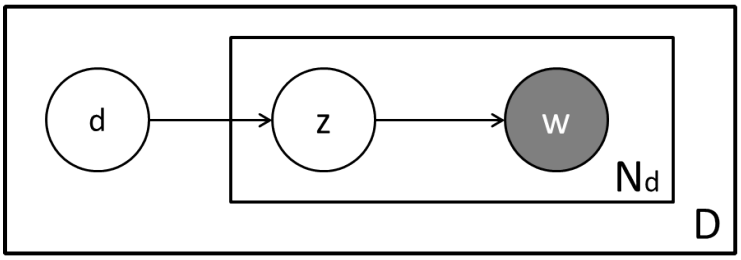

Figure 2: PLSA generative process in plate notation.

we aggregate the θ_d distributions of the individual tweets from an information source s.

The next subsections analyze the 12 text-based models we examine in this paper, organized in the categorization that we described above. We summarize their basic functionality and put emphasis on their internal parameters.

4.2 Order-Agnostic Models

4.2.1 Probabilistic Latent Semantic Analysis

Probabilistic Latent Semantic Analysis (PLSA) [22] is an unsupervised probabilistic topic model. PLSA generates each word in a document from a single topic from a number of predefined topics Z. In turn, it models each document as a list of mixing proportions for these topics. The word-document pair (w,d) of observed variables are considered independent from the latent topic variable z: $P(w,d) = \sum_z P(z)P(z|d)P(w|z)$. $P(z/d)$ are the mixing proportions for document d.

Figure 2 describes the generative process of PLSA in plate notation. Shaded and unshaded nodes indicate observed and latent variables respectively. Arrows between nodes depict conditional dependencies between variables. The plates show repetition in sampling for the variables inside, with the number of repetitions included in the right bottom corner:

1. Select a document d with probability $P(d)$
2. For each word w in document d:
   (a) Select a topic z with probability conditioned on d (i.e. probability $P(z|d)$).
   (b) Select a word given the previously selected topic z (i.e. probability $P(w|z)$).

For Z topics, D documents and a vocabulary of size V, PLSA needs to estimate Z distributions of size V for topics over words and D distributions of size Z for documents

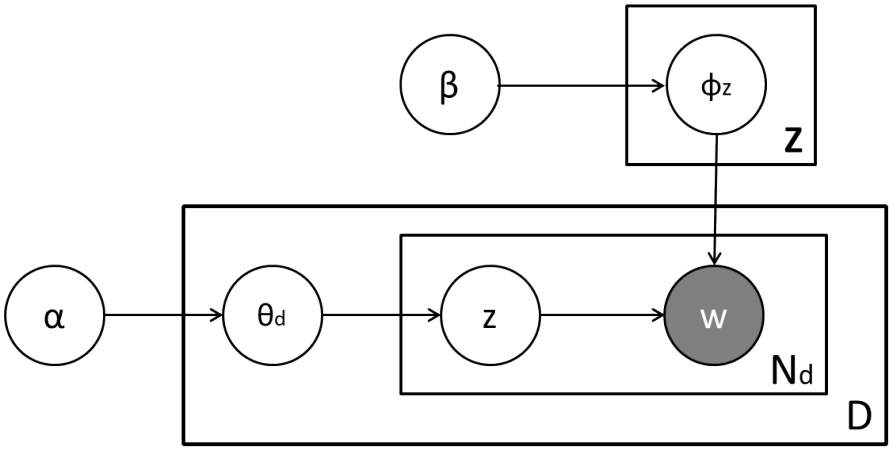

Figure 3: LDA generative process in plate notation.

over topics, leading to $ZV + DZ$ total parameters for estimation and so linear growth in D. This is a major drawback of PLSA, since the linear growth results in the overfitting of the model [7].

4.2.2 Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) [7] extends PLSA by assigning a Dirichlet prior on the distribution of a document over topics. Unlike PLSA which merely regards a document as a list of numbers (the mixing proportions for topics), LDA includes a generative process for documents themselves and models the proportions as a k-parameter latent random variable. The number of topics is given as a parameter to the model and can seriously impact its performance. A small number leads to too broad topics, failing to capture the diverse themes discussed in many text documents, while a large one results in too complex and uninterpretable topics [45].

Figure 3 shows the generative process of LDA which is defined as follows:

1. For each topic z in $1 \dots Z$, draw a multinomial distribution ϕ_z from symmetric Dirichlet prior β.
2. For each document d:
   (a) Select a multinomial distribution θ_d over the Z topics from symmetric Dirichlet prior α.
   (b) For each word w in document d:
      i. Draw a topic z from θ_d.
      ii. Draw a word w from the multinomial distribution ϕ_z of topic z over words.

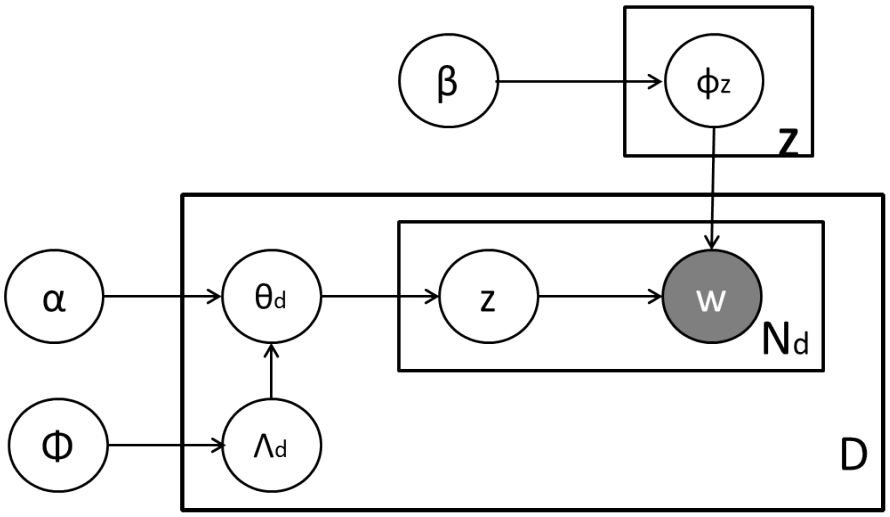

Figure 4: Labeled LDA generative process in plate notation.

For inference of the distributions θ_d and ϕ_z, approximate methods are used such as mean-field variational expectation maximization and Gibbs sampling. Significant parameters, which distinguish LDA from PLSA, are the α and β of the Dirichlet prior on θ and ϕ respectively. The former was introduced in the original paper for LDA while the latter was added in a later variant of LDA [19].

The Dirichlet distribution is a multivariate generalization of the beta distribution. It is used in Bayesian statistics as the conjugate prior of multinomial distribution. The Dirichlet is parametrized by a vector of real numbers $\vec{\alpha} = \{\alpha_1, \ldots, \alpha_K\}$. In case of documents topic modelling, the hyperparameter α_j denotes the number of times topic j is drawn for a document, before actually observing any words of the document itself. Very often, the hyperparameters are considered as $\alpha_1 = \cdots = \alpha_K = \alpha$, resulting in the symmetric Dirichlet prior. On θ_d and ϕ_k are placed symmetric Dirichlet priors by LDA. The parameter β can be interpreted as the counts of a word in a topic prior to the actual observation of any word in the corpus.

4.2.3 Labeled LDA

Labeled LDA (LLDA) [40] is an extension of LDA that incorporates supervision in the learning process. Unlike LDA which assumes the existence of unobserved topics underlying a document collection, a set of observed domain-specific labels Λ are employed by LLDA to characterize the collection. Each document is modeled as a multinomial distribution of labels from a subset Λ_d of Λ. Subsequently, each word of document d is picked from

a distribution ϕ_z of some label z contained in the set Λ_d. Together with the specific labelled dimensions, Labeled LDA can also use some latent dimensions characterizing all documents by adding the labels “Topic 1” to “Topic Z” to Λ_d for each d [39].

Figure 4 presents the generative process of LLDA:

1. For each topic z in $1 \dots Z$, draw a multinomial distribution ϕ_z from symmetric Dirichlet prior β.
2. For each document d:
   (a) Construct a labelled set $\Lambda_d \in Z$ from the deterministic prior Φ.
   (b) Select a multinomial distribution θ_d over the subset Λ_d from symmetric Dirichlet prior α.
   (c) For each word w in document d:
      i. Draw a label z from θ_d.
      ii. Draw a word from the multinomial distribution β_z of label z over words.

4.2.4 Nonparametric Topic Models

Nonparametric models aim at imposing the fewest possible assumptions to the distributions of the data and let the parameters to adapt to the structure of the data. Unlike parametric models which are given a fixed number of parameters from the beginning of the training, the parameters of nonparametric models grow as more training data are accumulated. The topic models of this class assume that the number of topic is a-priori unknown; it is indicated from the documents themselves during posterior inference.

Hierarchical Dirichlet Process

Hierarchical Dirichlet Process (HDP) [46] is a Bayesian unsupervised nonparametric model for analyzing data subdivided in groups. Particularly, HDP is designed for applications, where each observation within a group is drawn from a mixture model and it is required for the mixture components to be shared between the different groups. The number of mixture components is not known a-priori and is inferred from the data during the learning process. The Dirichlet Process (DP) constitutes the core component of the HDP model. To each group, a random measure G_j is assigned, distributed according to the $DP(a, G_0)$. The G_0 is the base measure for all child DPs and is itself distributed according to a $DP(a, H)$. Because the draws from a DP are atomic probability distributions and mixing components are associated to atoms, by sharing the same base measure G_0, the

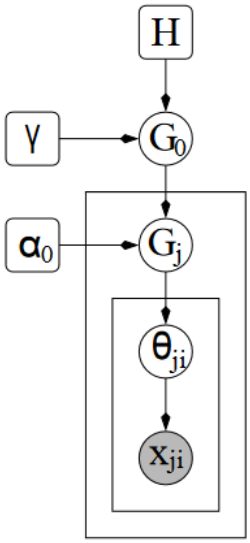

Figure 5: An HDP topic model of one hierarchical level.

groups share the same mixing components as well. More formally the HDP is defined as:

$$G_0|\gamma, H \sim DP(\gamma, H)$$
$$G_j|\gamma, G_0 \sim DP(\gamma, G_0) \qquad \text{for each j}$$

In our application scenario, each document corresponds to a group, the words of the document constitute the observations within the group and the topics which are distributions over words comprise the mixture components. In document topic modeling, HDP is used as a nonparametric extension of LDA in which the number of topics is not known beforehand. Figure 5 indicates the HDP extention of LDA of one hierarchical level:

1. From the measure H, draw a random measure G_0 to provide an infinite number of possible topics .
2. For each document d:
   (a) Select a subset of topics by drawing G_j from$DP(\alpha_0, G_0)$
   (b) For each word in d:
      i. Pick a topic θ_{ji} from G_j.
      ii. Draw a word x_{ji} from the distribution $F_{\theta_{ji}}$ over words.

Note that, due to its recursive nature, it is straightforward to add more hierarchical levels to the HDP model. Such an extension, for example, facilitates cases when documents

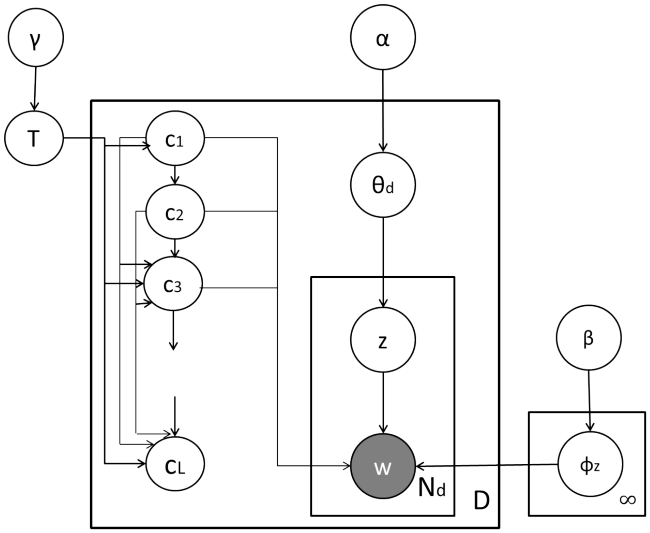

Figure 6: HLDA generative process.

are grouped into broader categories by allowing the discovery of shared topics between categories and comparing with those topics shared between individual documents.

Hierarchical LDA

In LDA documents are treated as a flat, unordered set of distributions across topics without defining any relation between them, thus being unable to capture the level of abstraction of each topic. Hierarchical LDA (HLDA) [18] is a variant of LDA which introduces an hierarchical tree for the topics of the document collection. The higher levels include the broader topics, which become more specific while we are moving closer to the leaves of the tree. HLDA belongs to the non-parametric generative models, since the inference process determines both the topics and the distributions of topics for each document. The HLDA model considers the hierarchies as random variables and produce the hierarchy of the topics as the data arises, through a generative process.

