Exploiting Social Media Network Structure to Improve User Profiles for Short-Text-Based Recommender Systems

Abdullah Alshammari

A thesis submitted in partial fulfilment of the requirement of the University of Brighton for the degree of Doctor of Philosophy

May 2019

Abstract

Social media platforms have today become an important part of people's lives. People share their daily activities and interests through these platforms and also read about others' activities. The increasing use of social media has led to rapid growth in the amount of shared information, and consequently has caused the challenge of information overload. This causes difficulties for users in filtering this huge amount of shared data in order to find and extract the information they need. This consumes the user's time, increases information clutter and reduces their satisfaction. To manage this challenge, there is a need for personalisation services such as recommender systems. One popular social networking site is Twitter; it provides a rich environment for shared information that can help with recommender system research. However, short-text-based recommender systems based on Twitter activity suffer from a lack of reliable data. One solution to this problem is to build more powerful user profiles that reflect the recent interests of the user and consequently provide more accurate recommendations. A prompt way of building such a profile is to gather the user's recent activities from their Twitter timeline. However, many users do not provide enough up to date data to build such a profile. Several researchers have tried to overcome this problem by enriching the profile via external textual sources such as Wikipedia. However, external sources may not be able to provide valid data that reflect the user's interests and consequently the recommendation quality can be affected .

This research studies short-text social media user modelling by utilising explicit and implicit relationships. This approach aims to build personal profiles through a different method, using tweets from the user's Twitter network to provide more accurate recommendations. Our method exploits a Twitter user's explicit and implicit relationships in order to extract other users' important tweets that will help in building the personal profile. These relationships were identified by our proposed influence algorithm, which is a new way of measuring the influence rule from the user's perspective rather than the influencers' perspective. The usefulness of this proposed method is validated by implementing a tweet recommendation service based on the content-based recommender system mechanism, and by performing offline evaluation on a real dataset of 40 users collected from Twitter. The proposed user profiles are compared against

other profiles, used as a baseline, based on similarity and distance metrics to check the effectiveness of the proposed method. Our proposed method shows increased performance gains in reflecting the user's interests by recommending more accurate items.

The main contribution of this work is an influence algorithm and a framework that is able to identify influential friends of a user (people that the user follows) more accurately than other influence and similarity algorithms, and then use them as sources in enriching the user's profile via their tweets. It also helped us to enrich the profile with tweets that were collected from other sources, which were found via explicit and implicit relationships. As a result, more accurate recommendations can be delivered.

Acknowledgments

Finishing the journey of my PhD, I would like to take a chance to express my appreciation to my parents, family members and my friends who helped me reaching this goal and for their continuous support, encouragement, trust, love and motivation through my study.

I would also like to express my sincere gratitude to my great supervisors Dr Stelios Kapetanakis, Dr Roger Evans and Dr Nikolaos Polatidis for continuous support during my PhD study and related research, and for their patience, motivation and immense knowledge. Their guidance helped me during all the time of research and writing of this thesis.

During the PhD journey, I had cancer and I was having treatment for about two years. I would like to take this occasion to thank the King Fahad Medical City in Riyadh for their comprehensive work in helping cure me of the cancer. I would like to say many thanks to Dr Abdulaziz Algahtani, who conducted the surgery in a professional manner.

I would like to thank the University of Brighton for supporting me in conducting this study and providing me with the facilities to create this thesis. Also, many thanks to the university of Hail and the Saudi Cultural Bureau for their financial support during this journey.

Declaration

I certify that this work has not been accepted in substance for any degree and is not concurrently being submitted for any degree other than that of Doctor of Philosophy (Ph.D.) being studied at the University of Brighton. I also declare that this work is the result of my own investigations except where otherwise identified by references and that I have not plagiarised the work of others.

_____

Abdullah Alshammari

Supervisors

_____

Dr. Stelios Kapetanakis

_____

Dr. Roger Evans

_____

Dr. Nikolaos Polatidis

TABLE OF CONTENTS

List of Publications

Alshammari, A., Kapetanakis, S., Evans, R., Polatidis, N., & Alshammari, G. (2018). User modeling on Twitter with exploiting explicit relationships for personalized recommendations. Paper presented to the 18th International Conference on Hybrid Intelligent Systems. Porto. 13-15 December.

Alshammari, A., Polatidis, N., Kapetanakis, S., Evans, R., & Alshammari, G. (2019). Personalized recommendations on Twitter based on explicit user relationships modelling. Paper presented to the 24th UK Academy for Information Systems International Conference. Oxford. 9-10 April.

List of Figures

List of Tables

1. Chapter One

1.1. Introduction

Real-time web application systems have grown rapidly over the years by attracting millions of users and generating substantial volumes of content and traffic. Such growth has caused a ripple effect in information comprehension and management, since such systems can lead to information overload. A lack of mechanisms for personal customisation, content filtering and user-oriented selecting criteria tends to lead to user dissatisfaction with the interactive process, due to the poor quality of selected items. Interaction is frustrating, time-consuming and unproductive. The experience and knowledge of people are often not sufficient for dealing with the large amount of usable information. Recommender systems have been created to solve these problems (Ricci, Rokach and Shapira, 2015).

Recommender systems are important and their usage can be seen in several real-time web applications. They are beneficial in offering users items or objects that match their needs and interests. Using a recommender system in a website or social platform can have many benefits, such as personalisation, cross-selling, keeping clients informed and increase client retention. Recommender systems can achieve user satisfaction and save users time by providing them with useful objects (Ricci, Rokach and Shapira, 2015).

Today, the real-time web is growing as a new technology that allows users to communicate and share information in multi-dimensional contexts such as Twitter, which is a very well-known social media micro-blogging platform used by hundreds of millions of users. It is a social network that allows users to post and exchange short messages, called tweets, of up to 280 characters (Vosoughi, 2015). It has become a very important source of shared information and breaking news, and it works quickly and effectively. It can also be considered the new face of social networking platforms that present relationships based on the following strategy, and this makes it different from classic platforms which are based on reciprocal relationships, such as Facebook. This strategy is that relationships in Twitter can be social, informational or both,

because users not only follow other users to maintain social links, but also to receive interesting information generated by others. These features of Twitter lead to the possibility that it could be used as the main source for characterising a specific user who participates in a network of relationships and interactions such as building a profile that reflects his interests (Abel, Gao, Houben and Tao, 2011; Vosoughi, 2015).

Other social web services offer specific information based on their domain, such as a social media platform that provides only music or events. On Twitter, tweets are not specified to a certain domain but they can participate in different topics and interests, and this makes Twitter a valuable resource for research, such as for example into recommender systems (Abel et al., 2011). Much research has exploited Twitter to model users and also to build user profiles, in order to use them in recommender systems. The increasing number of users and shared messages on Twitter encourage such services, which help to recommend useful tweets and users to follow, thus solving the overload problem.

Abel, Gao, Houben and Tao (2013) found that short-term tweets reflect a user's interests better than long-term tweets, when user profiles are built. They also suggest that using external textual sources can raise the performance of a recommender system because tweet length is short and some users do not provide enough recent tweets to build a profile. This research focuses on the possibility of exploiting explicit and implicit relationships between Twitter users (social networks around users) in order to enrich the user's profile with more recent tweets and then improve the performance of short-text-based recommender systems as Dias-Agudo, Jimenez-Diaz and Recio-García (2018) reported that exploiting social network information with feedback data (e.g. ratings) can raise the accuracy of recommendations. It explores the generation of a strategy for building user profiles that can be used in different applications.

This chapter describes the motivation, aims and objectives of the current study, the research questions, and its contributions to knowledge.

1.2. Motivation

The challenge of information overload attracts researchers from different disciplines both within computer science, including in human computer interaction (HCI), and outside, such as in social science and cognitive science, to tackle the problem and provide different solutions. One field that needs to be investigated is exploiting networks of relationships between users in short-text social media (i.e. Twitter) in order to characterise specific users and improve the performance of recommender systems that are based on short-text activities (i.e. tweets). The short-text-based recommender systems are the systems that rely on short-text documents such as tweets. In the obvious way, characterising a single user who generates time series of short-text data based on their history and behaviour can be achieved by collecting history data that the user has generated themselves (see Figure 1). However, in order to acquire sufficient information for profiling, this method may need to go a long way back into the past, and thus might not form a very coherent set of information.

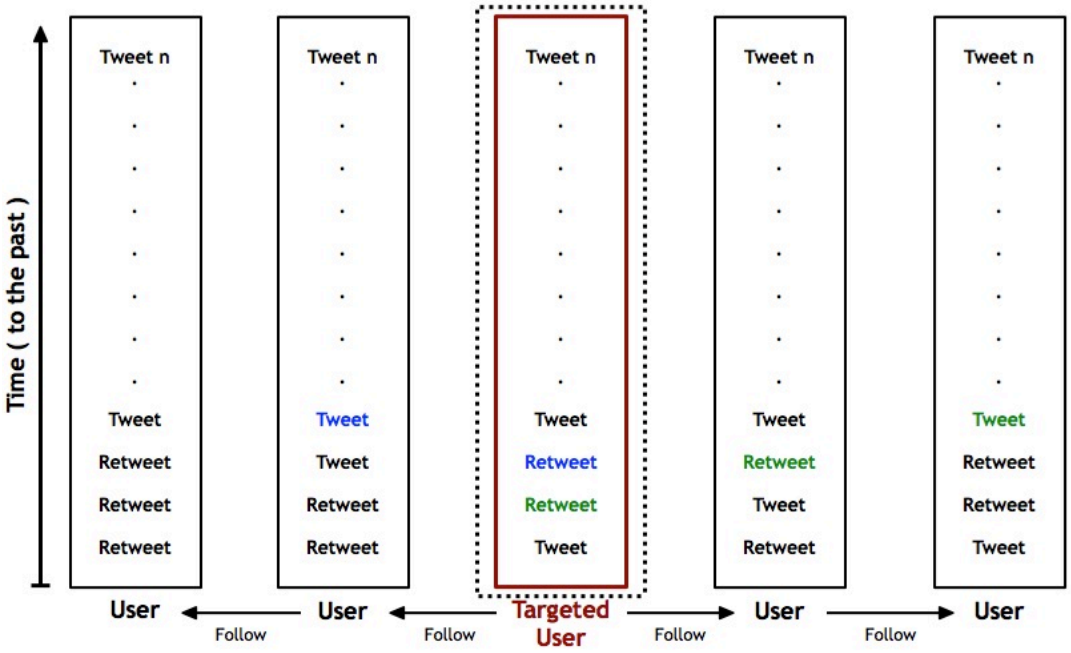

Figure 1. The typical way of building the profile of a specific user.

According to Abel, Gao, Houben and Tao (2013), short-term user profiles – from the last week, for instance – perform better than the complete profile in recommender systems. Piao and

Breslin (2016) show that using a decay function in long-term profiles, which gives a higher weight to recent interests than to older interests, showed better performance for delivering recommendations than long-term profiles without the decay function. The problem is that for many users there is not enough data in their recent activities from which to create a reliable user profile. To solve the problem of a lack of data in a user's Twitter profile, some research has exploited external sources to enrich user profiles, such as Wikipedia, and other studies exploit the shared URLs in a user's timeline to enrich the data in the user profile. These methods can be useful to supply a user profile with more information in order to improve the accuracy of a recommender system. However, data gained from external sources might have no relevance to the user's interests and this might affect the performance of the recommender system. Also, many users do not provide enough URLs in their tweets to enhance their profiles.

In order to address this lack of data and to make sure the collected data are relevant to the user, this research proposes to use explicit links between users (e.g. following links) to expand the set of recent relevant activities (see Figure 2). It will then explore implicit relationships between users (e.g. user similarity) to expand the set with further recent relevant data. Implicit relationships refer to artificial links between users that do not have clear explicit links (e.g. following each other) but might have similar behaviour or common interests (e.g. they follow the same influential people; see Figure 3). The advantage of this method is that there is more recent data to choose from and there is a more coherent story about why it is related, which will allow us to improve the performance of short-text-based recommender systems.

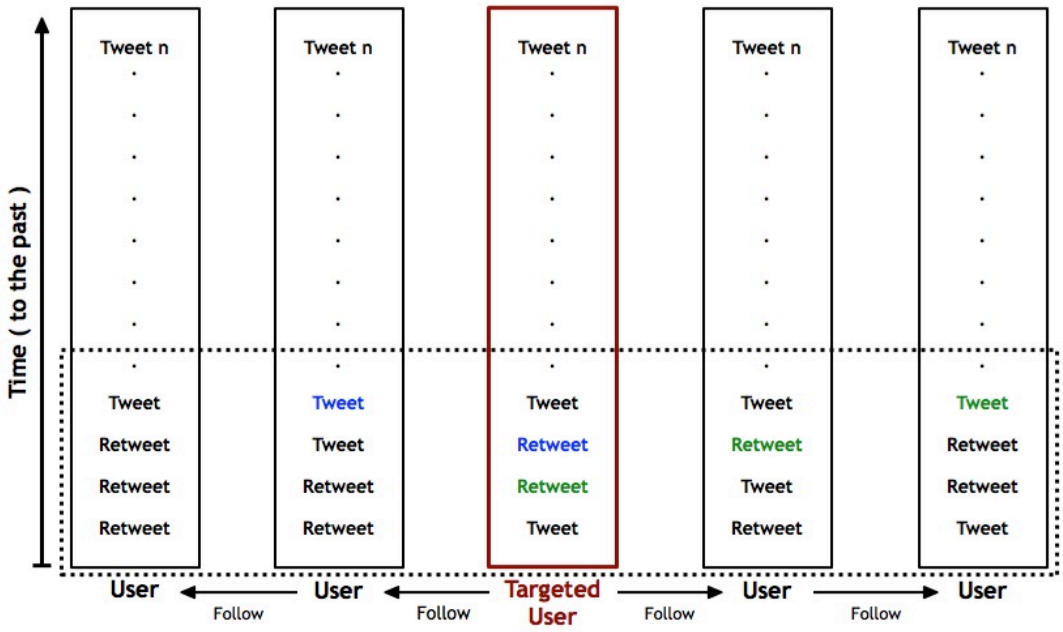

Figure 2. Exploiting explicit links between users in the proposed technique in order to collect more recent data related to the targeted user.

This approach was developed using Twitter, which provides a rich space of explicit relationships in which to develop these ideas. There is also a range of previous research with which to compare our research, and other content-based recommender systems that can be used to evaluate the effectiveness of these new user profiling techniques.

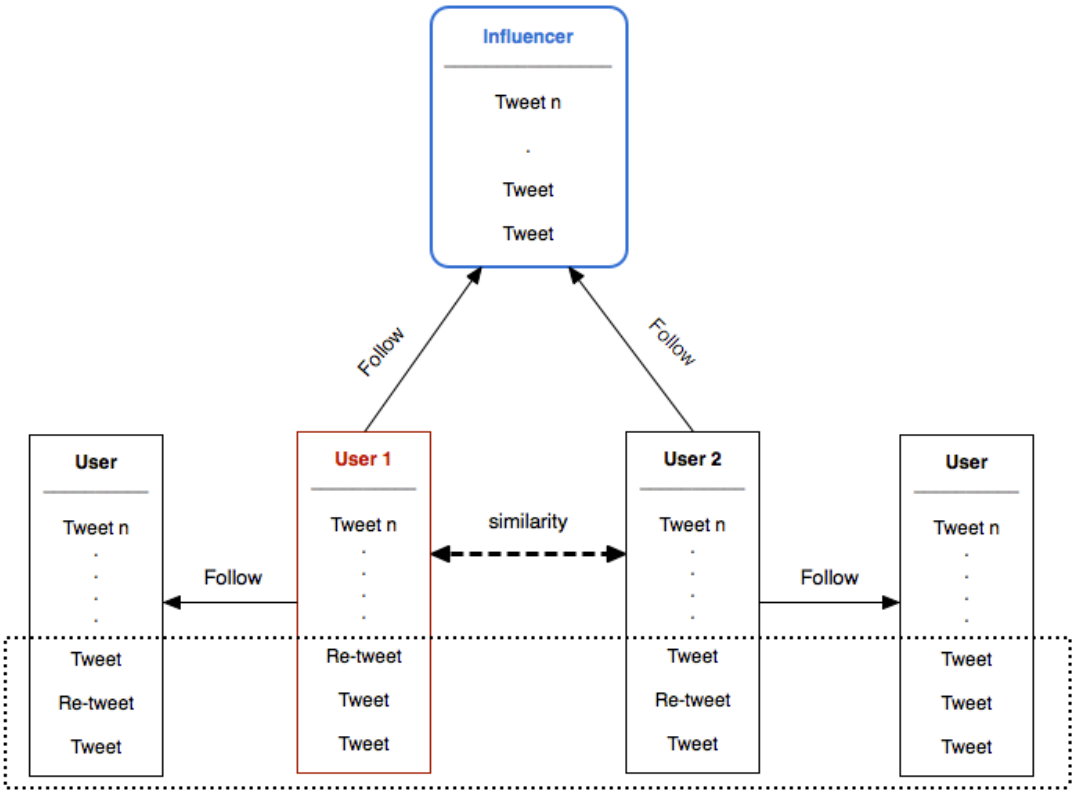

Figure 3. Types of implicit relationship between users that the proposed techniques consider.

1.3. Research questions

Is it possible to improve the performance of short-text-based recommender systems by using explicit and implicit relationships between social media users?

1. Is it possible to extract and utilise explicit relationships among social media users to improve the performance of short-text-based recommender systems?
2. Is it possible to extend such an approach to exploit the implicit properties of a wider social media network around its users?

1.4. Objectives

1. To improve the performance of short-text-based recommender systems by exploiting explicit relationships between social media users.
2. To improve the performance of a short-text recommender system by exploiting the implicit properties of the wider network around the user.

1.5. Main contribution to knowledge

Recommender systems based on short-text activities, Twitter activities for instance, face challenges in building a strong profile and then delivering more accurate recommendations. This research makes a contribution to existing knowledge by improving the accuracy of short-text-based recommender systems and solving certain limitations that have been raised in the literature. To the best of our knowledge, this research is the first approach to mine and exploit different relationships between users within a network in order to improve the accuracy of short-text-based recommender systems. The key contributions are as follows:

1. Redefining the influence rule between users in order to help in collecting relative data that helps to raise the accuracy of recommender systems.
2. Using the new influence rule to find hidden networks around the user (implicit relationships) and then using its members' activities to build the user profile.
3. Modelling users' network activities (explicit and implicit networks) by building different profiles to find the profile that best reflects the user's interests.

1.6. Thesis outline

Chapter 2 provides information about the related work and the background of the areas that the thesis is based on. Chapter 3 provides full details about the proposed methodology. After that, chapter 4 clarifies the experiment, the results and the discussion. Finally, the chapter 5 concludes the thesis and the future work is provided in this chapter.

2. Chapter Two: Related Work and Background

2.1. Recommender systems

Recommender systems have become a very important field of research since the first papers on collaborative filtering were published in the mid-1990s. Recommender systems' roots can be traced back to extended work in many fields, such as information retrieval, cognitive science, forecasting theories, approximation theory, consumer choice modelling in marketing and management science. Recommender systems appeared as an independent research area when researchers started concentrating on the problems of recommendations that depend on a rating structure. The recommendation problem was decreased to the problem of rating estimation for items that a user has not seen. Such estimations are usually based on other item ratings given by the user and some other information. Once the ratings of unrated items can be estimated, items with the highest estimated rating can be recommended to the user (Adomavicius and Tuzhilin, 2005).

2.1.1. Definition

Recommender systems are tools that help users find relevant items. They can help users to recommend similar products to other users. An idea was proposed of how artificial intelligence assists people in sharing relevant information; in reality, people usually ask others for recommendations based on their experience (Resnick and Varian 1997). Burke (2002) described a recommender system as a system that can effectively lead users to interesting products by personalising recommendations based on the user's interests. In the development of recommender systems, developers have produced many systems within various domains in order to improve the recommendations. Theoretically, however, recommender systems suffer from certain limitations that need to be improved in order to raise the quality of the recommendations they provide.

Xiao and Benbasat (2007) describe recommender systems as software agents that extract individual consumers' interests and preferences and then make recommendations accordingly. They aim to support and improve the quality of the decisions that consumers make while searching for and selecting products online.

Ricci, Rokach and Shapira (2015) define recommender systems as software tools and mechanisms that are used to provide suggestions to users in order to help them in decision-making processes, such as deciding what products to buy, what book to purchase or what music to listen to.

Recommender systems are software tools and techniques that provide suggestions for items that are most likely to be in the circle of interest of a particular user. They try to predict what the most interesting products or services are for users based on their preferences. In order to complete this computational task, recommender systems gather information from users regarding their preferences, which are explicitly extracted (e.g. by the user rating items) or implicitly derived by interpreting the user's actions. For example, navigation to a specific product page can be considered an implicit sign of interest for the items shown on that page (Ricci, Rokach and Shapira, 2015).

2.1.2. Motivation

The increasing amount of information available over the Internet has led to an increased focus on research into recommender systems. Understanding user characteristics and knowing item information can help to recommend items to a user after reasoning the user's needs. Recommender systems have proved their strength in overcoming the information overload problem by filtering and matching aspects that are likely to be in the area of interest of a particular user. User profiles are applied in order to predict items based on similar users who have rated the items in the past. User profiles allow recommender systems to search a user's history and recognise their interests and preferences. Recommender systems have been shown to work very well at predicting appropriate items that match a user's interests. They have been used in many different areas, such as e-learning, e-tourism, e-government and e-commerce (Ricci, Rokach and Shapira, 2015; Lu et al., 2015).

Various techniques have been applied and considered in terms of recommendation methods. However, some challenges remain unsolved due to the increasing number of users and items over time. By considering similarity functions, recommender systems have the ability to personalise recommendations. As a result, on-demand applications might profit from this type of system in terms of earning customer satisfaction. This might have a considerable impact on commercial applications by helping users to find relevant items. For instance, 60% of items streamed on Netflix are chosen based on personalised recommendations (Lü et al., 2012).

2.1.3. The aim of recommender systems

One of the main aims of recommender systems is to help users find relevant items based on their interactions and behaviours. In addition, they aim to increase user satisfaction by recommending relevant items. In order to achieve this, recommender systems find different ways to search for and deliver interesting items, and thus solve the information overload problem correlated with increased use of the Web. They also provide a chance for companies which apply recommendations to increase their profits (Lü et al., 2012).

2.1.4. The importance of recommender systems

The main function of recommender systems is to determine documents relevant to users and then suggest recommendations that match their needs. Recommender systems also have other functions, such as checking the importance of a web page (e.g. looking at page position in the results list of a query) or exploring the different uses of a word in a collection of documents. With the rapid growth of shared information in recent years, recommender systems play an increasingly important role in solving the problem of information overload. The user of a recommender system can benefit from personalising his/her needs and filtering items in order to receive recommendations based on his/her actions and information (Ricci, Rokach and Shapira, 2015).

The importance of recommender systems differs between service providers and user perspectives. Ricci, Rokach and Shapira (2015) outline different reasons why service providers may want to use the technology of recommender systems:

- **Increase the number of sold items:** Being able to sell additional items compared to those items that are sold without any type of recommendation is very popular for commercial recommender systems. Increasing the number of sold items can be achieved because the recommended items can match the user's needs and wants. In general, the main goal of using a recommender system is to increase the conversion rate.
- **Sell more diverse items:** In this function, the recommender system enables the user to choose items that may be difficult to find without a recommendation. For example, in a film recommender system such as Netflix, it could be difficult to rent all the items in the catalogue without a recommender system, as opposed to just the most popular ones. Thus, a recommender system recommends films that are not so popular to users who may have an interest in them.
- **Increase user satisfaction:** The experience of the user can be improved by a well-designed recommender system. Finding the recommendations interesting, relevant and properly designed will make the user enjoy using the system. The integration of effective (i.e. accurate) recommendations and a usable interface will increase the user's satisfaction. As a result, system usage and the probability of accepting the recommendations will increase.
- **Increase user fidelity:** When a regular user visits a website, the user should be recognised as an old customer and the website should treat them as a valuable visitor. This is a normal feature that is produced by recommendation systems, which take the information obtained from previous interactions to provide recommendations. Then, the longer the user interacts with the system, the more accurate and specialised the user's model becomes. Recommended items can be effectively customised to correspond to the user's preferences.
- **Better understand what the user wants:** Another significant function of recommender systems, which can be useful in other applications, is a description of the user's preferences, which is either gathered explicitly or predicted by the system. This knowledge might be re-used by the service provider for other goals, such as management of stock or production. For example, in the travel field, travel organisations can use and

analyse data collected by a recommender system (user transactions) to advertise a particular region to new customers or to provide a specific type of promotional message.

Important motives are mentioned above as to why service providers want to include recommender systems in their systems. However, users may also want recommender systems that support their tasks or goals. As a result, such a recommender system has to balance the needs of these two sides and produce a valuable service for both. Ricci, Rokach and Shapira (2015) outline eleven common tasks through which a recommender system can help users to achieve their needs:

- **Find some good items:** Giving a user recommended items as a ranked list with predictions of how much the user would like them, for instance on a scale from one to five stars. Many commercial systems use this as their main task. However, the predicted items are not shown in some systems.
- **Find all good items:** Some users are satisfied when a system recommends to them all the items that match their needs. In such cases it not enough to just find some good items. This happens when the number of items is comparatively small, or when the recommender system's mission is critical, such as in medical or financial applications. In these cases, all possible recommended items are examined carefully, and consequently the user might take advantage of the recommender system's item rankings or of further explanations that are provided by the recommender system.
- **Annotation in context:** Relying on a user's long-term preferences, a recommender system provides an existent context (e.g. a list of items) and emphasises some of the entries. For example, a TV recommender system might give notice of which TV shows are worth watching, to be written in the electronic program guide.
- **Recommend a sequence:** Giving recommendations of a sequence of items is much better than focusing on just generating a single recommendation. For example, recommending a book on recommender systems after having recommended a book on data mining.
- **Recommend a bundle:** A group of items can be suggested by a recommender system because they fit well together. For example, a travel plan may consist of different

attractions, destinations and accommodation services that are situated in a limited area. Users then consider and select a single travel destination from these diverse alternatives.

- **Just browsing:** In this task, the user looks at a catalogue without any intention of purchasing an item. The recommender system's task is to help the user to browse the items that interest him/her during that particular browsing session. Adaptive hypermedia techniques also support this task.
- **Find credible recommenders:** Some users play with recommender systems to see how good they are at making recommendations. As a result, some recommender systems might also offer certain functions that let the users examine their behaviour.
- **Improve a user profile:** This task is linked to the ability of the user to input data to a recommender system about what he/she likes and dislikes. This task is necessary for providing personalised recommendations. The user would be given the same recommendations that would be given to an 'average' user if the system does not have specified knowledge about him/her.
- **Express oneself:** Some users have no interest in recommendations at all. The important thing to them is that they are allowed to participate in rating items and to express their beliefs and opinions. The satisfaction of the user from that activity may still perform the role of holding the user to the application.
- **Help others:** Some users feel happy providing information (e.g. their ratings for items) because they believe that the community profits from their contribution. This could be a key motivation for entering information into a recommender system that is not used routinely. For example, in car recommender systems, a user who has already bought a new car might give his/her rating to the system to be used by other users, rather than using them for the next time he/she buys a car.
- **Influence others:** In web-based recommender systems, there are users who influence other users to purchase specific products. Otherwise, the system could be used by malicious users to locate specific items.

2.1.5. Major challenges in recommender systems

2.1.5.1. Sparsity

The rapidly increasing usage of recommender systems has led many commercial recommender systems to use large datasets. This large amount of data, however, causes the sparsity problem, which affects the performance of the recommender system. One of the examples of the sparsity problem is in the collaborative filtering method, in which recommendations are made based on the past preferences of users. New users need to rate enough items in order to allow the system to give them accurate recommendations based on their preferences; if they have not done so, the sparsity problem arises. Some researchers overcome this problem by introducing another algorithm to the system or using hybrid systems (Thorat, Goudar and Barve, 2015; Lü et al., 2012).

2.1.5.2. Scalability

The scalability problem happens when the number of users and items both increase rapidly. For example, Twitter has millions of users who browse it daily, providing massive amounts of information. The system must respond to a user's requirements immediately and then scale their needs and preferences to give them a high level of recommendations (Thorat et al., 2015; Lü et al., 2012).

2.1.5.3. Diversity

A good recommender system should recommend diverse items in order to help users explore new and different items that might interest them. However, relying on a single filtering algorithm (e.g. recommending only highly rated items) in such a system may lower the accuracy of the recommendation process. Researchers have solved this problem by developing hybrid systems that use more than one filtering algorithm, thus improving the accuracy of the recommendation process (Thorat et al., 2015; Lü et al., 2012).

2.2. Modern social media platforms (Twitter as example)

Twitter is a popular communication tool due its features such as accessibility, speed and ease of use. It is used for many purposes such as reading about breaking news, sharing interests and discussing events. In addition to posting messages ('tweeting'), Twitter's key features are retweeting, favouriting (liking) , followers, followees (friends) and verified users. A retweet is a repost or forward of a tweet by another user, whereas a favourite (like) is a signal that a user likes a tweet. Followers are the users who see a user's tweets and retweets, whereas followees (friends) are other people who the user follows, and as a result sees their tweets and updates. These features create explicit relationships between users in Twitter's social network. A verified user is a user that is confirmed by Twitter to be a real person (Vosoughi, 2015).

2.2.1. Social network sites

Hughes, Rowe, Batey and Lee (2012) state that social network sites are the most popular and the fastest growing type of Internet site; they have changed online information transfer and social interaction. They can be defined as collections of user profiles that can be shared with others. The main idea behind social network sites is that users generate content by themselves that expresses their interests, opinions, activities and so on. This user-generated content helps to allow users to communicate with others online and also to express themselves (Ellison, 2007). Social media includes various types such as social network sites, blogs and content communities. Social network sites include a range of different forms, such as tweets on Twitter, status updates on Facebook and videos on YouTube (Smith, Fischer and Yongjian, 2012).

Social networking sites can be divided into two types in the term of interactions in the social relationship network structure: two-sided (reciprocal) and one-sided. Facebook is an example of the two-sided type and Twitter is the one-sided type. The social relationships network of Facebook is considered an old form, whereas Twitter is considered a modern form (Hughes et al., 2012).

On Facebook, interactions occur among the members of a controlled network of relationships (e.g. a network of friends), and to join this network, a friend request is needed and has to be

accepted. These interactions include posting status updates, posting images, posting comments and tagging others. For example, user A and user B cannot see each other's status updates until they become friends and the relationship between them is reciprocal. Recently, Facebook have slightly changed the way of interacting between users as they added the following strategy feature alongside with the reciprocal feature. However, not all ordinary users using this feature but it is used by popular users or organisations pages such as newspapers (Davenport, Bergman, Bergman and Fearrington, 2014). Facebook allows users to build full profiles as a first step that include occupation, religious and political views, relationship status, favourite films and songs, and so on (Hughes et al., 2012). Facebook allows users to create or maintain social capital, keep updated with other users' lives and communicate with others (Smith, Fischer and Yongjian, 2012). Twitter is different from Facebook. Links on this social network are directed (one way) and it is not necessary to be reciprocal. The core of Twitter is different from Facebook as users do not build full profiles and interactions can be seen as one-way. For example, user A follows user B but user B does not follow user A; in this case, user A can see the posts by user B whereas user B cannot see the posts by user A. The information shared on Twitter is public and everyone can see it, unless the user makes their account private. Twitter is predominantly used to share information, daily activities, opinions and news, while users of Facebook tend to post information about themselves to find friends. Twitter allows huge space for users to form various social networks that matter to people, made out of the patterns of interaction between a user and their friends or acquaintances, rather than a list created only from declared friends. Furthermore, there might be hidden social networks among users (no clear or direct links between users) on Twitter, and these networks need to be discovered and then use them in how they are useful for modelling users. The most widely shared posts on Twitter are text, and this give researchers a rich field for developing technologies based on text, such as text mining (Huberman, Romero and Wu, 2008; Hughes et al., 2012; Smith, Fischer and Yongjian, 2012; Davenport et al., 2014).

2.3. User profiling (modelling)

User profiling or modelling is a special component of any recommender system that represents each user's important information in order to provide personalised recommendations (Brusilovsky and Millán, 2007). The procedure of collecting the information about a single user or a group of users alongside their interests, preferences and/or usage data is commonly called user modelling (Brusilovsky and Millán, 2007; Gauch, Speretta, Chandramouli and Micarelli, 2007).

Brusilovsky and Millán (2007) explain that the term user model sometimes appears as user profile in the literature. Consequently, they are often interchangeable. However, the difference between a user model and a user profile is that a user model includes more general and varied information about the user (i.e. different user features), whereas a user profile commonly contains personal information about the user alongside his/her interests. Gauch et al. (2007) distinguish user profile as an instance of a user model.

Mobasher (2007) clarifies that user modelling/profiling is the first step in personalisation and is one of the fundamental factors in the success of a personalisation system. Users' behaviour, interaction patterns and feedback with the system have to be constantly collected and accurately processed for the purpose of providing up-to-date, accurate and useful adaptive services such as recommendations and search result filtering.

2.3.1. User modelling dimensions

Brusilovsky and Millán (2007) illustrate that users' interests, knowledge, goals/tasks, background and style are the most common features of a user model. In addition, the user's context is significant when the concern is with mobile adaptive systems.

2.3.1.1. Interests

The user's interests are the most significant feature of a user model/profile, especially in adaptive information filtering and retrieval systems, as well as in web recommender systems (Brusilovsky and Millán, 2007). The user's interests are generally extracted from web document

content which the user interacts with, and the user profile is updated in response by adding new interests or modifying existing ones (Gauch, Speretta, Chandramouli and Micarelli, 2007).

2.3.1.2. Knowledge

According to Brusilovsky and Millán (2007), the user's knowledge is applied in recommender systems with the aim of providing personalised content, navigation and presentation to the user. This feature is mutable as it could be increased (i.e. learning new concepts) or decreased (i.e. forgetting or misunderstanding some concepts). Therefore, continual change in the user's knowledge should be taken into account if the recommender system relies broadly on the user's knowledge. User knowledge is a substantial part of most existing recommender systems.

2.3.1.3. Goals/tasks

Kaplan, Fenwick and Chen (1993) demonstrate that the user's goal and task information answer the question of exactly what the user's needs are within the system in their current task (i.e. the purpose of their current interaction with the system). This is the most frequently changeable feature of a user model due to the changing of the interaction goal in every session or even several times in a single session (Brusilovsky and Millán, 2007).

2.3.1.4. Background

Brusilovsky and Millán (2007) define user background as a user's previous experience, and it is a constant feature. Moreover, it cannot be changed, or might be changed over a very long period of time. Background information is used in many recommender systems that include interactions between the user and the system. This feature is used repeatedly for providing personalised content (Gates, Lawhead and Wilkins, 1998) and adaptive search (Vassileva, 1996).

2.3.1.5. Style

According to Brusilovsky and Millán (2007), user style is a set of user traits that define a user as an individual. Learning style and cognitive style are popular examples of style in regards to personalisation.

Stash, Cristea and De Bra (2004) define learning style as the preferred learning approach of the user, and its scope is somewhat limited to personalised educational systems as it focuses on user learning. User learning style is worthy of further consideration, however, and the question needs to be answered as to what the relationship is between learning style and personalisation. And can learning style be integrated in contexts other than e-learning and educational systems, for instance in a generic user-adaptive system?

Cognitive style is defined as organising and representing information by individual preference and habitual approach. Research on cognitive style in personalised systems has concentrated on the navigation aspect of adaptivity (Brusilovsky and Millán, 2007).

2.3.2. Profiling

User profiling can be classified into two classes, static or dynamic. Static means that the user profile information is not changed over time. This would include, for example, the user's name, birth date and language. This information can be gained during the registration process on a website. In contrast, a dynamic user profile is always updated. Dynamic user profiles have two forms, short-term and long-term. The former represents the user's current interests while the latter represents information on the user's interests which is not often changed. On the other hand, there are two basic profiling strategies in the recommender systems which are explicit and implicit feedback (Kelly and Teevan, 2003; Sugiyama, Hatano and Yoshikawa, 2004).

2.3.2.1. Explicit feedback

Explicit feedback is gained from direct input by users. Recommender systems in this case ask the user to rate items using scales or to fill in forms in order to explore the user's interests about different items. The collected data might include some personal information such as name, age, location and background level (i.e. demographic information). General interest fields could be

predefined as an example, and then the user could tick checkboxes that indicate specific interests. Popular explicit feedback methods include where a user shows his/her opinion by rating/voting for an object (e.g. a web document, book, video clip or picture) with the form of like/dislike or a value with a range (m to n), and also other methods such as questionnaires and keywords (Gauch, Speretta, Chandramouli and Micarelli, 2007; Jawaheer, Weller and Kostkova, 2014).