Each node of the tree corresponds to a single topic. The hierarchy is constrained by the number of levels, however the branching factor for nodes is determined by the inference procedure. For each document, a single path from root to a leaf is drawn and the text is generated by the topics across this path. Consequently, it is derived by the topic of the leaf and the topic's abstractions up to the root. To the topics of a path words are assigned, drawn from the documents assigned to that path. The fact that texts are restricted to contain topics from a single path and cannot use the whole topic collection, renders HLDA more inflexible compared to LDA.

HLDA is based on the *Chinese Restaurant Process* (CRP) which is a distribution on

partitions of integers. The CRP assumes the existence of a Chinese restaurant with infinite number of tables. The first customer always selects the first table. The n^{th} customer who arrives at the restaurant, picks a table to sit according to the following distribution:

$$P(\text{sit at an occupied table i|previous customers}) = \frac{n_i}{n-1+\alpha}$$
$$P(\text{sit at the first unoccupied table i|previous customers}) = \frac{\alpha}{n-1+\alpha} \quad (1)$$

where α is a parameter controlling the possibility for a new customer to sit to an occupied table or to select an empty one, and n_i are the customers already seated on table i. After the placement of M customers, a partition of M integers has been derived.

HLDA actually relies on an hierarchical extension of CRP, the *nested Chinese Restaurant Process* (nCRP). The nCRP builds on CRP to organize a countably infinitely many restaurants (with an infinite number of tables) of a city, in a hierarchy. One restaurant has the role of the root restaurant. To the tables of all restaurants, a label is placed to point to another restaurant. Each restaurant can appear to the label of only one table across all restaurants.

Now consider a customer who has visited the above city for L days. The first day chooses a table at the root restaurant in terms of equation (1). The second day, he goes at the restaurant determined by the label of the table he had picked the previous day and draws a table according to (1). After L days, the customer will have drawn a path of tables of length L, starting at the root. After M customers has spent L days in the city, an L-level hierarchy for tables has been constructed. nCRP can be used as a prior for modeling topic hierarchies.

Figure 6 presents the generative process of HLDA in the basis of the nCRP process:

1. Assume c_1 as the root restaurant.
2. For each level l in $1 \dots L$, draw a table from restaurant c_{l-1} using equation (1) and set c_{l-1} to refer to table c_l.
3. For each document d:
   (a) Select a multinomial distribution θ_d over the L levels from symmetric Dirichlet prior α.
   (b) For each word w in document d:
      i. Draw a topic z in $1 \dots L$ from θ_d.
      ii. Draw a word w from the topic corresponding to restaurant c_z.

4.2.5 Twitter-LDA

Twitter-LDA [51] (TLDA) is an extension of LDA designed to handle the shortness of tweets. It assumes that each tweet talks about a single topic. It also hypothesizes that tweets contain topic and background words while the choice between them is ruled by a Bernoulli distribution. The Twitter-LDA generative process is shown in Figure 7 and its steps are described below:

1. Draw a word distribution for background words $\phi^B \sim Dir(\beta)$ and $\pi \sim Dir(\gamma)$
2. For each topic z in $1 \dots Z$ draw a multinomial distribution $_z$ from symmetric Dirichlet prior β.
3. For each user u in $1 \dots U$:
   (a) Select a multinomial distribution θ_u over the Z topics from $Dir(\alpha)$.
   (b) For each tweet d in $1 \dots N_u$
      i. Draw a topic $z_{u,d}$ from θ_u.
      ii. For each word w in $1 \dots N_{u,d}$ in tweet d:
         A. Draw a background word $y_{u,d,n}$ from pi.
         B. Draw a word $w_{u,d,n}$ from ϕ_B if $y_{u,d,n} = 0$ and $w_{u,d,n}$ from $\phi_{z_{u,d}}$ if $y_{u,d,n} = 1$.
         C. Draw a word from the multinomial distribution β_z of topic z over words.

Twitter-LDA is not suitable for the recommendation task we are considering in this work. Even though it associates every user with multiple topics, the individual tweets are assigned to a single topic. Thus, all tweets with the same inferred topic would have the same similarity with the user model, inevitably resulting in numerous ties. The lower the number of distinct topics, the higher the portion of ties in the ranked list. Given that we do not consider any external, contextual information, these ties can only be resolved arbitrarily. As a result, Twitter-LDA is expected to exhibit a performance similar to the random ordering of tweets in the recommendation task we are considering.

4.2.6 Dirichlet Multinomial Mixture

The Dirichlet Multinomial Mixture model (DMM) [37] is a probabilistic topic model. Similarly to TLDA, it establishes a one-to-one correspondence between documents and topics. Figure 8 illustrates the generative process of DMM:

1. For each topic z in $1 \dots Z$, draw a multinomial distribution ϕ_z from symmetric Dirichlet prior β.

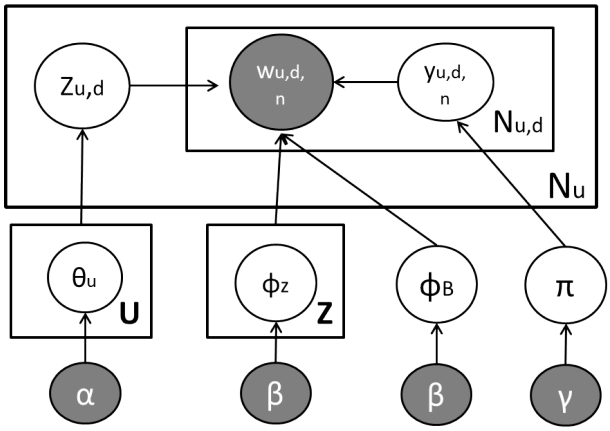

Figure 7: Twitter-LDA generative process.

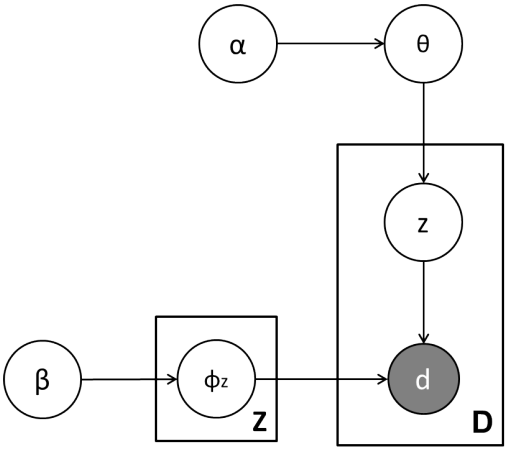

Figure 8: DMM generative process.

2. Select a multinomial distribution θ_d over the Z topics from symmetric Dirichlet prior α.
3. For each document d:
   (a) Draw a topic z from θ.
   (b) Draw a word w from the multinomial distribution ϕ_z of topic z over words.

DMM has also been successfully applied to short texts. In particular, Yin and Wang [50] introduce a Gibbs Sampling algorithm for DMM to solve a short clustering problem. Their model (GSDMM) deals with the intricacies of short texts and infer the number of clusters automatically. However, DMM or GSDMM are not suitable in our recommendation scenario since the same argument with that of TLDA holds.

4.3 Partially Order-Preserving Models

4.3.1 Token N-Grams

A token n-gram is a contiguous sequence of n words. In the token n-grams vector model (TN), the tweet vector comprises one dimension for each of the token n-grams extracted from the entire tweet collection. The value of each dimension is determined by a weighting scheme, where the j-th weight quantifies the importance of the j-th token n-gram for the tweet. For n=1, we have the well-known ”bag-of-words” model which is completely order agnostic as it ignores the relative position of the words in the tweet. On the contrary, for n>2 the model counts the occurrences of consecutive words, thus preserving the order to some extent.

The most common weighting schemes are the following:

1. The *Boolean Frequency* method gives a binary value to the weights, indicating the absence or presence of the corresponding token in the tweet. So, for the tweet t_i we define:
$$w_{ij} = \begin{cases} 1 & \text{,if token } k_{ij} \text{ exists in } t_i \\ 0 & \text{,otherwise} \end{cases}$$

2. The *Term Frequency (TF)* method treats as a weight the occurrence frequency of a token in a document. More formally, for the tweet t_i the w_{ij} can be defined as:
$$w_{ij} = \frac{f_{ij}}{n_t}$$
where f_{ij} is the frequency of the j^{th} token in tweet t_i and n_t is the length of t_i. The frequency is normalized by the tweet length in order to avoid the bias towards longer tweets.

3. The *Term Frequency Inverse Document Frequency* (TFIDF) scheme downscales the term frequencies for the most common tokens in the whole collection which, albeit more popular, they convey less information. For token k, the inverse document frequency can be defined as:
$$idf(k, T) = \log \frac{N_T}{n_k+1}$$
where N_T is the total number of tweets and $n_k = |\{t \in T : k \in t\}|$ Then, for tweet t_i and token k_{ij} we have $w_{ij} = tf(k_{ij}) \times idf(k_{ij}, T)$. A high tfidf-weight is obtained

when the frequency of occurrences of the token in the tweet is high but the token is rare in the whole collection and the weight is getting lower as the token appears in more tweets.

For n-grams comparisons, the most common similarity measures are the following:

1. **Cosine Similarity (CS).** It measures the angle between the two weighted vectors actually normalizing the document length during comparison. For two tweet vector models $M(t_i) = (w_{i1}, \ldots, w_{im})$, $M(t_j) = (w_{j1}, \ldots, w_{jm})$, it is defined as:

$$CS(M(t_i), M(t_j)) = \frac{\sum_{k=1}^{m} w_{ik} w_{jk}}{||M(t_i)||||M(t_j)||}$$

2. **Jaccard Similarity (JS).** It calculates the similarity between two finite sets by dividing the intersection of the sets by their union. Two tweet vectors with boolean weights can be considered as two sets and each vector component indicates the presence or absence of the corresponding token in the set. More formally, for boolean weights, we have:

$$\sum_{e \in G_i} \frac{min(w_{ik}, w_{jk})}{max(w_{ik}, w_{jk})}$$

3. **Generalized Jaccard Similarity (GJS)**. It computes the similarity between two finite sets like Jaccard but also, it accounts for the number of times each element appears in both sets. Two tweet vectors with Term Frequency weights can be considered as two sets and each vector component indicates the frequency of occurrence of the corresponding token in the set; for boolean weights it is the same as Jaccard. It is formally defined exactly as Jaccard does.

4.3.2 Character N-Grams

The character n-gram vector model (CN) is formed of all the substrings of length n contained in the text. The vectors again consist of one weight for each character n-gram encapsulating its significance in the documents. This model is more robust than the token models with respect to spelling mistakes. For example, let us consider two documents [”tweet”,”twete”] with a misspelling in the second one. These texts completely differ in their bag-of-words representations while their 2-gram character models match in 3 out of 4 dimensions conveying more precisely the actual texts’ similarity. The weighting techniques most commonly used are the *Term Frequency* and the *Boolean Frequency*.

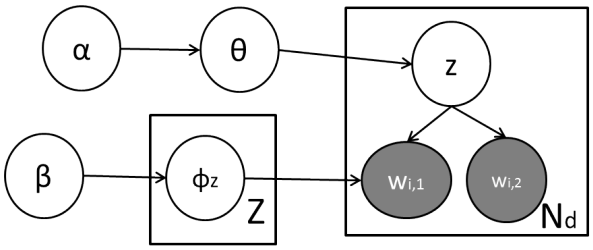

Figure 9: BTM generation process.

4.3.3 Biterm Topic Model

The topic models described above, in order to discover a document's topics, implicitly uncover the word co-occurrence patterns within a document. In short texts like tweets, the sparsity of the patters affects the traditional models' performance. The Biterm topic model [49] (BTM) copes with this problem by explicitly modeling the word co-occurrence for topic learning as well as by utilizing the aggregated word patterns in the whole collection. BTM assumes that the collection of documents (rather than each document itself) consist of a mixture of topics and directly models the biterm generation from these topics. A biterm is an unordered word-pair co-occurring in a short context. In short texts, each document is considered as a context unit while in longer texts a window is given as a parameter to the model.

Figure 9 shows the generative process of BTM:

1. For each topic z in $1 \dots Z$, draw a multinomial distribution ϕ_z from symmetric Dirichlet prior β.
2. For the whole collection, select a multinomial distribution θ_d over the Z topics from symmetric Dirichlet prior α.
3. For each biterm b in the set of biterms B:
   (a) Draw a topic z from θ_d.
   (b) Draw two words from the multinomial distribution ϕ_z of topic z over words.

For the inference of θ and φ, Gibbs Sampling is used. Note that BTM does not contain a generation process for documents. To infer the distribution θ_d for individual documents, Yan et al. employ the formula $P(z/d) = \sum_b P(z/b)P(b/d)$, presuming that the document-level topic proportions can be derived from the topic proportions of the biterms generated from the document.

4.4 Fully Order-Preserving Models

4.4.1 N-gram graphs

The n-gram graphs model represents each document d_i as an undirected document graph G_{di}, containing one vertex for each n-gram derived from the document. The n-grams can be in the form of tokens or characters. Across this work, we use the abbreviations CNG and TNG for token and character n-gram graphs respectively. The vertices are connected with edges, having weights which denote the frequency of co-occurrence of the corresponding n-grams. In that way, the n-gram graphs take into account the closeness of n-grams and add contextual information to the model.

The parameters characterizing a n-gram graph are the following [15]: (i) the minimum n-gram rank L_{min} (ii) the maximum n-gram rank L_{max} and (iii) the maximum neighborhood distance D_{win}. In our experiments we consider only the configuration values Lmin= Lmax= Dwin=n where n $\in \{2, 3, 4\}$ since it has been experimentally proven that they result in graphs conveying enough information and with limited noise [15]. A graph is constructed by connecting with edges the n-grams located within a window D_{win} in the original text. A user model can be derived from the merge of the graphs of the individual documents that represent the user's interests, through the update operator [16].

For graph comparison, we use the following proposed similarity measures [15]:

1. **Containment Similarity (CGS).** It measures the number of common edges contained in two graphs G_i and G_j. It indicates the existence of common sequences of tokens or characters in the original texts and therefore it corresponds to the cosine similarity of vectors with boolean frequencies. More formally, the containment similarity is defined as:

$$CGS(G_i, G_j) = \frac{\sum_{e \in G_i} \mu(e, G_j)}{min(|G_i|, |G_j|)}$$

   where $m(e, G) = 1$ if and only if the edge $e \in G_i$ and 0 otherwise and |G| is the number of edges of graph G.

2. **Value Similarity (GVS).** Besides counting the shared edges, value similarity takes into account their weights as well. It is related to the cosine similarity between vectors

with term frequency weights. It is given by the formula:

$$GVS(G_i, G_j) = \frac{\sum_{e \in G_i} \frac{min(w_e^i, w_e^j)}{max(w_e^i, w_e^j)}}{max(|G_i|, |G_j|)}$$

where w_e^i is the weight of edge e in G_i and $\sum_{e \in G_i} \frac{min(w_e^i, w_e^j)}{max(w_e^i, w_e^j)}$ is a symmetric scaling factor

3. **Normalized Value Similarity (NGVS.** It is a normalized version of value similarity in order to reduce the impact of larger graphs:

$$NGVS(G_i, G_j) = \frac{\sum_{e \in G_i} \frac{min(w_e^i, w_e^j)}{max(w_e^i, w_e^j)}}{min(|G_i|, |G_j|)}$$

4.5 Summary

In this Section, we introduce a novel taxonomy that classifies 12 text-based models as either order-agnostic, partially order-preserving and fully order-preserving models. Whether the model considers the ordering of tokens (or characters) , determines its assignment to each of the three categories. Then, the distinctive features of the 12 models were analyzed in detail; emphasis was put to their internal parameters.

5 EXPERIMENTAL SETUP

5.1 Dataset

All methods and experiments were implemented in Java, version 8. All experiments were conducted in a server with 120GB RAM and Xeon ES-4603 (2.20GHz, 32 cores), running Ubuntu 14.04. They were carried out on a Twitter data set, containing more than 476 million tweets and 71 million retweets, posted by more than 17 million users during the 7-month time period from June, 1 2009 to December 31, 2009 [33]. It includes the complete content of tweets along with the corresponding usernames and timestamps. We recovered the user connections from the publicly available[8] social graph crawled by [31]. We removed the users (together with their tweets) not appearing in the social graph and constrained the remaining users to have more than 3 followers and followees as well as to have retweeted more than 400 posts.

Then, we defined the following three categories of Twitter users: (i) *Information Producers (IP)* are those users who tweet and retweet more frequently than they receive updates from followees, (ii) *Information Seekers (IS)* are those users who are less active compared to their followees, and (iii) *Common Users (CU)* are those who exhibit a balance between the received and posted messages. The ratio of a user's outgoing to her incoming tweets, called *posting ratio*, determines the classification of a user to one of these three categories. From each category, we selected the top 20 users, thus building a dataset with 60 users in total.

In more detail, the higher the posting ratio of a user, the larger the number of her outgoing tweets relative to the number of her incoming tweets. Thus, we defined as IP the 20 users with the higher ratio; ideally, this ratio should be higher than 2, thus indicating that they post twice as many tweets as they receive. For IS, we selected the 20 users with the lowest ratio, which in this case should be lower than 0.5; thus, IS comprises those users that received at least twice as many tweets as those that they published themselves. For CU, we selected the 20 users with a ratio closer to 1.

Note that in practice, the lower ratio among IP users is 1.20, while the higher one among CU users is 1.16, due to scarcity of information providers in our dataset. This means that the two categories are too close, probably introducing noise to the results. For this reason, we also examine a fourth user group, the Pure Information Producers (PIP). PIP includes only those IP users with a posting ratio higher than 2 (9 in total).

[8] https://an.kaist.ac.kr/traces/WWW2010.html

User type	Outgoing messages			Incoming messages		Followers' tweets and retweets	
	total	tweets (%)	average	total	average	total	average
IP	95,833	16	4,792 ± 4,466.41	44,494	2,225 ± 2,435.46	559,568	27,978.4±99,311.70
CU	48,836	33	2,442±4,416.44	49,566	2,478 ± 1,958.82	166,233	8,311.65±12,552.95
IS	47,659	43	2,383±1,388.58	390,638	19,532 ± 11,515.76	665,778	33,288.9±38,436.22
PIP	42,566	11	4,730±5010.33	10,285	1,143±602.42	50,330	5,592±10870.92

Table 2: The dataset used in our experiments.

User Type	Total		Average	
	Negative	Positive	Negative	Positive
IP	1,807	453	90.35 ± 197.48	22.65± 49.63
CU	1,796	465	89.8 ± 156.09	23.25±40.32
IS	2,016	504	100.8 ± 91.63	25.2±22.91
PIP	136	34	15.11±10.15	3.78±2.54
Total (IP+IS+CU)	5,619	1,422		

Table 3: Test dataset

Table 2 shows the technical characteristics of our dataset. We report the total outgoing, incoming and followers' messages for each user type and the tweets percent of the total outgoing posts. We also show the average number of messages per user along with the standard deviation.

As test set for each user, we considered her followees' tweets in our Twitter dataset, i.e., her incoming tweets. The retweeted tweets are considered as *positive examples* and the rest as *negative examples*. In forming the test set, the main difficulty we faced was the sparseness of the positive samples and the class imbalance across time. To retain a reasonable proportion between the test positive and negative examples, we proceeded as follows: for each user, we defined as the training phase the time span that includes 80% of all positive tweets received by the user. All posts within that period compose the training set for that user, while the posterior tweets form the testing set. The testing phase begins when the first one of the 20% most recent positive examples is found in the incoming tweets. In that way, the test set is guaranteed to contain the 20% of all positive tweets. We then sampled the negative data from the testing phase, following the same approach as in [12], in order to reduce the impact of the sparsity of positive examples: for each positive tweet in the testing set, we randomly added four negative ones from the testing phase. The technical characteristics of the test sets per user type are summarized

User Type	T		R		TR		C		CT	
	total	average	total	average	total	average	total	average	total	average
IP	2,524	126.2 ±227.08	5,633	281.65±684.29	8,157	407.85±815.96	5,332	266.6±298.88	7,871	393.55 ± 458.72
IS	2,035	101.75±155.05	2,211	110.55±167.22	4,246	212.3±297.93	4,357	217.85±199.62	6,395	319.75 ± 310.11
CU	1,258	62.9±87.34	1,407	70.35±117.01	2,665	133.25±185.19	26,011	1,300.55±17,50.45	27,271	1,363.55 ± 1800.58
PIP	735	81.67±143.17	3,647	405.22±1016.13	4,382	486.89±1157.25	2,024	224.89±171.17	2,760	306.67±267.86
Total (IP+IS+CU)	5,817		9,251		15,068		35,700		41,537	

(a)

User Type	CR		E		ET		ER	
	total	average	total	average	total	average	total	average
IP	10,965	548.25 ± 807.46	13,113	655.65±919.55	15,637	781.85 ± 1037.87	18,746	1,8746± 937.3
IS	6,568	328.4 ± 280.37	12,791	639.55±716.68	14,826	741.3 ± 804.23	15,002	15,002± 750.1
CU	27,418	1,370.9 ± 1839.35	50,380	2,519±2,818.6	51,638	2,581.9± 2877.71	51,787	51,787± 2589.35
PIP	5,671	630.11±1097.32	7213	801.44±1336.04	7,948	883.11±1475.79	10,860	1,206.67±2345.23
Total (IP+IS+CU)	44,951		76,284		82,101		85,535	

(b)

User Type	F		FT		FR		EF	
	total	average	total	average	total	average	total	average
IP	22,756	1,137.8±1,328.34	25,280	1,264 ± 1400.80	28,389	1,419.45 ± 1542.82	35,869	1,793.45 ± 1676.95
IS	27,752	1,387.6±1,788.65	29,787	1,489.35 ± 1862.14	29,963	1,498.15 ± 1886.20	40,543	2,027.15 ± 2110.25
IC	61,083	3,054.15±2,976.28	62,341	3,117.05 ± 3048.85	62,490	3,124.5 ± 3067.22	111,463	5,573.15 ± 5472.53
PIP	6,359	706.56±535.79	7,094	788.22±619.12	10,006	1,111.78±1356.97	13572	1,508.00±1664.04
Total (IP+IS+CU)	111,591		117,408		120,842		187,875	

(c)

Table 4: Training datasets ordered by the train set size starting from the top table. Each row corresponds to one user type.

Pooling Scheme	T	R	E	F	C	TR	ET	ER	FT	FR	CT	CR	EF
UP	56	60	1,665	7,776	870	60	1,721	1,725	7,832	7,836	926	930	8,564
HP	26,685	70,304	291,659	757,818	163,814	96,354	317,679	360,256	783,721	825,899	190,190	232,866	1,042,758

Table 5: Train sets for each pooling scheme and information source.

in Table 3.

For every combination of a user u, a representation model M and an information source s, we trained a user model $M(u, s)$. For the bag and graph models, i.e., $M \in \{TN, CN, TNG, CNG\}$, we train a separate model per user using the corresponding training set. Thus, the users are represented as weighted vectors (for the TN and CN) or graphs (for the TNG and CNG). For the topic models, i.e., $M \in \{LDA, LLDA, HDP, HLDA, BTM\}$, we train a single model using the training set of all 60 selected users. Then, we take into account only the tweets of u coming from s and infer their distributions over topics using the trained topic model M. Then, the model for the individual user u, $M(u, s)$, is constructed by aggregating these distributions.

Tables 4(a), (b) and (c) summarize the training sets of our experiments. For each information source, we present how many training examples come from each user type. We also show the average number per user along with the standard deviation, for each user type. The last row reports the total number of training tweets per information source, across all users.

Note that for the topic models, we examine three different pooling schemes of the training data: (i) the aggregation on users, called *User Pooling* (UP), where all tweets posted by the same user are considered as a single document, (ii) the aggregation on hashtags, called *Hashtag Pooling* (HP), where all tweets annotated with the same hashtag form a single document (the tweets without any hashtag are treated as individual documents), and (iii) the unpooled technique, called *No Pooling* (NP), where each tweet is considered as an individual document. The pooling method determines the size of the training set. The train set of the unpooled scheme is shown in the last row of Table 4. For HP and UP, Table 5 indicates the train set sizes. Observe that HP has a very large training set size, by far exceeding the total number of tweets for all information sources. This is because the same tweet is assigned to as many aggregated documents as the number of hashtags it comprises.

Concerning pre-processing, we tokenized the raw tweet text on white spaces and punctuation for the token-based models, i.e., all the topic models and the token bag and graph

n-grams. Special rules were applied to squeeze repeated letters and to keep together URLs, hashtags, mentions and emoticons. Also, the 100 most frequent tokens of the training set were removed from the data, as they practically correspond to stop words. Note that we did not apply any language-specific pre-processing technique, such as stemming, lemmatization and part-of-speech tagging, as the dataset we used is multilingual (challenge C2). Note also that we applied the same pre-processing workflow to the character-based models, but we noticed that it significantly degraded their performance. Thus, we did not employ any special pre-processing for them.

5.2 Evaluation Measures

To assess the effectiveness of the recommendation models, we use the Mean Average Precision (MAP) measure exactly as defined in [12]. In particular, the Average Precision (AP) of a user model is the average Precition-at-n (P@n) of all re-tweeted tweets, where P@n is the proportion of the top-n ranked tweets that have been re-tweeted. AP is formally defined as:

$$AP = \frac{\sum_{n=1}^{N} P@n \times RT(n)}{|R|}$$

where RT(n)=1 if and only if n is a re-tweet and 0 otherwise, N is the number of testing tweets and R is the total number of re-tweets. To calculate MAP for a specific user category, we average the AP values over all corresponding users in our dataset.