An example is the Skyskill and Webert system, which was proposed by Pazzani et al. (1996), one of the earliest systems in the personalisation domain; it recommends relevant web pages based on user ratings. Another example is the Wisconsin Adaptive Web Assistant system, which was proposed by Shavlik, Calcari, Eliassi-Rad and Solock (1998) and Shavlik and Eliassi-Rad (1998), and which uses explicit feedback to help users during browsing. Another good example is the WIFS system, which was proposed by Micarelli and Sciarrone (2004), and which allows users to rate the degree of relevance of computer science topics using a rating scale from -10 to +10. Examples of commercial systems that use explicit feedback are Netflix and Amazon.

According to Amatriain, Pujol and Oliver (2009), the accuracy and reliability of explicit feedback are both very high. Capturing both positive and negative feedback is possible with this kind of system. However, additional effort and time are required for explicit feedback, and consequently the burden on the user can become high. Another problem is that users are inconsistent when they provide feedback and sometimes wrongly or incompletely report data such as personal information or their interests. As a result, explicit feedback should not be considered an absolute truth about a user's interests, and a good profile should not rely 100% on explicit feedback in the process of modelling users (Gauch, Speretta, Chandramouli and Micarelli, 2007; Gasparetti and Micarelli, 2007; Jawaheer, Weller and Kostkova, 2014).

2.3.2.2. Implicit feedback

Implicit feedback is the process of monitoring the behaviour of a user in order to collect usage data, with the goal of deriving the user's interests in order to create a user profile. The user does not need to assign his/her interests explicitly. In fact, the system will derive and identify those interests by monitoring the user's interaction and/or by analysing usage data (Kelly and Teevan,

2003). It is argued that recommender systems in the future will rely more on implicit user feedback and will also have less need to be intrusive (Adomavicius and Tuzhilin, 2005).

Nichols (1998) suggested the first classification of user behaviour via implicit feedback by classifying various actions during the interaction between a user and a system. Oard and Kim (2001) later improved that classification and proposed a framework of user behaviour to make monitoring the user's interests possible by using two axes: the first axis is the behaviour category (examine, retain, reference and annotate) which classifies the user's behaviour. The second axis is minimum scope (segment, object and class) which sorts items that the user interacts with. Kelly and Teevan (2003) then evolved that framework by inserting a fifth behaviour category which is create.

A number of studies, tools and systems use implicit feedback. Two old systems that utilise implicit feedback interactively are Letizia, which was proposed by Lieberman (1995), and WebWatcher, which was proposed by Joachims et al. (1997).

Chien (2000) proposed the Haystack system, which utilises past search queries (i.e. query history) which are comparable to the current query to express better results. Later on, Dumais et al. (2003) developed a system that was called Stuff I've Seen (SIS). It simplifies information re-use based on past viewing of various kinds of documents and objects. Qiu and Cho (2006) suggested a user model for learning a user's interests in web pages based on past clicks in search results. Similarly, Stamou and Ntoulas (2009) studied the extraction of preferences from the click history of each user with the purpose of filtering search results. Recently, Vallet, Cantador and Jose (2010) conducted a study that takes advantage of the user profiles built from social tagging systems such as Delicious (www.delicious.com) to provide personalised web search result re-ranking. Zhou, Lawless and Wade (2012) proposed a method of creating individual user profiles based on bookmarking for the purpose of query extension (i.e. improving search results and retrieval performance by reformulating a query).

Agichtein, Brill and Dumais (2006) clarify that the advantage of implicit feedback is that the user does not need to complete actions that waste time and effort, and it allows the user to concentrate completely on the current task. Gathering and analysing a massive amount of implicit data is easier as a result. However, implicit feedback is rated to be less accurate than

explicit feedback because it only captures positive feedback (Amatriain, Pujol and Oliver, 2009; Jawaheer, Szomszor and Kostkova 2010; Jawaheer, Weller and Kostkova, 2014).

2.3.2.3. Comparison of user profile methods

There is a necessity to compare user profiling techniques (i.e. explicit feedback and implicit feedback) to help in answering the question of which is preferable: explicit feedback, implicit feedback or hybrid feedback (a combination of both), and when and how. In this comparison, there are certain factors and criteria that should be considered: accuracy, data noise, cognitive load, data abundance and, finally, expressivity of feedback.

- **Accuracy**

Indicating a direct preference makes the accuracy of the explicit feedback value for an item high, whereas indicating frequency of action (e.g. viewing an item many times) makes the accuracy of the implicit feedback value of an item low (Amatriain, Pujol and Oliver, 2009). For example, in implicit feedback, if the user has purchased some items, that does not mean that it is accurate to give preference to those items, as they might be gifts for friends or the user may not be satisfied with the purchased items (Hu, Koren and Volinsky, 2008).

- **Data noise**

Data noise means that the data might be unclear, undefined or inaccurate. This is caused by unknown problems, such as rating an item by mistake (i.e. inaccurate rating), leaving the system for a period and the system calculates the time spent viewing a web page (i.e. unclear view time), or the crashing of web pages in a browser or proxy server (i.e. undefined view frequency). Both explicit and implicit feedback have some data noise, though implicit feedback data may suffer more from noise than explicit feedback data (Anand, Kearney and Shapcott, 2007; Hu, Koren and Volinsky, 2008).

- **Cognitive load**

Cognitive load means that the user makes an extra effort to finish a specific task. The explicit feedback method has a high degree of cognitive load, whereas implicit feedback does not require any effort from the user other than the effort put into the current task. For example, where a user enters his/her interests manually, here the situation of entering and thinking is an example of cognitive load. In contrast, in implicit feedback, the user just visits a web page and the system infers from this visit (Gauch, Speretta, Chandramouli and Micarelli, 2007; Gasparetti and Micarelli, 2007).

- **Data abundance**

Data abundance means that the amount of data collected via the implicit feedback approach is rich because the system gathers the data rather than the user, based on the user's interactions and behaviour. This is in contrast to data gathered in the explicit feedback approach, which is difficult to collect (Amatriain, Pujol and Oliver, 2009; Jawaheer, Szomszor and Kostkova, 2010).

- **Expressivity of feedback**

This includes positive feedback such as 'I like it', and negative feedback such as 'I do not like it'. Explicit feedback can take the form of both positive and negative feedback, whereas only positive feedback can be captured by implicit feedback (Amatriain, Pujol and Oliver, 2009).

To conclude the comparison, a hybrid feedback system should be used for building user profiles for the general purpose of most recommender systems. One reason is that this feedback can overcome some of the limitations of each feedback method. Also, hybrid feedback can show rich interaction techniques to the user if the explicit feedback is implemented at the same time as the implicit feedback is exploited. The following table provides a summary.

Table 1. Comparison of explicit feedback and implicit feedback

Criterion	Explicit Feedback	Implicit Feedback
Accuracy	High	Low
Data Noise	Yes (Low)	Yes (High)
Cognitive Load	High	Low
Data Abundance	Low	High
Expressivity of Feedback	Positive and Negative	Positive Only

2.3.3. User profile representation (feature-based)

According to Gauch et al. (2007), representing user profiles can be done using feature-based representation by several data structures and techniques. Feature-based representation analyse the items data in the aim of converting the item representation from the original information space to the target one such as converting web pages as keywords vectors. There are three fundamental ways of representing user profiles: sets of keywords, concepts and semantic networks. Ghorab, Zhou, O'Connor and Wade (2012) extend this classification to represent users' interests by adding two extra dimensions: data structure and content.

2.3.3.1. Keyword-based profile

A keywords profile includes a bag-of-words representing user interests (Barla, 2011). These keywords can be either provided by the user or collected from the items that the user interacts with (e.g. web documents). Numerical weight is given to each keyword and this weight refers to the importance of the keyword in the user profile. A popular weighting system is TF*IDF

(term frequency * inverse document frequency). This is very simple and easy to build. However, a keyword might have various meanings (i.e. polysemy), which leads to a degree of negative accuracy on the user profile (Salton and Buckley, 1988).

2.3.3.2. Semantic network profile

A user profile can be represented as a semantic network. A semantic network contains nodes, which represent concepts or interests, and edges between those nodes, which indicate the relationship between them. WordNet (http://wordnet.princeton.edu), a large lexical database for English, is an example of a semantic network. This representation has an advantage over keyword-based representation due to its ability to solve the polysemy problem. However, it is difficult to implement such a system. Moreover, a personalised system which uses a semantic network-based profile cannot involve a large semantic network because the system cannot sustain the time and space to use it (Gauch et al., 2007).

2.3.3.3. Concept-based profile

There is a similarity between a semantic network profile and a concept-based profile in that they both contain nodes with relationships between them. However, in concept-based profiles, the nodes represent abstract topics, which reflect interests to the user rather than a set of keywords. Hierarchical concepts can be used in the concept-based profile to boost the system in order to make generalisations. According to the user's interests, levels in the concept hierarchy give options, which are fixed-level or dynamic-level. Building such a concept-based profile from a concept hierarchy can be simple, and built from taxonomy, or can be complicated, and built from rich ontology (Gauch et al., 2007).

There are some considerations which should be taken into account when implementing a concept-based profile and using an existing concept hierarchy, such as the fact that not all linked parent-child nodes are conceptual because there might be a mechanism for organising them alphabetically or geographically. Furthermore, some directories have dozens of children for a topic, whereas others may have few or none at all (Gauch et al., 2007).

2.4. Document modelling (representation)

There are a number of different representations that can be implemented by different techniques in order to extract important information that can satisfy the user.

There are a variety of reasons for extracting important information from the document in the form of keywords. One of the reasons is to find information relevant to the user. Another reason is that the Web contains a huge number of documents in different structures, some of which are well-designed whereas others have very bad structures. Therefore, a useful representation of documents is necessary for the personalised recommender system to handle any type of document (Micarelli, Sciarrone and Marinilli, 2007). Documents in our experiment are the tweets which contain only text and text contains terms. Term weighting is a step to assess the value of each term to the document. In addition, it is assignment of numerical values to terms that represent their importance in a document. There are different methods to represent document which are Boolean Model, Vector Space Model and Probabilistic Model.

2.4.1. Boolean model

Micarelli et al. (2007) explain that the Boolean model is represented by sets of terms which are extracted from a collection of documents. This model is considered simple, and thus makes the computing process easy. However, it does not allow the document to be ranked and, as a result, there is a problem with its effectiveness. Moreover, a large amount of memory will be consumed because it has a sparse matrix (i.e. a lot of zeros), in comparison to its limited advantages.

As seen in equation 1, this model simply contains a collection of terms (T) that are extracted from the collection of documents (D). Each term is represented by one value, either 1 or 0. The 1 value is granted to the term Ti if it exists in the document Dj, and 0 value is granted if the term does not exist. Assume that there are n documents, D1, D2, …, Dn, and m terms, T1,T2, …,Tm. Let us say that Cij is the count of existence of the term Ti in the document Dj. Thus, the Boolean representation of the document Dj as m components vector D_J = (D1j D2j ... Dmj), where:

$$D_j^i = \begin{cases} 0, & C_{ij} = 0 \\ 1, & C_{ij} > 0 \end{cases} \quad (1)$$

2.4.2. Vector space model

Documents and queries are both represented as vectors in the vector space model representation. For example, document i is represented as $\overrightarrow{d_i} = (w_{i,1}, w_{i,2}, \ldots, w_{i,j})$ where $w_{i,j}$ is a weight for the term j (e.g. TF-IDF weighting system) in document i. Cosine similarity $\mathrm{sim}(d_i, q)$ is usually used to compute the angle between the document vector d_i and the query vector q. The equation to calculate the cosine similarity is provided in equation 2.

$$\mathrm{sim}(d_i, q) = \cos\theta = \frac{d_i.q}{|d_i||q|} = \frac{\sum_i w_{i,j}\, w_{q,j}}{\sqrt{\sum_j w_{i,j}^2}\sqrt{\sum_j w_{q,j}^2}} \quad (2)$$

This is considered an appropriate way of ranking documents based on computing how close their vectors are to a query vector. For instance, assume that the point $Q(x1, y1)$ is represented as a query and the points $A(x2, y2), B(x3, y3)$, ... etc. are represented as documents. Therefore, the cosine angle (i.e. cosine similarity) can be computed between Q and each document, and then they can simply be sorted in descending order based on the value of the cosine angles (Micarelli et al., 2007).

The step of document length normalisation is important because long documents have higher term frequencies and the same terms might appear often, and this increases the number of matches between the document and the query. Moreover, the system retrieves the longest documents in most cases, and the cosine similarity can show the effect of long documents (Micarelli et al., 2007).

Applying the vector space model has a number of advantages such as enhancing response quality by using term weighting systems and ranking documents based on their relevance to the user's query. However, this model considers terms as independent (i.e. there is no connection between terms) (Micarelli et al., 2007).

2.4.3. Probabilistic model

There is a similarity between the probabilistic model and the Boolean model in terms of representing documents and queries as binary weight vectors (Micarelli et al., 2007). The

difference between them is the similarity function between the query or the user model (e.g. user interest) and the document. The main idea in this model is to indicate the probability that a specific document is relevant to the query or the user model (Jones, Walker and Robertson, 2000). Its main purpose is for rating documents based on the probability of relevance. Therefore, the system's effectiveness is raised. The function $\mathrm{sim}(d_i, q)$ that computes the similarity between the document d_i and the query q is provided in equation 3:

$$sim(d_i, q) = \frac{P(R/d_i)}{P(\overline{R}/d_i)} \tag{3}$$

Document and query are both represented by the value of 0 (does not exist) and 1 (exists). In the equation above, R represents the set of relevant documents whereas $\overline{R}$ represents the set of non-relevant documents. $P(R/d_i)$ denotes the probability that document d_i is relevant to the query and $P(\overline{R}/d_i)$ indicates the probability that document d_i is not relevant to the query q (Micarelli et al., 2007).

The probabilistic model is able to rank documents based on relevance probability in descending order. However, Micarelli et al. (2007) clarify that there is a main disadvantage of this model, which is the term frequency in documents is not taken into consideration in this model as all weights are binary.

2.5. Recommendation methods

Various filtering techniques are used in recommender systems. The domain requirements can identify which techniques are appropriate in order to recommend items that meet the user's interests. The most commonly used recommendation techniques are collaborative filtering and content-based filtering. The following section discusses each technique in detail.

2.5.1. Collaborative filtering

Collaborative filtering recommender systems recommend items that other users who have similar tastes to the active user have liked previously. The similarity in taste of the active user to the others is calculated relying on the similarity in the rating history of the users. This is a

reason why collaborative filtering is called “people-to-people correlation”. The basic idea is that users who had similar tastes in the past will have similar tastes in the future. Collaborative filtering is the most commonly used technique in recommender systems (Ricci, Rokach and Shapira, 2015).

It is argued that the Grundy system, which uses stereotypes as a method of building models of users based on a limited amount of information on each user, was the first recommender system to use the collaborative filtering technique. By using stereotypes, user models can be used to recommend relevant books for each user individually. Later on, the Tapestry system was proposed to identify like-minded users manually, relying on each user. Ringo, GroupLens and Video Recommender were the first systems to use collaborative filtering algorithms for automating prediction (Adomavicius and Tuzhilin, 2005).

Algorithms used by collaborative recommender systems can be categorised into two general classes: memory-based (or heuristic-based) and model-based. Memory-based algorithms collect all the information about ratings, items and users. The most commonly used techniques in this class are: nearest neighbour (cosine, correlation), clustering and graph theory. Model-based algorithms, which are applied offline, use the collected information to learn a model that is then used to make rating predictions. The most commonly used techniques in this class are: Bayesian networks, clustering, artificial neural networks, linear regression and probabilistic models (Adomavicius and Tuzhilin, 2005; Schafer, Frankowski, Herlocker and Sen, 2007).

Adomavicius and Tuzhilin (2005) explain that collaborative recommender systems have several limitations:

- **Cold start problem:** cold start problem can happen in two different ways, which are new user problem and new item problem. In the new user problem, it is the same problem that content-based recommender systems have. For making accurate recommendations, the system first has to learn and understand the user’s preferences from past ratings the user has made. Many techniques have been presented to solve this problem and most of them use hybrid systems. For instance, combining content-based and collaborative techniques. In the new item problem, Collaborative recommender systems depend solely

on users' preferences for making recommendations. Subsequently, the recommender system will not be able to recommend a new item until it has been rated by a considerable number of users. Hybrid systems are also able to solve this problem.

- **Sparsity:** This is a result of the absence of sufficient obtainable ratings. The number of ratings obtained is usually quite small compared to the number of ratings that needs to be predicted. For example, in a recommender system that specialises in films, there might be many films that have been rated by only a small number of users, and therefore these films would be recommended too rarely. This problem can be solved by using user profile information when user similarity is being calculated.

2.5.2. Content-based filtering

In the content-based filtering technique, the system learns how to recommend items based on how similar they are to those that the user has liked or purchased in the past. The calculation of the similarity of items is based on the features that the compared items have. For example, if a user has rated a film positively, the genre of which is comedy, the system can learn to recommend to the user other films in the comedy genre (Ricci, Rokach and Shapira, 2015). The content-based filtering technique has methods and strategies for representing items and building the user profile, then comparing the user profile with item representation in order to give appropriate recommendations. Research on content-based recommender systems has been done at the intersection of many computer science topics, particularly information retrieval and artificial intelligence (Lops, De Gemmis and Semeraro, 2011).

The recommendation process in content-based recommender systems is performed in three steps: content analyser, profile learner and filtering component. In the content analyser step, a further pre-processing step is needed to elicit structured and relevant information when the information does not have structure, such as text. The main purpose of the process is representing the content of items (e.g. product description, document, news, web pages and so on) that are coming from information sources in a form that is appropriate for the next processing steps. Feature extraction techniques analyse the data items in order to convert item representation from the original information space to the target one; for instance, representing

web pages as keyword vectors. This process is the input for the profile learner and filtering component steps. In the second step, which is profile learner, data representative of the user's preferences is collected and generalised through machine learning techniques, which have the ability to infer a model of the user's interests from items liked or disliked previously, in order to build the user profile. In the final step, which is the filtering component, the user profile is exploited to suggest relevant items by matching the profile representation with the items that will be recommended (Lops, De Gemmis and Semeraro, 2011).

Many approaches have used the content-based approach to recommend interesting tweets. Recommending items based on content is not an easy job due to the size limit of tweets. Previous approaches to recommending tweets and representing user interests have generally used content analysis, such as Latent Dirichlet Allocation (LDA) or TF-IDF metrics (Karidi, Stavrakas and Vassiliou, 2016).

Adomavicius and Tuzhilin (2005) state that content-based recommender systems have several advantages when compared to collaborative filtering systems:

- **User independence:** Content-based recommenders use only ratings that are given by the active user for building his/her profile, whereas collaborative filtering systems need ratings that are provided by other users in order to find the nearest neighbour of the active user. Therefore, the system will only recommend items that are most liked by the neighbours of the active user.
- **Transparency:** Giving explanations of how the recommender system works might be achieved by listing content attributes or descriptions. These features can be useful for deciding whether to trust a recommendation. However, collaborative systems are mysterious in terms of the recommended item explanation, because unknown users, who have similar tastes, have liked that item.
- **New item:** Content-based systems can recommend items that have not yet been rated by any user. As a result, they do not suffer from the first-rater problem, which impacts collaborative systems that depend only on user preferences to give recommendations –

consequently, the system will be able to recommend new items only after they have been rated by a considerable number of users.

On the other hand, as any system does, content-based recommender systems have shortcomings and drawbacks, as proved by Lops, De Gemmis and Semeraro (2011):

- **Limited content analysis:** The number and type of the features, which are associated with the recommended objects either automatically or manually, in content-based techniques are limited, and domain knowledge may be needed. For example, a system providing recommendations for films needs to know other information such as director and actors. Suitable suggestions may not be provided in content-based systems if the analysed content does not contain enough information to reduce the selection of items that the user likes from items that the user dislikes. Only specific aspects of the content can be captured by some representations. However, many other aspects might influence a user's experience. Moreover, for web pages, the techniques that extract features from text totally dismiss aesthetic and extra multimedia information. In general, both automatic and manual association of features to items may not be adequate to distinguish between items' aspects that could be useful for deriving user interests.
- **Over-specialisation:** There is no inherit method in content-based recommenders for finding something unexpected. Items are recommended based on matching them against the user profile. Therefore, the user is recommended items that are similar to those that he/she has already rated. For example, when a user has only rated films which are directed by James Cameron, he/she will be given recommendations for only that type of film. A perfect content-based technique can limit the range of applications from what would be useful.
- **New user:** Content-based recommender systems give recommendations based on collecting enough ratings about the user and understanding his/her preferences in order to provide accurate recommendations. However, when a new user has not rated any items and has not built his/her profile yet, the system will not be able to give him/her reliable recommendations.

2.5.3. Demographic filtering

In these systems, items are recommended based on the demographic profile of the user, such as language, country, age and so on. Different recommendations are generated for various niches. For example, users are forwarded to specific web sites according to their language or country. Another example is that recommendations might be customised based on the age of the user. This demographic data is usually provided during the registration process by the users themselves, and then may be updated from the user's purchasing history (Ricci, Rokach and Shapira, 2015). Recommended items are provided based on how similar people (in terms of their demographic data) rated a specific item (Tintarev, 2007). Pazzani and Billsus (2007) propose an approach that uses demographic attributes such as age, gender or postcode to build user profiles as vectors in order to find relationships with other users and then compute similarities between them to make the final predictions.

Demographic filtering has similarity with collaborative filtering in that both use similarity calculations between users in order to generate recommendations. The difference is that here demographic features are exploited to find similarity between users rather than their previous ratings of items (Tintarev, 2007).

2.5.4. Link-based filtering

A link-based technique in such a system finds the relations between items and user graph-theoretical algorithms to find groups of items that are either the most ideal or most indicated by other users. For example, a link-based system simplifies Google's good search results (McDonald, 2003).

2.5.5. Knowledge-based filtering

In this type of filtering, systems recommend items according to particular domain knowledge about how a specific item's attributes match the user's preferences and how the item is useful for the user. In other words, knowledge-based recommender systems are case-based. These systems have a similarity function, which estimates how much the user needs to meet the

recommendations. Then, and for the user, the similarity degree can be used as the likely interest in the recommendation (Ricci, Rokach and Shapira, 2015).

2.5.6. Community-based filtering

The recommendations in these systems are given based on the preferences of the users' friends. This method is based on the proverb "Tell me who your friends are, and I will tell you who you are". It is suggested that users tend to be more likely to trust recommendations from their friends than recommendations from anonymous users who have similar tastes. This technique is based on the growth of social networks. The recommendations given in this technique are based on the ratings provided by the user's friends. Research on this area is still in its early stages and the performance results of such systems are mixed (Ricci, Rokach and Shapira, 2015).

2.5.7. Simplified statistical filtering

In this type of filtering, systems show items to users based on certain statistical facts, which are generally common measurements. For example, such a recommender system recommends a user the best rated items or the best items that have been purchased by other users (Kazienko and Kolodziejski, 2006).

2.5.8. Hybrid filtering systems

To avoid some of the limitations and solve some of the problems of the standard content-based and collaborative filtering systems, many recommender systems use a hybrid technique that combines content-based and collaborative methods. This combination can be classified as follows: (i) Applying collaborative and content-based techniques separately and then combining their results. This method can choose items that are better than others according to some recommendation quality metric. (ii) Integrating some content-based features into a collaborative approach. This method can maintain the content-based profiles for each user and can also overcome the sparsity-related problems of the standard collaborative technique. (iii) Integrating some collaborative features into a content-based approach. This method results in improved performance compared to the standard content-based technique. (iv) Building a general unifying

model that integrates both collaborative and content-based features (Adomavicius and Tuzhilin, 2005).

Ricci, Rokach and Shapira (2015) clarify that hybrid filtering recommender systems can be a combination of all the above-mentioned techniques. In general, a hybrid system combines techniques i and ii, attempting to use the advantages of i to solve the disadvantages of ii, and vice-versa.

2.6. Recommender system evaluation

Methods for evaluating recommender systems differ depending on the goal of the system. Many researchers have discussed the classification of recommender system evaluation methods into different classes, such as: offline and online, data-centric and user-centric, live user experiments and offline analyses, and finally offline, online and user studies (Beel and Langer, 2015; Ricci et al., 2015). There are three main evaluation types, which are explained in detail in the following sections.

2.6.1. Offline evaluation

Offline evaluation is considered the easiest and more flexible method in the sense of performing in experiments. It can be used in cases where a dataset is available, and it contains user information and their ratings of some items. It provides a good space for developers to run many candidate algorithms and compare them without interacting with real participants. As a result, experiments can be performed and conducted more cost-effectively (Ricci et al., 2015).

Based on ground-truth, offline evaluation is typically used to measure the accuracy of recommender systems and is also sometimes used to evaluate recommendations' novelty and serendipity. The common metrics of this type of evaluation are: precision, recall, F-measure, mean reciprocal rank (MRR), mean average precision (MAP), normalised discounted cumulative gain (nDCG), mean absolute error and root mean square error (Beel and Langer, 2015).

Beel and Langer (2015) define three types of dataset that can be considered ground-truths:

- **Explicit ground-truths:** these have information on particular items rated by the user. In the evaluation of this type, some ratings are removed from the dataset and the recommender system has to predict the ratings of the removed items based on the remaining data. The recommender system can be considered more accurate if its predictions are closer to the original ratings.
- **Inferred ground-truths:** these have a collection of the personal items of a particular user, for example a collection of books that a specific user bought. This collection of items might be good recommendations. In the evaluation of this type, random items of this collection are removed and then the recommender system provides recommended items based on the remaining items. The recommender system can be considered more accurate if it recommends most of the removed items.
- **Expert ground-truths:** in this type of dataset, experts gather and classify items manually. For instance, classifying research papers into different categories such as computing, education, business and so on. In the evaluation, some papers of one category are taken as input. The more papers of the same category that are recommended, the more accurate the recommender system should be.

2.6.2. Online evaluation

In the many realistic recommender systems, one of the main goals is to capture and understand the behaviour of users while using it, and this can be achieved by using online evaluations, which are unique in the sense that they can measure system aims such as long-term benefits or user retention. They also provide an understanding of how system properties affect system aims, such as recommendation accuracy and recommendation diversity. They also allow researchers to understand the correlations between these properties. However, it is believed that running candidate algorithms and comparing them in online evaluations is expensive (Ricci et al., 2015).

Online advertisements and e-commerce helped to create online evaluations. In real-world recommender systems, acceptance rates can be measured by online evaluation. Acceptance rates are simply measured by click-through rate (CTR); for example, the ratio of clicked advertisements to displayed advertisements. For instance, if 10,000 recommendations are

provided by the recommender system and 120 are clicked, CTR in this case is 1.2%. The ratio of downloaded or bought items can be used in other metrics. The hypothesis about CTR metrics is that when a user clicks, downloads or buys a recommended item, that means the user liked the recommendation. However, this hypothesis is not often dependable, as users, for instance, might buy a book and after reading it they might not like it and might rate it negatively (Beel and Langer, 2015).

2.6.3. User studies

If the goal of a recommender system is to measure the satisfaction of users, user studies evaluations are a typical way of achieving this goal via explicit ratings. In this type of evaluation, different recommender systems recommend different items to users and then the users are asked to rate the recommendations. The highest average rated recommender system can be considered the most effective. In general, participants are asked to rate their overall satisfaction with the recommended items. Otherwise, individual aspects of the recommender system might be rated by users, such as recommendation novelty or suitability for non-experts (Beel and Langer, 2015; Ricci et al., 2015).

Beel and Langer (2015) divide user studies into “lab” and “real-world”. In the former, participants are aware that they are part of a user study, and this might allow other factors to impact their behaviour and therefore the evaluation results. In the latter, participants are not aware that they are part of the study, and this can result in rating the recommendations for their own benefit. For example, depending on ratings, a recommender system provides more accurate items. Another example is that in some recommender systems rating items is required in order to create suitable recommendations (Beel and Langer, 2015).

2.7. Exploiting short-text social media platforms in recommender systems

Over the years, there have been lots of studies carried out on Twitter. These studies have led to the availability of sufficient scientific work looking at different aspects of Twitter as well as microblogging in general. The studies carried out in this area are grouped here into various categories: general studies on social networks, searching with and also in microblogging,

conversations, topic modelling and detection, contextualisation, user influence, activity, popularity, user recommendations, predicting based on tweets, user modelling, users similarity, collaborative and re-tweetability, event detection and news stream aggregation. Some of the influence, similarity and popularity algorithms will be used in the experiment as evaluation candidates against the proposed algorithm. They are chosen as they can be applied on our dataset. These algorithms are: signal strength, retweet impact, follower rank, in-degree (number of followers). Moreover, Cosine, Jaccard, Euclidean and Manhattan metrics are also used in the evaluation.

2.7.1. General studies on social networks and microblogging

After Twitter was launched, lots of scientific studies were carried out on it. Java, Song, Finin and Tseng (2007) looked at the properties (topological and geographical) of Twitter with a view to analysing the intentions of users. It was discovered that Twitter is primarily used by people for the sharing of information and also general activities which take place every day. A quantitative study was performed by Kwak, Lee, Park and Moon (2010) with the aim of information diffusion. A network study was performed by Yang and Counts (2010) in order to analyse information diffusion on Twitter. The model they used captured the scale, speed and range of information diffusion. Nagarajan, Purohit and Sheth (2010) looked at Twitter through the perspective of different events, such as the International Semantic Web Conference, the healthcare reform debate and an election in Iran, with a view to identifying retweet behaviour properties. They looked for the most viral messages and the most commonly tweeted. Myers and Leskovec (2012) analysed information diffusion by looking at the way that various contagions interact with one another as they spread through the network of Twitter. It was discovered that interactions bring about changes in dissemination probability. On average, this was about 71%. According to Huberman, Romero and Wu (2008), usage of Twitter seems to be driven by sparse as well as hidden network connections, and this is under the friends and followers set.

2.7.2. Searching with and in microblogging

A model was developed by Duan et al. (2010) aimed at real-time tweet search. This not only uses content that is relevant to a particular tweet but also makes use of the authority of the user as well as features related to the tweet. An example is whether the tweet includes a URL. With the use of machine learning for ranking and also BM25 as a method for retrieving information, experiments were carried out on 500 tweets. These have predefined query terms and relevance annotation.

Apart from using a microblogging platform to search, it can also be used to increase the novelty of search results, as explained in the work of Dong et al. (2010). It was predicted by Huang, Yang and Zhu (2011) that there are several features which can be used in providing a quality label for a tweet. These could be authority, ratio of unique words, term similarity and tweet length. An assumption which is vital to this approach is that documents which have the same content possess similar qualities. In order to carry out an evaluation, about 9,000 tweets were annotated using quality labels. This was used for testing and training.

Alonso et al. (2010) categorised a set of tweets as interesting or uninteresting using crowdsourcing. The method showed that the existence of URLs is a successful feature in selecting interesting tweets with high accuracy. However, a shortcoming of this factor is that it might wrongly categorise an uninteresting tweet, which has a link to useless content, as interesting (Karidi, Stavrakas and Vassiliou, 2016).

2.7.3. Conversation

Apart from sharing information, Twitter can also be used for the purpose of conversation. The conversational aspect of retweeting is explained in the work of Boyd, Golder and Lotan (2010). For conversations that are ongoing, identifying the dialogical act (statement, answer, question) that tweets represent can be very useful. A model which is unsupervised that addresses this problem is discussed in the work of Ritter, Cherry and Dolan (2010). Chen, Nairn and Chi (2011) clarify that recommending conversations, for which there are various algorithms, is the next step such as recommending top related answers when the user is searching about a good answer for his question.

2.7.4. Topic modelling and detection

Topic modelling of temporally-sequenced documents on Twitter is reported in the work of Kawamae (2011), along with attempted modelling of contentious topics over time. Based on word allocations of tweets, this approach learns topic shifts. Ramage, Dumais and Liebling (2010) used labelled-LDA to classify a tweet, exploiting its labelled information, and after that the probability distribution vector of latent topics is built in order to represent the tweet's content. Therefore, and based on similarity between the topic vectors, incoming tweets in the timeline are classified as interesting or uninteresting.

In order to follow up tasks on Twitter effectively, it is important to understand what people are discussing. Michelson and Macskassy (2010) used entities stated in tweets to build a user's topic profile. This was done through employing a knowledge base both for entity classification and disambiguation. Ramage, Dumais and Liebling (2010) adopted another approach which partially supervised the learning model explained in Ramage, Hall, Nallapati and Manning (2009). This was used in mapping users and tweets into various dimensions which roughly correspond to topics such as a social post's characteristics, status, style and substance. Through the mappings, the timeline of a user is re-ranked. They can also be used in making recommendations about which new users should be followed.

2.7.5. Contextualisation

Entity, which refers to names, places, organisations and so on, is a particular notion of concept. Performances of traditional Natural Language Processing (NLP) tools such as Named Entity Recognition (NER), which seem to have had their training on news corpora, are not strong on Twitter corpora. Also, the NLP pipeline was rebuilt, and this was from part-of-speech tagging to NER. It was aimed at addressing problems of short text (Ritter, Clark and Etzioni, 2011). Jung (2012) put another NER approach forward whereby clusters of microtexts are built using temporal association, semantic, social and contextual. A solution was proposed by Meij, Weerkamp and De Rijke (2012) to tackle the issue of what each microblog post entails. Semantics were added to posts through automatic identification of concepts semantically related to them. Links were generated to articles from Wikipedia. An additional step is identifying

various areas of interest of an entity like competitors, services and products. Spina et al. (2012) looked at the methods through which these aspects are analysed.

2.7.6. User influence

A user is called influential when their actions inside a network are able to affect many other users in the same network. In other words, influential users are those who are able to spread information through a network (Morone and Makse, 2015).

Riquelme and González-Cantergiani (2016) state that there is a conceptual problem in defining the meaning of a user being influential because there is no agreement on the definition of influence. As a result, new influence measures are continuously appearing, and various measurement criteria have been provided. These different criteria have led to categorising influential users into new types of user, such as opinion leaders, innovators, and prestigious or authoritative actors (Bouguessa and Romdhane, 2015; Chai, Xu, Zuo and Wen, 2013). Some research classifies influential users into different groups: for instance, Jabeur, Tamine and Boughanem (2012) classify influential users, based on activity and impact, into opinion leaders, influencers and discussers, while del Fresno Garcia, Daly and Sanchez-Cabezudo (2016) classify them as disseminators (users who expand their influence), engagers (users who simplify relationships with a third party) and leaders (users who are top disseminator-engagers).

In terms of social influence, and in accordance with the work of Quercia, Ellis, Capra and Crowcroft (2011), there are two points of view. The first considers that a small number of users, who are highly connected and convincing, exert the influence rule. The second is believing that many unpredictable circumstances might make many users influential by accident. It is argued by others that Twitter offers the possibility of developing different influence criteria that can be measured.

Chen, Gao, Li and Hou (2014) proposed a multi-view influence rule clustering algorithm (MIRC) to partition users of Twitter into five clusters: fans, information disseminators, experts, celebrities and others. They used three features to partition users into the clusters: topic view, sentiment view and popularity view. In the evaluation, the proposed model (MIRC) was compared against: (i) the baseline K-means, (ii) two existing multi-view clustering approaches,

and (iii) K-means clustering (SC) on the three single views they designed. The clustering results showed that the MIRC outperformed the other compared approaches in all three metrics.

Riquelme and González-Cantergiani (2016) clarify that some research classifies users based on their popularity into broadcasters (users who have many followers and few friends), acquaintances (users who have similar numbers of followers and friends) and miscreants (those who have few followers and many friends, such as spammers). They proposed dividing the existing metrics into three different types of measures: activity, popularity and influence. In addition, they state that their classification is valid because there is no direct relationship between activity, popularity and influence, based on the work of Romero, Galuba, Asur and Huberman (2011).

To be influential on a platform such as Twitter, it is important to identify other users' influence and passivity. Romero, Galuba, Asur and Huberman (2011) explain these factors via an approach based on web link information forwarding activity on Twitter. Three influence measures were analysed, mentions, retweets and in-degree (number of followers). They were compared with various topics as well as over time. The metrics introduced by others directly measure users' influence, such as TwitterRank in the work of Weng, Lim, Jiang and He (2010), considered an upgraded version of Page Rank. Influence metrics are provided as services by companies such as Klout, peerIndex, Influence Tracker and so on (Riquelme and González-Cantergiani, 2016).