For each combination of a representation model and an information source, we only report the average MAP over all configurations, defined as the *Mean MAP*, and the minimum and maximum values. We present the MAP values for every user type separately and collectively for all users. A *best configuration* per model achieves the highest MAP computed over all users, across all information sources. For the *best information source*, we calculate per user type, the average MAP across all models for each information source. In that way, we can estimate the effectiveness of each source on each user type. The best source is the one with the highest average effectiveness over all user types. The most *robust* representation model is the one with the smallest MAP deviation across all information sources. As MAP deviation we define the maximum range of MAP values, i.e., the maximum difference between the highest and lowest values, across all information sources.

To estimate the efficiency of each recommendation model, we measured the training time ($TTime$) for each configuration of its parameters that we consider and report the average time. In that way, we show the relative complexity of the recommendation models

Smile	:) , :-) , :) , =) , (: , (: , (-:
Frown	:(, :-(,): ,) : , : (,)-: , >:[, :-c , :c , :-< , :< , :[, :-[, :{
Wink	;-) , ;) , *-) , *) , ;-] , ;] , ;D , :-,
Big grin	:-D , 8-D , 8D , :D , x-D , xD , X-D, XD , =-D , =D , =-3 , =3 , BD
Tongue	>:P , :-P , :P , X-P , x-p , xp , XP , :-p , :p , =p , :-b , :b ,d:
Heart	<3 , </3 , :-* , :*
Surprise	>:O , :-O , :O , :-o , :o , 8-0 , O_ O , o-o , o_ o , O-O
Awkward	>:\\, >:/ , :/ , :\\, =/ , =\\
Confused	% -) , %) , O_ o , o _ O , >:\\, >:/ , :-/ , :-. , :/ , :\\, =/ , =\\, :L , =L , :S , >.<

Table 6: Emoticon labels for LLDA along with their variations.

in practice.

5.3 Parameter Tuning

We applied the representation models to our dataset of 60 users for each distinctive information source, trying a wide range of meaningful parameter configurations, as presented in Table 7. In total, we considered 2873 combinations of user models, information sources and internal configurations. For every combination, we established the following requirements: it should consume less than 32GB RAM (*memory threshold*) and its training should last less than 5 days (*time threshold*). Models not satisfying either of these constraints were considered invalid and, thus, they were ignored. As a result, we totally excluded PLSA from our analysis, since it exceeds the memory threshold. Also we avoided some configuration combinations for HLDA, as their training running time exceeds the time threshold.

For LDA, LLDA and BTM, we set a=50/Z and β=0.01, where Z is the number of topics, since it is reported in [39] that these configurations work well in many different text collections. Regarding LLDA, the selection of tweet-specific labels was based on [39]. We created one label for each hashtag of the training collection, assigned only to the tweets that contain it. Additionally, we treated as labels nine categories of emoticons, i.e., smile, frown, wink, big grin, heart, surprise, awkward and confused; they are presented in Table 6 along with their variations. The @user label was associated to every post in which a user is mentioned as the first word, indicating a direct messaging action. We did not consider the @reply label that is proposed in [39], as our dataset does not contain reply information. The question label was applied to messages comprising a question mark. The questions, the @user labels and all the emoticons except for the big grin, the heart, the surprise and the confused emoticons were frequent in our corpus and thus we considered 10 variations

for each of them as [39] does; for example, the frown label was factored in: :(-0 to :(-9. Because the train set of all information sources include too many hashtags, we filtered out those with high occurrence frequency; for the sources T, R, E and C, we removed those hashtags with frequency less than 15, while for all other sources, we excluded hashtags with frequency less than 30, as these sources involve a larger set of hashtags.

For HLDA, we only varied the values of α and γ. We set fixed values for the hierarchical levels and the pooling schemes, since any other value exceeded the time threshold. For BTM, we selected a fixed number of iterations, namely 1,000, following [49]. For individual tweets, we set the context window for considering two words as a biterm, equal to the size of the tweet itself; for larger texts in user and hashtag pooling, we set the window size equal to 30, because this threshold was the first one for which BTM outperformed LDA in [49]. After that value, the improvement slows down indicating that the larger the distance between two words, the more irrelevant the words of the generated biterms are, corresponding to different topics. Another reason for setting this threshold (30) is that the larger the window size is, the higher the training time gets, degrading efficiency.

Concerning token and character n-grams (TN and CN respectively), the main parameters we fine-tune are the size of n-grams, i.e, the n, the similarity measure for comparing the user model with the individual tweets and the aggregation strategy for building the user model. Note that, not all combinations of parameters are valid for bag models. For example, Jaccard Similarity is applied only with Boolean weights, Generalized Jaccard only with Term Frequency and TFIDF is not combined with character n-grams. Also, the Rocchio algorithm is only applied in combination with those information sources that contain both positive and negative examples, i.e, C, E, TE, RE, TC, RC and, EF.

	#Topics	#Iterations	Pooling	α	β	n	Weighting	Aggregation strategy	Similarity	Fixed parameters	Total
LDA	{50,100, 150,200}	{1000, 2000}	{NP, UP, HP}	50/#Topics	0.01			{centroid, Rocchio}	CS		48
LLDA	{50,100, 150,200}	{1000, 2000}	{NP, UP, HP}	50/#Topics	0.01			{centroid, Rocchio}	CS		48
HDP	{50,100, 150,200}	1000	{NP, UP, HP}	1.0				{centroid, Rocchio}	CS	γ=1.0	24
HLDA		1000	UP	{10,20}	{0.1,0.5}			{centroid, Rocchio}	CS	levels=3	8
BTM	{50,100, 150,200}	1000	{NP, UP, HP}	50/#Topics	0.01			{centroid, Rocchio}	CS	r=30	24
TN						{1,2,3}	{B,TF, TFIDF}	{sum, centroid, Rocchio}	{CS,JS, GJS}		30
CN						{2,3,4}	{B,TF}	{sum, centroid, Rocchio}	{CS,JS, GJS}		21
TNG						{1,2,3}			{CGS, GVS,NGVS}		9
CNG						{2,3,4}			{CGS, GVS,NGVS}		9
											221

Table 7: Configurations of the representation models parameters. NP, UP and HP stand for the unpooled technique, pooling on user and pooling on hashtag respectively. B is the Boolean weighting scheme, TF the Term Frequency and TFIDF the Term Frequency-Inverse Document Frequency. CS, JS and GJS correspond to Cosine, Jaccard and Generalized Jaccard similarity respectively , while CGS, GVS and NGVS to Containment, Graph Value and Normalized Graph Value similarity respectively.

6 EXPERIMENTAL ANALYSIS

In this section, we analyze and compare the performance of all 9 representation models in combination with all 13 tweet information sources in terms of effectiveness (Section 6.1) and time efficiency (Section 6.2). In our evaluation, we take into account the categorization of models as order-agnostic, partially order-preserving and fully order-preserving. We also consider the Twitter challenges (C1) to (C4) that are listed in Section 2.2.2. Finally, we employ two baselines in our analysis, which are independent of the information source: (i) the Chronological algorithm (CHRON), which ranks the testing set (i.e., incoming tweets) in chronological order, and (ii) the Random method (RAND), which orders the tweets randomly.

6.1 Effectiveness

Table 8 presents the MAP values for CHRON and RAND, while the performance of the recommendation models for each user type appears in Figures 10-13. In the latter case, all diagrams have the same scale so as to facilitate comparison. Remember that higher MAP values indicate a better performance, i.e., higher effectiveness.

Starting with all users collectively in Figure 10, we observe that the two fully-order preserving models do not perform equivalently. TNG outperforms all other models across all information sources with respect to the average MAP; all its MAP values are also significantly above the two baselines. The highest average MAP for TNG is achieved by T (0.784), with R (0.741) and TR (0.744) being almost equally effective. Most importantly, TNG maintains a robust performance across all information sources, with its average MAP consistently fluctuating between 0.625 and 0.784. However, for a specific information source, TNG is less robust with respect to its internal configuration, as its difference between its minimum and maximum MAP values rising to 0.281. CNG, though, scores much lower values: 0.368$\leq$mean MAP$\leq$0.477; all MAP values within the range outperform the baselines. Yet, it is more robust than TNG with respect to its internal configuration, as its smallest MAP deviation is just 0.114. This pattern indicates that most configurations of CNG fail to capture distinctive information about the real interests of a user: the aggregation of multiple character n-gram graphs into a user model results in a graph that bears strong similarities with the n-gram graph of many irrelevant, unseen documents. The lower n is, the more intensive is this phenomenon; more bigrams are expected to be shared by positive and negative examples, thus explaining the poor performance for n=2.

	MAP	
	CHRON	RAND
All users	0.140	0.284
IP	0.145	0.322
PIP	0.144	0.353
CU	0.143	0.284
IS	0.132	0.270

Table 8: Effectiveness of baselines in terms of MAP.

The dominance of TNG should be attributed to its ability to represent the relations between token n-grams, thus adding enough contextual information to the model in order to substantially alleviate challenge (C1). Note that TNG deals with challenges (C2) and (C4) only to some extent; it considers misspelled or non-standard token n-grams as totally different n-grams but at the same time, it adds contextual information to them by connecting the n-grams with edges denoting their frequency of co-occurrence. Nonetheless, it is the best model with respect to effectiveness and thus the positive effect of the fully order-preserving property compensates for the limited caution of the Twitter challenges (C2) and (C4). On the other hand, the performance of CNG is counter-intuitive. It partially accounts for (C1) and it fully copes with (C2) and (C4), extracting sequences of characters instead of words, but its effectiveness is much lower than TNG.

Among partially order-preserving models, we observe that TN significantly outperforms the other two models over all information sources. Its highest average MAP amounts to 0.673 (in combination with T), while the highest average MAP for CN is just 0.438 (in combination with RC); for BTM, the highest MAP is even lower. Regarding robustness, there is a large deviation between the maximum and the minimum MAP values for both TN and CN across most information sources. This means that the two models are quite sensitive in terms of their parameter configuration.

It should be stressed at this point that TN is order-agnostic for n=1, while for $n \geq 2$, it is partially order-preserving. Table 9 presents the average MAP values of all users for TN, after separating the configurations with n=1 from those with $n \geq 2$. We observe that, by excluding the bag of words property, TN becomes more effective for all information sources; the highest average MAP for $n \geq 2$ (0.779) is even higher than that of TNG (0.744), albeit to a minor extent.

Regarding BTM, we observe that it exhibits equivalent performance with CN across all sources, while exhibits a significantly more robust performance. Compared to the rest

	Mean MAP, all users												
	T	R	F	E	C	RT	TF	RF	TE	RE	EF	TC	RC
all n	0.644	0.673	0.549	0.558	0.592	0.641	0.549	0.555	0.558	0.568	0.540	0.596	0.602
n=1	0.484	0.460	0.401	0.451	0.475	0.471	0.404	0.426	0.451	0.475	0.437	0.462	0.482
n≥2	0.724	0.779	0.624	0.611	0.651	0.726	0.621	0.619	0.611	0.615	0.592	0.662	0.661

Table 9: Comparison of TNG over configurations with n=1 and n≥2 in terms of the average MAP values of all users.

topic models, BTM achieves slightly better scores for all sources. This should be attributed to its ability to account for word ordering by creating biterms. Another advantage is that it bypasses challenge (C1), capturing topic patterns at the level of entire corpora, rather than the document-level.

Concerning the order-agnostic topic models, we observe uniform patterns across all information sources. Their average MAP values are very low, ranging from 0.265 (for HDP and F) to 0.360 (for LLDA and R); they are also too close, even worse in some cases, to the performance of the RAND baseline. The less robust order-agnostic model is LLDA, as its minimum MAP deviation over its configurations is equal to 0.264; the most robust is HLDA with a minimum MAP deviation of just 0.106. This is also the smallest MAP deviation across all models and information sources; BTM and CNG follow in close distance, with a smallest MAP deviation of 0.109 and 0.114, respectively. The poor performance of order-agnostic models can be attributed to two factors: (i) the loss of contextual information encapsulated in the word ordering and (ii) the sparseness of tweets, i.e., challenge (C1). The traditional topic models count word co-occurrences in document-level to capture topics; thus, the sparse co-occurrence patterns in the short text of tweets reduce drastically their effectiveness.

On average, across all information sources, the partially order-preserving models significantly outperform the average MAP of order-agnostic models by 47%. Similarly, the fully order-preserving models outperform the order-agnostic models by 76% and the partially order-preserving models by 20%. The substantially lower performance of order-agnostic models should be attributed to the order-preserving factor introduced by the other two categories. The fully order-preserving models also increase the effectiveness over partially order-preserving models since they introduce a stronger order-preserving factor; they account for the ordering of pairs of n-grams themselves rather than just the ordering between individual tokens or characters.