Anger and Kittl (2011) proposed a method that can measure influence on Twitter by calculating social network potential (SNP) as a percentage. This calculation is based on interaction ratio (conversation-oriented) and tweet and mention ratio (content-ratio). They compared their proposed method with Klout score, which is a tool based on unpublished features that gives a score to a user that reflects his/her activity on Twitter, in order to find some similarities. They analysed the top 10 Austrian Twitter users in their evaluation. The results showed that the proposed method has some similarities to Klout score. Differences were observed especially with users who have a very large number of followers.

The Retweet Impact algorithm was presented by Pal and Counts (2011); its estimates the impact of a user's tweets in terms of retweets; the algorithm is presented in equation 4.

$$Retweet\ Impact = OT * \log(uRT) \quad (4)$$

OT refers to the number of original tweets posted by author and retweeted by others, while uRT is the number of users who have retweeted the author's tweets.

The signal strength algorithm was presented by Pal and Counts (2011); it measures the strength of the author's topical signal; it is shown in equation 5. It is believed that an active user, who participates in a network constantly and frequently, might be considered a topical influencer (Riquelme and González-Cantergiani, 2016).

$$Signal\ Strength = \frac{OT}{OT+RT} \quad (5)$$

OT represents the number of original tweets of the author, whereas RT refers to the number of retweets accomplished by the author.

These are not all the influence classification systems that have been devised, but this discussion shows that there is no standard definition of or classification for influential users. Most of them focus on how influential the user is based on popularity via factors such as number of followers and followees, or interaction with others. However, from the perspective of a normal user who follows other identified influential users, it is believed that the influence rule can vary from one user to another in the following list. For example, some users who the user follows, such as close active friends, are not identified as influential, but can have more influence than influential users who have millions of followers. Therefore, there is a need to create a method that can give an influence score from the user's perspective, relative to the user's behaviour and interactions.

2.7.7. Activity

Riquelme and González-Cantergiani (2016) state that users in a social network can be considered active users when they participate in the network constantly and frequently within a period of time. This participation can involve measurable actions such as tweeting, retweeting, replying and so on. However, they also assume that there might be some users who are active readers but who do not leave any mark on the network and thus cannot be observed by any

metric. To address this problem, Yin and Zhang (2012) define user activity as the likelihood that a user will see a tweet. It cannot be determined when a user has seen a tweet until they perform an action such as retweeting, and then it can be determined that the user has read the tweet. A more active user is more able to see new tweets and then interact with them. Riquelme and González-Cantergiani (2016) argue that activity is an ambiguous concept as it is unknown how many tweets a user should post or retweet in order to be considered active, and they believe that it depends on the definition of the concept. They define general activity as combining the number of a user's original tweets, replies posted by the user, retweets accomplished by the user and tweets by other users that are marked as favourite (it is called "like" nowadays) by the user. The time factor is not considered in their general activity equation. Similarly, Vosoughi (2015) uses an engagement measure that calculates how active a user has been on Twitter since joining, as in equation 6.

$$Engagement\ (Activity) = \frac{Tweets+Retweets+Replies+Favourite}{AccountAge} \quad (6)$$

2.7.8. Popularity

Riquelme and González-Cantergiani (2016) define a user as popular inside a network when he/she is recognised by a large number of other users who are members of the same network. Despite this popularity, such a user does not need to be active and influential. An example would be when a celebrity has an account with thousands of followers, and has not followed anyone or even tweeted.

Popularity can be measured simply by considering follow-up links between users inside a network. One of the famous measures is FollowerRank, which was presented by Nagmoti, Teredesai and De Cock (2010) and which is an improved version of the Hajian and White (2011) traditional in-degree measure for social networks, as in equation 7.

$$FollowerRank(i) = \frac{Number\ of\ Followers}{Number\ of\ Followers+Number\ of\ Friends} \quad (7)$$

2.7.9. User recommendation and social links

Prediction of links within social networks, as Liben-Nowell and Kleinberg (2007) predicted, is also explained within the context of Twitter. New users are provided with recommendations in Armentano, Godoy and Amandi's (2012) approach through exploring a network's typology in the neighbourhood of a target user. Rather than networks being considered, Hannon, Bennett and Smyth (2010) suggest profiling of users through tweets which they have made.

2.7.10. Predictions based on people's tweets

Tweets are being used to make predictions about certain events in the world. An example of such outcomes is IMDb movie ratings in the work of Oghina, Breuss, Tsagkias and de Rijke (2012), where a linear regression model was used to predict values. Asur and Huberman (2010) predict box-office revenues through the use of a sample model from a popular topic's tweet rate. A German federal election was observed on Twitter by Tumasjan, Sprenger, Sandner and Welpe (2010); they indicate that election results can be determined or predicted based on how many times a party is mentioned.

2.7.11. User modelling

Abel, Gao, Houben and Tao (2011) analysed temporal dynamics in Twitter profiles for personalised recommendations in the social Web. They built two different types of profile, based on hashtags and entities (e.g. places and celebrities), taking into consideration time-sensitivity, enrichments (using external sources such as Wikipedia) and the activity of the user. They used a news recommender system for evaluation in order to compare the quality of the proposed profiles after considering time stamps. The results showed that the entity-based profile, which was built with time and enrichment, performed better than profiles that were hashtag-based (time), entity-based (enrichment only) and hashtag-based with no time constraints. The results also showed that with continuously active users, hashtag-based (time) profiles performed better than entity-based (time). Moreover, with sporadically active users, entity-based (time) profiles performed better than hashtag-based (time). They also noticed that when modelling a user based

on hashtags, the bigger the profile the better results are gained. However, an entity-based profile is independent from profile size.

Abel, Gao, Houben and Tao (2013) modelled user profiles in Twitter with different dimensions and compared the quality of each one by using a simple news recommendation system. The dimensions are topic modelling (hashtag-based, category-based or entity-based), enrichments (Twitter-only-based or external resources), temporal constraints (specific time period, temporal patterns such as weekends or no constraints) and the weighting scheme (TF, TF-IDF or time-sensitive weighting schemes). TF-IDF is explained in the proposed research section. Their results showed that in the dimension of topic modelling, entity-based profiles performed better than category and hashtag-based profiles. In the enrichment dimension, using external sources, such as news articles, is better than relying solely on Twitter. In temporal constraints, a short-term profile (from the last week for example) is better than a complete profile (the whole profile). The results also showed that a weekend profile is better than a complete profile.

Garcia Esparza, O'Mahony and Smyth (2013) proposed a CatStream system that uses traditional classification techniques to profile Twitter users, based on URLs they post in tweets, and which then filters their timelines based on topics that matter to them. The implementation showed that the system is able to profile users effectively and it shows a good performance in categorising tweets in order to filter users' timelines. The predicted categories were correct for 64% of tweets. Moreover, feedback from live users was extremely positive. However, the system only focuses on URLs in tweets for the purpose of profiling users and it is not suitable for users who do not have a sufficient number of tweets with URLs.

Karidi, Stavrakas and Vassiliou (2016) proposed a system that constructs a new ranked, personalised Twitter user timeline based on Concept Graph. User profile in their work was built from semantic information in the user's previous tweets based on Concept Graph. Each tweet was assigned to a specific topic. The relatedness between the user profile and tweets is calculated by graph theory algorithms, and then the tweets are ranked in descending order. Offline evaluation was conducted, and their work was compared against most popular state-of-the-art approaches. Their proposed work showed effectiveness in tweet recommendations.

A TRUPI system was proposed by Elmongui et al. (2015) that combines the history of a user's tweets with social features, and also catches the dynamic level of a user's interests in several topics in order to measure the user's interests as they change over time.

2.7.12. User similarity

Razis and Anagnostopoulos (2016) proposed a method that can discover and find similar Twitter accounts using semantic features. For calculation of the similarity metric, four Twitter entities were used as comparison coefficients: hashtags, mentions, URLs and the domain of the URLs. The more entities the accounts have in common, the more similar their content tends to become. In the evaluation, they used an active well-known influential user, and different depths and top-k (the top most similar users) were discovered. As a result, 80% of top-15 accounts were similar to the examined account in the depth of 1 (similar users to the examined user they started with). In the depth of 2 (similar users for each user that was discovered as similar in depth 1), fewer similar accounts were discovered. Moreover, when the values of top-k and depth were increased, the number of unique accounts (similar users) rapidly decreased. However, their methodology did not take into consideration the content of the tweets in the similarity metric.

Different similarity and distance metrics are used in this research as candidate algorithms against the proposed influence algorithm: cosine similarity, Jaccard similarity, Euclidean distance and Manhattan distance.

2.7.12.1. Cosine similarity

In this similarity metric, documents are represented as term vectors and the similarity existing between documents will be equal to how the vectors correlate with each other. This is expressed by quantifying the angle between the vectors' cosine, which is called the cosine similarity. Cosine similarity is a common similarity measure that is applied to various text documents, as exhibited by many applications that are used for information retrieval and clustering (Huang, 2008). The cosine similarity between two documents $\overrightarrow{t_a}$ and $\overrightarrow{t_b}$ is shown in equation 8.

$$\text{Cosine similarity } \left(\overrightarrow{t_a}, \overrightarrow{t_b}\right) = \frac{\overrightarrow{t_a}.\overrightarrow{t_b}}{|\overrightarrow{t_a}|\times|\overrightarrow{t_b}|} \tag{8}$$

2.7.12.2. Jaccard similarity

The Jaccard coefficient, also known as the Tanimoto coefficient, implements a similarity measurement by dividing the intersection by the union of objects. For any text document, the Jaccard coefficient strictly compares the weight sum of shared terms to the weight sum of terms present in either document but not the actual shared terms (Huang, 2008). Equation 9 shows the calculation of Jaccard similarity where the measure ranges between 0 and 1, where 1 means the two documents are the same and 0 means the documents are completely different.

$$Jaccard\left(\overrightarrow{t_a}, \overrightarrow{t_b}\right) = \frac{\left|\overrightarrow{t_a} \cap \overrightarrow{t_b}\right|}{\left|\overrightarrow{t_a} \cup \overrightarrow{t_b}\right|} \tag{9}$$

2.7.12.3. Euclidean distance

Euclidean distance is a reference for calculating geometric problems. It is simply the normal distance between two locations, and is measurable with a ruler in a two-dimensional or three-dimensional space. It is equally regarded as the default distance in the K-means algorithm (Huang, 2008). In order to measure the distance existing between two text documents d_a and d_b that are represented by their term vectors $\overrightarrow{t_a}$ and $\overrightarrow{t_b}$, respectively, both documents will have a Euclidean distance calculated as in equation 10.

$$Euclidean\ Distance(\overrightarrow{t_a}, \overrightarrow{t_b}) = \left(\sum_{t=1}^{m} \left|w_{t,a} - w_{t,b}\right|^2\right)^{1/2} \tag{10}$$

Where the term set is $T = \{t_1, \ldots, t_m\}$, and $w_{t,a}$ and $w_{t,b}$ refer to the term weights.

2.7.12.4. Manhattan distance

Manhattan distance is also referred to as boxcar distance, absolute value distance, block distance, city block distance, and L1 distance. It calculates how far will be covered between two points should a grid-like route be followed. Hence, the block distance between any two items can be calculated by adding the differences existing between corresponding components (Gomaa and Fahmy, 2013).

2.7.13. Collaborative and re-tweetability

Exploratory data analysis was conducted by Suh, Hong, Pirolli and Chi (2010) using principal component analysis as well as a generalised linear model. The aim was to find out how certain features can influence a tweet from being retweeted or not, such as if a tweet contains mentions, hashtags or a URL, the age of the user's account, favourite tweets, how many shared tweets, frequency of shared tweets, and how many followers as well as friends a user has. According to the authors, the features influence retweetability to a great extent and should be considered important. Retweetability means that a user's tweet is likely to be reposted by other users.

A probabilistic collaborative filtering model was developed by Zaman, Herbrich, Van Gael and Stern (2010) to ensure that future tweets are predicted. The most important features discovered were tweet as well as retweeter source. Data was built based on tweets collected within one hour. Retweets made within this time (after the tweet was made) were also gathered. Retweets are used as means of positive feedbacks in this model. Negative feedback was observed from the network of retweets, which consists of users that are active who have been retweeted or have retweeted.

A graph model was proposed by Yang et al. (2010) in order to predict the retweeting behaviour of users; the authors' paper covered retweeting in two aspects. The first is whether a given tweet is retweeted by users to their friends once it has been viewed. The second is how a new tweet spreads. Statistical analysis is provided based on users' behaviour in retweeting and tweeting within a dataset of their own.

Features of Twitter posts which are popular were studied by Hong, Dan and Davison (2011) in predicting whether a posts is going to be retweeted, and if it is, then how often it is going to be. During training, the initial n-1 retweets were used to represent positive. Final retweets and every other message not retweeted represented negative. Retweet number was used in training a multiclass classifier for the prediction about the possible number of retweets which a message will have. Feature combinations which performed very well were used are message content, as well as topical information, user graph structural properties and user retweet chain temporal dynamics.

A human experiment was conducted by Petrovic, Osborne and Lavrenko (2011) on predictions regarding the chances of a tweet being retweeted. There were test subjects of randomly ordered tweets in pairs, of which one tweet got retweeted. The experiment performed by a machine learning approach. This involved a passive aggressive algorithm based on the work of Crammer et al. (2006). The method uses a local hourly model and a global model. They are trained based on tweets which happen at specific times of the day. Social features were made use of, such as words contained in the tweet, if such a tweet is a response, novelty, tweet length, trending words, URLs, mentions, number of hashtags and if it is written in English. A major aspect of this work is using the sample stream of Twitter in acquiring the data. There is the assumption that lots of retweets went missing as a result of the sample stream being used as source.

The conditional random fields method was used by Peng et al. (2011) in predicting users' decision to retweet within a targeted network. The computational costs of this approach were reduced through the data being partitioned, with minor loss in regards to performance. Although good results were reported via this method, the paper tends to be very vague, especially when the data used for the experiment is being described.

Naveed, Gottron, Kunegis and Alhadi (2011) proposed a technique to predict the likelihood of a tweet being retweeted based on low and high levels of content-based features from a large collection from Twitter. They trained a machine learning method to forecast whether a tweet has a likelihood of being retweeted, based solely on content features.

Chen et al. (2012) proposed a method based on collaborative tweet ranking (CTR) in order to recommend useful tweets that the user might be interested in as a solution to information overload on Twitter. Their method considers three major elements: tweet topic level factors, social relations factors and explicit features. In the evaluation, they compared their method with chronological order (ordering tweets based on putting the recently posted tweets on top of the list) , retweet time, profiling, LDA, RankSVM and JOINTMF by using Mean Average Precision (MAP). The results showed that the proposed method outperformed other strategies.

Uysal and Croft (2011) proposed a personalised tweet ranking method based on retweet behaviour. They trained a model using a coordinate ascent (CA) algorithm to rank incoming

tweets based on the likelihood that the user will retweet them. They used four sets of features to rank incoming tweets: author-based, tweet-based, content-based and user-based. They used a decision tree-based classifier (J48) to classify tweets as retweetable or not. The CA algorithm was used to rank the tweets that are retweetable and display them in descending order. The results showed that a tweet-based feature set had better performance among the sets of features. However, combining all sets of features showed better performance still.

2.7.14. Event detection

With the real-time nature of Twitter, events can be detected or determined. As explained by Sakaki, Okazaki and Matsuo (2010), for example, users of Twitter are sensors which can be used in estimation of earthquake centres. Events are not only detected but also relevant data is filtered, collected and semantically enriched to ensure that faceted search as well as real-time analytics are provided (Abel, Hauff, Houben, Stronkman and Tao, 2012).

2.7.15. News stream aggregation

Other services as well as networks make use of Twitter to ensure that news is hot (Sankaranarayanan et al., 2009). This is evident as Twitter has often been used in capturing of breaking news. An example of a service in this respect is Tweeted Times, in which personalisation is achieved through the use of a Twitter user's account. Twitter history as well as Google Read are used by Zite to ensure that a personalised experience is provided. News is aggregated from various channels on the Fever website, while content that is personalised is provided by My6sense from social feeds and RSS feeds (Breuss, 2013).

Lee, Oh, Lim and Choi (2014) proposed a method to improve the accuracy of recommended news articles based on tweets. A user profile was built by extracting the nouns in a user's tweets and retweets. The user profile was represented by bag-of-words and normalised by TF-IDF. Cosine similarity was used in their research to recommend top-k news articles to the user. In the evaluation, they used a live user study, consisting of eight users, and hit ratio was used to measure accuracy. Two different sets of recommendations were shown to the users: recommendations based on Twitter activities and randomly selected recommendations. Their

results showed that Twitter-based recommendations had more accuracy than the random recommendations.

2.8. Classification algorithms

Generally, in text classification, and according to Hotho, Nürnberger and Paaß (2005), the aim is to assign a text document to a predefined class, for example labelling incoming news articles with a specific topic such as sports, economics or technology. Such classifications can be achieved by various algorithms and techniques. Generally, the classification task in any method starts with a training set $D = (d_1, \dots, d_n)$ of documents that have already been labelled with a class $L \in \mathbb{L}$ (e.g. art, economics). Then, the task is to define a classification model, as in equation 14. As a result, it is able to correctly locate the class to a new document d.

$$f : D \rightarrow \mathbb{L} \qquad f(d) = L \tag{14}$$

The performance of a classification model can be measured by taking a set of random fractions of the labelled document aside and not using them in the training step. This set is commonly called the Test Set; its document can be classified and then a comparison can be made of the estimated labels with the real labels.

2.8.1. Naïve Bayes

The naïve Bayes classifier is known as a simple Bayesian classification algorithm. It is also considered a simple probabilistic classifier that is based on Bayes theorem, with strong naïve independence assumptions between attributes (i.e. features). The main idea is to calculate the probability that a document $d \in D$, which is represented as a bag-of-words and by each word $w \in W$, belongs to a particular class $c \in C$ based on Bayes' theorem, as in equation 15.

$$P(c|d) = \frac{P(d|c)P(c)}{P(d)} \tag{15}$$

The combination between the naïve independence assumption and the Bayes theorem gives the naïve Bayes classifier, as in equation 16.

$$P(c|d) = P(c) \prod_{w \in d} P(w|c) \quad (16)$$

In the context of naïve Bayes, the key feature is that the words $w \in W$ in the document $d \in D$ are independently given the class $c \in C$ value, where W denotes the set of all feature words, D denotes the set of training documents and C denotes the set of class labels. For example, a fruit can be considered an orange if it is orange, round and about 9 cm in diameter. In the naïve Bayes classification, these features are considered independent from each other in terms of their contribution to the probability that this fruit is an orange, ignoring any possible connection between the colour, roundness and diameter features. Furthermore, the naïve Bayes classifier does not consider a specific order of the words in the document as an important factor (Hotho, Nürnberger and Paaß, 2005; Kim, Han, Rim and Myaeng, 2006; Dai, Xue, Yang and Yu, 2007).

According to Kim et al. (2006) and Aggarwal and Zhai (2012), naïve Bayes has been one of the more popular machine learning techniques for many years due to its simplicity, which enables different tasks and achieves reasonable performance in these tasks based on independence assumptions. Many researches have shown that the performance of the naïve Bayes is very strong in the task of the classification.

Aggarwal and Zhai (2012) explain that for naïve Bayes classification, two classes of models, the multivariate Bernoulli model and the multinomial model, are usually used; both of them calculate the posterior probability of a class based on the appearance of words in the document. Both models work with bag-of-words assumptions, regardless of the actual position of the words in the document. The main difference between the models regards whether they take word frequency into consideration.

- **Multivariate Bernoulli model**: the presence or absence of words in a text document are used as features in terms of representing the document, and the word features in the text are assigned a binary of one of two values, indicating the presence or absence of a word in the text. However, word frequency is not used in modelling the document.
- **Multinomial model**: in this model, the document is represented by a bag-of-words, and thus word (term) frequency is calculated in the document. In each class, documents are

modelled as samples made from a multinomial word distribution. Therefore, a document can be given a class based on the probability of a word occurring in the document.

2.8.2. Random forest

The random forest classifier is comprised of tree classifiers, with the generation of each classifier based on a random vector that is sampled separately from the input vector. The classification of an input vector is defined when each tree casts one vote for the most popular class. The random forest classifier comprises a combination of features at every node, or randomly selected features, to grow a tree. Examples are usually classified by taking the most popularly voted class from amongst all the tree forecasters that are in the forest. To design a decision tree, the choice of an attribute or feature selection measure and a pruning method will be needed. Various approaches are adopted in the selection of attributes that can be employed in the decision tree induction. Most of the approaches directly assign a quality measure to the attribute. In a decision tree induction, the two most commonly used attribute selection measures are the Gini index and information grain ratio criterion. The random forest classifier implements the Gini Index as a measure of attribute selection that measures the impurity of an attribute in regards to the existing classes (Pal, 2005).

With the help of new training data and a combination of other features, trees are successfully grown to their maximum depth. These fully-grown trees are usually not pruned. This is one of the advantages of the random forest classifier compared to the other decision tree methods. As indicated by previous studies, the pruning method is what affects the performance of the tree-based classifier, not the attribute selection measures. It is speculated that as the number of trees increases, there is a convergence of the generalisation error, even if the tree is not pruned. And due to the strong law of large numbers, over-fitting has never been a problem. To generate a random forest classifier, two user-defined parameters are usually needed: the number of trees to be grown and the number of features used at individual nodes to generate a tree. At individual nodes, only selected features are considered for the best split. Hence, the random test classifier comprises N trees, where N represents the number of trees to be grown. To classify a new

dataset, every case of the data set has to be passed to each of the N trees. The forest then selects a class that has the most votes out of the N votes for that particular case (Pal, 2005).

2.8.3. Artificial neural networks

An artificial neural network (ANN) is a model powered by natural neurons in living things. Signals are passed to natural neurons by synapses that are situated on the neurone membranes or dendrites in animals (Kriger, 2007). In any computer application, the basics are similar to synapses, and they are the inputs. These are eventually multiplied by the strength of the signal that is received, which is the weight, and lastly calculated by a specific mathematical function that is meant to activate the neuron (Gershenson, 2014).

In a regular implementation of ANN, between artificial neurons the signal is always a real number, just as the product of every artificial neuron is calculated by a non-linear function that happens to be the sum of its units. Edges are connecting points between artificial neurons. Edges and artificial neurons adjust in weight as learning progresses. The resulting weight affects the signal strength at a connection by either decreasing it or increasing it. Artificial neurons may be operating with a threshold, such that signals will only be sent if the aggregate signals exceed that threshold. Normally, artificial neurons are separated into layers. All layers may have to carry out different types of transformations on their respective inputs. Signals travel from the input layer to the output layer, after possibly traversing many layers several times.

The initial focus of the ANN method was to have problems solved in a similar way to that of a human brain. Nevertheless, as time passed, attention was shifted slightly to executing other tasks, resulting in deviations from the initial biology. ANNs have been implemented for various purposes, including speech recognition, computer vision, social network filtering, machine translation, medical diagnosis and video games (Sharma, Rai and Dev, 2012).

2.8.4. Support vector machine

For a linearly separable task that is two-class, the goal of a support vector machine (SVM) is to find a hyperplane that can split the two classes of specific samples with a maximum margin that is capable of displaying the best generalisation ability. Generalisation ability means that a

classifier not only has an excellent classification performance, such as accuracy on training data, but also ensures high predictive accuracy for future data from similar distribution of the training data (Xue, Yang and Chen, 2009).

By intuition, a margin is the amount of space as defined by the hyperplane between the two classes. By geometry, the margin is the shortest distance measured between closest data points to any point on the hyperplane. Figure 15 shows a geometric reproduction of the corresponding optimal hyperplane concerning a two-dimension input space (Xue, Yang and Chen, 2009).

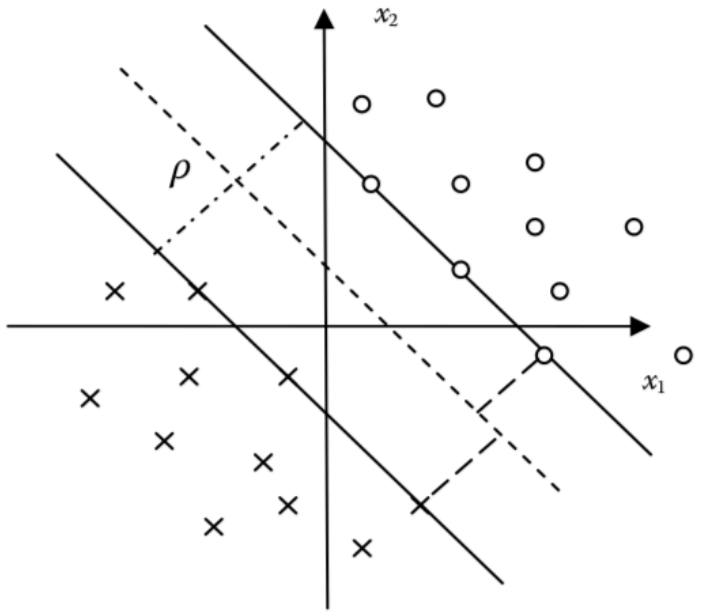

Figure 4. Illustration of the optimal hyperplane in SVC for a linearly separable case (Xue, Yang and Chen, 2009).

2.8.5. Decision tree

One of the most practical and commonly used methods for inductive inference is decision tree learning. This is an appropriate method for calculating discrete-valued functions, which is robust with respect to noisy data and capable of effectively learning disjunctive expressions. It is a non-parametric and supervised learning method that is used for regression and classification. The

objective is to create a model that could suggestively predict the value of a certain variable by mastering easy decision rules that are gathered from available data features (Polat and Güneş, 2007).

Decision tree learning is an effective method for calculating discrete-valued functions in which a decision tree illustrates the learned function. Learned trees are equally represented by various if-then set rules to enhance human readability. The deeper a tree is, the more advanced its decision rules, translating to a much fitter model. The learning methods are the most popular inductive inference algorithms. These learning methods have been employed in a variety of tasks, such as learning how to diagnose medical cases, learning how to evaluate the risks of giving out loans to applicants and many more. Decision tree learning can be regarded as a one-step look ahead (hill climbing, heuristic and non-backtracking search via the space of all the possible decision trees (Polat and Güneş, 2007).

2.8.6. K-nearest neighbour

The logic behind near neighbour classification is quite easy, as cases are classified based on the class of their nearest neighbours. It is a lot more useful to consider more than one neighbour, so the method is known as k-nearest neighbour classification (k-NN) where k-nearest neighbour is the class determinant. Since the data for these training sets are required at runtime, it could be referred to as memory-based classification, because induction is normally delayed until runtime, and it is regarded as a "lazy learning" technique. When a classification is founded directly on training examples, it is also called case-based classification or example-based classification (Cunningham and Delany, 2007).

The fundamental idea is demonstrated in Figure 16, which shows a 3-nearest neighbour classifier on a two-class problem, given a two-dimensional feature space. In this instance, the decision for q1 is quite straightforward. All three of its neighbours are of class O; hence, it is classified as an O. The case for q2 is bit more advanced because it has one neighbour of class O and two neighbours of class □. This can be usually decided by simple majority voting (Cunningham and Delany, 2007)

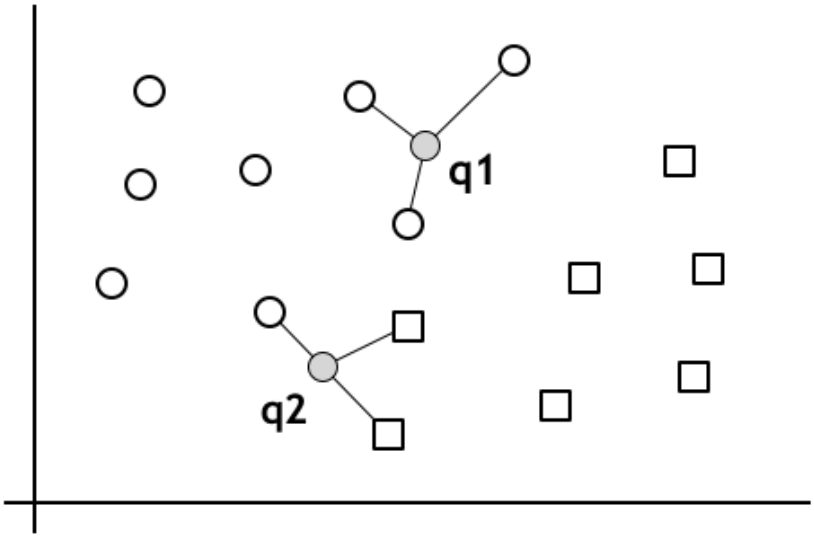

Figure 5. A simple example of 3-nearest neighbour classification (Cunningham and Delany, 2007).

As a result, k-NN classification can be divided into two stages, namely the determination of nearest neighbours and the classification of the nearest neighbours (Cunningham and Delany, 2007).

2.9. Clustering algorithms

The clustering method is considered an unsupervised classification that can partition data into a number of clusters (groups, subsets or categories). The main aims of clustering algorithms are finding insight into data, classifying data and compressing data. The difference between classifying and clustering algorithms is that the former needs labelled data to work on, whereas there is no need for labelled data in the latter. Clustering algorithms are powerful in separating a limited unlabelled dataset into a different number of partitions, with each partition including data patterns that have a reasonable correlation (Xu and Wunsch, 2005; Celebi, Kingravi and Vela, 2013).

Clustering algorithms can generally be divided into two types: hierarchical and partitional. The former algorithms discover nested clusters via a divisive (top-down) or agglomerative (bottom-up) approach, whereas the latter finds all clusters together as partitions of the data.

Hierarchical algorithms have high complexity, while partitional algorithms have lower complexity in the number of data points (Celebi, Kingravi and Vela, 2013).

2.9.1. K-means clustering algorithm

The K-means algorithm is the most commonly used partitional clustering algorithm. Several reasons make the K-means the most popular: first, it is very simple and easy to implement in solving many practical problems. Second, it is multilateral and every part in the algorithm can be modified. Third, it is invariable to data ordering, such as random shuffling of data points (Celebi, Kingravi and Vela, 2013). K-means was used in this research to classify users as influential or non-influential based on influence score.

3. Chapter Three: Methodology

3.1. Proposed research methodology

A great amount of research has been proposed to improve the performance of short-text-based recommender systems, such as recommending items based on Twitter accounts. Most of this research focuses on time and enrichment. In regards to the time factor, some studies have proved that short-term profiles, which are based on recent activity, such as from the last week, give more accurate recommendations than complete profiles that are built from all user activities. Other studies have proved that using a decay function that gives higher weight to recent activities than old activities can improve recommendation accuracy. In regards to enrichment, some studies have exploited external sources to enrich user profiles, such as Wikipedia, while others exploit the URL links shared on Twitter to enrich data on user profiles. These methods can be useful to supply user profiles with more information in order to improve the accuracy of recommender systems. However, some data gained from external sources might have no relevance to a user's interests and might then affect the performance of the recommender system. Also, many users do not provide enough links in their tweets to enhance their profiles. Furthermore, some users do not provide recent enough tweets to form sufficient information in order to give them better recommendations.

There is a need for better information sources that can feed user profiles with related information in the short-term (the last week for instance), in order to improve the performance of short-text-based recommender systems. One of the fields is to exploit the network of relationships a user has, and build their profile from related tweets from other accounts that are members of this network. To create a reliable profile, this approach uses explicit links between users (e.g. following links) to collect relevant recent activity from other related users. Implicit relationships between users were also explored (e.g. user similarity) and relative data was gathered in order to expand the user profiles with useful information. The intuition behind this method is that there are thus more recent activities to choose from that might let us avoid irrelevant external data and provide a more coherent story about why it is related, which thus allows us to improve the performance of short-text-based recommender systems.

The experiment of this research was conducted offline. Different users with different interests and different activity statuses (e.g. active or not active) were chosen randomly from Twitter. By using Twitter's API, information about a particular user can be gathered, such as following lists (e.g. friends and followers), timeline tweets (including retweets), favourited tweets and details about each tweet. All this information helped us to address the research questions. The experiment was evaluated and tested via users' provided activities; the details of the evaluation are explained in the next chapter. The experiment was carried out using the Python programming language.[1] Various algorithms from the Scikit-learn library[2] were also used in this approach.

The methodology consists of two main stages: explicit and implicit relationships (see Figure 4). The profiles constructed in these stages were evaluated by using them in recommender systems and comparing the results with state-of-the-art profiles. The two stages, in more detail, are as follows:

1- Discovering and examining tweets that have explicit relationships and classifying them into representative and not representative. The representative tweets were chosen and used in building the user profile, as they represent the user's interests. This kind of relationship is considered explicit, based on the manner of receiving tweets. There are two types of received tweets, direct and indirect (incoming tweets that appear in the user's timeline). Different user profiles were built from the direct and indirect tweets. These profiles were then evaluated by using them in the recommender system.
2- Exploring the properties of implicit relationships in order to improve the approach. After finding similar users who have implicit relationships, tweets were collected from them and then classified as representative or not representative in order to enhance the user profile with more relevant tweets. The profile of this step was evaluated using a recommender system.

[1] www.python.org
[2] www.scikit-learn.org

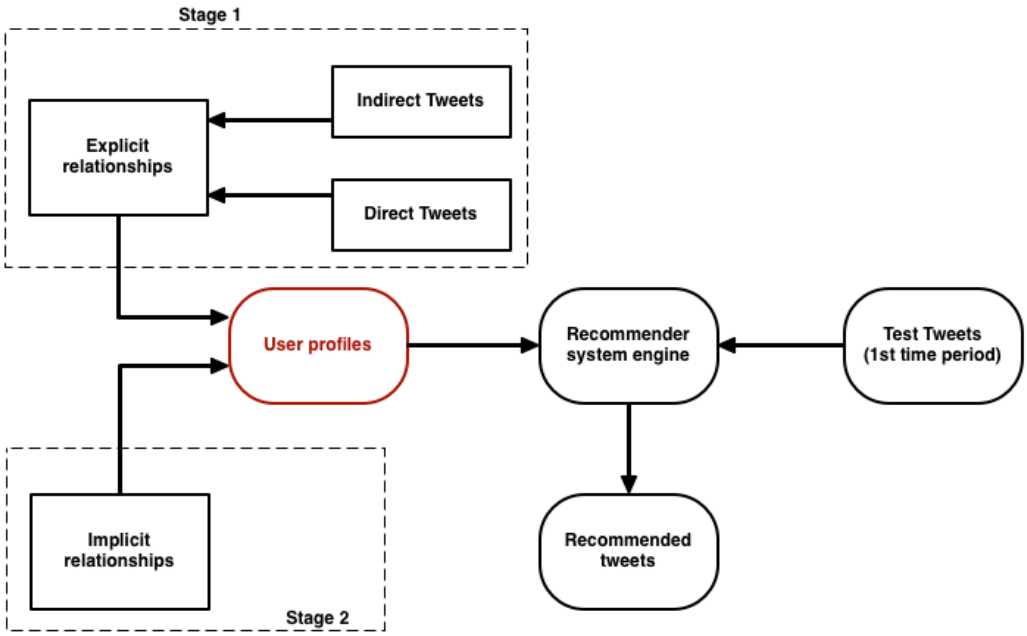

Figure 6. The main architecture of the proposed methodology.

In general, the main goal of this research was to find whether it is possible to improve the performance of short-text-based recommender systems by using tweets collected from explicit and implicit relationships among social media users. In this research user profiles were built as bag-of-words profiles, because Twitter posts are very short and have limited words. Different user profiles were built and examined from explicit and implicit relationships in order to explore which set of tweets represents the user profile better. The following section explains the main points of each stage.

3.1.1. Stage one

This stage involves finding sets of tweets from explicit relationships. Explicit relationships in this stage are direct and indirect. The former represents friends in the friends list (following list), whose tweets are received directly by the user (direct tweets). The latter represents accounts whose tweets are received indirectly; for instance the user receives their tweets when friends retweet them (indirect tweets). Indirect relationships can be from two sources in terms of receiving tweets: influential friends of influential friends and travelling tweets accounts.

3.1.1.1. Explicit direct relationships

1. Based on the user's timeline history, friends are clustered into three groups based on influence score: influential, less influential and non-influential.
2. Collecting influential friends' tweets from a short-term period and storing them in the user profile.
3. Using tweets from the user's timeline history (representative) and tweets of non-influential friends (not representative) to label the dataset and then train the machine learning algorithms. Representative tweets are the tweets that reflect the user interests such as tweets posted or retweeted by himself.
4. Applying machine learning classification on the less influential tweets using the labelled dataset from the previous step. Tweets are classified as representative (retweetable) or not representative (not retweetable). The tweets classified as representative are stored in the user profile (see Figure 5). The classifier data is saved in order to use it later.
5. Plugging the new profile, which contains influential friends' tweets and the classified less influential friends' tweets, into the recommender system and using test tweets for evaluation. The results are compared against other profiles and the best profile is extended and compared to profiles from other sources.

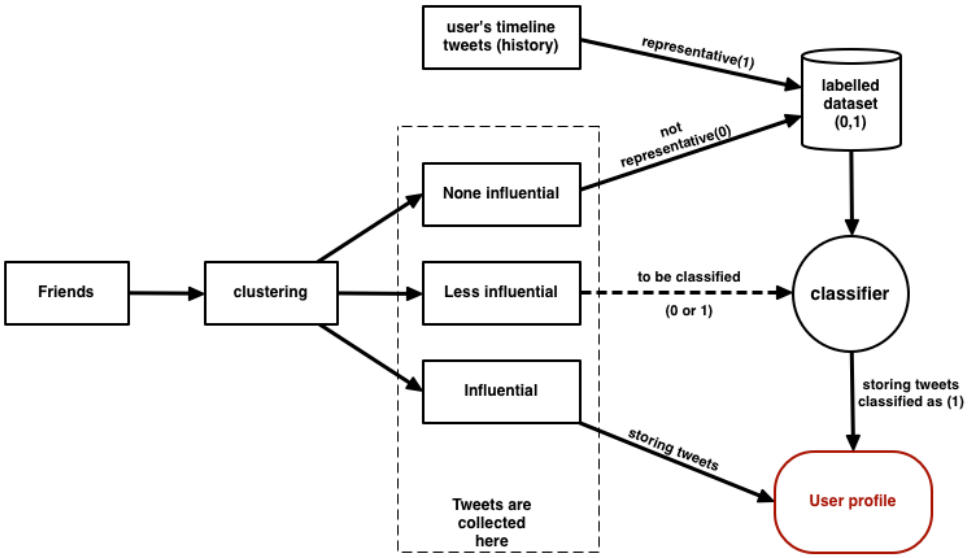

Figure 7. Building a user profile based on direct explicit relationships (Alshammari et al., 2018)

3.1.1.2. Explicit indirect relationships

1. Influencers of influential friends (Inf of Inf):
   a. Calculating the influence score between the influential friends and their friends in order to identify the influencers of influential friends.
   b. Clustering the friends of the influential friends into: influential, less influential and non-influential.
   c. Collecting the influential friends' tweets from within a short-term time and storing them in the profile.
   d. Using the saved classifier to classify the tweets of less influential friends. The tweets that are classified as retweetable are stored in the profile.
   e. The profile of this stage is tested and evaluated by plugging it into the recommender system. See Figure 6.

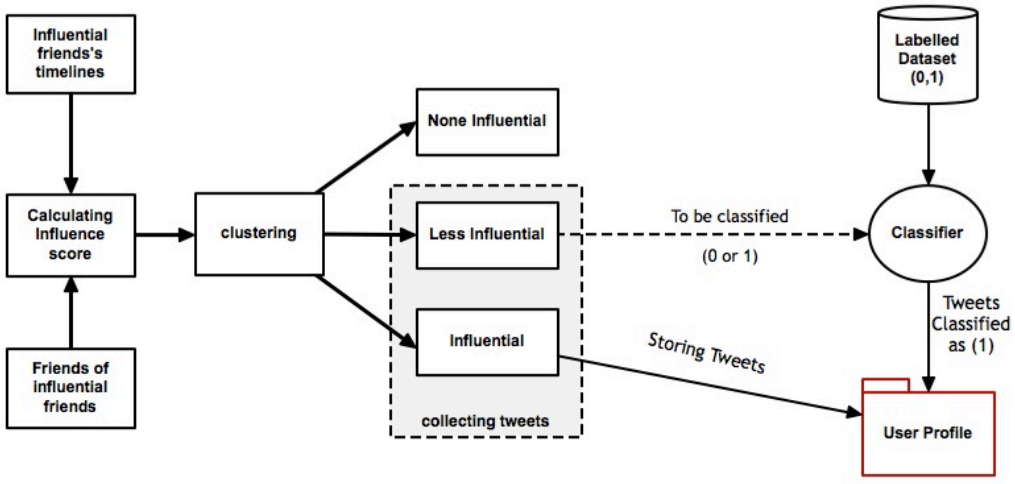

Figure 8. Building a user profile based on indirect explicit relationships (Inf of Inf).

2. Travelling tweets accounts (TTA):
   a. Based on user timeline, extracting the accounts that do not exist in the friends list.
   b. Collect tweets from within a short-term time from these accounts.

c. Using the saved classifier to classify collected tweets as retweetable or not retweetable. The retweetable tweets are stored in the profile. See Figure 7.
d. Plug the profile into the recommender system to evaluate it and compare its performance against other profiles.

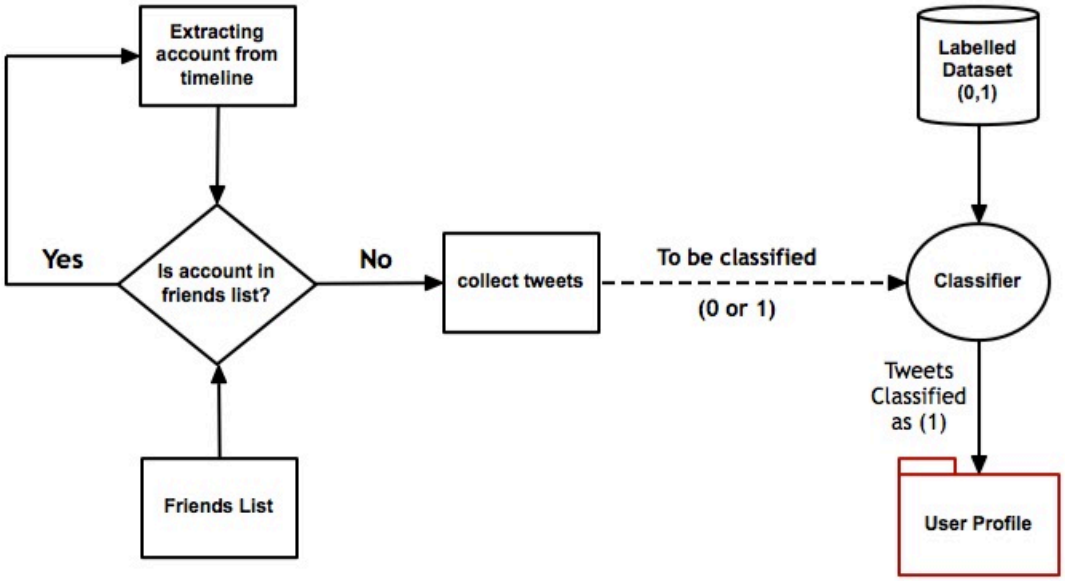

Figure 9. Building a user profile based on indirect explicit relationships (TTA).

3.1.2. Stage two

This stage involves finding related set of tweets posted by users within implicit relationships. Similar users are identified within a wider implicit network. Implicit relationships in this experiment are based on the idea that if a user sees an implicitly similar account, he/she will show some interest, such as following or retweeting. This kind of relationship is implicit because in the Twitter following structure there is no chance of receiving any tweets from these accounts because there is no explicit link between the user and the implicitly similar accounts. One of the hypotheses is that if two users follow similar influential friends, they are likely to be similar to each other even if they do not have any explicit links. Building the profile from this kind of relationship is explained in the following steps:

1. Finding implicitly similar accounts by collecting the followers of influential friends only.

2. Collecting tweets of the identified similar accounts.
3. Classifying collected tweets using the saved classifier into retweetable or not retweetable. The tweets classified as retweetable are stored in the profile. See Figure 8.
4. Using the recommender system to evaluate the profile from this stage and compare its performance against other profiles from the literature and from stage 1.

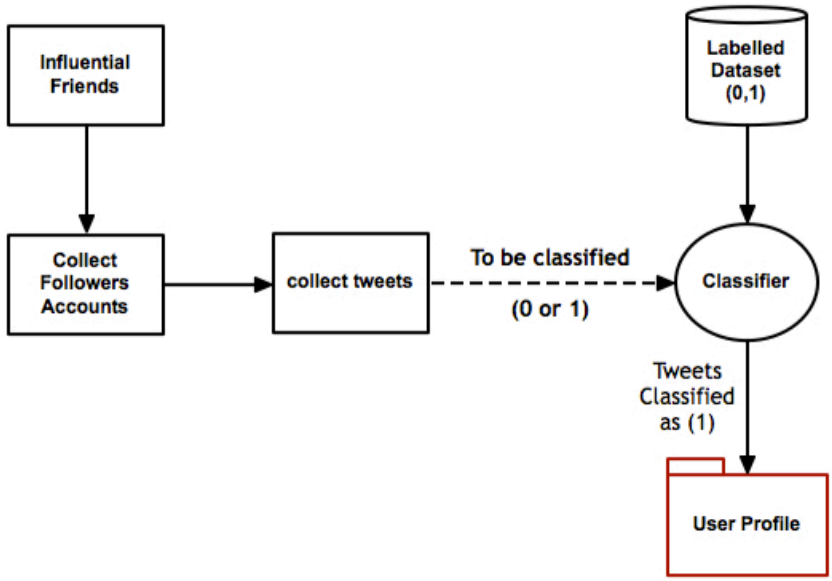

Figure 10. Applying stage 2 to exploit the implicit relationships to build a user profile.

3.2. Recommender system main architecture

The content-based recommender system used in our experiment has three main parts: user profile, recommender system engine and the recommendation items (see Figure 9). In this research, for a particular user, different profiles were used and compared in the same recommender system. Moreover, the recommender system engine and the recommendation items were not changed. The following sections explain each part of the recommender system in detail. The steps involved in building user profiles from the explicit and implicit relationships are also explained in detail.

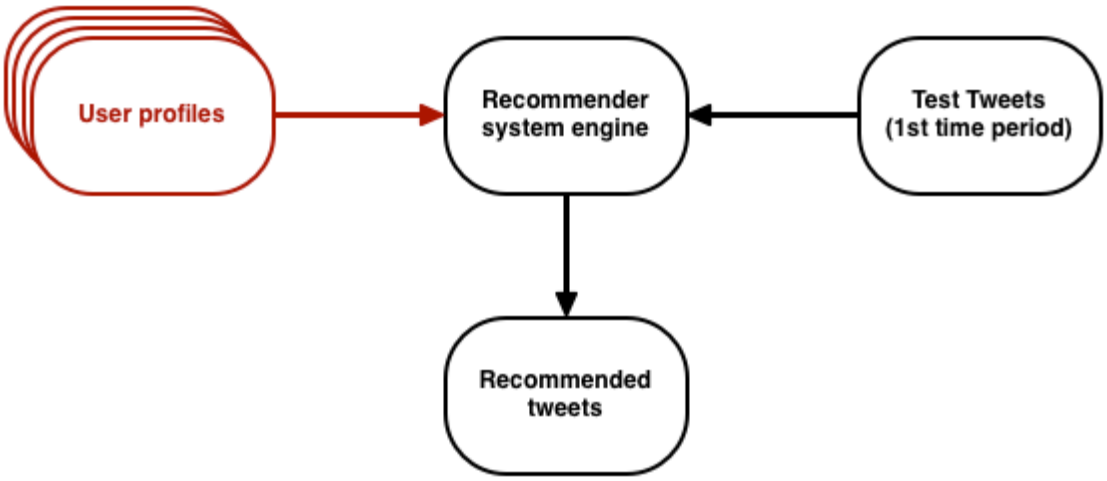

Figure 11. The main architecture of the content-based recommender system.

3.2.1. User profiles

A user profile contains important information about the user. A good profile can help the recommender system to deliver more accurate recommendations to the user. This research attempted to find which is the best profile in terms of helping the recommender system to recommend interesting items. These profiles were built from tweets from different sources, such as implicit and explicit relationships in Twitter. All profiles were built as keyword profiles (bag-of-words) and pre-processing steps were conducted before the recommendation process.

Micarelli et al. (2007) suggest that there is a number of steps that should be applied in order to build a document model. The aim is to filter the document and then extract important content. General steps are recommended as follows:

Step 1: Converting all letters in the document to lowercase or uppercase. The reason is to generalise the text into one consistent case to avoid mismatching of similar words.

Step 2: Remove any punctuation symbol such as ?, ^ and %. They add noise and are not necessary in the pre-processed document.

Step 3: Remove common linking words as they are unnecessary, such as stop words (e.g. the, then and because) and pronouns (e.g. I, we and they), because they are added several times to the document, which will cause a problem of weighting "important words" in the document.

Step 4: Stem every single word in the document. This step is done to return the word to its morphological root in order to recognise word variations. Porter (1980) proposed a very useful stemming algorithm, which is used without having a dictionary to compare each word in the document.

Step 5: Weight every term in the document to indicate its importance. A very common weighting technique is TF-IDF (term frequency * inverse document frequency). It is a weighting technique that gives each user/document profile a proper weight that indicates its importance. The equation for TF-IDF (W_i) is provided in equation 11, where t_i is the term i frequency in the document m and d_m is the number of all the terms in the document m, the result of that times the logarithm of $|D|$, which is the number of documents in the collection D divided by $DT(t_i)$, the number of documents D in which t_i appears.

$$W_i = \frac{t_i}{d_m} \text{Log} \frac{|D|}{DT(t_i)} \tag{11}$$

3.2.1.1. Building the profiles

From a targeted user perspective, examined tweets were collected from explicit and implicit relationship sources. They also had to be within a specific time limit, the last two weeks for instance, as in the work of Abel, Gao, Houben and Tao (2013). In the explicit relationship sources, the collected tweets were divided into directly received and indirectly received. The directly received tweets are those from the following list of the user, and the indirectly received tweets come from users outside the following list (i.e. tweets from users that a friend follows; see figures 10 and 11). The received tweets (direct and indirect) are the tweets that appear in the user's timeline, and so the user might show some interest by retweeting them. In the implicit relationship sources, tweets were collected from similar users, who do not have any explicit relationship with the targeted user, and if the targeted user sees them the user might show some interest by retweeting them.

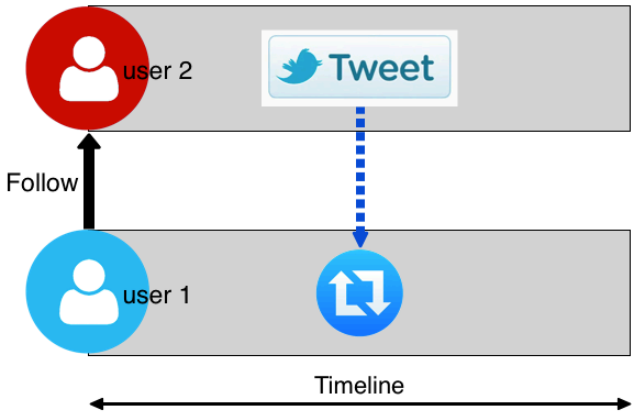

Figure 12. Directly received tweets. The blue user (user 1) follows the red one (user 2) and when the red user posts a tweet, it can be seen (received) and retweeted by the blue user.

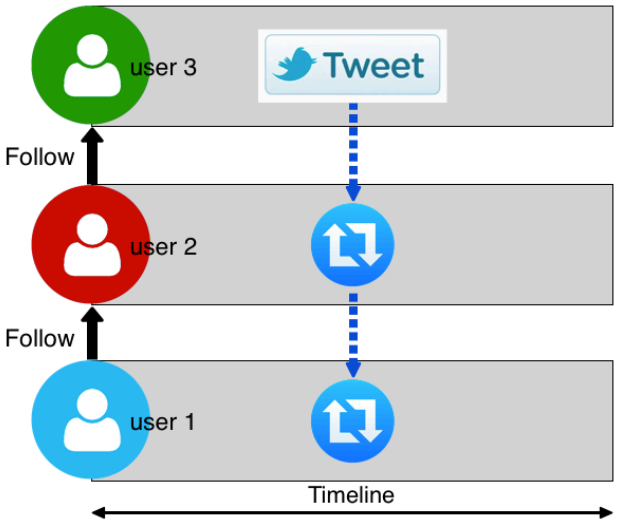

Figure 13. Indirectly received tweets. The blue user (user 1) follows the red one (user 2) and the red one follows the green user (user 3). However, the blue does not follow the green, but when the green posts a tweet and the red retweets the tweet, the blue can see (receive) and retweet it.

The generated tweets (posted by the user themselves) and retweeted tweets represent the user's interests obviously, whereas the received and collected tweets from explicit and implicit links need to be examined and classified.

3.2.1.1.1. Influence Score

Before building the profiles, data about the user was collected from the user's Twitter timeline, including: friends list, timeline tweets (posted and retweeted tweets) and favourited tweets. After that, the influence score between the user and his/her friends was calculated in order to help us choose and collect the appropriate content from users within different kinds of relationship. Influence score, which considers actions such as following, retweeting, replying and favouriting, might be a good attribute to find explicit and implicit relationships. Influential friends, which are related to the examined user and in their following list, have to be found using the influence score from the user's perspective. This score has to consider indicators of similarity, relationships and interactions. These indicators can be identified as: retweet, reply (mention) and favourite. Equation 12 calculates the influence score between the user and their friends. User 1 (u1) follows user 2 (u2):

$$\text{Influence Score} = \left(\frac{\sum \text{RTs (u2)}}{\sum \text{RTs p(u1)}} + \frac{\sum \text{RTs (u2)}}{\sum \text{Ts p(u2)}} + \frac{\sum \text{MT(u2)}}{\sum \text{MT p(u1)}} + \frac{\sum \text{FV(u2)}}{\sum \text{FV p(u1)}}\right) \times \frac{1}{4} \qquad (12)$$

- $\sum \text{RTs (u2)}$ represents the total number of tweets posted by user 2 and retweeted by user 1.
- $\sum \text{RTs p(u1)}$ represents the total number of retweets in user 1's profile.
- $\sum \text{Ts p(u2)}$ represents the number of tweets in user 2's profile.
- $\sum \text{MT(u2)}$ represents the number of replies (mentions) that user 1 posts to user 2.
- $\sum \text{MT p(u1)}$ represents the total number of mentions in user 1's profile.
- $\sum \text{FV(u2)}$ represents the total number of times user 1 favourites tweets from user 2.
- $\sum \text{FV p(u1)}$ represents the total number of favourite tweets in user 1's profile.
- $\frac{1}{4}$ is used to normalise the score between 0 and 1.

After calculating the influence score of the user's friends and before mining the relationships of the user, the friends list was divided into three groups based on influence score: influential users, less influential users and non-influential users. In this step, a K-means clustering algorithm was used to cluster the friends list into the three mentioned groups. This algorithm is explained in section 3.4.. The reason to choose K-means algorithm for this step is due its

advantages as it return the 3 clusters automatically and in very short time. Some other statistical methods could be used in this step such as standard deviation and quartile. However, the challenges are that the former is complicated to apply and consume time whereas K-means can be applied easier, can save time and the gives similar results. In quartile, the set of influential friends are not fixed for 25% as it could be form more or less than this percentage. Also, it cannot be assumed that most of less influential friends form 50%. Furthermore, most of the followed users (friends) are non-influential friends and their influence scores are 0. Therefore, K-means can divide the remaining data to influential and less influential friends.

3.2.1.2. Explicit relationships

The goal in regards to explicit relationships is to find sets of tweets that have explicit relationships to the user from within a short-term time, such as the last two weeks. In explicit relationships, there are two types of tweets: direct and indirect. Direct tweets are tweets that come from the following list (friends) of the examined user, whereas indirect tweets are tweets that come from users from outside the following list, but who have a relationship with the user's friends.

Direct tweets: All tweets from the influential group are taken to the user profile, whereas tweets from the non-influential group are not taken. Tweets from less influential users are classified as representative (retweetable) or not representative (not retweetable). Different classifiers are used in this process: naïve Bayes, random forest, support vector machine, decision tree, k-nearest neighbour and neural networks. All classifiers were trained using a labelled dataset from the user's timeline (the history of the user's tweets and retweets) and the tweets by non-influential users. The tweets from the user's timeline in the dataset are labelled as representative (retweetable) and the tweets from non-influential friends are labelled as not representative (not retweetable). The dataset was divided into training and testing sets. The latter was used to calculate the accuracy of each classifier. The classifier with the highest accuracy is chosen automatically to classify less influential friends' tweets. This step is to make sure the tweets are classified with the most accurate classifier. After classifying the less influential users'

tweets, the tweets that the machine learning classified as representative are stored in the user profile alongside the influential users' tweets.

Indirect tweets: these are the tweets from outside the friends list. There are two types of these tweets in terms of the manner of receiving them: those from friends of friends and travelling tweets. The former reach the user from his/her friends when they retweet something posted by someone in their friends list. The manner of exploiting this type of tweets is to apply the same methodology as for direct tweets. Influence score was calculated between each influential friend and their friends. The intuition behind this was to find whether the influence impact continues to affect the user profile even from outside their friends list. After calculating the influence score between each influential friend and their friends, these friends were clustered into influential, less influential and non-influential, based on a k-means algorithm (see Figure 12). After that, tweets of influential friends of influential friends within the short-term time were stored in the profile, while non-influential tweets were excluded. Tweets from less influential friends of influential friends were collected and then classified into retweetable or not retweetable using the same saved and trained classifier from the direct tweets stage. The tweets that were classified as retweetable were added to the user profile. This profile is named Inf of Inf, which refers to tweets by influential friends of influential friends.

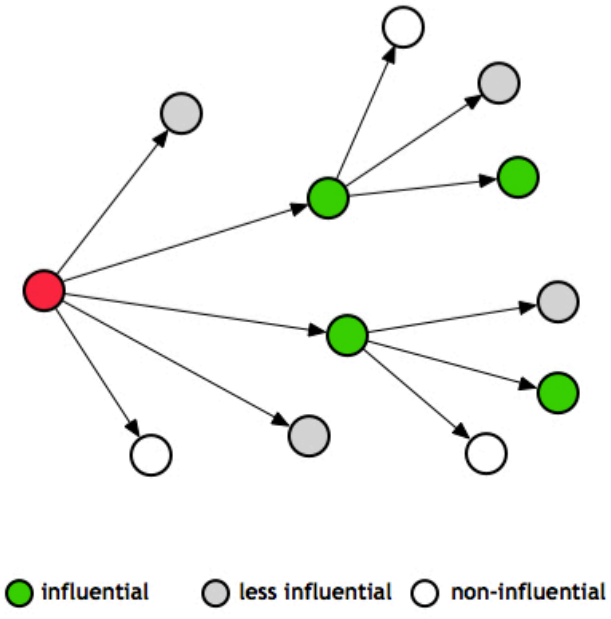

Figure 14. Inf of Inf profile sources.

Travelling tweets are tweets that reach the user after travelling from a long distance. They travel by having been retweeted by many Twitter users until they reach the examined user. In other words, they reach the user from outside his/her friends and their friends (see Figure 13). These tweets can be found in the user's timeline and, therefore, the accounts that posted these tweets are extracted and tweets are collected from those accounts from within a short-term time. The saved trained classifier from the direct tweets stage was used to classify these tweets into r-tweetable or not retweetable. The retweetable tweets were added to the user's profile, whereas not retweetable tweets were excluded. This profile is named TTA, which refers to travelling tweets accounts.

Both the Inf of Inf and TTA profiles are extended versions of the direct tweets profile. Tweets by both types were merged into one profile in order to determine whether this step can perform better.

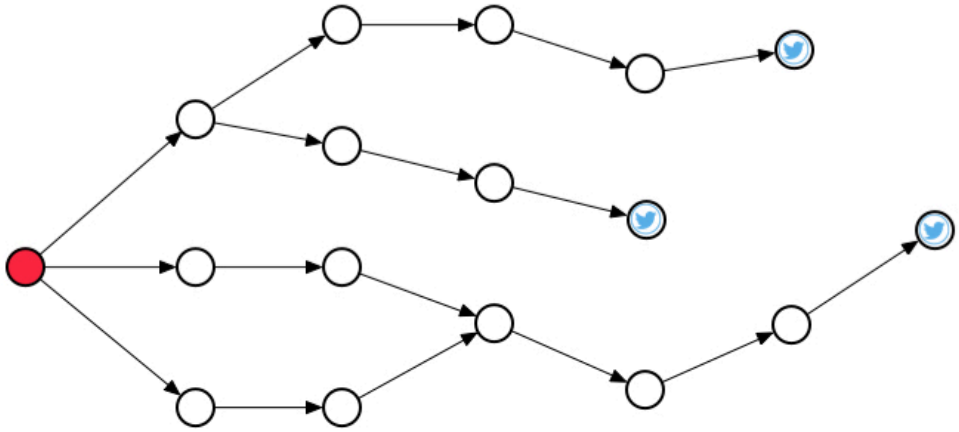

Figure 15. TTA profile sources.

3.2.1.3. Implicit relationships

The goal of implicit relationships is to find further recent related tweets in order to use them in the user profile, by discovering and mining properties of the implicit relationships in the wider network around the user. One of the main keys to finding similar accounts in a wider network around the user is the influence score of the followed accounts. For example, if two users have more influential users in common, they might share similar interests. Different user profiles

were built from different implicit relationships to test different aspects of similarity. For example, similarity with users who have a high number of followers may not work well, but similarity with users who share influential users might work better. In this stage, follower accounts from the influential friends were collected. Their tweets from within the short-term time were then collected and classified using the classifier from stage 1. The tweets classified as retweetable were added to the direct tweets profile. The profile will be an extended version of the direct tweets profile and is named InfF, which refers to influential followers. Another profile was built at this stage based on the followers of friends who have a high number of followers (FCF). This profile was used to test the hypothesis that our proposed influence algorithm can find implicit similar accounts more successfully than the algorithms from previous studies (see Figure 14).

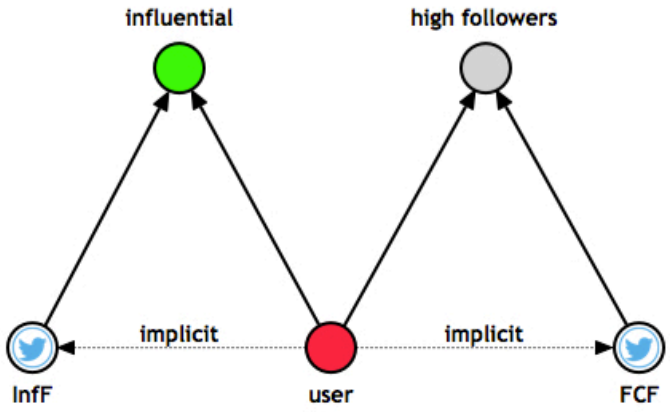

Figure 16. Implicit profiles sources.

3.2.1.4. Profiles to be compared

After mining and collecting tweets from different sources and from different relationships, user profiles were built, and their performance was then compared against each other and against profiles from the literature. The tweets which the user generated or retweeted within the short-term time were included in all profiles.

3.2.2. Recommender system engine

In this part, our system uses the vector space model representation, which considers user profiles and recommended items as vectors and then calculates the angle between them. The closer the items to the user profile, the more relevant they are considered. Cosine similarity was used at this stage to calculate the angles, as in equation 13.

$$\text{sim}(d_i, q) = \cos\theta = \frac{d_i.q}{|d_i||q|} = \frac{\sum_i w_{i,j}\, w_{q,j}}{\sqrt{\sum_j w_{i,j}^2}\sqrt{\sum_j w_{q,j}^2}} \quad (13)$$

Before calculating the similarity, documents in user profiles and recommended items must be pre-processed to avoid sparsity and low accuracy. Also, the keywords in them must be weighted using the TF-IDF weighting technique.

3.2.3. Recommendation items

In this part, the user was given a set of documents as recommendations. In our experiment, the recommendation items are sets of tweets that the user might show some interest in by retweeting or favouriting them. The details of how the tests tweets were chosen are explained in section 4.1.3.

3.3. Evaluating the approach

In this research, offline evaluation was used to measure the accuracy of the recommender system with different user profiles. Ricci et al. (2015) state that using offline evaluation is easy and more flexible in the case that developers want to run and compare different algorithms or approaches, because the experiments can be performed more cost-effectively. As in our methodology, various user profiles were used in the recommender system and were then compared. The metrics that were used in this research in order to measure the accuracy of the performance of the system were: precision, recall, average precision at top-k and mean average precision (MAP).

The proposed approach was evaluated by using test tweets that were plugged into the recommender system. This set of tweets was collected from the user's timeline, influential

friends, less influential friends and non-influential users. In other words, the timeline tweets, which are already retweeted by the user, were considered relevant tweets, whereas non-influential tweets were considered not relevant. The test tweets are explained in detail in the next chapter.

Each one of the profiles for a particular user was plugged into the recommender system with the same test tweets. As a result, the evaluation metrics can be calculated based on the set of recommendations that the recommender system provides.

4. Chapter Four: Experiment and Results

This chapter clarifies the experiment in order to evaluate the proposed methodology, which exploits relationships on Twitter to find sets of tweets that represent the interests of a particular user. These sets in turn raise the accuracy of the recommender system. This chapter explains in detail all the steps of the experiment and the preparation of the datasets.

4.1. Preparation of datasets and experiment setup

After collecting the timelines of the examined users and before calculating the influence score and clustering users into the three mentioned groups, the dataset was divided into three time frames. The tweets from the first time frame (two weeks ago) were used as test items (test tweets), and it is already known that the user showed interest in them by retweeting them. This is like going back in a time machine into the past in order to predict a future that is already known, and this can help in the evaluation process. The tweets from the second time frame (between two weeks ago and four weeks ago) were used in building user profiles from different sources, as this time frame represents only the last two weeks before evaluation. The third time frame (more than four weeks ago) was used alongside the second period to calculate the influence score only from the timeline of the examined user. The user's timeline in this period was also used in the machine learning classification (see Figure 17).

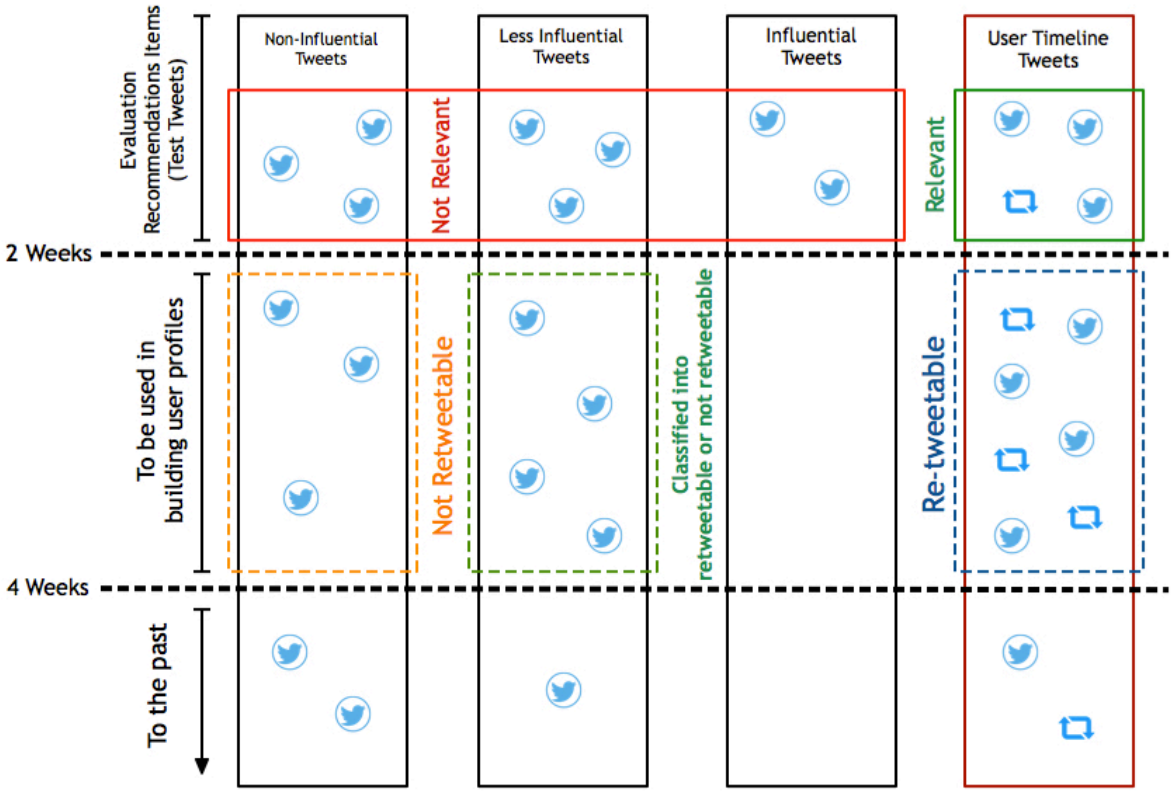

Figure 17. The three time frames and their uses in the experiment (Alshammari et al., 2019).

4.1.1. Profiles from explicit relationships

4.1.1.1. Profiles from direct relationships

After calculating the influence score for the examined user and their friends, and clustering the friends into influential, less influential and non-influential, tweets were collected from influential and less influential friends. Different profiles were then built, examined and compared in order to discover which profile performs better, with the aim of extending this later in other stages. These profiles are:

1- Baseline: includes all tweets from the user's timeline. Contains both tweets that the user posted and tweets that the user retweeted.

2- BLCinf: includes all tweets from the user's timeline, short-term tweets by influential friends (second time frame) and short-term tweets by less influential friends that are classified as representative (second time frame).
3- STBLCinf: includes tweets only from the second time frame (short-term), and includes tweets from the user's timeline, tweets by influential friends and tweets by less influential friends that are classified as retweetable.
4- STBLinf: includes tweets only from the second time frame (short-term), and includes tweets from the user's timeline, and all tweets by influential and less influential friends, without consideration of classification.
5- BLinf: includes all tweets from the user's timeline and all short-term tweets by influential and less influential friends. Classification of the tweets of less influential friends is not considered in this profile.

4.1.1.2. Profiles from indirect relationships

There are three profiles in this phase and all of them are extended versions of the best performing profile from the direct relationships phase. These profiles are:

1- Inf of Inf: includes tweets by influential friends of influential friends within the short-term time.
2- TTA: includes tweets from the travelling tweets accounts from within the short-term time frame and that are classified as retweetable.
3- Inf of Inf + TTA: includes all tweets from both previous profiles.

4.1.2. Profiles from implicit relationships

Profiles in this phase are built from accounts that have implicit relationships. These implicit relationships can be identified by the proposed influence algorithm. Additionally, the follower count strategy was used in building profiles in order to discover the efficiency of the proposed influence algorithm in finding implicit relationships. The profiles of this stage are extended versions of direct tweet profile, as follows:

1- InfF: includes the tweets of influential friends' followers that are classified as retweetable within the short-term time. Influential friends are identified by the proposed influence score.
2- FCF: includes the tweets of friends' followers who are identified as influential based on follower count. This profile is built within the short-term time.
3- InfF + FCF: includes the tweets from the InfF and FCF profiles. Duplicated tweets are excluded. For example, if a tweet is collected from both InfF and FCF, then it will be saved only once.

4.1.3. Test tweets

Test tweets are used to evaluate the accuracy of the recommender system. They are a collection of tweets from the first time frame (two weeks ago), as explained previously, and they are used as recommendation items in the recommender system. This collection contains both relevant and non-relevant tweets. From the timeline of the user, it is already known which items the user has shown some interest in by retweeting them, and these are considered relevant items. Tweets of friends in the same time frame, which the user did not retweet, can be used as non-relevant items. As a result, the set of recommended items contains a mixture of relevant and non-relevant items, which helps the recommender system to measure accuracy when it is run with different user profiles. It also allows a comparison between the built profiles and the baseline profiles.

4.2. Experiment setup and evaluation

The experiment was applied on one recommender system with different user profiles. In other words, for an individual user, different user profiles were plugged into the same recommender system using the same recommended items (test tweets). The recommendation results of each profile could then be evaluated (see Figure 18).