Now, we analyze the experimental results for the individual user categories. Note that for IP, we only report the diagrams for their subclass, the PIP (with ratio greater than 2),

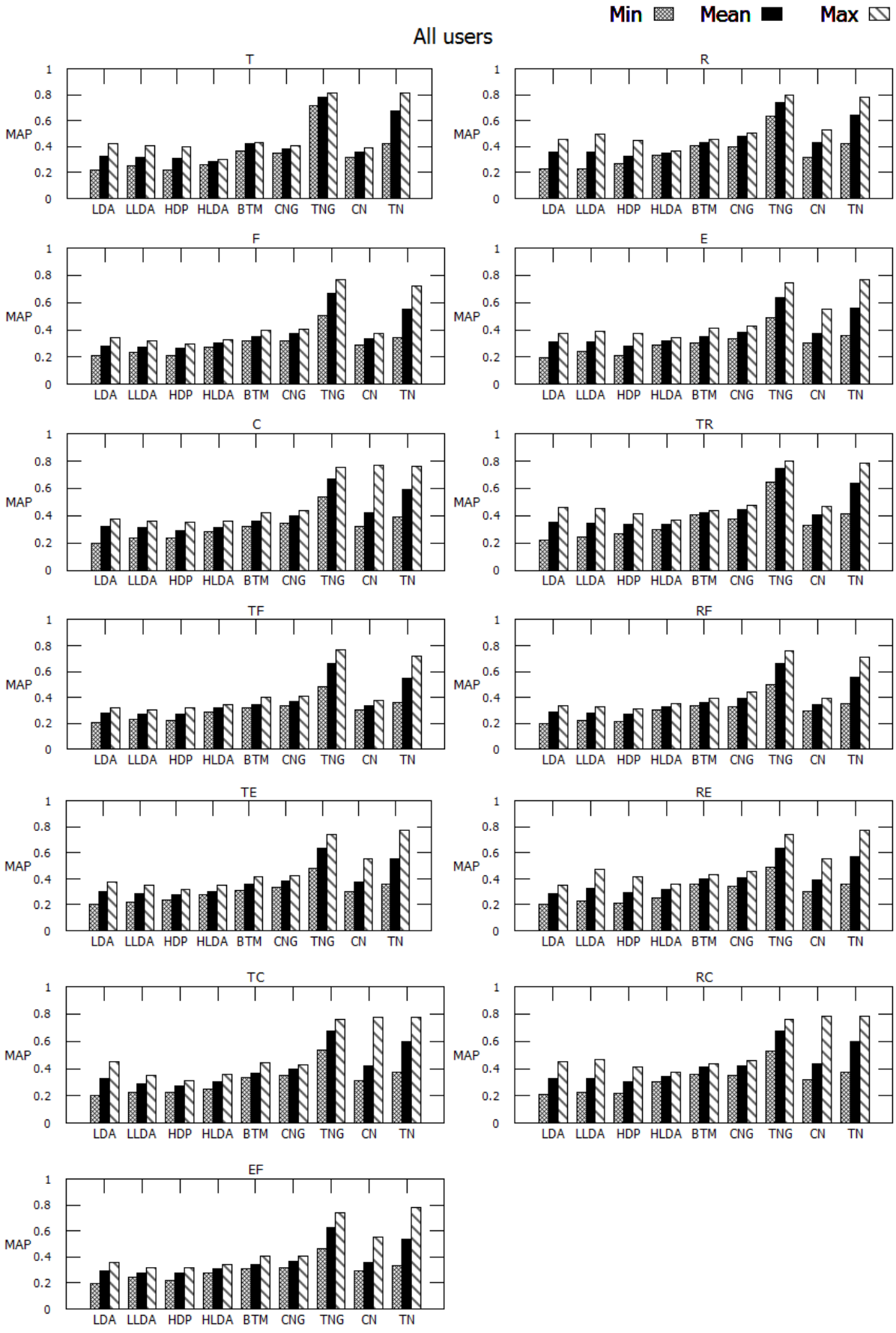

Figure 10: Effectiveness of the 9 representation models in combination with the 13 tweet information sources with respect to the Mean Average Precision across all users. Higher bars indicate better performance.

as their performance was slightly better. In particular, on average, across all models and information sources, the PIP users increase the average MAP value by 5%. Although the increase is not outstanding, it indicates that the users who tweet twice as many times as they receive messages, represent more effectively the IP user category.

The relative performance of models is approximately the same for all user types and for all users collectively; however, the absolute values vary. Firstly, in relation with all users, on average, over all models and information sources, PIP exhibits a higher Mean MAP by 12%, thus the other two user categories introduce noise to the recommendation process. On the other hand, IS degrades the performance by 14% over all users. CU user group have practically equivalent effectiveness with all users; CU reduces the Mean MAP by the minor quantity of 0.9%.

Now we examine the differences between the three user types. In particular, by measuring per user type, the average Mean MAP of each model over information sources, we get the following increases from CU to PIP, IS to PIP and CU to IS: LLDA scores higher values by 26%, 39% and 10% while LDA by 16%, 30% and 12% and BTM by 19%, 39% and 17% respectively. Both HDP and HLDA retain the same performance from IS to CU users. From IS to PIP and from CU to PIP, HLDA scores 15% higher while HDP 25%. CNG raises the average Mean MAP by 1.830%, 28.982% and 26.664%, TNG by 16.124%, 23.155% and 6.055 and TN by 15.910%, 32.156% and 14.016% from CU to PIP, IS to PIP and CU to IS respectively. While CN, similarly to other models, performs by 24.389% and 36.504% better from IS to PIP and CU to IS respectively, it decreases the effectiveness by 9% from CU to PIP.

	All users				IS				CU				PIP		
	Min	Mean	Max		Min	Mean	Max		Min	Mean	Max		Min	Mean	Max
R	0.323	0.457	0.741	**R**	0.290	0.415	0.715	**R**	0.354	0.456	0.691	**R**	0.412	0.525	0.810
TR	0.335	0.448	0.744	**TR**	0.319	0.407	0.723	**TR**	0.321	0.431	0.705	**RC**	0.369	0.497	0.771
T	0.288	0.430	0.784	**T**	0.263	0.383	0.769	**RC**	0.306	0.422	0.672	**TR**	0.359	0.494	0.795
RC	0.307	0.427	0.678	**RC**	0.278	0.373	0.596	**T**	0.263	0.403	0.737	**C**	0.351	0.488	0.786
C	0.293	0.410	0.672	**C**	0.267	0.363	0.600	**RE**	0.279	0.400	0.629	**RE**	0.327	0.470	0.712
TC	0.274	0.406	0.677	**RE**	0.279	0.357	0.573	**C**	0.290	0.397	0.672	**RF**	0.346	0.468	0.759
RE	0.284	0.402	0.639	**TC**	0.255	0.357	0.597	**TC**	0.259	0.383	0.660	**E**	0.357	0.460	0.720
E	0.282	0.392	0.638	**E**	0.266	0.351	0.572	**E**	0.266	0.379	0.632	**F**	0.327	0.459	0.784
TE	0.279	0.387	0.637	**TE**	0.260	0.347	0.572	**TE**	0.255	0.369	0.630	**T**	0.295	0.446	0.841
RF	0.271	0.386	0.660	**TF**	0.237	0.341	0.598	**RF**	0.251	0.368	0.646	**TC**	0.290	0.444	0.798
F	0.265	0.378	0.666	**RF**	0.237	0.339	0.598	**TF**	0.253	0.359	0.648	**TF**	0.322	0.444	0.772
EF	0.278	0.378	0.625	**EF**	0.245	0.336	0.566	**EF**	0.245	0.355	0.617	**EF**	0.340	0.440	0.704
TF	0.269	0.377	0.663	**F**	0.241	0.336	0.600	**F**	0.244	0.353	0.651	**TE**	0.301	0.429	0.718

(a)

R	**TR**	**RC**	**T**	**C**	**RE**	**TC**	**E**	**RF**	**TE**	**F**	**TF**	**EF**
0.463	0.445	0.430	0.415	0.415	0.407	0.398	0.395	0.390	0.383	0.381	0.380	0.377

(b)

Table 10: (a) Min, Mean and Max average MAP values across all representation models, for each combination of information source and user type. Per user type, the values are presented in descending order by the Mean MAP. (b) Average Mean MAP values of Table 10a per information source, over all user types. MAP values are presented in descending order.

Undoubtedly, PIP is the best performing user type, then follows the CU type and finally the IS. On average, across all information sources and user models, PIP exhibits a raise of 30% over IS and 13% over CU while CU 15% over IS. At this point, we have sufficient experimental proofs to point out the token n-gram graphs (TNG) as the best representation model in our recommendation scenario, in terms of effectiveness. TNG, for every combination of user type and information source, outperforms all the other models.

Then, take a look at Table 10. It illustrates, per combination of user type and information source, the average Mean MAP over all representation models. The results are ordered by MAP so as to facilitate the evaluation of each source for each user type.

Table 10b presents the average of the means per row of Table 10a. Actually, these are the average Mean MAP values across all models and user types, for every information source, ordered by the MAP. We deduce that on average, a user retweets (R) is

the best information source with a MAP of 0.463 and it follows the combination of tweets and retweets (TR) with 0.445. Tweets (T) themselves are fourth in the list with 0.415. This result is intuitive since users post and re-post messages that capture their interests. Retweeting is more effective and adds to the effectiveness of tweets; users are more careless as to the messages they post themselves, thus introducing some noise to their tweets, while they select to retweet only those followees' posts that truly reflect their preferences.

Notably, the posts of reciprocally connected users (C) and their combination with a user's retweets are within the five best sources. This adds to the intuition that reciprocally connection is similar to friendship and users are mutually connected when they share common interests to a large extent. We want to emphasize here that C and RC outperform the corresponding sources of followees' posts, i.e., the E and RE. Therefore, the one-way linkage in Twitter, is not an equivalently strong indication of common tastes among people. Yet, RE slightly outperforms TC, probably meaning that combining the followees' posts with the strong source of user retweets, the noise introduced by one-way connections is reduced. The less informative of a user's preferences source is, clearly, his followers' tweets. The three lowest MAP values belong to F (0.381), TF (0.38) and EF (0.377). Thus, our hypothesis on Section 2.2.1 that followers' posts add a lot of noise to the model since users do not actively select them, is verified.

Retweets are also the most effective source for each user type individually. Tweets, although scoring highly for all users, IS and CU, for PIP, they occupy the ninth position. Additionally, for PIP, while C and E are ranked practically in the same level as in the rest user types, when combined with tweets (i.e, the combinations TC and TE), their effectiveness is degraded. Consequently, the tweets of PIP users add noise to the model. This patterns indicates that users who tweet too frequently, often post careless, noisy messages, not actually reflecting their personality; while those with a lower posting ratio, tweet thoughtfully, when they have something important to say.

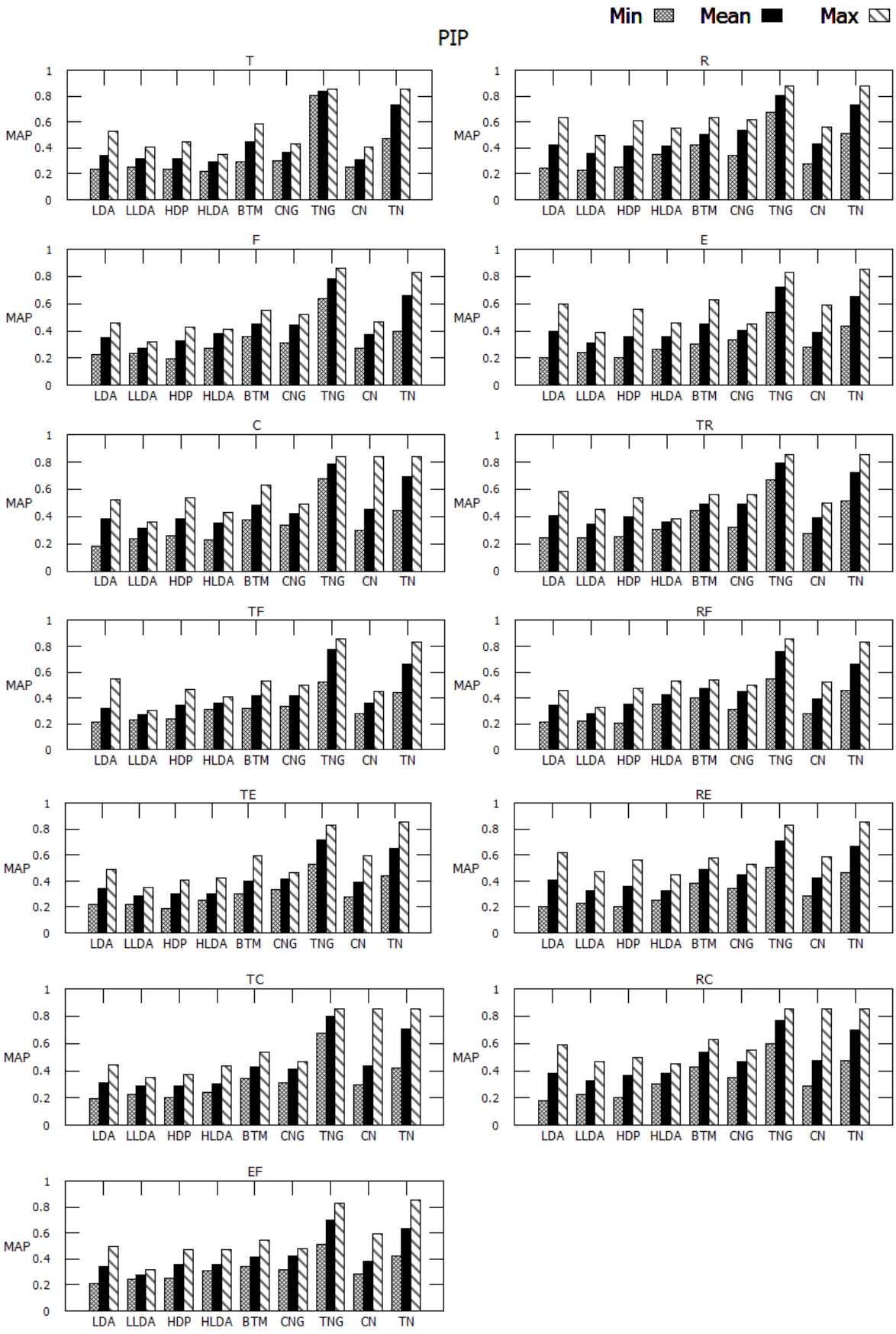

Figure 11: Effectiveness of the 9 representation models in combination with the 13 tweet information sources with respect to the average, minimum and maximum values (across all configurations) of MAP for those Information Producers with ratio greater or equal to 2. Higher bars indicate better performance.