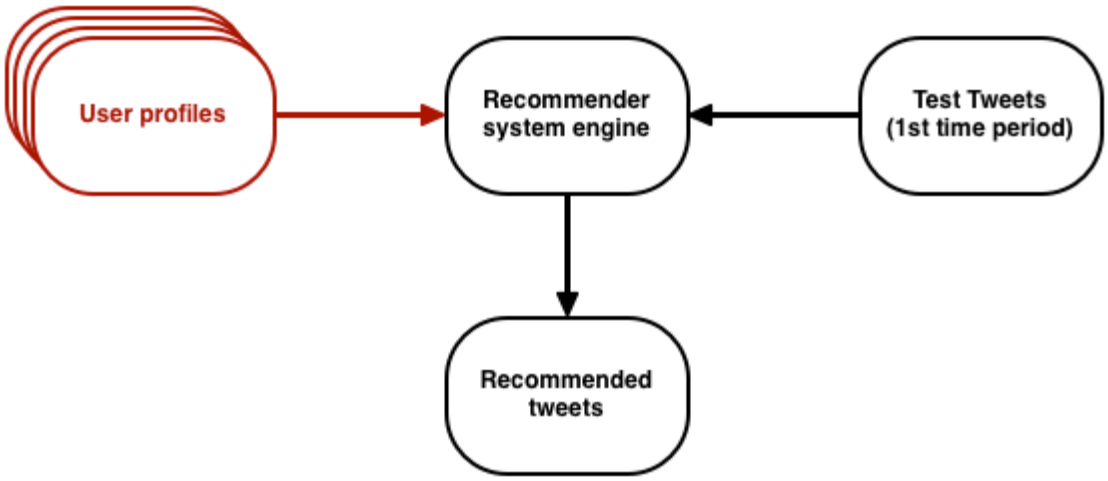

Figure 18. The main recommender system structure of the experiment, using the same recommender system engine and the same collection of test tweets. A different user profile is plugged into the recommender system each time.

A classifier can be used in classifying the relevant tweets and then recommend them instead of the cosine similarity. However, the classifier can recommend set of tweets but it cannot decide which tweet is more important than the others and then recommend it within the top 3 recommendations for example. On the other hand, in cosine similarity, the closest the tweet to the user profile it is recommended first. As a result, cosine similarity was used in the recommender system's engine; before that words in both user profile and the test tweets had to be weighted using TF or TF-IDF. It is known that tweets are short and also might have a connection with each other, which might affect the performance of the recommender system's engine. In order to choose the best weighting system for the experiment, the baseline profile was tested with TF and TF-IDF. The n-gram values also had to be tested alongside the weighting systems.

An n-gram represents a sequence of n items from a particular sample of speech or text. These items can be syllables, phonemes, words or letters, based on the application. Typically, n-grams are gathered from a speech or text corpus. In our experiment the items are words. Using numerical prefixes that are Latin, a unigram represents an n-gram of single size or size 1; a bigram is size 2; a trigram represents size 3 (Broder et al., 1997; Li et al., 2005).

The intuition behind choosing the baseline in this test was that it is the only profile that is considered standard and the results give a more reliable ground truth. However, the proposed profiles are yet to be tested at this stage, and are evaluated later against the baseline. Figure 19 shows the result of testing different weighting systems and also different n-gram values. The TF-IDF system proved itself significantly better than TF in all n-gram values. It performs best with trigrams. As a result, the recommender system engine will rely on the TF-IDF weighting system and also use trigrams.

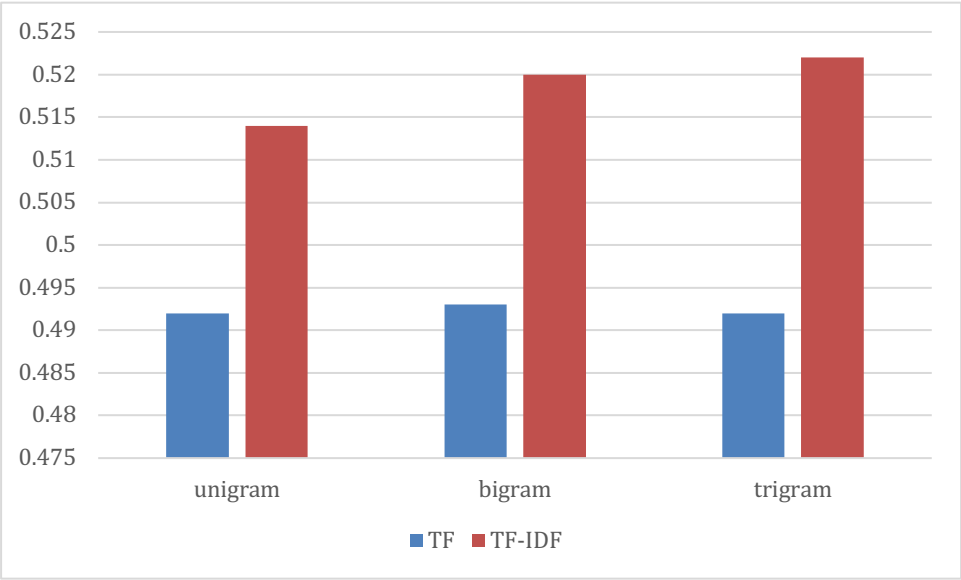

Figure 19. The TF and TF-IDF weighting systems with different n-gram values.

4.2.1. Evaluation metrics

Offline evaluation was used to measure the accuracy of the recommender system with different user profiles. Ricci et al. (2015) state that using offline evaluation is easy and more flexible in the case that developers want to run and compare different algorithms or approaches, because the experiments can be performed more cost-effectively. As in our methodology, various user profiles were used in the recommender system and were then compared. The metrics that were

used in this research in order to measure the accuracy of the performance of the system were: precision (P), recall (R), average precision (AP) and mean average precision (MAP).

In the task of recommending items in our approach, basically the focus is only on binary ratings: either a the tweet was retweeted (1) or not (0). Each tweet in the recommendation list was classified as one of the cells in Table 3. Preferred in our case refers to retweeted, and not preferred refers to not retweeted.

Table 2. Classification of the possible result of a recommendation of an item to a user (Gunawardana and Shani, 2009).

	Recommended	Not Recommended
Preferred	True-Positive (TP)	False-Negative (FN)
Not Preferred	False-Positive (FP)	True-Negative (TN)

It is possible to count the number of items (tweets) that fall into each cell in the table, and calculate precision and recall metrics, as in equations 17 and 18, respectively.

$$Precision\ (P) = \frac{TP}{TP+FP} \tag{17}$$

$$Recall\ (R) = \frac{TP}{TP+FN} \tag{18}$$

Precision measures how many of the items that the system retrieved were actually relevant to the user. In other words, it measures the number of relevant identified items as a percentage of the number of retrieved items. The higher the precision value, the better the system is at retrieving relevant items. Recall, on the other hand, measures the number of relevant identified items as a percentage of the total number of relevant items. The higher the recall rate, the better the system is at not missing items relevant to the user (Maynard, Peters and Li, 2006). In the case of multiple users and for a fixed list of recommendations, top 10 recommendations for

instance, precision and recall can be calculated for each user, and then the average precision and recall can be computed (Shani and Gunawardana, 2011).

Based on precision and recall, other evaluation metrics were used, namely average precision (AP) and mean average precision (MAP). The former measures the mean of all precision scores obtained after each relevant item is retrieved. The latter is used when there is a set of queries; it calculates the mean of the average precision scores for each query (Yue, Finley, Radlinski and Joachims, 2007; Buckley and Voorhees, 2017). The equations for both of these metrics are presented in equations 19 and 20.

$$Average\ Precision\ (AP) = \frac{\sum P@k}{R} \tag{19}$$

Where R is the total number of relevant items, and $P@k$ is the precision of the top-k retrieved items.

$$Mean\ average\ precision\ (MAP) = \frac{\sum_{q=1}^{Q} AP(q)}{Q} \tag{20}$$

Where Q is the total number of queries, which is the number of users in our case.

For multiple users, as in our approach, some users prefer a short list of recommendations and others prefer more recommendations. The number of items in the list of recommendations is not fixed. For example, such a system gives five recommended items. This number of recommendations is enough for one user but it is not enough for other users. Therefore, the recommendation list length k should be tested with different lengths between one and 10 (Gunawardana and Shani, 2009). Based on $P@k$, the average value ($AP@k$) is calculated according to equation 21.

$$Average\ precision@k\ (AP@k) = \frac{\sum P@k}{Q} \tag{21}$$

Where $\sum P@k$ represents the sum of precision $@k$ and Q represent the number of queries, which is the number of users in our case. For example, when k is set to 5, all precision@5 values of all users are aggregated and then divided by the number of users.

The two main measurements in the results section are the average of precision@k (AP@k) and the Mean Average Precision (MAP). The former measure the ability of the profiles in increasing the precision (increasing the relevant items in the top k) whereas the latter is to measure the ability of the profiles in ordering the relevant items in better way. In other words, it measure the ability of retrieving every relevant item in better way.

4.2.2. Choosing classifiers

During the classification stages, different classifiers were used in the experiment. After building the labelled dataset for each user, different classifiers were tested in order to measure their accuracy, and at the end the best classifier was chosen to classify friends' tweets. For example, one classifier can achieve the highest accuracy for a particular user but cannot achieve the highest accuracy for another user. Six classifiers were applied in our experiment: random forest, naïve Bayes, support vector machine (SVM), decision tree, k-nearest neighbour (k-NN) and neural networks. They were applied in their default settings and values in order to locate which classifiers will exhibit better performance. Table 4 shows the classifiers and the number of users for which the classifier achieved the highest accuracy.

Table 3. The number of users for which each classifier achieved the highest accuracy.

Classifier	Number of users
Random forest	12
Naïve Bayes	12
SVM	1
Decision tree	0
K-NN	0

Neural networks	15
Total	40

The default values of each classifier are explained in the subsequent subsections. Changes to the values were then applied to improve each classifier's performance. These changes were carried out using a pilot study consisting of three users in each classifier.

Random forest: The default settings values of the random forest classifier are:

- Number of estimators or number of trees = 10.
- Maximum depth of the tree = none. The nodes are expanded until all leaves are pure.

The only parameter in this classifier that can be tested is the number of trees. Different values are tested and the results of this test are shown in Table 5. The appropriate number of trees is 120.

Table 4. Tested values of number of trees and the accuracy of the test.

Number of trees	**10**	**30**	**50**	**70**	**90**	**100**	**120**	**130**	**140**
User 1	0.802	0.811	0.82	0.814	0.814	0.821	0.825	0.82	0.819
User 2	0.911	0.909	0.909	0.911	0.912	0.911	0.909	0.911	0.912
User 3	0.91	0.92	0.917	0.923	0.924	0.924	0.924	0.926	0.923
Average accuracy	0.874	0.88	0.882	0.882	0.883	0.885	**0.886**	0.885	0.884

Naïve Bayes: The only important parameter of the naïve Bayes classifier is alpha, and the default value is 1. Table 6 shows different values that were tested in order to find out which is the right alpha value. The best alpha value of this study was 0.2.

Table 5. Different alpha values and their accuracy.

Alpha	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**	**1**
User 1	0.848	0.847	0.844	0.843	0.842	0.839	0.84	0.839	0.84	0.839
User 2	0.863	0.866	0.866	0.867	0.866	0.865	0.865	0.862	0.861	0.861
User 3	0.864	0.871	0.871	0.864	0.864	0.864	0.857	0.857	0.85	0.85
Average accuracy	0.858	**0.861**	0.860	0.858	0.857	0.856	0.854	0.852	0.850	0.85

Artificial neural networks: The important parameters of artificial neural networks are how many hidden layers there are and how many neurons are inside each hidden layer. The default value is one hidden layer containing 100 neurons. Table 7 shows the results of a different number of neurons in one hidden layer.

Table 6. The accuracy of different tested values of neurons.

Number of neurons	**10**	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**
User 1	0.888	0.876	0.888	0.881	0.882	0.887	0.884	0.887	0.884	0.882
User 2	0.745	0.762	0.752	0.762	0.757	0.762	0.760	0.757	0.755	0.753
User 3	0.819	0.820	0.827	0.837	0.804	0.820	0.824	0.823	0.826	0.826
Average accuracy	0.817	0.819	0.822	**0.827**	0.814	0.823	0.823	0.822	0.821	0.820

As shown above, 40 neurons achieved the highest average accuracy value. However, it was necessary to find the appropriate value around 40. Therefore, values between 35 and 45 were tested to see if there was a possibility of finding better values of neurons around the number 40. Table 8 shows the tested values, and the best value remains 40.

Table 7. Tested values between 35 and 45 and their accuracy.

Number of neurons	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**	**43**	**44**	**45**
User 1	0.873	0.881	0.882	0.881	0.887	0.881	0.881	0.881	0.879	0.881	0.879
User 2	0.762	0.757	0.755	0.741	0.755	0.762	0.760	0.758	0.755	0.748	0.755
User 3	0.815	0.831	0.819	0.818	0.824	0.837	0.819	0.827	0.823	0.820	0.826
Average accuracy	0.817	0.823	0.819	0.813	0.822	**0.827**	0.820	0.822	0.819	0.816	0.820

The next step was to find which is the best number of hidden layers that can achieve the best average accuracy. Therefore, different values between 1 and 10 were tested; Table 9 shows that one hidden layer has the highest average accuracy.

Table 8. Tested values of hidden layers and their accuracy.

Hidden layers	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
User 1	0.881	0.873	0.882	0.881	0.850	0.850	0.870	0.856	0.855	0.785
User 2	0.762	0.758	0.764	0.753	0.764	0.764	0.762	0.760	0.623	0.770
User 3	0.837	0.828	0.823	0.832	0.819	0.828	0.822	0.831	0.811	0.773
Average accuracy	**0.827**	0.820	0.823	0.822	0.811	0.814	0.818	0.816	0.763	0.776

K-nearest neighbour (k-NN): The default values of the k-NN classifier are:

- Number of neighbours = 3.
- Weight function used in prediction = ‘uniform’.
- Distance metric = ‘minkowski’.

The previous pilot study showed the best parameters settings and values of the best three classifiers in our dataset. The best three classifiers with their new parameters were applied on all users in the dataset alongside the other three classifiers with their default values. Table 10 shows the new results of the classifiers and the number of users that the classifier achieved the highest accuracy with. Surprisingly, the naïve Bayes classifier with the new settings achieved the highest number of users with 60%. The second-best classifier was random forest, achieving 25%, while the neural networks classifier scored 15%. The other three classifiers did not achieve any better performance.

Table 9. Number of users of each classifier that achieved the highest accuracy after modifying the parameters of the classifiers.

Classifier	**Users**	**Percentage %**
Random forest	10	25%
Naïve Bayes	24	60%
SVM	0	0
Decision tree	0	0
K-NN	0	0
Neural networks	6	15%

Total	40	100%

4.3. Results

This section presents and explains the results obtained from the experiment. The experiment was carried out on 40 different users that were chosen randomly. Different metrics were used to measure the accuracy of the recommender system after plugging in different profiles. The evaluation set contains 20% of relevant items and 80% of irrelevant items. For example, such a user has 10 relevant items, will be mixed with 40 irrelevant items. As a result, the total set number is 50. The next sub-sections present the results in detail. The results of each experiment are divided into two types: general results and dividing dataset into different categories. The direct explicit relationship profiles are compared against the baseline profile, profiles from the literature and similarity profiles. Indirect explicit and implicit relationship profiles are compared against the best performing profile from the direct explicit relationship profiles.

4.3.1. Direct explicit relationships

4.3.1.1. General results

After building profiles, as explained previously, from tweets that came through explicit direct relationships between the user and their following list (friends), there is a need to know which profile performs the best. Moreover, is this profile performing better than the baseline profile? It will be used later for comparison with profiles from previous research. It will also be used to build other profiles from indirect explicit and implicit relationships. Results were obtained from the metrics of average precision at top-k (AP@k) recommendations and mean average precision (MAP). The tested values of k are from 1 to 10. This results were published on the work of Alshammari et al. (2018).

Figure 20 shows that the STBLCinf profile achieved the highest average precision seven times in the top 10 recommendations, which are: 1, 2, 3, 4, 5, 6 and 8, whereas the BLCinf

profile came second by achieving the highest value three times in k = 7, 9 and 10. Therefore, the former profile has the best performance with 70% of the k values, while the latter is second with 30%. The STBLinf profile, which is the most similar to the STBLCinf profile, achieved the worst performance among all the profiles, including the baseline one.

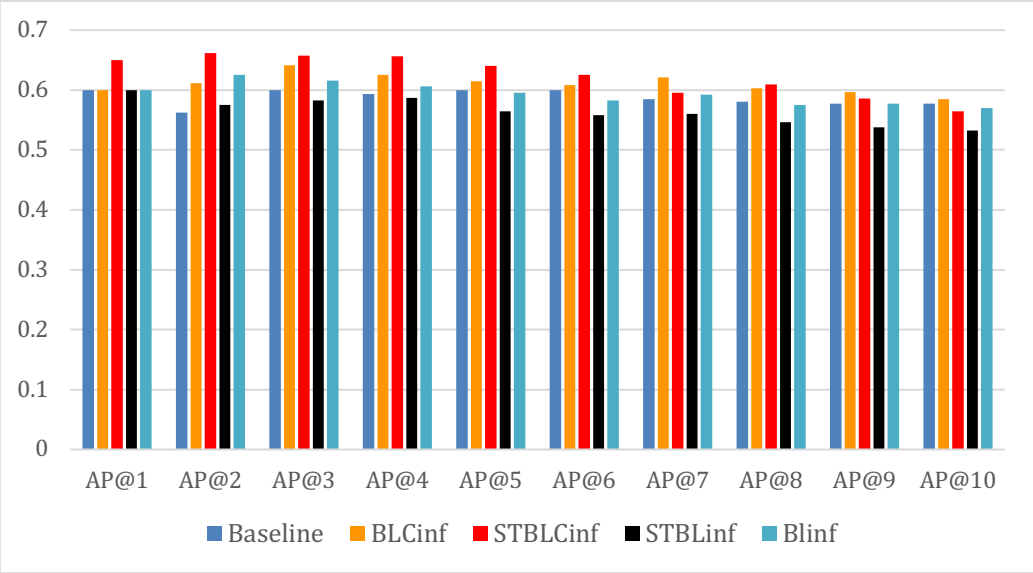

Figure 20. Average precision (AP) @ top-10 recommendations of the tested profiles.

In regards to the MAP metric , Figure 21 shows that the STBLCinf profile has the best MAP value in comparison to the other profiles with 0.55, whereas the STBLinf profile has the worst MAP with a value of 0.50. Both profiles were built in a similar way but one of them achieved the best performance and the other achieved the lowest MAP. The BLCinf profile, which is built based on the baseline and the short-term tweets from influential and less influential friends, achieved better performance with a MAP value of 0.53 than the baseline profile, which has a MAP value of 0.52, and came second after the STBLCinf profile. The BLinf profile, which was built similarly to BLCinf, achieved the second lowest MAP; the baseline profile was better.

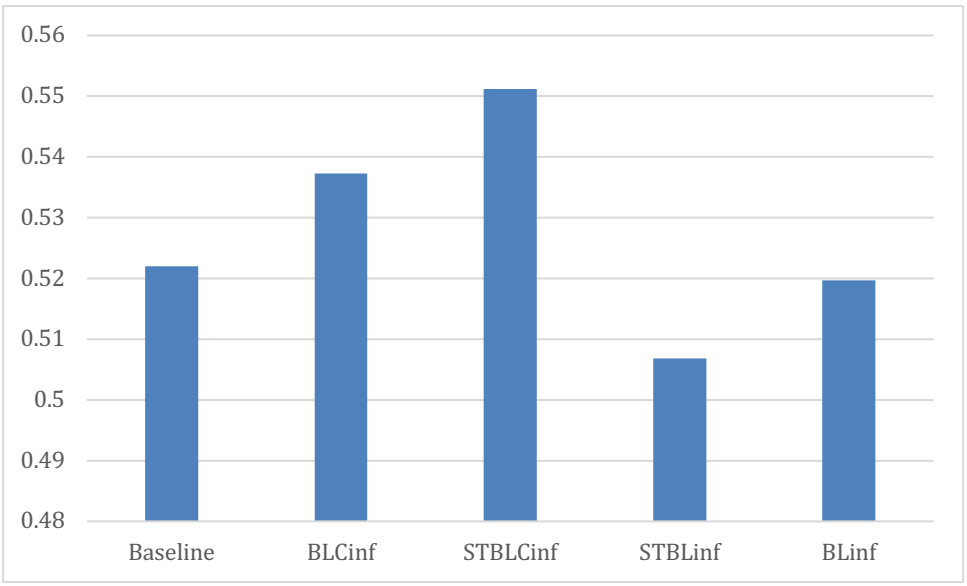

Figure 21. Mean average precision (MAP) of the tested profiles against the baseline.

Table 11 shows the average number of tweets in each profile across all users. The baseline profile has the lowest average number of tweets. The STBLCinf profile, which is the profile that has the highest performance, has only 3577 tweets in average and is the second lowest profile in terms of the average number of tweets. The difference between it and the baseline profile is 1114 tweets. In other words, the percentage difference in number of tweets is about 45%. Surprisingly, the number of average tweets in the STBLinf profile, which showed the worst performance and is the profile most similar to STBLCinf, had more than the best performance profile by 784 and a percentage of 22%. As a result, this small number of tweets affected the performance of the profile.

As explained earlier, the BLCinf and BLinf profiles were built similarly and the difference consists of the former classifying less influential friends' tweets. The difference between them in terms of the average number of tweets is 13.6%, and this extra number of tweets affected the performance of the BLinf profile and led it to perform worse than the baseline. These profiles are the extended versions of the baseline. The BLCinf has 133.6% more tweets than the baseline and this amount resulted in the profile performing better than the baseline, whereas the BLinf profile has 165.4% more tweets than the baseline and this led it to perform worse than the baseline.

Table 10. Average number of tweets in each profile.

Profiles	Baseline	BLCinf	STBLCinf	STBLinf	BLinf
Average number of tweets	2463	5754	3577	4361	6538

Having determined which profile performed the best, the STBLCinf profile was compared against other profiles that were built based on different strategies from the literature, in order to see how good the proposed profile is in terms of modelling the user and also in improving recommendations. The profiles from previous research are: follower count, follower rank, signal strength and retweet impact. All these profiles were built in the same way as STBLCinf. In contrast, and for each literature profile, clustering was applied based on its measurement (algorithm). For example, the users in the follower count profile were clustered into three groups based on the number of followers of each user. As can be seen in Figure 22, the results show that our proposed profile, STBLCinf, achieved the best performance against all other profiles in the metric of average precision at top-k when the k was set from 1 to 10. The follower rank strategy achieved the worst performance in comparison to all other strategies. At AP@4, the proposed profile was better by 14% than the closest profiles, which are follower count and retweet impact. It was also better by 12% than the closest profile at AP@5 and AP@6.

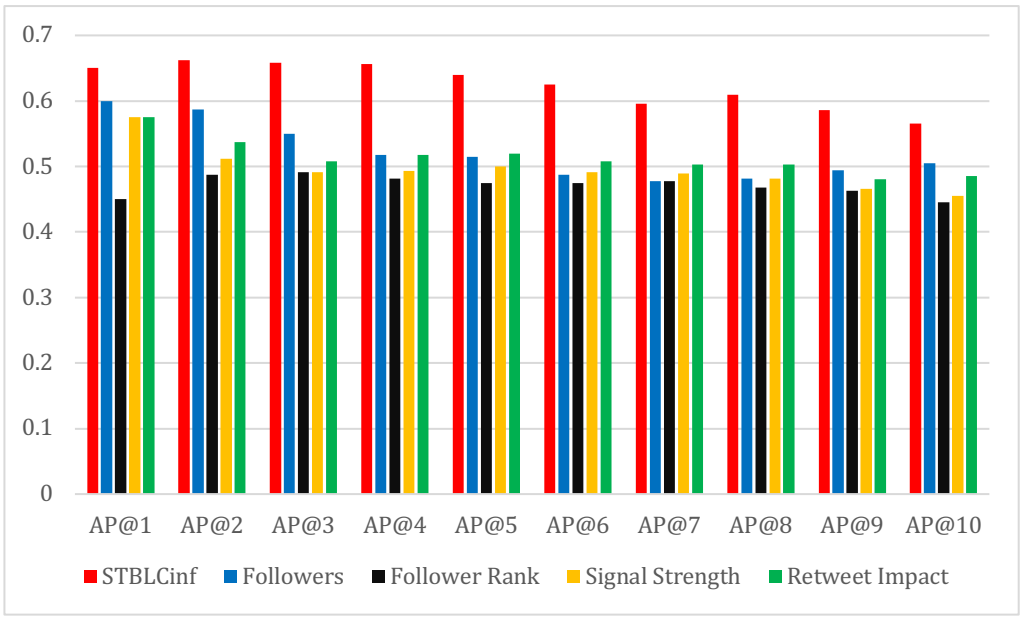

Figure 22. AP@top-10 recommendations of the proposed profile in comparison with the profiles from the literature.

In the metric of mean average precision (MAP), as Figure 23 shows, the STBLCinf profile outperformed the other strategies by 6%. It achieved the value of 0.55, whereas the closest strategy was the follower count profile with 0.49.

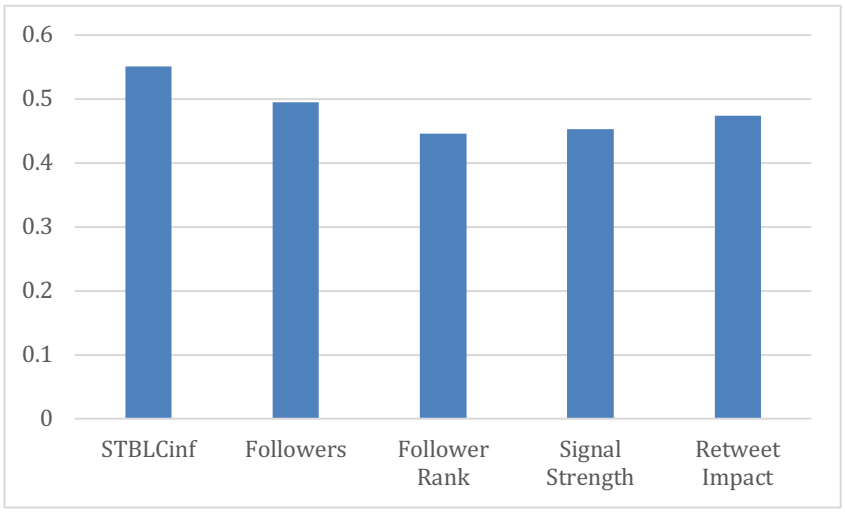

Figure 23. MAP of the proposed profile in comparison with the profiles from the literature.

After measuring the performance of the proposed profile and comparing it against other profiles from the literature, there was a need to see the average number of tweets in each profile in order to have a clear view inside each profile. Table 12 shows the average number of tweets in each profile across all users. The follower count profile has the lowest number of tweets, whereas the follower rank profile has the highest number of tweets. In regards to the two highest performing profiles, namely STBLCinf and follower count, the former has 2335 more tweets than the latter, a percentage of 188%. Both of them have the lowest number of tweets by a big margin in comparison to all other profiles. In comparison between the highest and the lowest profiles in terms of performance, the difference is 16,324 tweets and a percentage of 456%, which is a very large amount. This large number of tweets affected the profile's performance and led to it being in last place.

Table 11. Average number of tweets of the proposed profile and the literature profiles.

Profiles	STBLCinf	Follower count	Follower rank	Signal strength	Retweet impact
Average number of tweets	3577	1242	19901	18543	8672

The proposed profile performance was also compared against other similarity metrics: cosine and Jaccard. It was also compared to two distance metrics: Euclidean and Manhattan. Those profiles were built according to the same strategy as the proposed profile. For example, clustering was applied to friends based on the cosine similarity score between them and the examined user, in order to divide them into three groups. The results were published in the work of Alshammari et al. (2019). Figure 24 shows the results of the AP@top-k metric when the proposed profile was compared against the similarity and distance strategy profiles. The proposed profile achieved the highest AP@k value four times when the k was set to 5, 6, 8 and 9. Therefore, its performance was the best in 40% of the tested k values. Euclidean distance came in second place by having the highest AP@k three times, which means that it was the

highest in 30% of the tested k values. Both the proposed profile and the Euclidean profile achieved the highest AP@k two times when the k is set to 2 and 10. Manhattan distance achieved the lowest average precision in all k values.

Figure 24. AP@top-10 recommendations of the proposed profile in comparison with the similarity and distance profiles.

When the profiles were compared by using the MAP metric, Figure 25 shows that the proposed profile outperformed all other similarity and distance strategies. Euclidean distance achieved the second-best performance, whereas Manhattan achieved the worst MAP value.

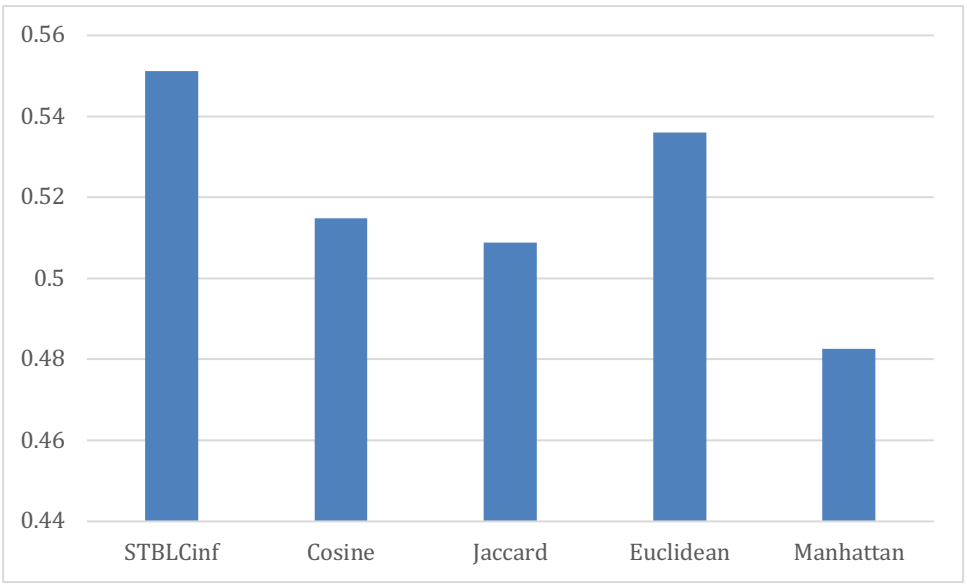

Figure 25. MAP of the proposed profile in comparison with the similarity and distance profiles.

After measuring the performance of the proposed profile against the similarity and distance metrics profiles, it was necessary to clarify the number of tweets in each profile. Table 13 shows that the proposed profile has the lowest number of tweets and is the best performing profile, whereas the Manhattan profile has the highest number of tweets and is the worst performing profile. The closest profile to the proposed one is the Euclidean, which has 3878 more tweets with a percentage of 108%, and also showed the second-best performance in the AP@top-k and MAP metrics. It is clear that enriching the profile with more tweets can affect its performance, especially when the tweets have no relevance to the user.

Table 12. Average number of tweets of the proposed profile and other similarity and distance profiles.

Profiles	STBLCinf	Cosine	Jaccard	Euclidean	Manhattan
Average number of tweets	3577	11821	11143	7455	14301

As the previous figures and results show, the proposed method of building the user profile based on the influence score between the user and their friends performs better than the baseline, metrics from the literature, and the similarity and distance metrics.

4.3.1.2. Dividing the dataset into categories

In this section, the dataset was divided into different categories with the aim of providing a deep analysis of the performance of the proposed method. The categories are: activity (engagement), number of friends and originality (receiver or producer). The proposed method was compared against the literature profiles that were mentioned previously. Its performance was also analysed in all different categories in order to understand the outcomes of the experiment.

4.3.1.2.1. Activity (engagement)

Based on equation 6, the users were divided in this stage into two categories, less active and active users. Firstly, the users were placed in ascending order based on their engagement score. Secondly, the users were divided into the two mentioned groups. Then the proposed profile's performance was compared against the profiles from the literature in both groups.

In the less active group of users and by using the metric of AP@top-k, as Figure 26 shows, the proposed profile (STBLCinf) outperformed all other profiles (baseline, follower count, follower rank, signal strength and retweet impact), in all top-k recommendations with a percentage of 100%. Its best performance was when the k was set to 2, as it achieved an AP 15% higher than the follower count profile, which came second. It performed better than the baseline profile by 20%. As in the previous comparisons, the follower rank profile had the worst performance. The signal strength profile achieved the lowest AP@3.

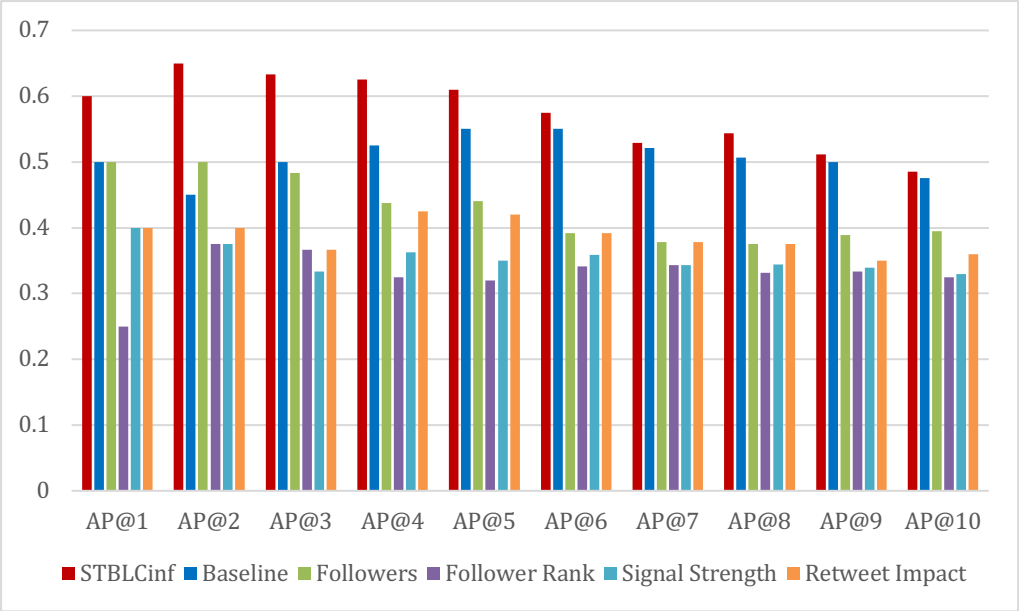

Figure 26. AP@top-10 recommendations of less active users. The proposed profile is compared to the baseline and profiles from the literature.

The previous figure shows the performance of all profiles when the AP of the top 10 recommendations was calculated. In order to see the overall performance, Figure 27 shows the performance of all profiles when the MAP metric was used. As can be seen, the proposed profile outperformed all other profiles. The baseline came second, less than the proposed profile by 7%. And as proved before, the follower rank profile achieved the lowest MAP, less than STBLCinf by 17%.

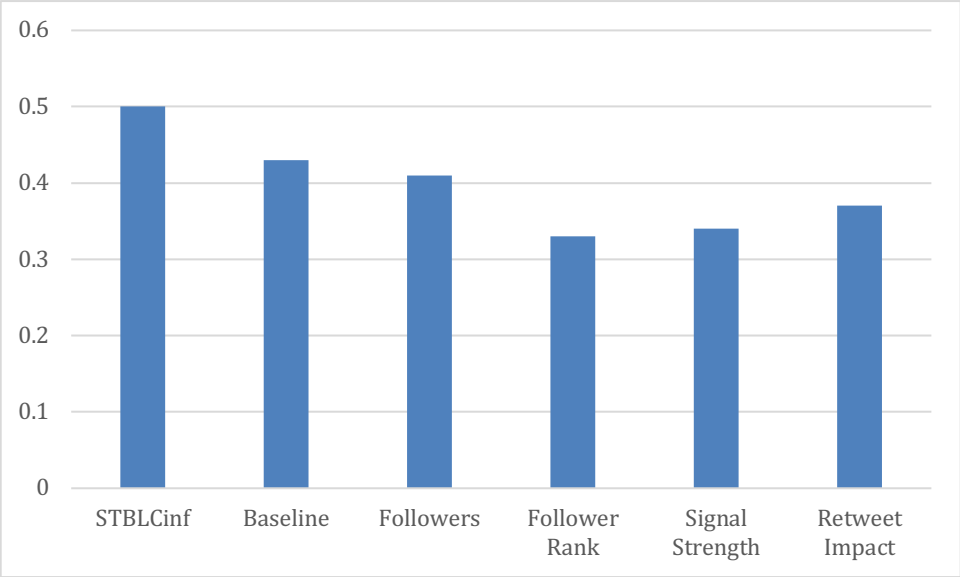

Figure 27. MAP of less active users. The proposed profile is compared to the baseline and profiles from the literature.

In order to see the proposed profile's performance against the similarity and distance metrics, the comparison results are shown in Figure 28, which is based on the AP of the top 10 recommendations. The proposed profile outperformed the other strategies by 60%; the second profile which exhibited a good performance was Euclidean distance, which achieved 10% of the highest values when the top k was set at 7. Both of these profiles achieved the best performance by 30%, as both have the same AP for the top 10 recommendations. The proposed profile's performance is thus suitable for less active users when the top 10 recommendations are delivered.

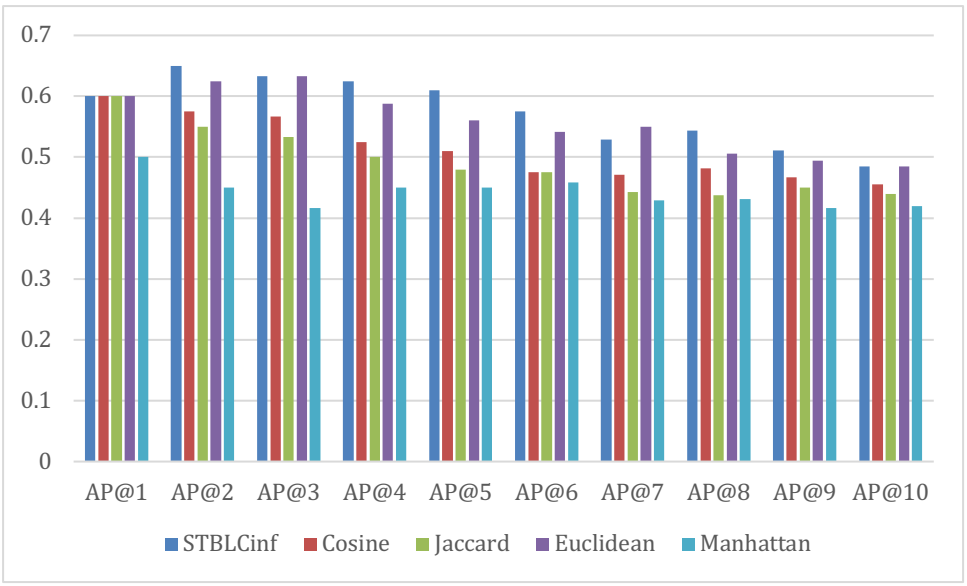

Figure 28. AP@top-10 recommendations of less active users. The proposed profile is compared to the similarity and distance profiles.

To see the overall picture of the performance of the proposed profile in comparison to the similarity and distance metrics, the MAP metric was used for this purpose. Figure 29 shows the results of the comparison; the STBLCinf profile outperformed all other metrics.

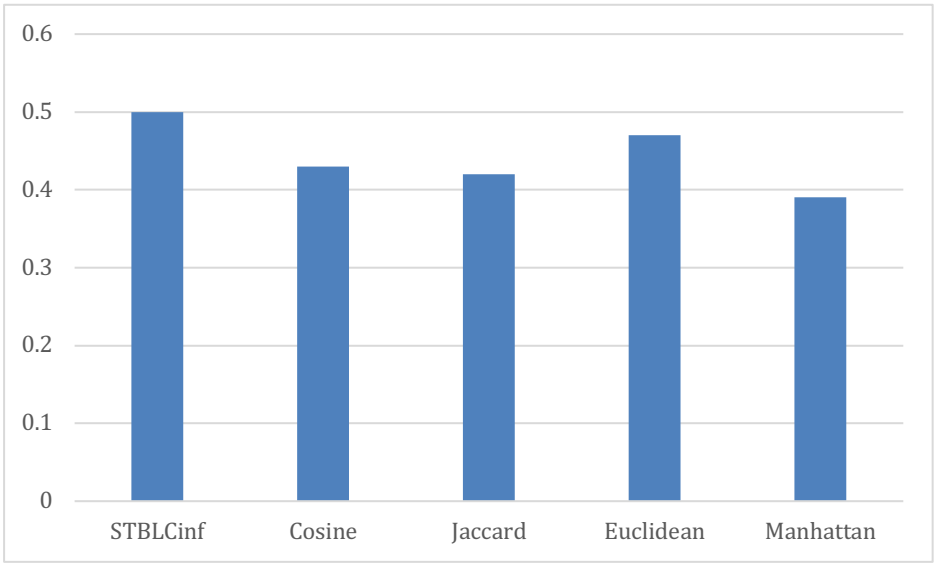

Figure 29. MAP of less active users. The proposed profile is compared to the similarity and distance profiles.

As seen above, our proposed profile proved suitable for less active users. In order to see the overall performance of the proposed profile, the same comparison steps were applied for active users. Figure 30 shows the results of the AP@top-k recommendations when active users were tested, and shows that the proposed profile was better in 60% of the cases and outperformed the other profiles six times when the k was set between 4 and 9. The baseline came second by outperforming others by 20% of the cases. The follower count profile achieved the worst performance among the other profiles.

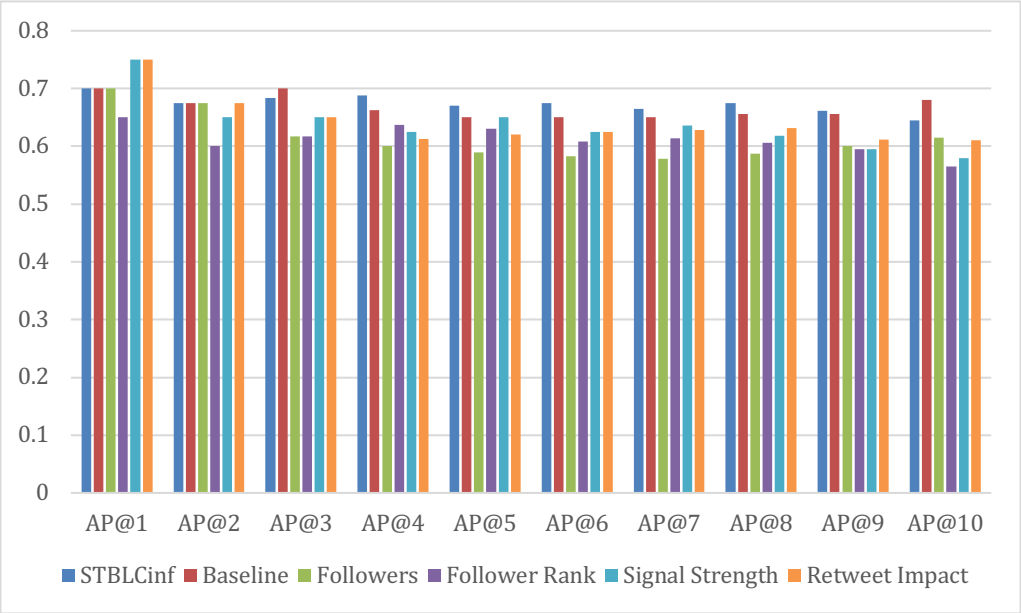

Figure 30. AP@top-10 recommendations of active users. The proposed profile is compared to the baseline and profiles from the literature.

The above figure shows the performance of all profiles when the AP was computed on only the top 10 recommendations. The MAP metric was used with the aim of seeing the overall performance of all profiles. Figure 31 shows that the STBLCinf and baseline profile achieved the same MAP value. The follower rank profile came in last place.

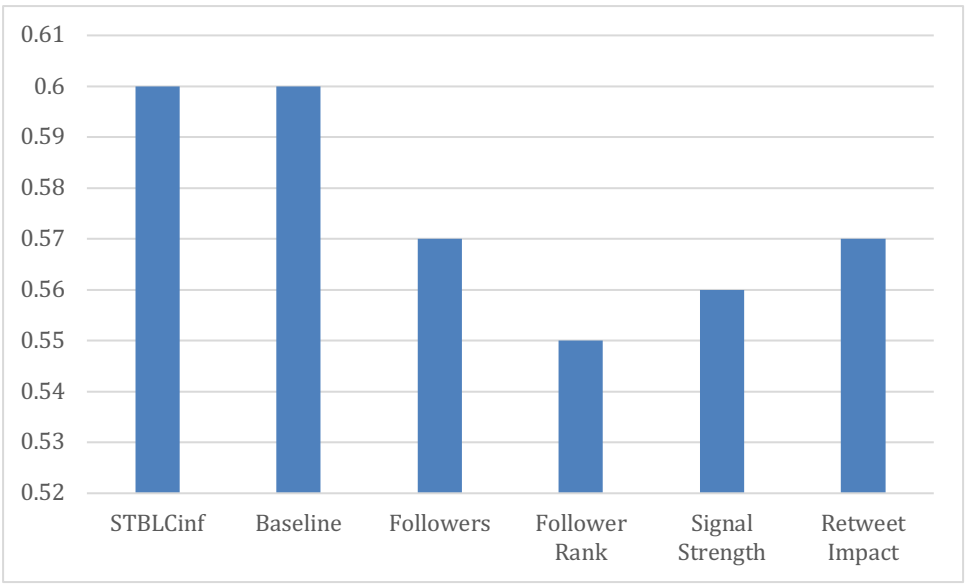

Figure 31. MAP of active users. The proposed profile is compared to the baseline and profiles from the literature.

Previous results and figures show that our hypothesis that the proposed profile works better for users who are less active when compared to other profiles is correct. As in the results for less active users, the performance of this profile achieved the highest AP of all top 10 recommendations and also achieved the highest MAP. In comparison to the results for active users and in terms of the top 10 recommendations, the proposed method works better with less active users by 40%.

The next step was to compare the performance of the proposed profile in both categories (active and less active). Figure 32 shows the AP@top-k recommendations of the STBLCinf profile in both categories. The profile in the active user group achieved better AP values in the top 10 recommendations.

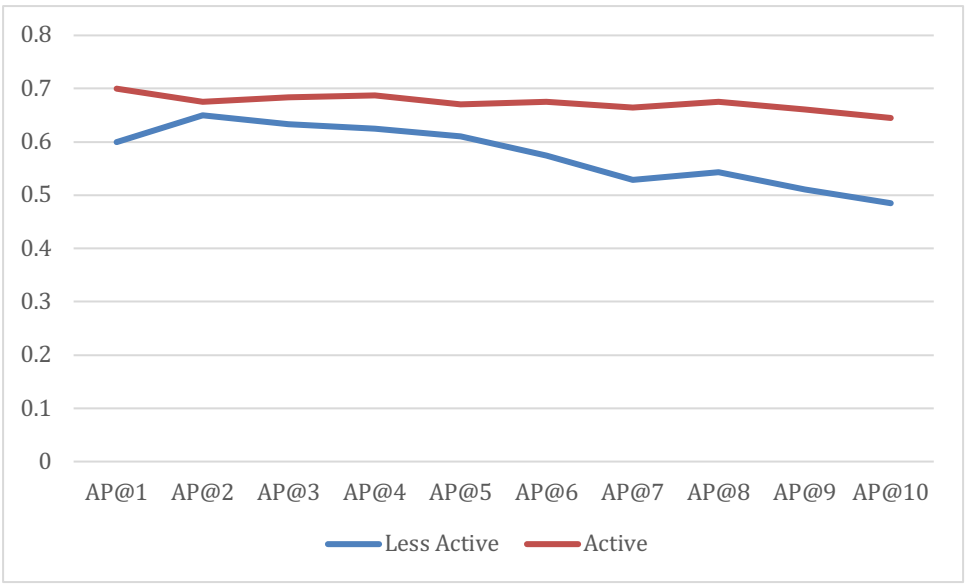

Figure 32. AP@top-10 recommendations of the proposed profile based on activity.

Figure 33 shows that the proposed profile also achieved a better MAP value with the active users group, more than the less active group by 10%. However, when the MAP is high with active users, the other profiles from the literature were higher as well, especially the baseline profile, as can be seen in the figure.

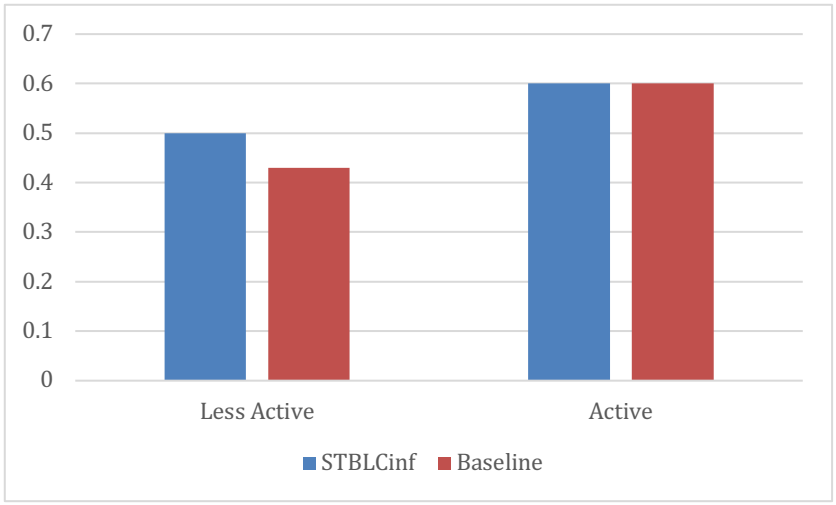

Figure 33. MAP@top-10 recommendations of the proposed profile and baseline profile based on activity.

4.3.1.2.2. Originality

The users were divided in this stage into two categories: receivers and producers. The former group contains the users who retweet more than posting tweets. The producers group contains the users who post tweets more than retweeting. After calculating the originality score of each user, a user with a score of less than 1 is considered a receiver, while user with a score of more than 1 is included in the producers group. Then the proposed profile's performance was compared against the profiles form the literature in both groups.

In the receivers group and by using the AP@top-k metric, as Figure 34 shows, the proposed profile (STBLCinf) outperformed the baseline six times out of 10, whereas the baseline was successful four times. In the comparison to all other profiles (baseline, follower count, follower rank, signal strength and retweet impact), the STBLCinf and baseline profiles shared the best performance, with six and four times, respectively. Surprisingly, the baseline achieved the lowest value in AP@2. Other profiles did not show any better performance.

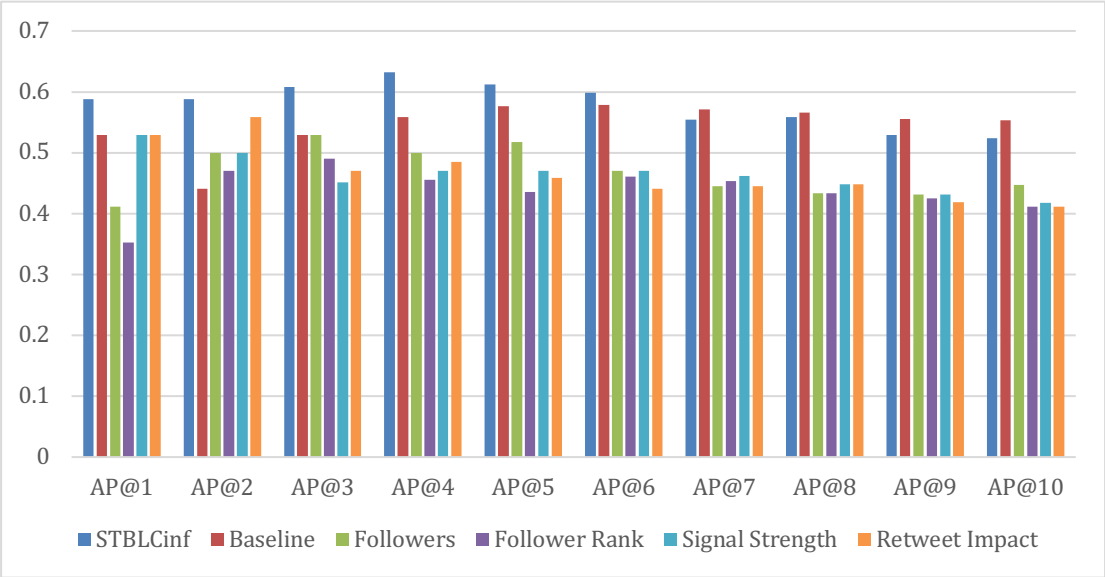

Figure 34. AP@top-10 recommendations of receivers. The proposed profile is was compared to the baseline and profiles from the literature.

To see the overall performance of all profiles including the proposed one, Figure 35 shows the MAP values of all profiles. The proposed profile and the baseline profiles continued their

outstanding performance, as the former achieved the highest MAP value and the latter came in second place. The follower count profile came third by achieving MAP value less than the STBLCinf and baseline by 5% and 4%, respectively.

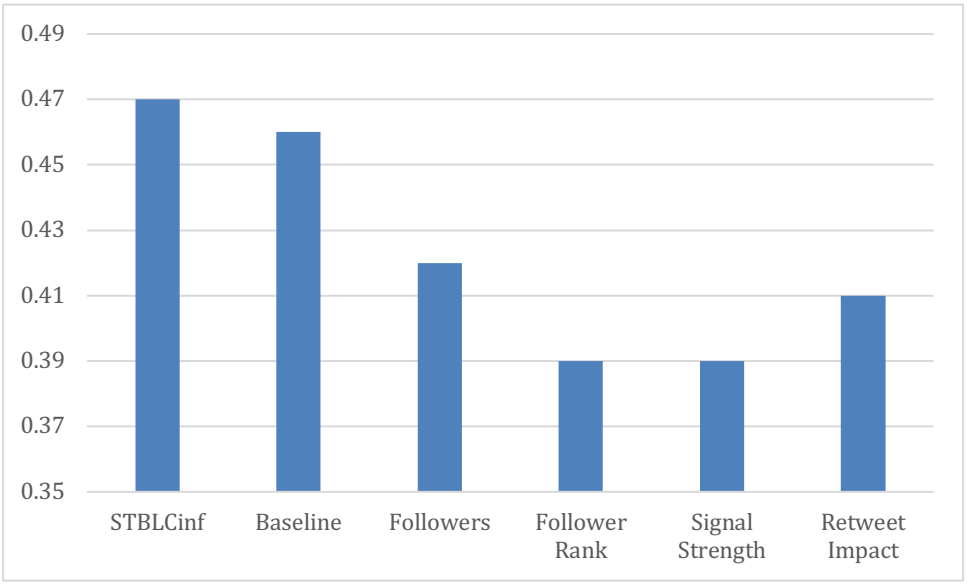

Figure 35. MAP of receivers. The proposed profile is compared to the baseline and profiles from the literature.

Figure 36 clarifies the comparison between the proposed profile and other similarity and distance metrics. As can be seen, the Euclidean profile was successful in 40% of the cases when the top 10 recommendations were delivered, whereas the proposed profile was successful in 30%. It also had the highest value alongside other profiles two times when the k was set to 2 and 8. It outperformed the baseline profile in 50% of cases and both had the same value four times. The baseline outperformed the proposed profile only once, when the k was set to 7.

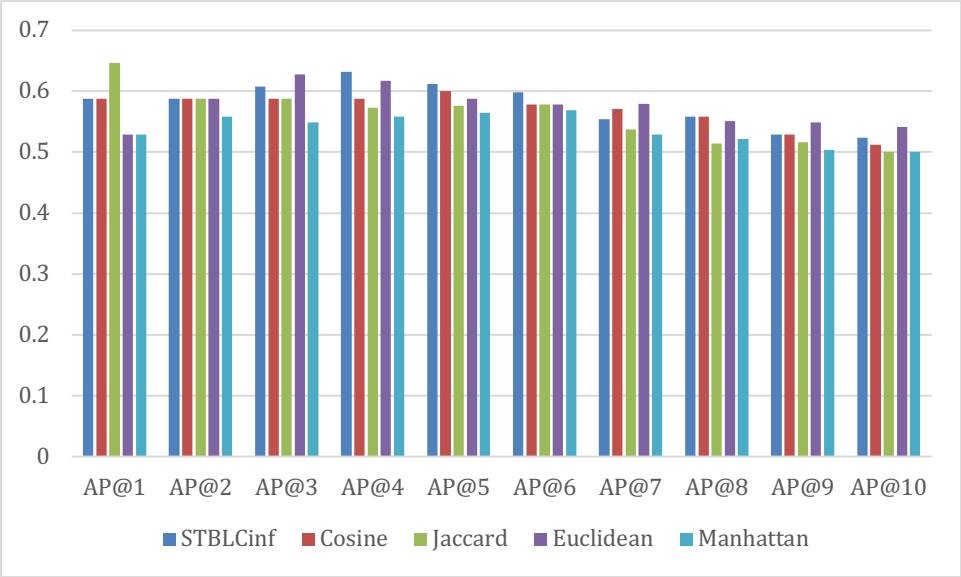

Figure 36. AP@top-10 recommendations of receivers. The proposed profile is compared to the similarity and distance profiles.

To see the overall performance of the proposed profile and other similarity and distance metrics, Figure 37 shows that the STBLCinf profile has the highest mean average MAP value against all other profiles. The cosine, Jaccard and Euclidean profiles achieved the same mean average value. The Manhattan profile showed the worst performance across all profiles.

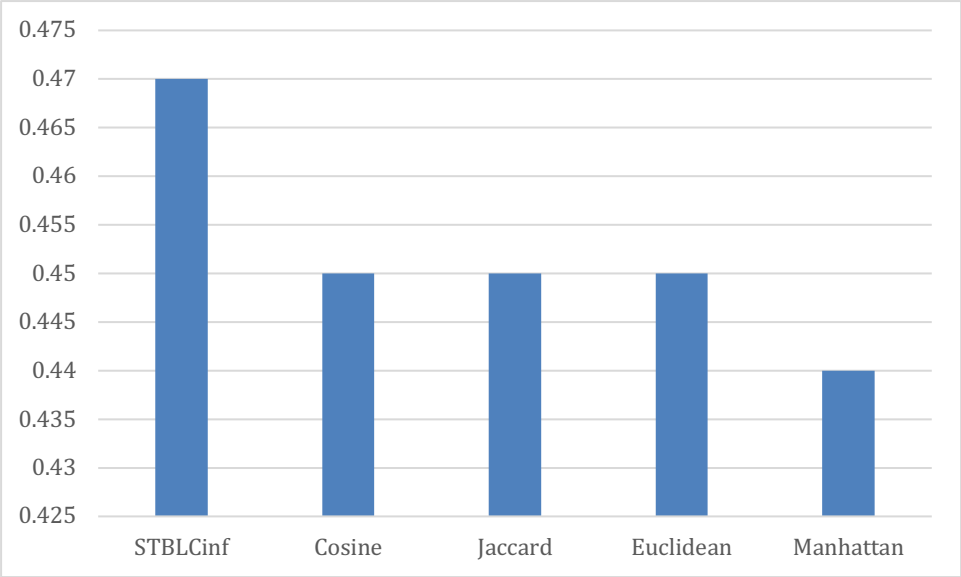

Figure 37. MAP of receivers. The proposed profile is compared to the similarity and distance profiles.

As the figures and results shown above demonstrate in regards to analysis of the receivers group, the proposed profile was as successful as the follower count profile, as both achieved the same performance when the MAP value was calculated based on the successful recommendations and their order to the users. Moreover, the proposed profile was successful against all similarity and distance profiles. Therefore, its performance is quite suitable for the receivers group. The next step was to see its performance with the producers group.

Figure 38 clarifies the comparison between the proposed profile, the baseline and the profiles from the literature. As can be seen, the proposed profile was successful in 80% of the cases when the top 10 recommendations were delivered. Moreover, it shared the same performance with the baseline and follower count profiles in 20% of cases. The proposed profile was successful against the baseline profile in 90% of cases and both were equal in 10%. Except for the STBLCinf profile, the baseline outperformed all other profiles in 40% of the cases when the top 10 recommendations were retrieved. The follower rank profile achieved the worst performance across all profiles.

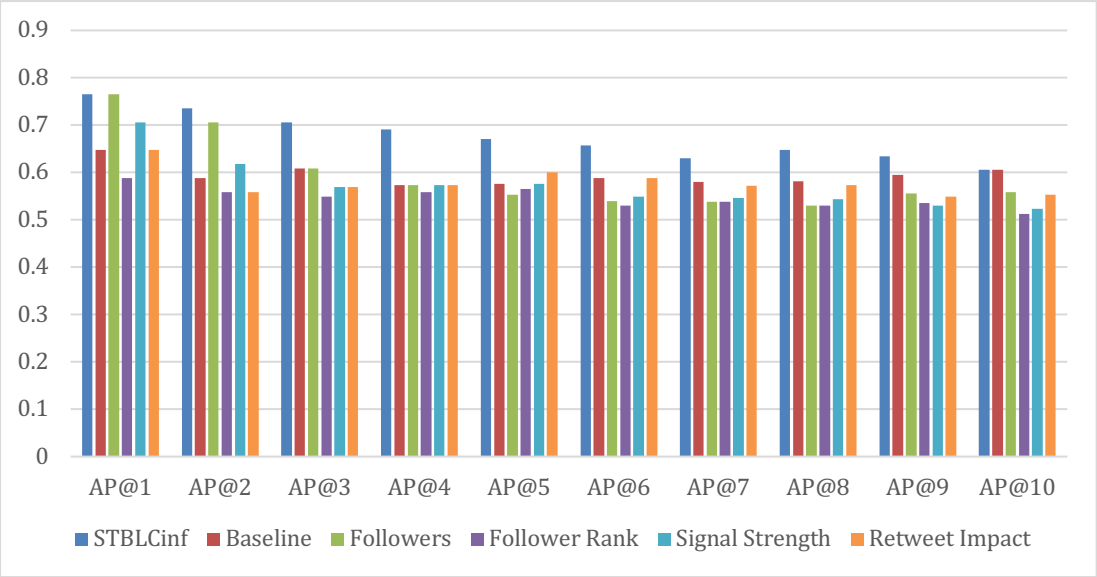

Figure 38. AP@top-10 recommendations of producers. The proposed profile is compared to the baseline and profiles from the literature.

Figure 39 shows the results of the MAP of all profiles. The proposed profile achieved the highest MAP value in comparison to the baseline and profiles from the literature. Surprisingly, the follower count profile had a slightly lower value than the baseline. Furthermore, both profiles were behind the proposed one by 4.7%. The follower rank profile again showed the worst performance across all other profiles.

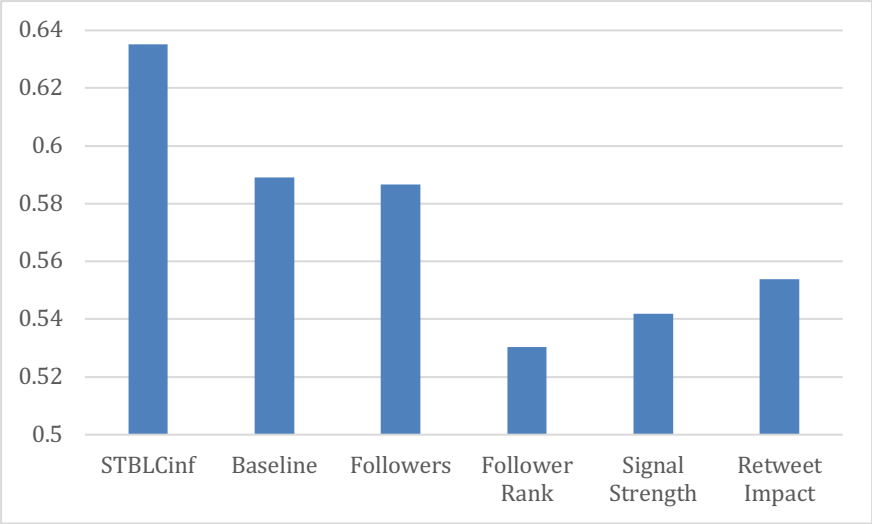

Figure 39. MAP of producers. The proposed profile is compared to the baseline and the profiles from literature.

In the next step, the similarity and distance metrics profiles were compared with the proposed profile. As can be seen in Figure 40, the proposed profile had the highest AP value when the top 10 recommendations were recommended in 60% of the recommendations. The competitive Euclidean profile had the highest value in 10% of the recommendations. It was equal to the proposed method in 20% of the cases.

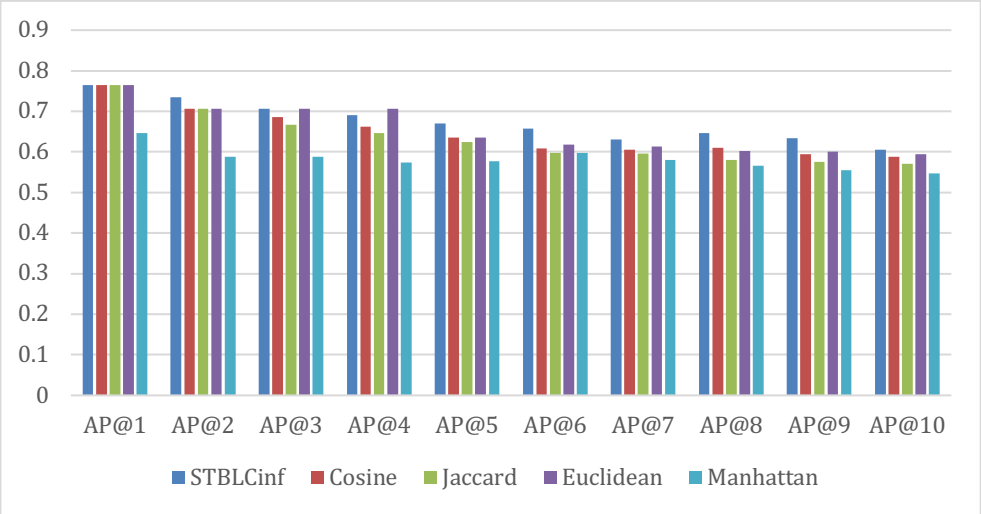

Figure 40. AP@top-10 recommendations of producers. The proposed profile is compared to the similarity and distance profiles.

In regards to the MAP metric, the proposed model had the highest value across other profiles, whereas the Euclidean profile had the second-best performance (see Figure 41). The Manhattan profile had the lowest MAP value in the producers groups, as in the receivers group.

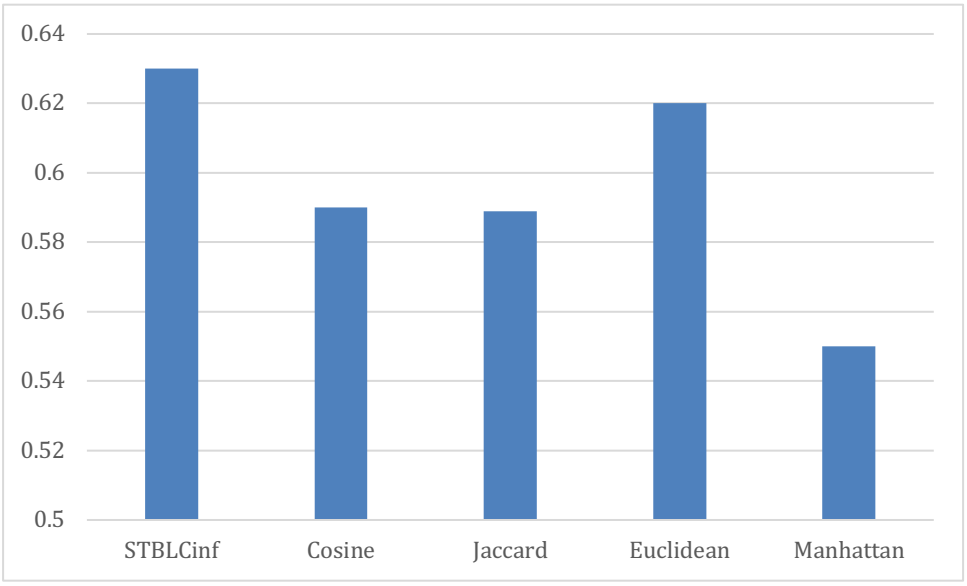

Figure 41. MAP of producers. The proposed profile is compared to the similarity and distance profiles.

In the previous results and figures, the proposed profile outperformed the baseline profile, similarity metrics and all other profiles, which are follower rank, signal strength and retweet impact, except the follower count profile, which was as good as the proposed profile especially in the MAP metric. The improvement in the follower count profile might be related to the distribution of active and less active users inside each of the receivers and producers groups. Figure 42 shows that the receivers group had 71% of less active users, with 29% of users being active. The producers groups is 76% active users and 24% less active users.

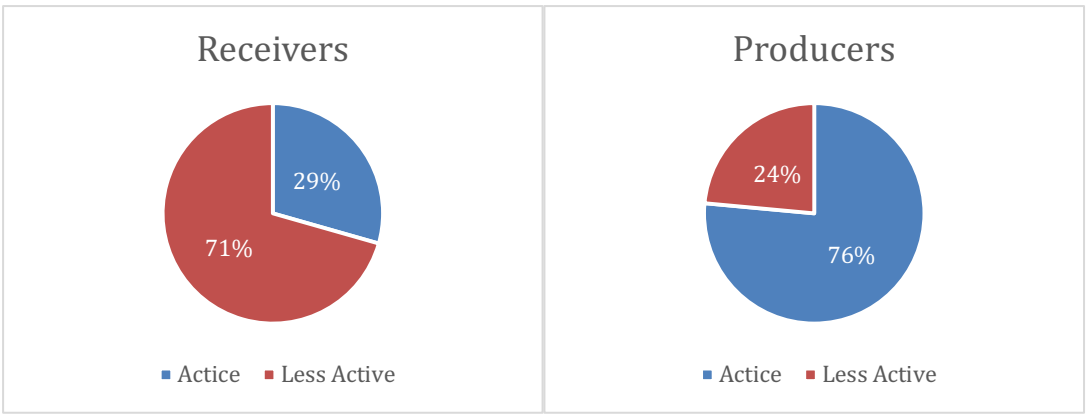

Figure 42. Activity distribution in the groups of receivers and producers.

Our hypothesis about the outstanding performance of the follower count profile has a connection with activity distribution, as seen above. Furthermore, it also affects the performance of the proposed profile. As a result, the receivers groups was cleaned from the noisy users, which are the active ones. In other words, the groups now only contained less active users. Less active users were removed from the producers group, in order to have the group contain only active users. Then the recommender system was tested with the newly shaped groups. The evaluation results of this step were measured by MAP. Figure 43 shows the results of the receivers groups with only less active users. As is clear, the proposed method outperformed all other profiles. The baseline profile achieved a higher MAP value than the follower count profile.

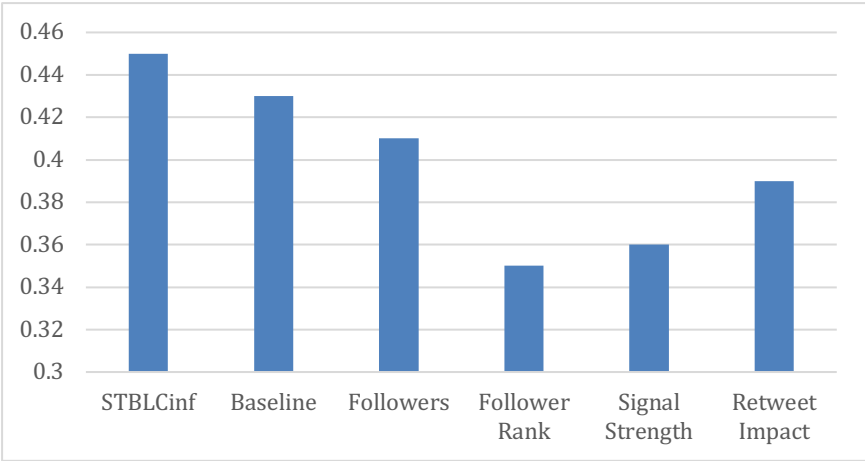

Figure 43. MAP of the receivers group with only less active users. The proposed profile is compared to the baseline and profiles from the literature.

The results of testing the producers group with only active users are shown in Figure 44. As expected, and as before in the section on activity, the proposed profile achieved the best performance alongside the baseline. As a result, activity is key to performance, and even more so than originality. However, the receivers group tends to contain more less active users, whereas the producers group tends to contain more active users.