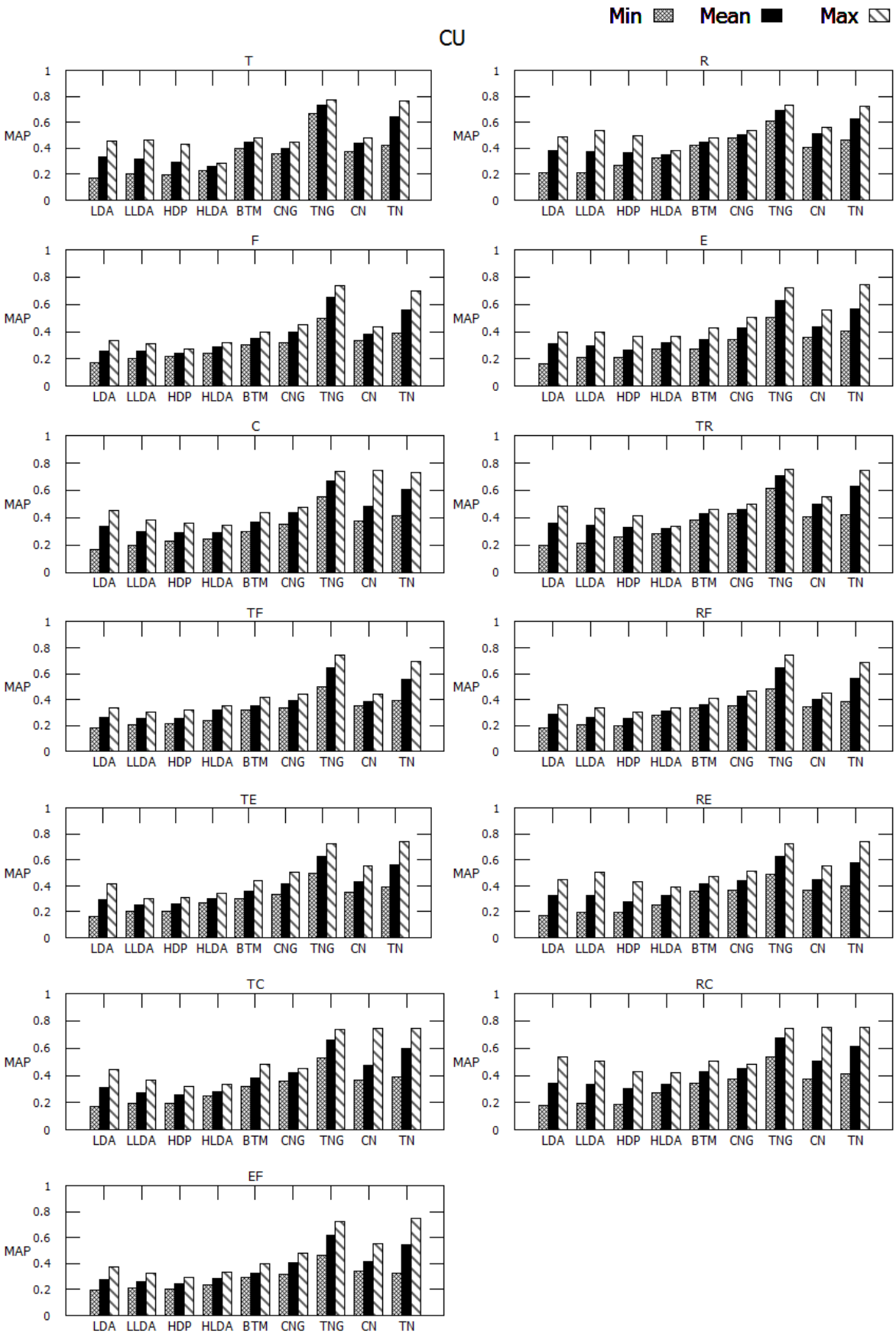

Figure 12: Effectiveness of the 9 representation models in combination with the 13 tweet information sources with respect to the average, minimum and maximum values (across all configurations) of MAP for all the Common Users. Higher bars indicate better performance.

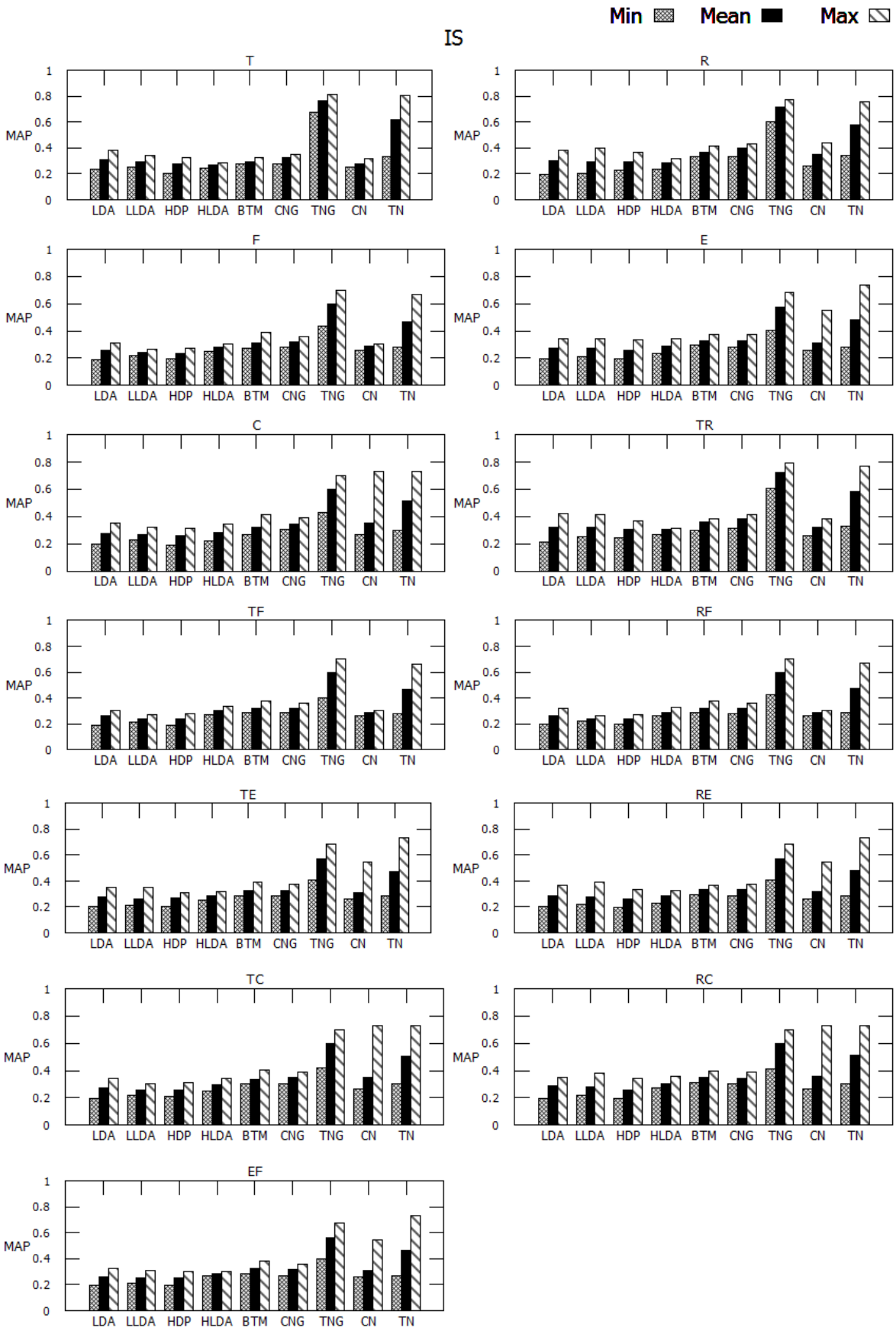

Figure 13: Effectiveness of the 9 representation models in combination with the 13 tweet information sources with respect to the average, minimum and maximum values (across all configurations) of MAP for all the Information Seekers. Higher bars indicate better performance.

	R		T		F	
	BC	MAP	BC	MAP	BC	MAP
LDA	**UP # it=2000 # topic=150**	**0.460**	UP, # it=2000 # topic=100	0.423	HP # it=1000 # topic=100	0.343
LLDA	**UP # it=1000 # topic=200**	**0.494**	UP, it=2000 # topic=50	0.407	UP # it=2000 # topic=150	0.323
HDP	**UP β=0.1 # topic=150**	**0.452**	UP β=0.1 # topic=50	0.400	(1)UP, β=0.1, topic=100 (2)HP, β=0.1, # topic=50	0.298
HLDA	**α=10,γ=1 η=0.1**	**0.368**	α∈{10,20}, γ=0.5 η=0.1	0.304	α10=,γ=1 η=0.1	0.329
BTM	**NP # topic=150**	**0.455**	NP # topic=200	0.436	UP # topic=200	0.396
CN	n=4, TF, CS Sum,Centroid	0.534	n=4, TF, GJS Sum	0.393	n=4, CS {Sum,Centroid}	0.376
TN	n=3 (1) B, sim∈{JS,CS}, Sum (2) w∈{TF,TFIDF}, CS,{ Sum,Centroid}	0.785	**n=3 (1) B, sim∈{JS,CS}, Sum (2) TF, sim∈{GJS,CS},{Sum,Centroid} (3) TFIDF, CS, {Sum,Centroid}**	**0.812**	n=3 (1) TF, CS, {Sum,Centroid} (2) TFIDF, CS, Centroid	0.72
CNG	**n=4 GNVS**	**0.507**	n=4 GVS	0.411	n=4 GCS	0.407
TNG	n=3 sim ∈{GCS,GVS,GNVS}	0.796	**n=3 sim∈{GCS,GVS,GNVS}**	**0.816**	n=3 GVS	0.764

	TR		TF		RF		TE	
	BC	MAP	BC	MAP	BC	MAP	BC	MAP

LDA	UP,# it=1000
	# topic=150
		0.458	UP,# it=2000,# topic=50	0.324	HP, # it=1000
					topic=150
						0.334	UP, # it=1000
							# topic=200
							Centroid
								0.372

LLDA	UP,# it=2000 # topic=150	0.450	UP, # it=1000 # topic=150	0.307	UP, # it=2000 # topic=200	0.329	UP, # it=1000 # topic=200	0.350
HDP	UP,β=0.1 # topic=50	0.414	UP,β=0.1 # topic=50	0.322	UP,β=0.1, # topic=200	0.310	UP,β=0.1 # topic=100 Rocchio	0.318
HLDA	α=20,γ=1 η=0.1	0.364	α=20,γ=1 η=0.1	0.342	α=20=,γ=0.5 η=0.1	0.355	α=20,γ=1 η=0.1 Centroid	0.347
BTM	UP # topic=150	0.437	UP # topic=150	0.399	UP # topic=200	0.395	UP topic=200,Centroid	0.419
CN	n=4, TF, CS, Sum n=4, TF, CS,Centroid	0.468	n=4, TF, CS, Sum n=4, TF, CS,Cent	0.379	n=4, TF, CS, Sum n=4, TF, CS,Cent	0.396	n=3, TF, GJS Rocchio	0.554
TN	n=3 B, sim∈ $\{JS, CS\}$, Sum TF, sim∈ $\{JS, CS\}$, {Sum, Centroid} TFIDF, CS,{ Sum,Centroid}	0.783	n=3 TF,CS,{ Sum,Centroid } TFIDF, CS, {Sum, Cenroid}	0.718	n=3 TF, CS,{ Sum,Centroid} TFIDF, CS, Sum, Centroid	0.715	n=3, TFIDF, CS Rocchio	0.775
CNG	n=4 sim ∈ {GCS,GNVS}	0.472	n=4 GCS	0.407	n=4 GCS	0.439	n=4 GCS	0.427
TNG	n=3 sim ∈ {GCS,GVS,GNVS}	0.797	n=3 GVS	0.764	n=3 sim ∈ {GCS,GVS,GNVS}	0.758	n=3 GVS	0.744