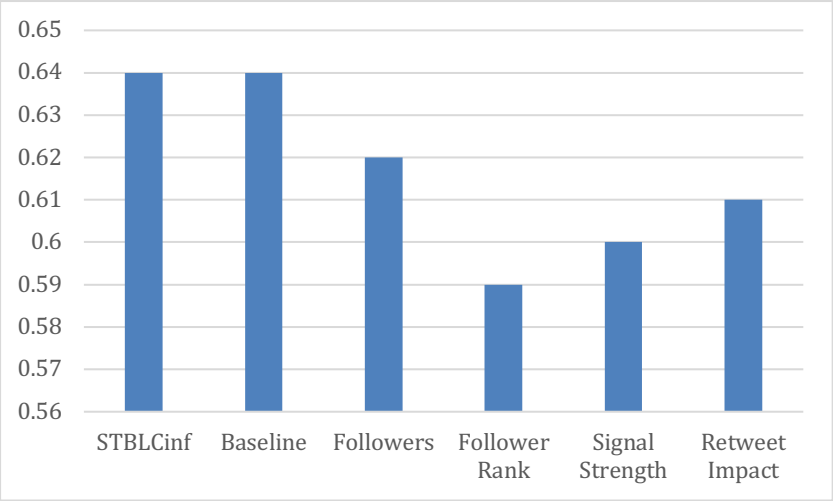

Figure 44. MAP of the producers group with only active users. The proposed profile is compared to the baseline and profiles from the literature.

The next step was to see the difference in performance of the proposed profile in both groups: receivers and producers. As can be seen in Figure 45, the proposed profile performed better with producers when the system recommends the top 10 recommendations.

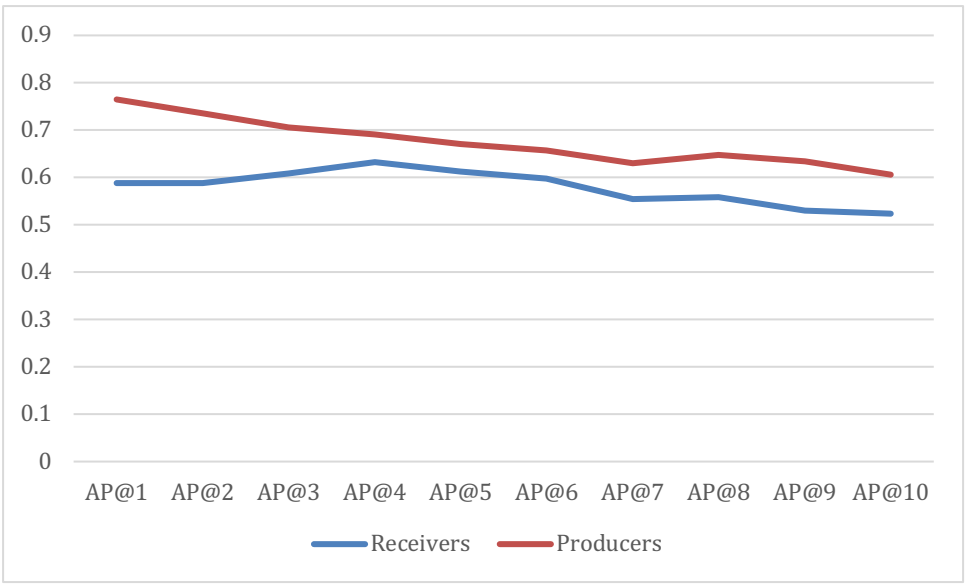

Figure 45. AP@top-10 recommendations of the proposed profile in the groups of receivers and producers.

In regards to MAP, the proposed profile had the highest value, and as a result, it works better with the group of producers, more than the receivers by 16%.

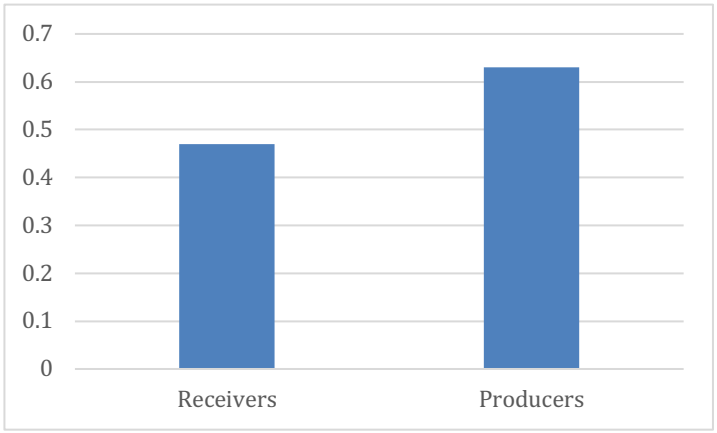

Figure 46. MAP of the proposed profile in the groups of receivers and producers.

4.3.1.2.3. Number of friends

At this stage, and in order to see how the number of the user's friends is correlated to the performance of the proposed profile, the users were divided into four groups based on the number of their following friends. The users at this stage were ordered in ascending order of number of following friends and then divided into four groups. Then proposed profile's performance was evaluated and compared to other candidate profiles from the literature. The four groups were determined as follows:

- Group 1: users with a number of friends between 0 and 171.
- Group 2: users with a number of friends between 172 and 300.
- Group 3: users with a number of friends between 300 and 440.
- Group 4: users with a number of friends between 441 and 800.

The results of the performance of group 1 are shown in Figure 47. It shows the AP for top 10 recommendations. Surprisingly, the follower count profile achieved the best AP value when the k was set to 1 by 10% better than the second-best performing profile, which was retweet impact. The proposed profile and the follower rank profile had the lowest AP value at 1 (AP@1). The baseline outperformed all other profiles five times, whereas the follower count profile had the best value twice. The proposed profile achieved the best performance only one time, when the k was set to 4, and it was equal highest to the follower count and baseline profiles twice.

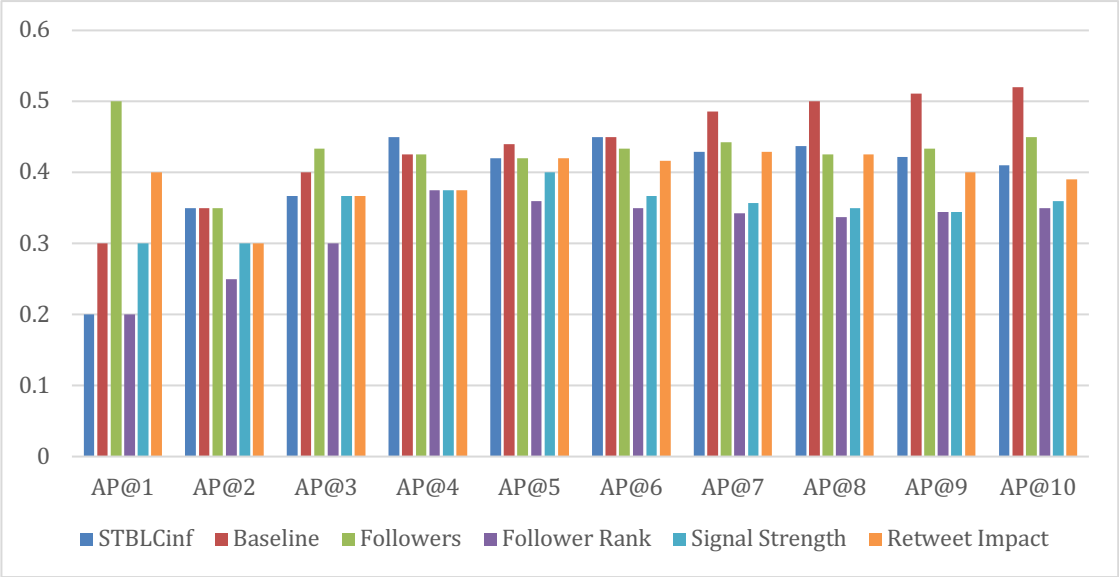

Figure 47. AP@top-10 recommendations of group 1. The proposed profile is compared to the baseline and profiles from the literature.

As seen in the above figure and based on the performance results, the baseline was the best performing profile in the top 10 recommendations, second was the follower count profile, while the proposed profile came in third place. In order to see the overall performance, the MAP metric was used for this purpose. Figure 48 shows that the proposed profile achieved the best performance in delivering all recommended items. The baseline and the follower count profiles achieved the same performance and they both came in second place. The follower rank profile achieved the lowest MAP vale and therefore had the worst performance.

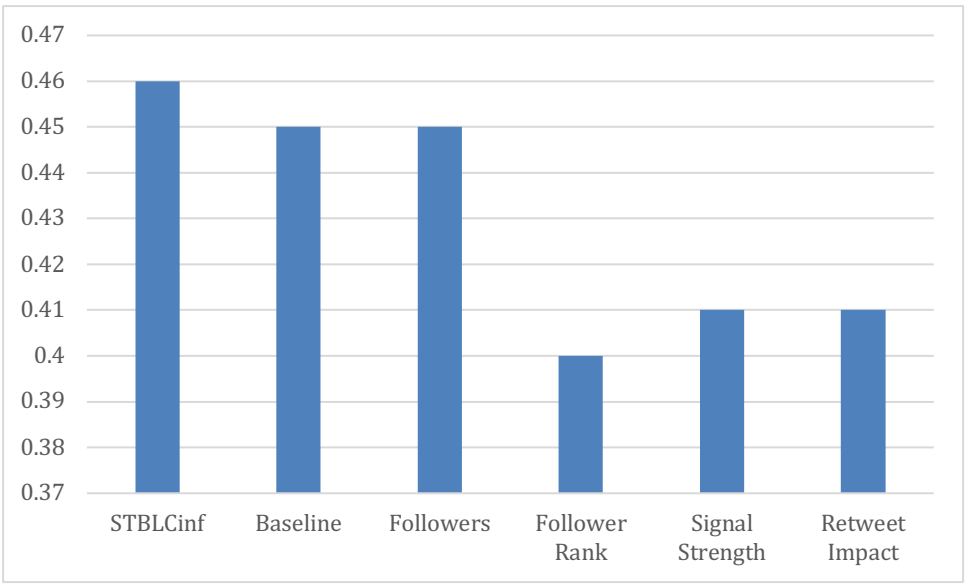

Figure 48. MAP of group 1. The proposed profile is compared to the baseline and profiles from the literature.

When the proposed profile is compared to other profiles built based on the similarity and distance metrics, the results show that the proposed profile outperformed other profiles three times when the k was set to 4, 6 and 7. In second place, the Euclidean profile outperformed other profiles only two times when the k was set to 9 and 10. The cosine and Jaccard profiles achieved the best performance equally only one time, when the k = 1. See Figure 49.

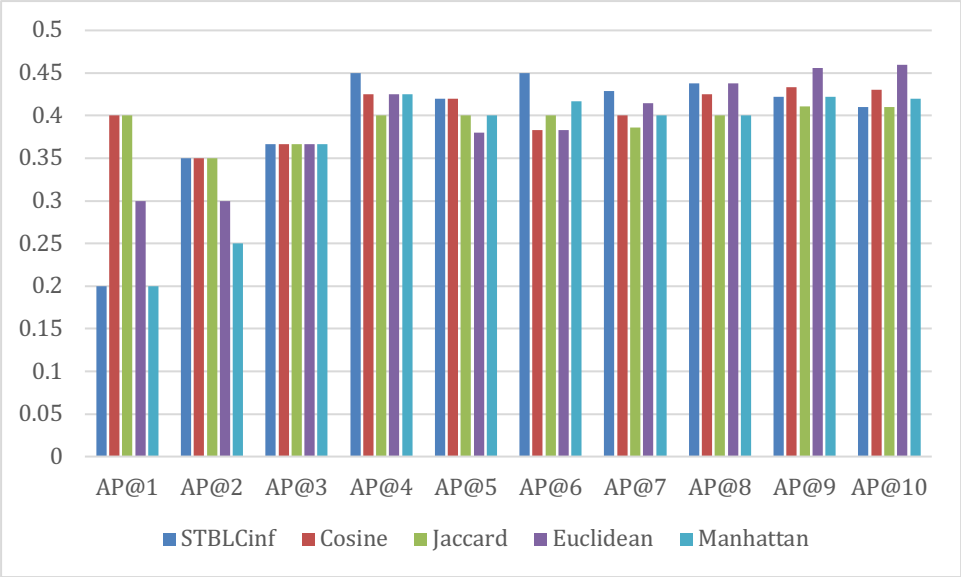

Figure 49. AP@top-10 recommendations of group 1. The proposed profile is compared to the similarity and distance profiles.

In the MAP metric, the proposed profile outperformed all other profiles which are based on similarity and distance metrics. The cosine, Jaccard and Euclidean profiles achieved the same MAP value. The Manhattan profile had the worst performance. See Figure 50.

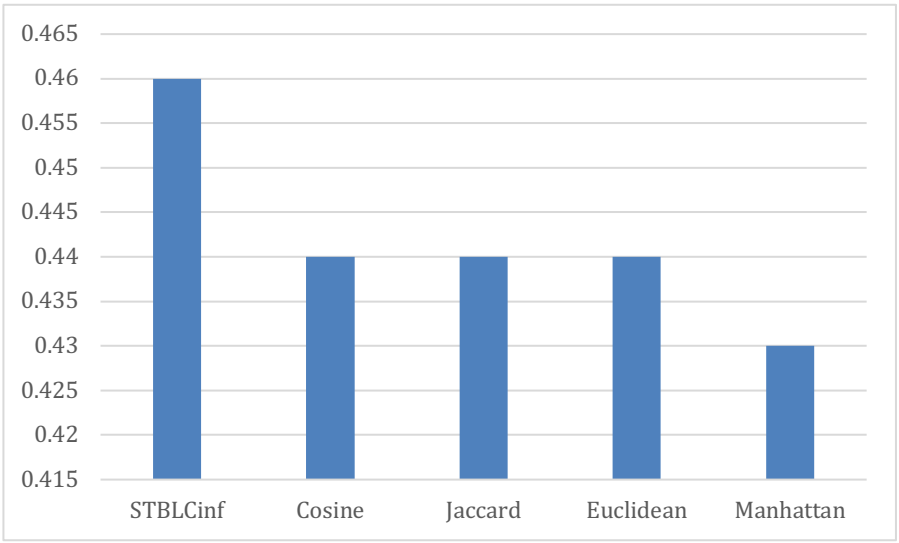

Figure 50. MAP of group 1. The proposed profile is compared to the similarity and distance profiles.

The results clarify that the proposed profile works well with this group in all MAP evaluations, as shown above. It outperformed the baseline, literature profiles and similarity profiles. This means that the proposed profile was able to deliver the related recommendations better than other profiles. However, as seen above, the results of the AP@top-10 show that the proposed method could not deliver more accurate related recommendations when compared to the baseline and literature profiles. The performance of the proposed profile in AP@top-10 recommendations might be affected by active users. Figure 51 shows the distribution of the activity and originality of the group members. As can be seen, the group contains 60% users who are active and 40% less active users. Therefore, and as our hypothesis mentioned previously, the majority of the users are active and this might affect the performance of the proposed profile. A majority of the users are producers, representing 67% of the group, whereas the receivers represent only 33%.

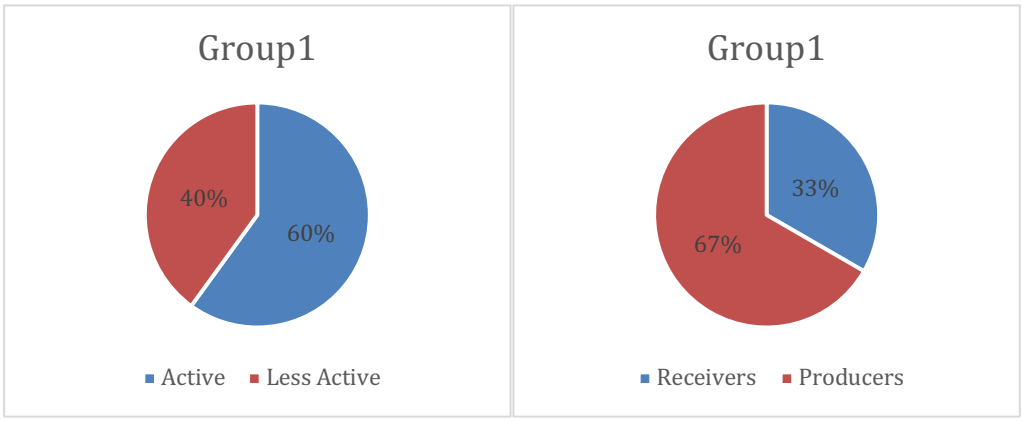

Figure 51. Activity and originality distributions in group 1.

Figure 52 shows the results of the performance of candidate profiles when the AP@top-k metric was used on group 2. The proposed profile outperformed all other profiles by 70% of the top 10 recommendations. It achieved the best performance equally with follower count profile in 20%. The follower count profile achieved the best performance one time when the k was set to 2. Generally, the performance of the proposed profile was very good when the top 10 recommendations were delivered.

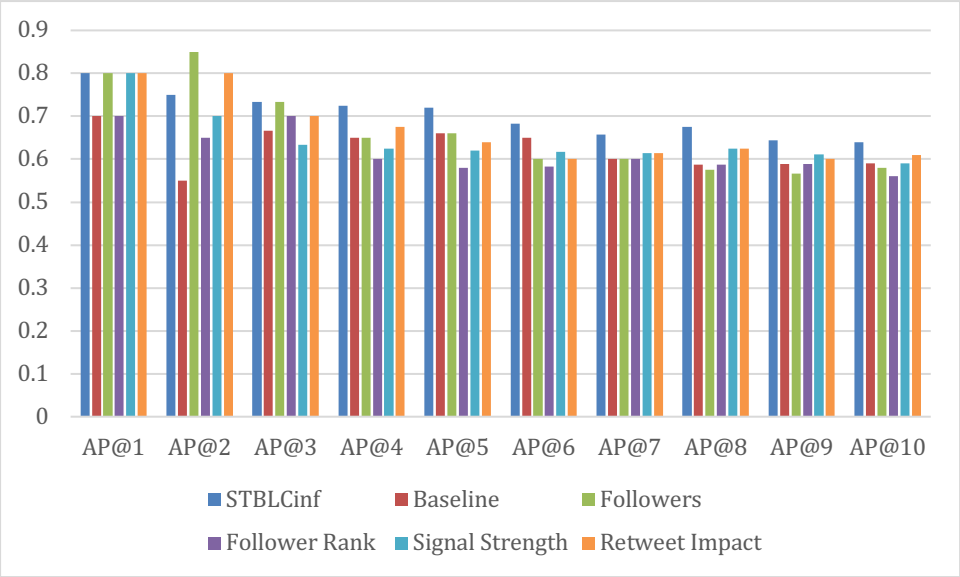

Figure 52. AP@top-10 recommendations of group 2. The proposed profile is compared to the baseline and profiles from the literature.

After determining that the proposed profile had the best performance in the metric of AP@top-10 recommendations, it was necessary to see the overall performance when all related recommendation items are retrieved; this was measured by using the MAP metric. As can be seen in Figure 53, the proposed profile and the baseline profile achieved the highest MAP value, while the follower rank and the signal strength profiles achieved the lowest value. The follower count profile, which performed slightly better against the baseline when the top 10 recommendations were delivered, achieved a lower value than the proposed profile and the baseline.

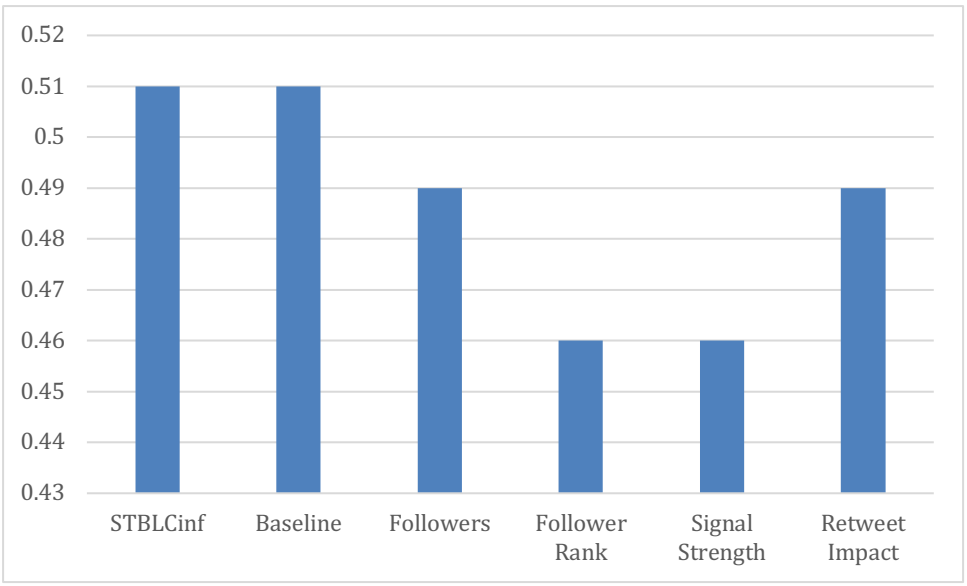

Figure 53. MAP of group 2. The proposed profile is compared to the baseline and profiles from the literature.

The next step was to compare the proposed profile performance against other similarity and distance profiles in this group. Figure 54 shows the results: both the STBLCinf and the Euclidean distance profiles achieved the best performance equally. Surprisingly, the other profiles, which are cosine similarity, Jaccard similarity and Manhattan distance, achieved the same MAP value.

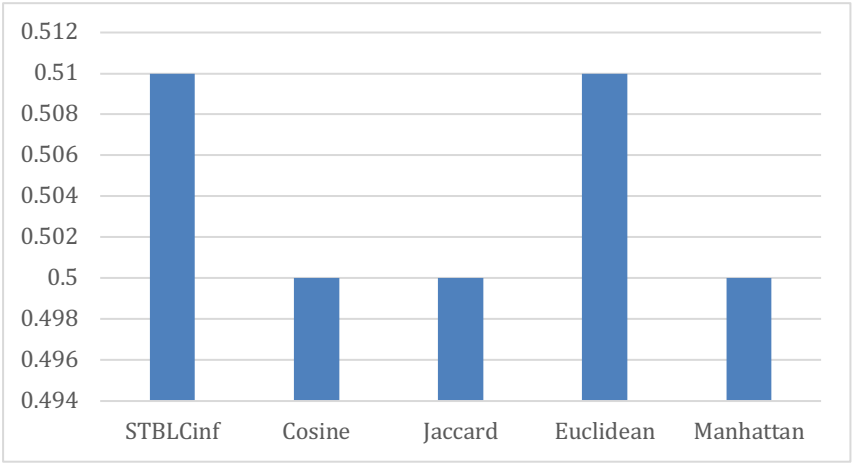

Figure 54. MAP of group 2. The proposed profile is compared to the similarity and distance profiles.

Figure 55 shows the distribution of the active and less active users inside group 2. It also shows the percentage of producers and receivers in the group. In the activity contribution, active users share the group equally with less active users, as both of them represent 50%. On the other side, the majority of the users inside the group are receivers by 67% and the producers represent only 33%. This contribution might be the reason why the performance of both Euclidean distance and baseline profiles achieved the same MAP value alongside STBLCinf.

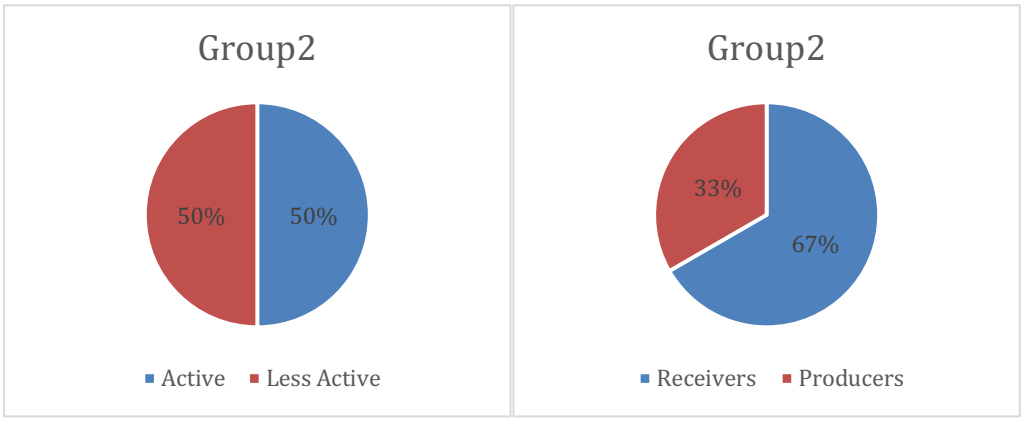

Figure 55. Activity and originality distributions in group 2.

The results of the performance of the proposed profile against the baseline and other literature profiles in group 3 when the AP@top-k metric is used are shown in Figure 56. The proposed profile achieved the highest performance nine times out of ten when the top 10 recommendations are retrieved. When the k value was set from 1 to 4, the proposed profile's performance was higher by 10% than the baseline and follower count profiles. The baseline and the proposed profile both achieved the highest and the same value when the k value was set to 10. Other profiles did not show any better performance at any point in the top 10 recommendations.

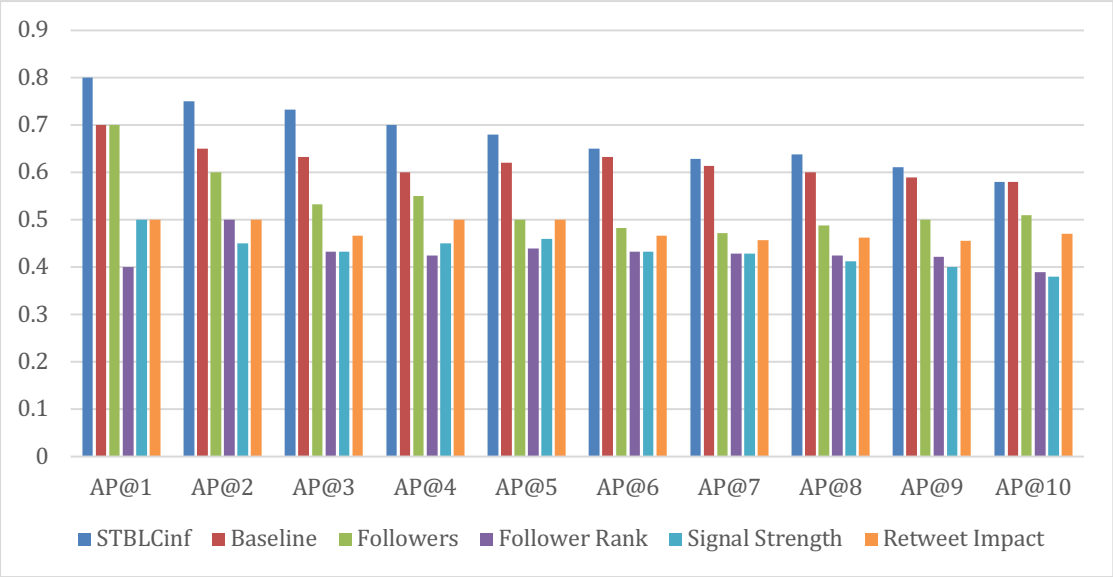

Figure 56. AP@top-10 recommendations of group 3. The proposed profile is compared to the baseline and profiles from the literature.

Overall performance was computed by MAP. The results of this evaluation are shown in Figure 57. The proposed profile outperformed all other profiles and achieved the highest MAP value. It was better than the baseline profile by 4%. It was 7% better than the follower count and retweet impact profiles, which both came in third place after the STBLCinf and baseline profiles. The follower rank profile achieved the lowest MAP value and was 14% worse than the STBLCinf profile.

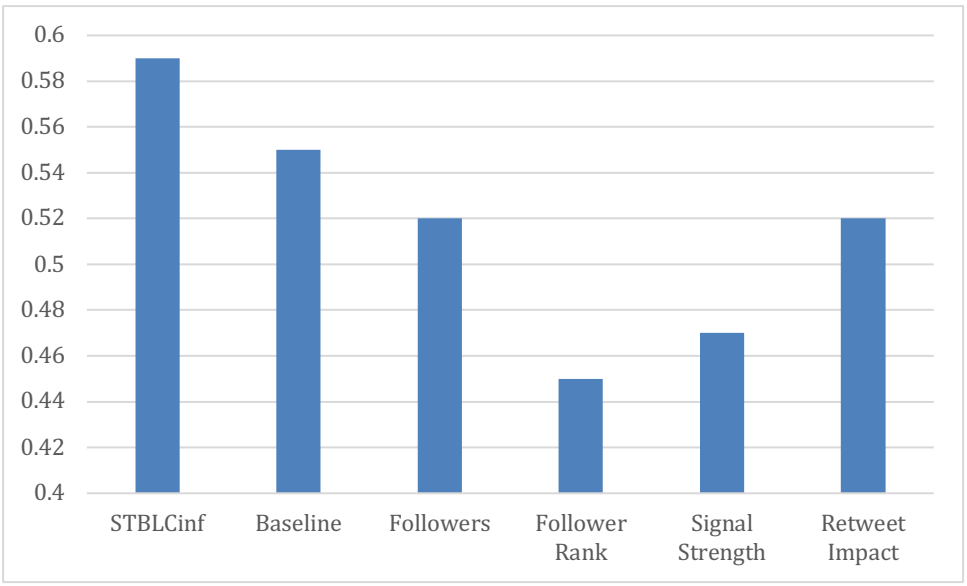

Figure 57. MAP of group 3. The proposed profile is compared to the baseline and profiles from the literature.

After evaluating the performance of the proposed profile in group 3 against the baseline and literature profiles, the next step was to evaluate it against other similarity and distance profiles. The AP@top-k recommendations metric was used and the tested k values ranged from 1 to 10. As shown in Figure 58, the STBLCinf profile achieved the highest performance in 70% of the top 10 recommendations. The Euclidean distance profile, which was a competitive profile in the previous evaluations, was the second best performing profile. It achieved the highest AP@top-k only once, when the k was set to 2. In AP@1, it shared the best performance with the Jaccard similarity profile. Unsurprisingly, the Manhattan profile had the worst performance in recommending related items. In general, the proposed profile proved its quality in delivering more related items within the top 10 recommendations when it was compared to other similarity metrics.

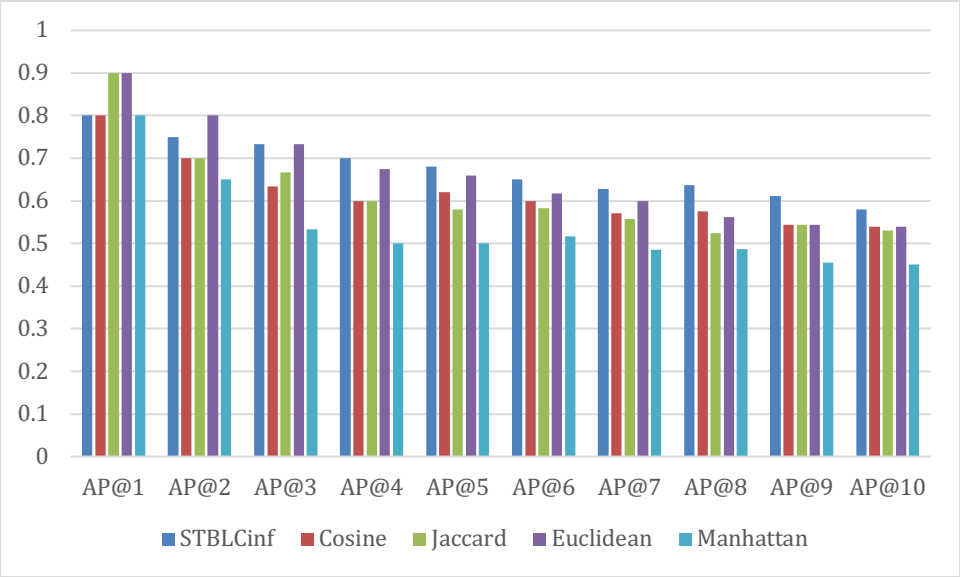

Figure 58. AP@top-10 recommendations of group 3. The proposed profile is compared to the similarity and distance profiles.

The MAP metric was used in order to evaluate the effectiveness of the proposed profile in delivering the related items more accurately (see Figure 59). The STBLCinf profile achieved the best MAP value of 0.59 in comparison to other similarity profiles, namely cosine, Jaccard, Euclidean and Manhattan. The Euclidean distance profile came in second place with 0.57. The STBLCinf profile was better than the Euclidean profile by 2%. The Manhattan profile had the lowest MAP value across all profiles.

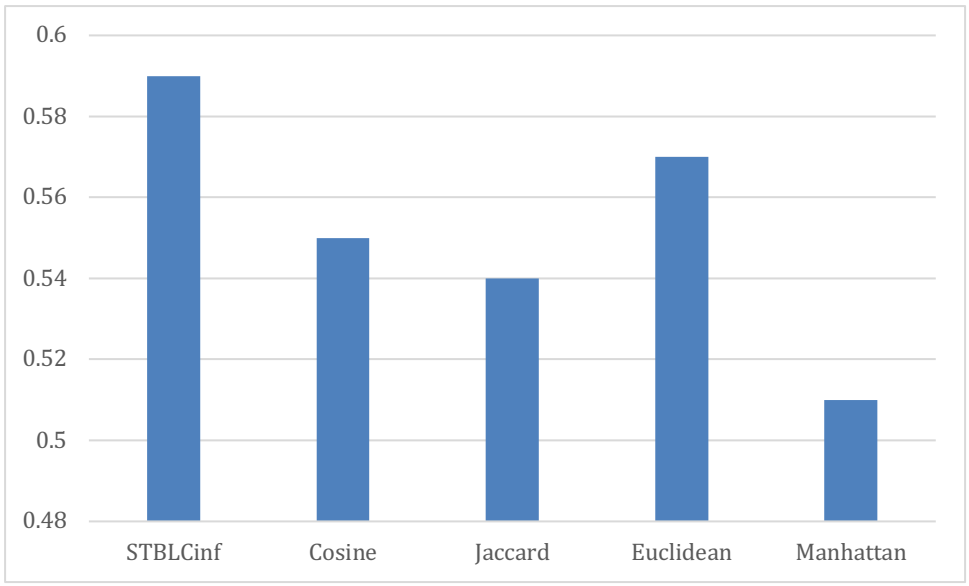

Figure 59. MAP of group 3. The proposed profile is compared to the similarity and distance profiles.

The next step was to see the distribution of activity and originality inside group 3. Figure 60 clarifies the percentage of activity and originality in the group. Active users share the group equally with less active users, as both represent 50% of the group. The percentage of users who are producers represents about 63% of the group whereas the receivers represents only 37%.

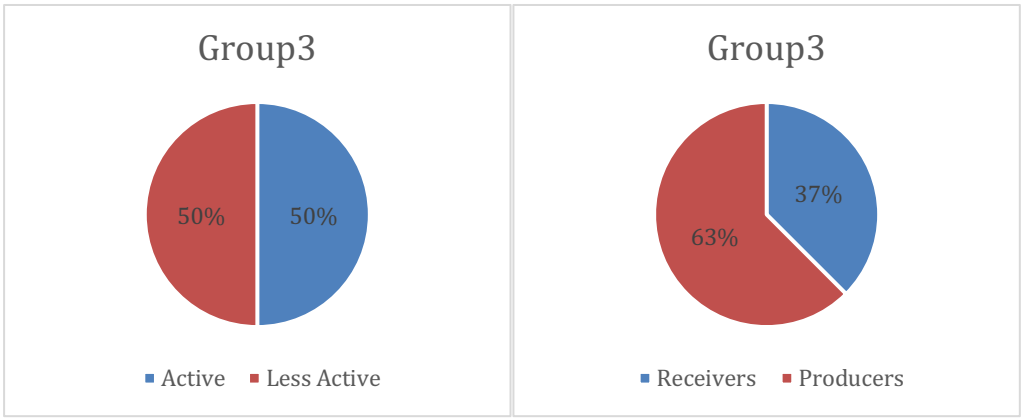

Figure 60. Activity and originality distributions in group 3.

In group 4, which has the users who follow a big number of followees (friends), the metric of AP@top-k was used; the results are shown in Figure 61. It can be clearly seen that the

STBLCinf profile achieved the best performance in recommending more related items to users within the top 10 recommendations. Furthermore, it outperformed the baseline and all other literature profiles by 100% of all k values. When the k value was set from 1 to 3, the STBLCinf profile was better than the baseline by 10%, as it achieved 0.80 MAP, whereas the baseline was 0.70. The baseline profile achieved better performance than the other profiles. Surprisingly, the follower count profile achieved the worst performance seven times out 10.