	RE		EF		TC		RC	
	BC	MAP	BC	MAP	BC	MAP	BC	MAP
LDA	HP, # it=1000 # topics=150 Centroid	0.348	HP # it=1000 # topics=150 Rocchio	0.360	UP, # it=2000 # topics=100 Rocchio	0.451	UP,# it=2000 # topics=100 Rocchio	0.451
LLDA	UP, # it=2000 # topics=200 Rocchio	0.470	UP,# it=2000 # topics=200 Centroid	0.318	UP, # it=1000 # topics=200 Rocchio	0.351	UP, # it=1000 # topics=50 Rocchio	0.465
HDP	UP,β=0.1 # topisc=50 Rocchio	0.420	UP,β=0.5 # topics=100 Centroid	0.319	HP,β=0.1, # topics=50 Centroid	0.310	UP,β=0.1 # topics=200 Rocchio	0.414
HLDA	α=10,γ=1 η=0.1 Rocchio	0.362	α=10,γ=0.5 η=0.1 Centroid	0.346	a=10=,γ=1 η=0.1 Centroid	0.356	α=20,γ=1 η=0.1 Rocchio	0.375
BTM	UP # topics=150 Centroid	0.435	UP # topics=150 Centroid	0.406	UP # topics=100 Centroid	0.444	HP # topics=200 Rocchio	0.439
CN	n=4,TF Rocchio	0.553	n=4,TF Rocchio	0.554	n∈ $\{2, 3\}$, TF CS, Rocchio	0.776	**n**∈ $\{2, 3, 4\}$ **TF, CS Rocchio**	0.782
TN	n=3,TFIDF CS,Rocchio	0.775	n=3,TFIDF CS,Rocchio	0.780	n=3,TFIDF CS,Rocchio	0.776	n=3,TFIDF CS,Rocchio	0.783
CNG	n=4 GCS	0.455	n=4 sim∈{GCS,GNVS}	0.404	n=4 GCS	0.431	n=4 GCS	0.459
TNG	n=3 GVS	0.744	n=3 sim=GVS	0.742	n=3 GVS	0.762	n=3 GVS	0.763

	E		C	
	BC	MAP	BC	MAP
LDA	HP, # it=2000, # topic=50	0.375	UP, # it=1000, # topic=50	0.379
LLDA	UP, # it=2000 topic=50, Rocchio	0.389	NP, it=1000, # topic=100, Rocchio	0.361
HDP	UP β=0.1 # topic=50 Rocchio	0.370	HP, β=0.1, # topic=200, Rocchio	0.356
HLDA	α=20, γ=0.5, η=0.5, Rocchio	0.339	α=10, γ=1, η=0.1, Rocchio	0.360
BTM	UP, # topic=150, Cenroid	0.410	UP, # topic=150, Centroid	0.419
CN	n=4, TF,CS, Rocchio	0.554	n∈{2,3}, TF ,CS, Rocchio	0.768
TN	n=3, TFIDF, CS, Rocchio	0.771	n=3, TFIDF, CS, Rocchio	0.763
CNG	n=4, GCS	0.425	n=4, GCS	0.438
TNG	n=3, GVS	0.744	n=3, GVS	0.754

Table 11: The most effective configuration (BC) per representation model and information source along with the corresponding MAP value. With bold are the the best configurations for each model, across all sources. # iter is the number of iterations and sim the similarity. NP, UP and HP stand for the unpooled technique, pooling on user and pooling on hashtag respectively. B is the Boolean weighting scheme, TF the Term Frequency and TFIDF the Term Frequency-Inverse Document Frequency. CS, JS and GJS correspond to Cosine, Jaccard and Generalized Jaccard similarity respectively , while CGS, GVS and NGVS to Containment, Graph Value and Normalized Graph Value similarity respectively.

Finally, Table 11 presents the best configuration per representation model and information source along with the corresponding MAP value. When configurations are numbered, both exhibit the highest MAP but their values cannot be grouped. Bold letters indicate the best configuration for a model over all information sources.

Starting with fully order-preserving models, both TNG and CNG score the highest MAP values for the highest size of n-grams n, i.e. token tri-gram graphs and character four-gram graphs respectively. It seems that the highest values of n capture more effectively distinguishing patterns in the short Twitter texts; they lead to longer n-grams that are also enhanced with patterns of co-occurrence. For four-gram graphs, the graph similarity does not significantly affect the performance; although in Table 11 we only report the best combinations of n and graph similarities for each source, the rest similarities differ by less than 0.004 from the best one.

The same pattern applies to TN and CN as well; the largest n values account more thoroughly for words or characters ordering than the lowest ones, thus enclosing more contextual information about the text. TF is the best weighting scheme for CNG over all information sources. For TNG, the Boolean is the worst among TF and TFIDF appearing only in 5 best configurations. TFIDF is slightly better than TF since it appears in 20 while TF in 15. Rocchio is the prevalent algorithm for building the user model from individual tweets in sources containing both negative and positive tweets, i.e, in E, C , TE, RE, EF, TC, E and C. It boosts the user similarity with his most preferable tweets while making

him dissimilar with the uninteresting ones; actually, it moves the user vector closer to the centroid of the positive examples and farther from the centroid of the negative ones.

Topic models exhibit a uniform behavior regarding the pooling strategy. User Pooling clearly outperforms the other two schemes over the majority of combinations of models and information sources. Hashtag Pooling appears in 7 combinations while the NP technique appears only in 3; in LLDA and C, in BTM and R, in BTM and T. This is the expected pattern, since the topic models are not robust to challenge (C1) of short texts. NP considers each short tweet as an individual documents. UP and HP form larger pseudo-documents, providing richer information to topic models. UP dominance over HP probably is due to the fact that HP forms an insufficient number of long aggregated documents; it considers the tweets not containing a hashtag as individual short documents. Also note that NP yields the best performance for BTM across all information sources (0.455, in R) since BTM is robust to the shortness of tweets. Yet, as with the rest topic models, in BTM, UP outperforms HP and NP. This is not totally unexpected since, as verified in [49], BTM is also effective for long documents.

6.2 Efficiency

Figure 14 depicts the time efficiency of 8 out of 9 representation models in combination with the information sources with respect to TTime. For TN, CN, TNG and CNG, where one model is built per user, TTime is the aggregated modeling time for all users. The TTime for topic models is the time that is requires for training once and collectively on all users' tweets. For each combination of an information source and a user model, we report the average, minimum and maximum TTime over all parameter configurations. Remember that lower values for TTime show a better performance, i.e., higher time efficiency. In general, we expect the TTime to increase with the size of the training set. All diagrams have the same scale to facilitate comparison. Note, though, that the performance of LDA is not directly comparable with those of the other models, since we used its parallel implementation from the open-source library MALLET[9]. Given that all other user models were serially implemented, we report the time efficiency of LDA separately, in the last diagram of Figure 14.

Starting with the fully order-preserving models, we observe a consistent pattern in their relative efficiency: across all information sources, CNG is much slower than TNG, by more than 2 orders of magnitude, on average. Among partially order-preserving models, TN is

[9]http://mallet.cs.umass.edu

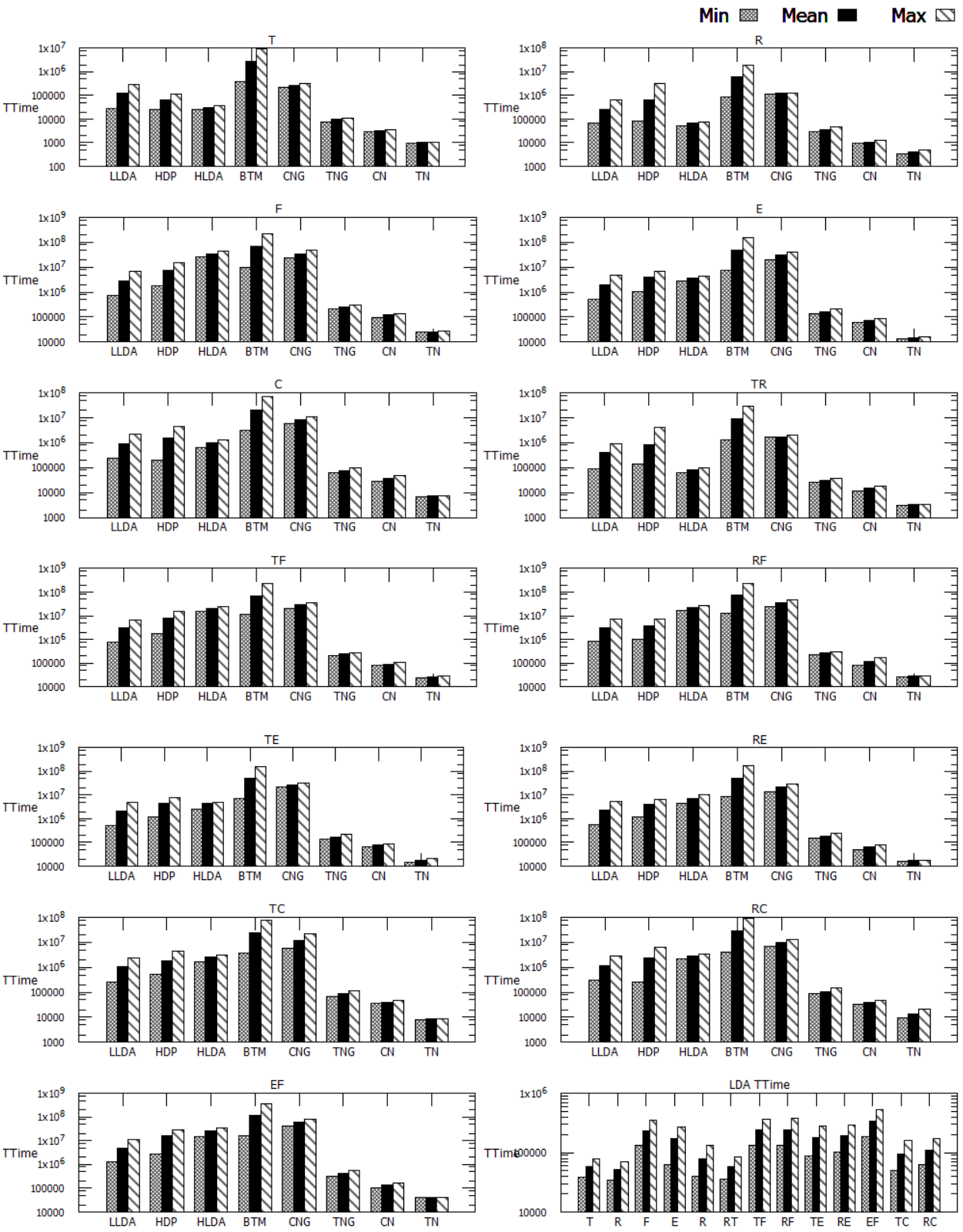

Figure 14: Time efficiency of 8 representation models in combination with the 13 tweet information sources with respect to the average, minimum and maximum values (across all configurations) of TTime in msec. The last diagram shows separately the TTime for LDA over all information sources. The y-axis is logarithmic. Lower bars indicate better performance.

consistently the fastest one over all information sources. It is also more efficient than TNG and all other models, due to the sparsity of tweets: the shorter a textual document, the fewer tokens it involves and the faster the processing gets. CN is slower than TN, since the character n-grams result in a larger feature space than token n-grams. Yet, it follows TN in a close distance. All bag and graph models are affected by the training set size in terms of efficiency; in general, as the number of training tweets for an information source increases, the number of the extracted token (or character) n-grams gets higher and TTime becomes larger. Refer to Table 4 for the ordered trainsets per information source. Only some information sources that differ by less that 1000 tweets are excepted from this rule; for example, CNG is faster in RC (44951) than in TC (41537).

BTM is significantly slower than TN and CN: in all cases its mean TTime is at least 3 orders of magnitude higher. It is also the most time-consuming model over all models and sources; the larger the training set, the worse its performance gets. This outcome is expected, since the main overhead of BTM is the topic selection for the biterms of the whole training document collection (see the generative process of BTM in Section 4.3.3): the more training tweets there are, the more biterms are generated.

Concerning the order-agnostic category, there is no model that dominates the others in terms of efficiency. LLDA is faster than HDP except for T, but their mean TTime differs by much less than 1 order of magnitude in all cases. HLDA is the slowest of the three in all sources apart from R, C, E, TR and TE. Note that for HLDA, we excluded the most time-consuming configurations, i.e., the values greater than 3 for the hierarchical levels and the pooling strategies NP and HP, due to the time threshold. Thus, the results favor HLDA and its actual time efficiency should be much worse.

We also observe that all topic models exhibit a higher TTime than the rest models, over all sources. The only exception is CNG which is slower than LLDA, HDP and HLDA for all sources except for F; yet, it is faster than BTM in all cases.

6.3 Summary

In this section, we thoroughly described the setup of our experiments. We reported the technical characteristics of our dataset in Table 2 as well as the training sets and testing sets sizes in Tables 4, 5 and 3. Table 7 shows all the configurations of the representation models we examine.

Further, we analyzed our experimental results in terms of effectiveness and time efficiency. Figures 10-13 illustrate the performance per model and information source for all

users collectively and each user category separately. Figure 14 depicts the time efficiency per user model and source with respect to TTime. We concluded that the best performing representation model is the token n-gram graphs while user type is the Pure Information Producer and the most effective information source is the user retweets.

7 CONCLUSIONS

Our systematic experimental analysis of 9 textual user representation models, in a popular recommendation scenario, leads to several remarkable conclusions.

First, the order preserving factor between words or characters, considerably affects the performance in terms of effectiveness. The strongest the order preservation, the greater the improvement on Twitter challenges and the higher the achieved effectiveness. This is verified by the increase in Mean MAP achieved by fully order-preserving models over the other two categories, as well as by partially order-preserving models over order-agnostic ones. Token N-gram graphs, a novel representation technique that adds contextual information to the model by connecting pairs of token n-grams, exhibits the highest Mean MAP across all information sources and user types.

Second, according to our results, the use of topic models is not justified in the ranking scenario we consider; their Mean MAP fluctuates between the low values of 0.265 and 0.360, being close to the random baseline. Besides, their computational cost is the highest with respect to training time (TTime). Their only advantage is their robustness across configurations; yet, it cannot pay off their significantly poor effectiveness and efficiency.

Third, the Twitter source that is used for modeling determines the recommendation performance. A user's retweets are the most effective source for accurately representing the user's interests. It outperforms the others, not only across all users, but also for each user type separately. Reciprocally connection, notably, yields better results than the one-way following linkage. The follower's tweets are the less effective source and perform poorly, even combined with other more effective sources like retweets or tweets.

Fourth, the relative performance of models across the three different user types exhibits similar patterns. However, the absolute MAP values differ; the pure information producers (PIP), i.e., users with posting ratio higher than 2, score the highest MAP value on average, over all sources. PIP also outperform all users collectively. The second best performing type is the common users (CU); though, CU perform equivalently with all users, on average, across information sources.

As future work, we intend to perform the same comparisons between representation models in curated documents such as news articels and scientific papers, which are long texts with standard formal vocabulary and low levels of noise. We could also examine user-generated content such as blog posts or forum posts, which are longer and less noisy that the raw tweet content. In addition, we plan to involve, in our recommendation scenario, more advanced topic models that go beyond the bag-of-words assumption. For

example, we could use the Topical N-grams [47] which models not only topics, but also topical phrases as well as the topic model presented in [20] which captures both semantic and syntactic relations between words.

ABBREVIATIONS-ACRONYMS

AP	Average Precision
BTM	Biterm Topic Model
BU	Balanced Users
C	Reciprocally connected users' messages
CN	Character N-grams
CNG	Character N-gram Graphs
CS	Cosine Similarity
DMM	Dirichlet Multinomial Mixture
DP	Dirichlet Process
E	Followees' messages
EF	Followees' and followers' messages
F	Followers' messages
GCS	Graph Containment Similarity
GJS	Generalized Jaccard Similarity
GNVS	Graph Normalized Value Similarity
GVS	Graph Value Similarity
HDP	Hierarchical Dirichlet Process
HLDA	Hierarchical Latent Dirichlet Allocation
HP	Hashtag Pooling
IP	Information Producers
IS	Information Seekers
JS	Jaccard Similarity
LDA	Latent Dirichlet Allocation
LLDA	Labeled Latent Dirichlet Allocation

MAP	Mean Average Precision
nCRP	nested Chinese Restaurant Process
NP	No Pooling
PIP	Pure Information Producers
R	A user’s retweets
RC	A user’s retweets and reciprocally connected users’ messages
RE	A user’s retweets and followees’ messages
RF	A user’s retweets and followers’ messages
T	A user’s tweets
TC	A user’s tweets and reciprocally connected users’ messages
TE	A user’s tweets and followees’ messages
TF	A user’s tweets and followers’ messages
TF	Term Frequency
TFIDF	Term Frequency Inverse Document Frequency
TN	Token N-grams
TNG	Token N-gram Graphs
TR	A user’s tweets and retweets
Twitter LDA	Twitter Latent Dirichlet Allocation
UP	User Pooling

BIBLIOGRAPHY

[1] Fabian Abel, Qi Gao, Geert-Jan Houben, and Ke Tao. Analyzing user modeling on twitter for personalized news recommendations. In *International Conference on User Modeling, Adaptation, and Personalization*, pages 1–12. Springer, 2011.

[2] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering*, 17(6):734–749, 2005.

[3] David Alvarez-Melis and Martin Saveski. Topic modeling in twitter: Aggregating tweets by conversations. In *Tenth International AAAI Conference on Web and Social Media*, 2016.

[4] Marko Balabanović and Yoav Shoham. Fab: content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72, 1997.

[5] Daniel Billsus and Michael J Pazzani. A personal news agent that talks, learns and explains. In *Proceedings of the third annual conference on Autonomous Agents*, pages 268–275. ACM, 1999.

[6] David M Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, 2012.

[7] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[8] Danah Boyd, Scott Golder, and Gilad Lotan. Tweet, tweet, retweet: Conversational aspects of retweeting on twitter. In *System Sciences (HICSS), 2010 43rd Hawaii International Conference on*, pages 1–10. IEEE, 2010.

[9] William B Cavnar, John M Trenkle, et al. N-gram-based text categorization. *Ann Arbor MI*, 48113(2):161–175, 1994.

[10] Jilin Chen, Werner Geyer, Casey Dugan, Michael Muller, and Ido Guy. Make new friends, but keep the old: recommending people on social networking sites. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 201–210. ACM, 2009.

[11] Jilin Chen, Rowan Nairn, Les Nelson, Michael Bernstein, and Ed Chi. Short and tweet: experiments on recommending content from information streams. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1185–1194. ACM, 2010.

[12] Kailong Chen, Tianqi Chen, Guoqing Zheng, Ou Jin, Enpeng Yao, and Yong Yu. Collaborative personalized tweet recommendation. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 661–670. ACM, 2012.

[13] Yajuan Duan, Long Jiang, Tao Qin, Ming Zhou, and Heung-Yeung Shum. An empirical study on learning to rank of tweets. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 295–303. Association for Computational Linguistics, 2010.

[14] Hugo Jair Escalante, Thamar Solorio, and Manuel Montes-y Gómez. Local histograms of character n-grams for authorship attribution. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 288–298. Association for Computational Linguistics, 2011.

[15] George Giannakopoulos, Vangelis Karkaletsis, George Vouros, and Panagiotis Stamatopoulos. Summarization system evaluation revisited: N-gram graphs. *ACM Transactions on Speech and Language Processing (TSLP)*, 5(3):5, 2008.

[16] George Giannakopoulos and Themis Palpanas. Content and type as orthogonal modeling features: a study on user interest awareness in entity subscription services. *International Journal of Advances on Networks and Services*, 3(2), 2010.

[17] Fréderic Godin, Viktor Slavkovikj, Wesley De Neve, Benjamin Schrauwen, and Rik Van de Walle. Using topic models for twitter hashtag recommendation. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 593–596. ACM, 2013.

[18] DMBTL Griffiths and MIJJB Tenenbaum. Hierarchical topic models and the nested chinese restaurant process. *Advances in neural information processing systems*, 16:17, 2004.

[19] Thomas L Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National academy of Sciences*, 101(suppl 1):5228–5235, 2004.

[20] Thomas L Griffiths, Mark Steyvers, David M Blei, and Joshua B Tenenbaum. Integrating topics and syntax. In *Advances in neural information processing systems*, pages 537–544, 2004.

[21] John Hannon, Mike Bennett, and Barry Smyth. Recommending twitter users to follow using content and collaborative filtering approaches. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 199–206. ACM, 2010.

[22] Thomas Hofmann. Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 289–296. Morgan Kaufmann Publishers Inc., 1999.

[23] Liangjie Hong and Brian D Davison. Empirical study of topic modeling in twitter. In *Proceedings of the first workshop on social media analytics*, pages 80–88. ACM, 2010.

[24] Wouter IJntema, Frank Goossen, Flavius Frasincar, and Frederik Hogenboom. Ontology-based news recommendation. In *Proceedings of the 2010 EDBT/ICDT Workshops*, page 16. ACM, 2010.

[25] Akshay Java, Xiaodan Song, Tim Finin, and Belle Tseng. Why we twitter: understanding microblogging usage and communities. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, pages 56–65. ACM, 2007.

[26] Ou Jin, Nathan N Liu, Kai Zhao, Yong Yu, and Qiang Yang. Transferring topical knowledge from auxiliary long texts for short text clustering. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 775–784. ACM, 2011.

[27] Rong Jin, Luo Si, and ChengXiang Zhai. Preference-based graphic models for collaborative filtering. In *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence*, pages 329–336. Morgan Kaufmann Publishers Inc., 2002.

[28] Ioannis Kanaris, Konstantinos Kanaris, Ioannis Houvardas, and Efstathios Stamatatos. Words versus character n-grams for anti-spam filtering. *International Journal on Artificial Intelligence Tools*, 16(06):1047–1067, 2007.

[29] Pavan Kapanipathi, Fabrizio Orlandi, Amit P Sheth, and Alexandre Passant. Personalized filtering of the twitter stream. 2011.

[30] Balachander Krishnamurthy, Phillipa Gill, and Martin Arlitt. A few chirps about twitter. In *Proceedings of the first workshop on Online social networks*, pages 19–24. ACM, 2008.

[31] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is Twitter, a social network or a news media? In *WWW '10: Proceedings of the 19th international conference on World wide web*, pages 591–600, New York, NY, USA, 2010. ACM.

[32] Su Mon Kywe, Tuan-Anh Hoang, Ee-Peng Lim, and Feida Zhu. On recommending hashtags in twitter networks. In *International Conference on Social Informatics*, pages 337–350. Springer, 2012.

[33] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, June 2014.

[34] Chunliang Lu, Wai Lam, and Yingxiao Zhang. Twitter user modeling and tweets recommendation based on wikipedia concept graph. In *Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[35] Rishabh Mehrotra, Scott Sanner, Wray Buntine, and Lexing Xie. Improving lda topic models for microblogs via tweet pooling and automatic labeling. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 889–892. ACM, 2013.

[36] Raymond J Mooney and Loriene Roy. Content-based book recommending using learning for text categorization. In *Proceedings of the fifth ACM conference on Digital libraries*, pages 195–204. ACM, 2000.

[37] Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. Text classification from labeled and unlabeled documents using em. *Machine learning*, 39(2-3):103–134, 2000.

[38] Michael J Pazzani, Jack Muramatsu, Daniel Billsus, et al. Syskill & webert: Identifying interesting web sites. In *AAAI/IAAI, Vol. 1*, pages 54–61, 1996.

[39] Daniel Ramage, Susan T Dumais, and Daniel J Liebling. Characterizing microblogs with topic models. *ICWSM*, 10:1–1, 2010.

[40] Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D Manning. Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 248–256. Association for Computational Linguistics, 2009.

[41] Krishnan Ramanathan and Komal Kapoor. Creating user profiles using wikipedia. In *International Conference on Conceptual Modeling*, pages 415–427. Springer, 2009.

[42] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186. ACM, 1994.

[43] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, 34(1):1–47, 2002.

[44] Bracha Shapira, Francesco Ricci, Paul B Kantor, and Lior Rokach. Recommender systems handbook. 2011.

[45] Mark Steyvers and Tom Griffiths. Probabilistic topic models. *Handbook of latent semantic analysis*, 427(7):424–440, 2007.

[46] Yee Whye Teh, Michael I Jordan, Matthew J Beal, and David M Blei. Hierarchical dirichlet processes. *Journal of the american statistical association*, 2012.

[47] Xuerui Wang, Andrew McCallum, and Xing Wei. Topical n-grams: Phrase and topic discovery, with an application to information retrieval. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 697–702. IEEE, 2007.

[48] Jianshu Weng, Ee-Peng Lim, Jing Jiang, and Qi He. Twitterrank: finding topic-sensitive influential twitterers. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 261–270. ACM, 2010.

[49] Xiaohui Yan, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. A biterm topic model for short texts. In *Proceedings of the 22nd international conference on World Wide Web*, pages 1445–1456. ACM, 2013.

[50] Jianhua Yin and Jianyong Wang. A dirichlet multinomial mixture model-based approach for short text clustering. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 233–242. ACM, 2014.

[51] Wayne Xin Zhao, Jing Jiang, Jianshu Weng, Jing He, Ee-Peng Lim, Hongfei Yan, and Xiaoming Li. Comparing twitter and traditional media using topic models. In *European Conference on Information Retrieval*, pages 338–349. Springer, 2011.

[52] Xianke Zhou, Sai Wu, Chun Chen, Gang Chen, and Shanshan Ying. Real-time recommendation for microblogs. *Information Sciences*, 279:301–325, 2014.