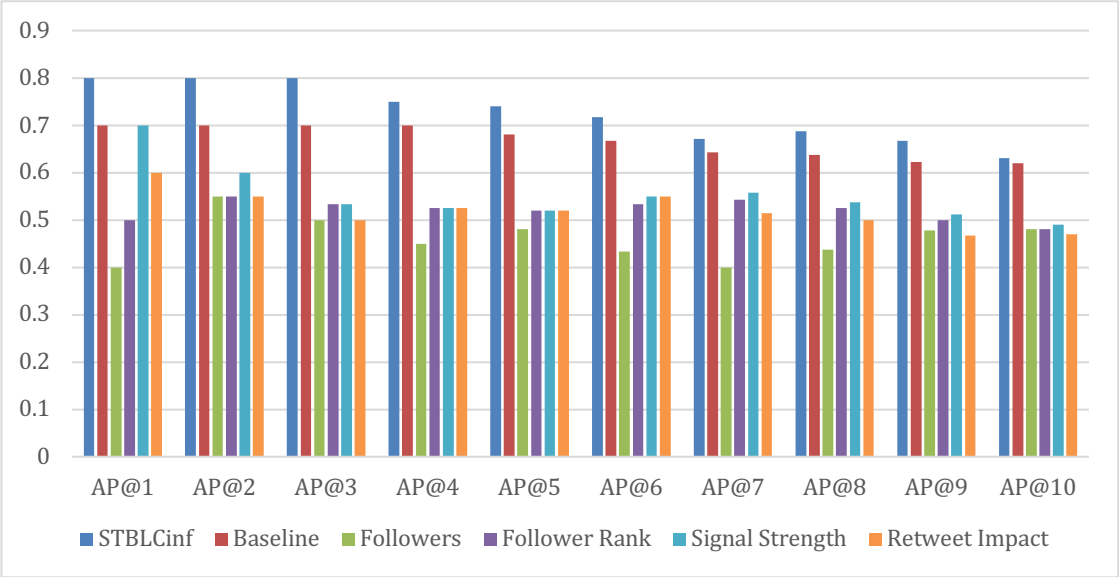

Figure 61. AP@top-10 recommendations of group 4. The proposed profile is compared to the baseline and profiles from the literature.

The next step was to use the MAP metric in order to evaluate the effectiveness of the proposed profile in delivering related items in comparison to the baseline and the literature profiles. The results show that the STBLCinf profile's performance was the best in comparison to the other profiles. It was better by 8% than the baseline profile, which came in second place. The follower count profile, which was a competitive profile to both STBLCinf and the baseline, achieved 13% MAP value, lower than the STBLCinf. The follower rank profile achieved the lowest MAP value across all other profiles. See Figure 62.

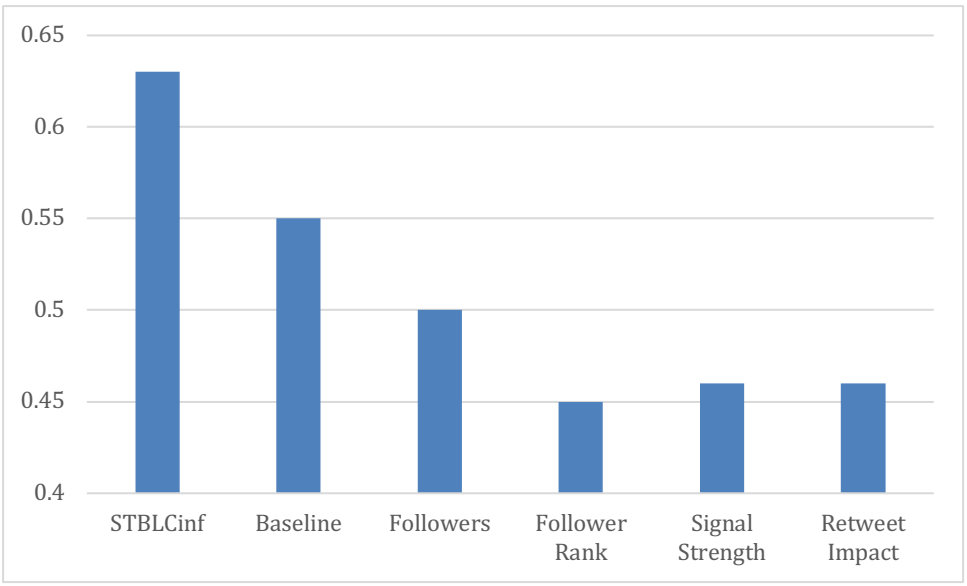

Figure 62. MAP of group 4. The proposed profile is compared to the baseline and profiles from the literature.

In using the MAP metric in order to see how good the system is in retrieving the related recommendation items, the proposed profile was compared to other profiles based on similarity and distance metrics. The results show that the STBLCinf profile was better than the Euclidean distance profile. The former achieved a 0.63 MAP value, whereas the latter achieved 0.62. The cosine and Jaccard profiles remained the same as before with no improvement on all comparisons. The Manhattan profile achieved the lowest MAP value. See Figure 63

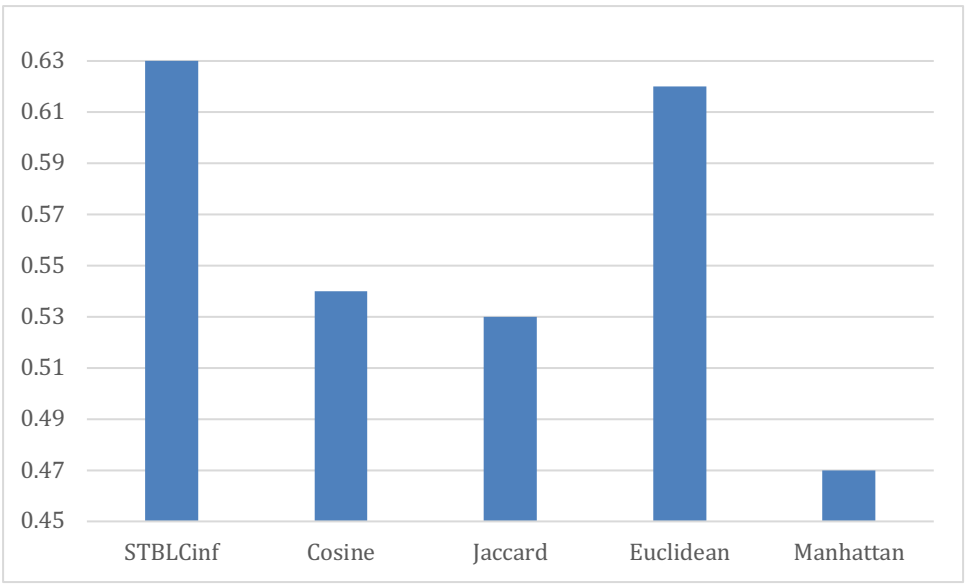

Figure 63. MAP of group 4. The proposed profile is compared to the similarity and distance profiles.

Figure 64 shows the activity and originality distribution inside group 4. As can be seen, the majority of users are less active users, who represent 60% of the group, whereas active users represent only 40%. In the originality distribution, users who are receivers represent 62% of the group and producers 38% of the group. This distribution clarifies the performance of the proposed profile inside group 4 in comparison to other profiles.

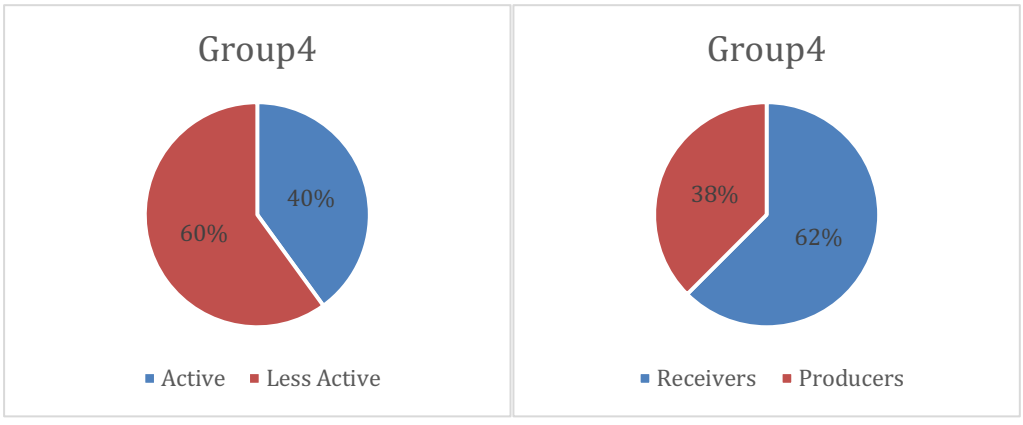

Figure 64. Activity and originality distributions in group 4.

In the next step, the performance of the STBLCinf profile was compared across all groups in order to obtain a clear view of its performance. The results of the metric of AP@top-10 recommendations are shown in Figure 65. When the k was set to 1, the proposed profile achieved the same AP value in groups 2, 3 and 4, as all of them had 0.80. However, it achieved a lower AP@1 value by 60% in group 1. The performance of the profile in group 4 remained steady in k = 1, 2 and 3, and then dropped by 17% in k = 10. The profile in group 1 increased from AP@1 to AP@10 by 21%. The profile in group 2 dropped by 16% at AP@10 and the profile in group 3 decreased by 22% in AP@10. The best performance of the STBLCinf profile was seen in group 4. The profile's performance in group 1 improves when the k value increases.

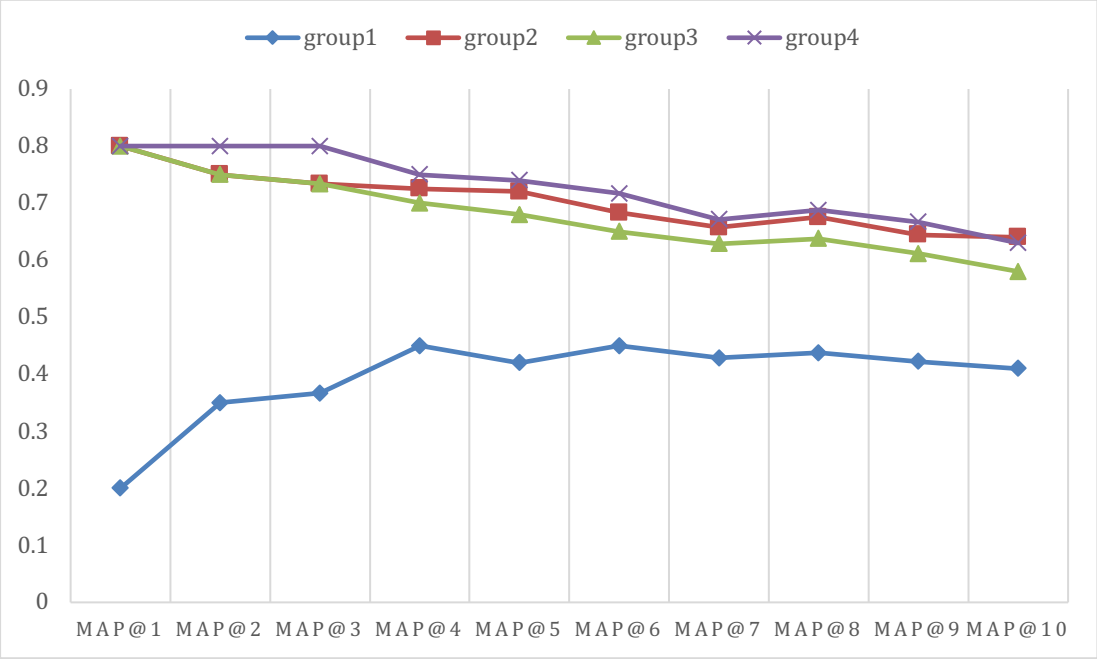

Figure 65. AP@top-10 recommendations of the proposed profile across all groups.

In regards to the MAP metric, it can be seen in Figure 66 that the proposed profile in group 4 has the highest MAP value in comparison to other groups. Also, it can be seen that the performance of the proposed profile increases when the number of friends increases. The MAP value in group 1 was 0.46 and in group 4 it was 0.63, which means that it increased by 17%. On the other hand, the baseline in group 1 achieved an MAP value of 0.45 and in group 4 of 0.55,

which means it only increased by 10%. The performance of the STBLCinf was better than the baseline by 4% in group 3 and 8% in group 4.

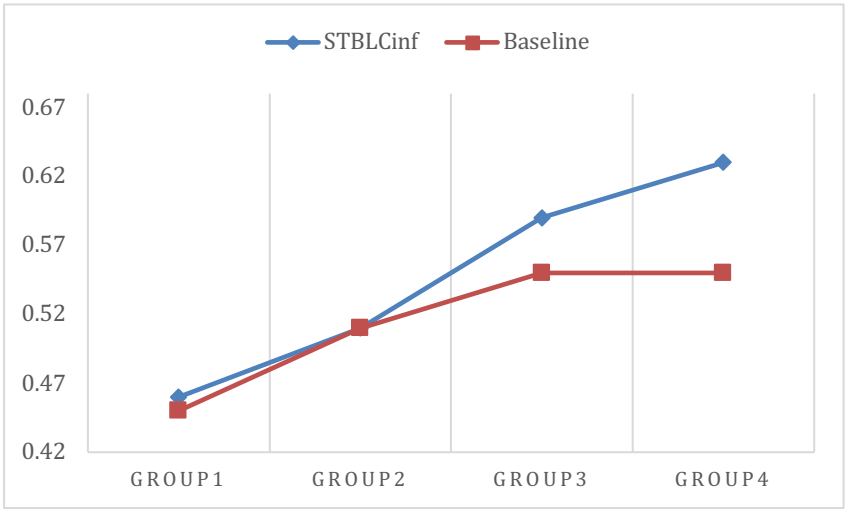

Figure 66. MAP of the proposed profile and the baseline across all groups.

The activity chart in previous Figures helps us to understand the results of the proposed profile's performance across all groups. As seen previously, the proposed profile works better with fewer active users. Group 4 has 60% less active users and group 1 has 40% less active users. Group 4 has 62% of users who are receivers and group 1 has 33%. When the activity distribution is equal in group 2 and group 3, the performance might be affected by the originality of the users. As seen before, the proposed profile works better with producers than receivers, and that might be what happened in groups 2 and 3. Producers made up 33% of group 2, whereas they were 63% of group 3.

4.3.2. Indirect explicit relationships

4.3.2.1. General results

In this subsection, the profiles built from tweets from different sources in indirect explicit relationships with the examined users are compared against each other and also against the profile from the direct explicit relationships, which is the STBLCinf profile. The aim of this step was to determine which profile performs better and to see how far we have to go to collect related tweets. Results in this stage were again obtained via the metrics of average precision (AP) at top-k (AP@k) recommendations and mean average precision (MAP). The tested values of k ranged from 1 to 10.

In the AP metric at top-k recommendations, the profiles built from the sources of travelling tweets accounts (TTA), influential friends of influential friends (Inf of Inf) and the mixed profile (TTA + Inf of Inf) were tested and their performance was evaluated and compared. They were also compared to the STBLCinf and baseline profiles in order to see the overall performance (see Figure 67). When the TTA and Inf of Inf were compared, the results show that the Inf of Inf profile outperformed the TTA profile in all AP@top-10 recommendations. When the k was set to 4, the Inf of Inf profile was at its best and the difference between them was 6%. When both were compared to the mixed profile, which is TTA + Inf of Inf, the results show that the Inf of Inf profile outperformed the others nine times out of the top 10 recommendations. The mixed profile's performance was the same in the AP@1 test. The performance of the mixed profile was better than the TTA profile as it achieved a higher AP value five times at k = 1, 3, 8, 9 and 10. The TTA profile achieved a higher AP value only three times when the k was set to 5, 6 and 7.

The TTA, Inf of Inf and TTA + Inf of Inf profiles are extended versions of the STBLCinf. Therefore, they should be compared to the STBLCinf profile in order to see how the performance changes, whether it increases or decreases. The results show that the Inf of Inf profile showed better performance than the other profiles, especially the STBLCinf one. It was better five times whereas the STBLCinf was better three times in the top 10 recommendations.

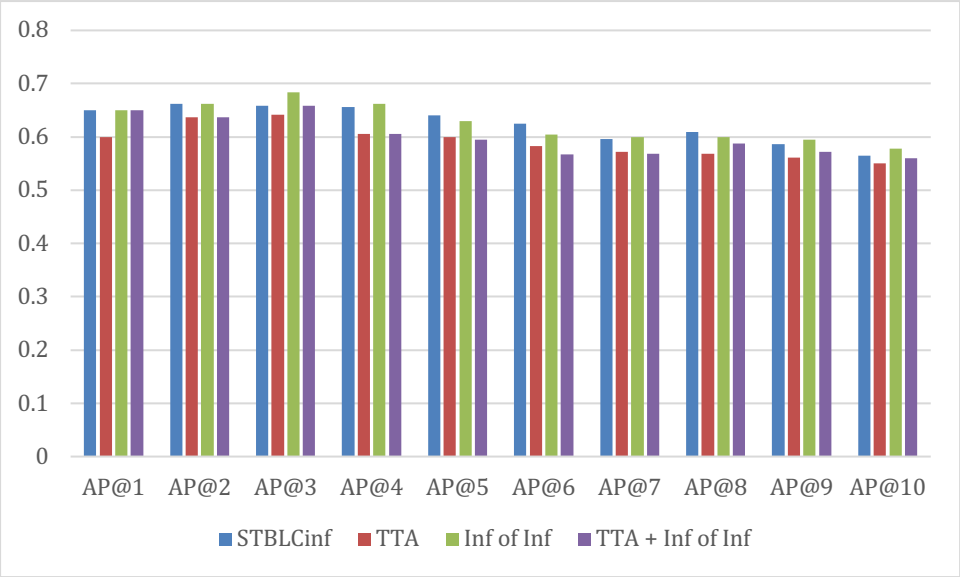

Figure 67. AP@top-10 recommendations of profiles built from indirect explicit relationships.

In regards to the MAP metric, the profiles were compared in order to see the overall performance. The performance of profiles was compared to the STBLCinf and the baseline. The baseline was chosen in order to see whether they can achieve better performance than the baseline or worse. The results show that the TTA profile achieved a higher MAP value than the Inf of Inf profile by 0.2%. The mixed profile achieved a lower MAP value than the TTA by 0.6% and lower than the Inf of Inf by 0.4%. When the profiles were compared to the baseline profile, the TTA, Inf of Inf and mixed profiles achieved a higher MAP than the baseline by 2%, 1.8% and 1.4%, respectively. The STBLCinf was also compared to the stage 2 profiles and the results show that the it was better than the TTA, Inf of Inf and mixed profiles by 1%, 1.2% and 1.6%, respectively (see Figure 68).

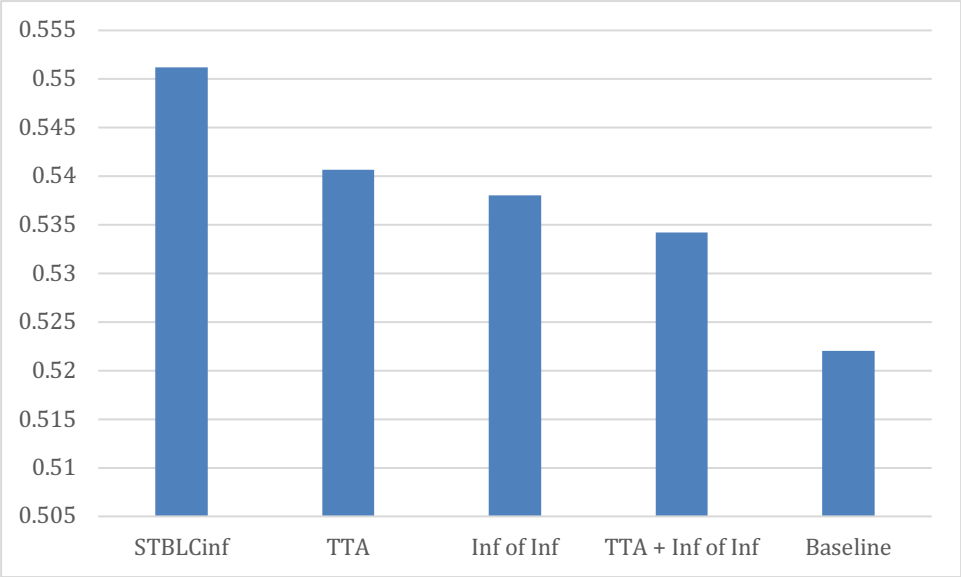

Figure 68. MAP values of profiles of indirect implicit relationships.

The above results show that the Inf of Inf profile was better in delivering the top 10 recommendations and also better than the STBLCinf. However, the STBLCinf was the best in delivering all related recommendation items and, additionally, the TTA was better than Inf of Inf, as proved by the MAP metric. It was necessary to look at the average tweet number of each profile and compare them to the STBLCinf profile. Table 14 shows that the TTA has 115% more tweets than the STBLCinf profile. The number of tweets on the Inf of Inf profile increased by 12.8% and the TTA + Inf of Inf profile increased the most by 127.7%. The low size of the increase in tweets in the Inf of Inf profile might explain that the new tweets enriched the profile with relative tweets and helped it perform better in retrieving more related items when the top 10 recommendations were delivered. This also proves that the influence rule is stronger than other methods in finding resources in order to enrich a user profile. Enriching a profile with multiple resources can affect the overall performance, such as in the case of the TTA + Inf of Inf profile, as it increases the number of tweets and consequently affects performance.

Table 13. Average number of tweets of the STBLCinf and indirect explicit relationship profiles.

Profiles	STBLCinf	TTA	Inf of Inf	TTA + Inf of Inf
Average number of tweets	3577	7693	4036	8144

4.3.2.2. Dividing the dataset

As explained previously, the dataset is divided into the mentioned categories in order to see deeper analysis and to understand the results in advanced way. The profiles of this stage will be compared to the STBLCinf in each category of activity, originality and following friends.

4.3.2.2.1. Activity (engagement)

As explained previously, in this category, the users were split into two categories, less active and active users. Then, the performance of the profiles, which are TTA, Inf of Inf and the mixed profile, were compared against the STBLCinf profile in both groups.

In the less active users group and using the metric of AP@top-10 recommendations, as Figure 69 shows, the Inf of Inf profile outperformed the TTA and the TTA + Inf of Inf profiles nine times out of 10 recommendations. In comparison to the STBLCinf profile, the Inf of Inf profile achieved a higher AP value three times, whereas the former achieved better performance four times. The mixed profile was slightly better than the TTA profile in delivering related item in the top 10 recommendations. Both demonstrated the lowest performance.

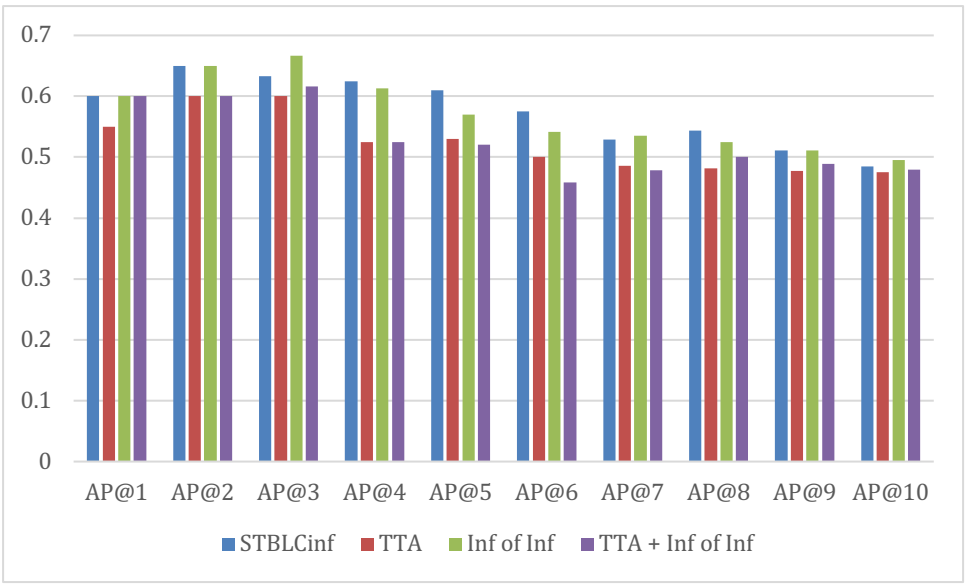

Figure 69. AP@top-10 recommendations of indirect explicit relationship profiles of less active users.

In the MAP metric, the result was similar to the general evaluation before the dataset was split. It showed that the TTA, Inf of Inf and TTA + Inf of Inf profiles achieved better performance than the baseline by 5.4%, 4.4% and 3.9%, respectively. However, the TTA, Inf of Inf and TTA + Inf of Inf profiles achieved a lower MAP value than the STBLCinf profile by 1.6%, 2.6% and 3.1%, respectively. See Figure 70.

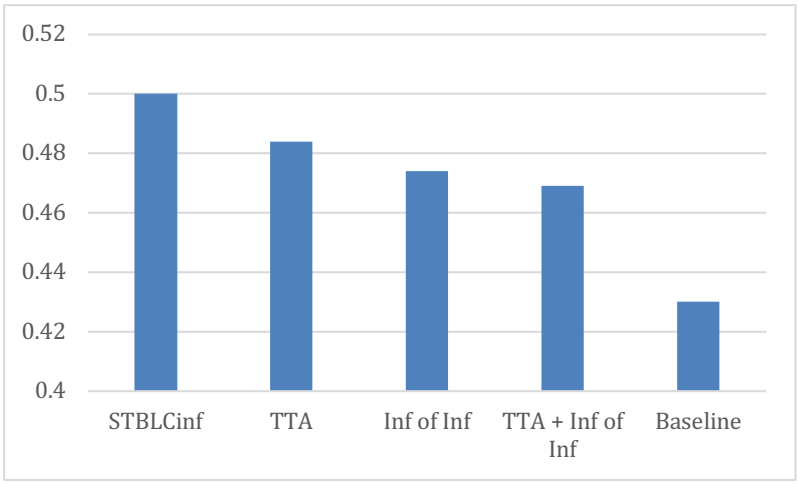

Figure 70. MAP values of indirect explicit relationship profiles of less active users.

In the active users group, the results show that the Inf of Inf profile outperformed all other profiles four times (see Figure 71). Surprisingly, the STBLCinf profile did not outperform the other profiles at any of the k values. The TTA profile achieved the lowest AP four times. When the top 10 recommendations were delivered, the results prove that the Inf of Inf profile is able to deliver related recommendations to active users more accurately than other profiles.

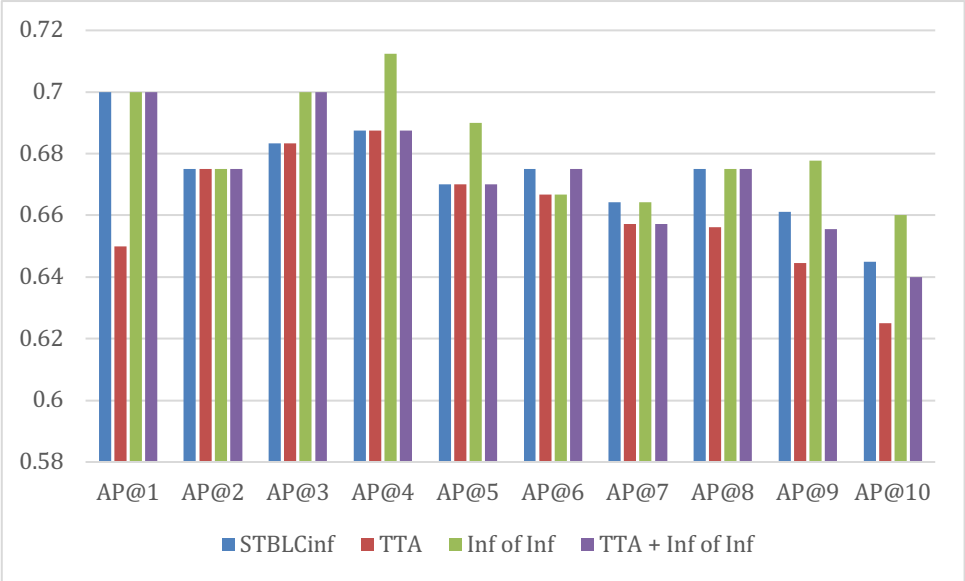

Figure 71. AP@top-10 of active users in the indirect explicit relationship profiles.

After seeing the performance of the profiles when the top 10 recommendations were recommended, it is necessary to display the results of the profiles when all related recommendations items were retrieved. This step was performed by using the MAP metric; the results are presented in Figure 72. Surprisingly, the Inf of Inf profile achieved the highest MAP value. It was slightly better than both the STBLCinf and the baseline. The TTA and TTA + Inf of Inf profiles achieved lower MAP values than the baseline. This might suggest that the Inf of Inf profile is suitable for active users more than less active ones.

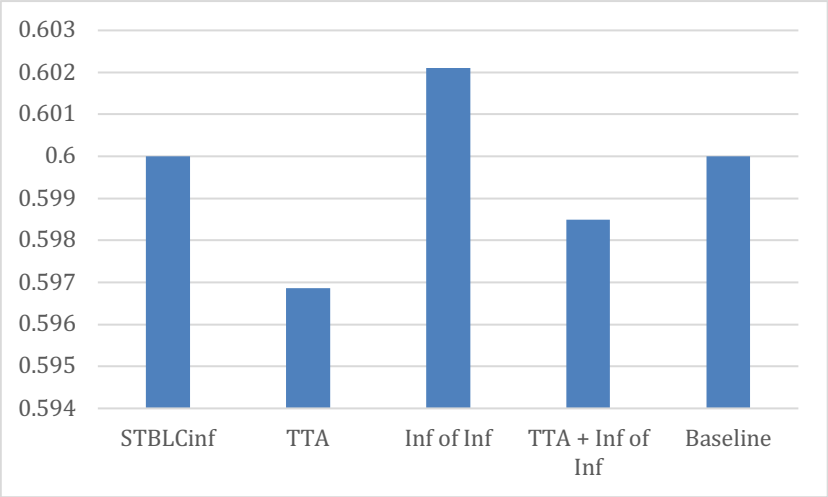

Figure 72. MAP values of active users profiles in indirect explicit relationships.

4.3.2.2.2. Originality

The users in this category were split into two categories, as explained previously: receivers and producers. After calculating the originality score of each user and splitting the users into the two categories, the performance of the stage 2 profiles was compared against the STBLCinf and baseline profiles in each group.

In the receivers group and by using the metric of AP@top-10 recommendations, as Figure 73 shows, the Inf of Inf profile had the highest AP five times and the STBLCinf profile achieved the highest AP four times. Consequently, the former is better than the latter by 10% in delivering more related items within the top 10 recommendations. The mixed profile (TTA + Inf of Inf) did not show any strength compared to other profiles. The TTA profile had the worst performance.

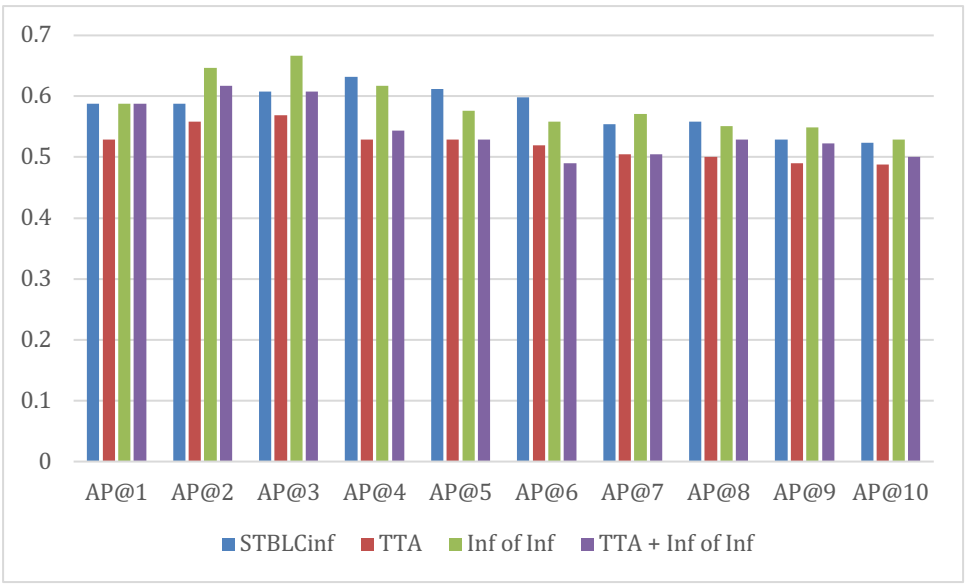

Figure 73. AP@top-10 values of receivers group in indirect explicit relationships.

In regards to the MAP metric, the STBLCinf profile outperformed all profiles and the Inf of Inf profile achieved a lower MAP value than the former by 0.4%. The Inf of Inf profile performed better than the baseline profile, while the TTA and TTA + Inf of Inf profiles performed worse than the baseline. See Figure 74.

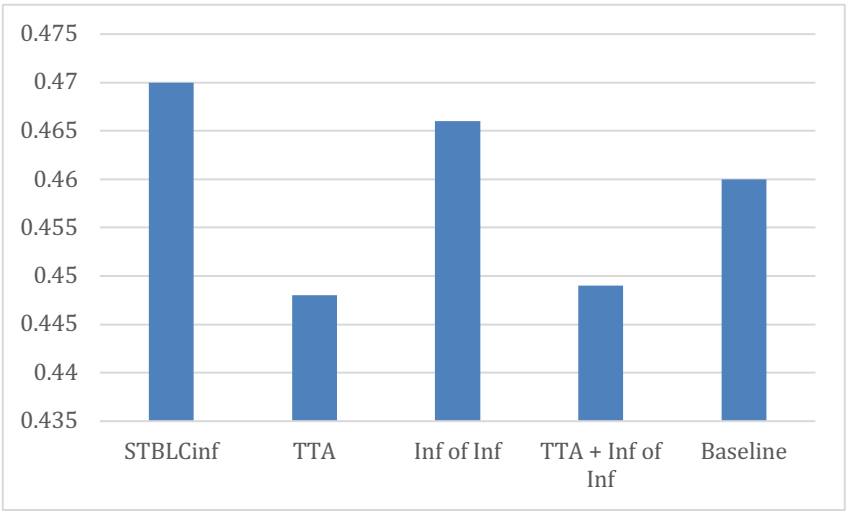

Figure 74. MAP values of the receivers group in indirect explicit relationships.

In the producers group, the metric of AP@top-10 recommendations was used in order to find which profile can deliver more accurate recommendations items within the top recommended ones. Figure 75 shows that the Inf of Inf profile outperformed the other profiles five times, which is better than the STBLCinf which outperformed the other profiles only two times. Comparing the TTA profile and the mixed profile, each one of them outperformed the other four times. Moreover, it can be seen that the performance of the TTA profile was better with the group of producers.

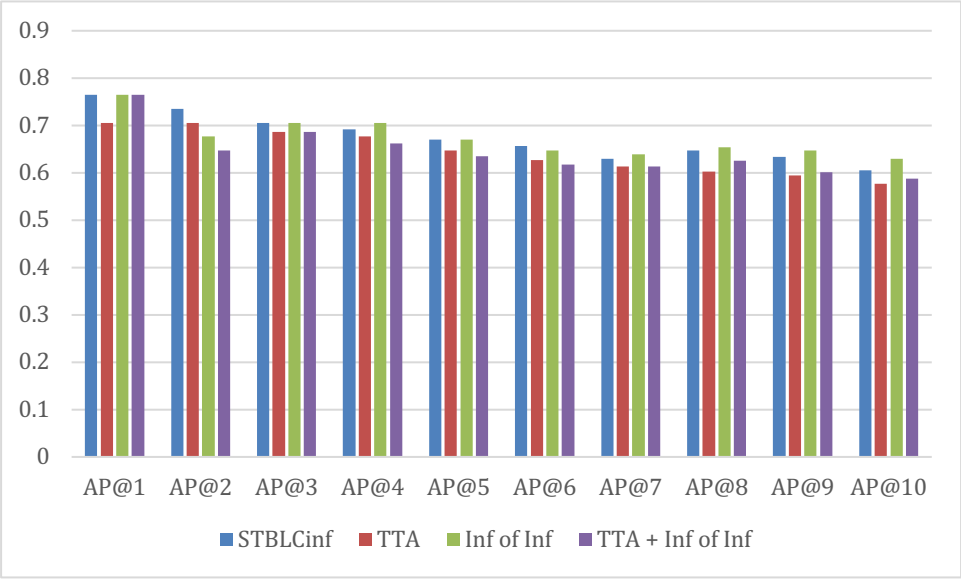

Figure 75. AP@top-10 of producers group in indirect explicit relationships.

In terms of overall performance, the MAP metric was used and the results are shown in Figure 76. Unexpectedly, the TTA profile showed better performance than the STBLCinf and Inf of Inf profiles by 0.1% and 0.7%, respectively. Another surprise was that all profiles achieved better performance than the baseline profile; the mixed profile was better by 4% and the Inf of Inf profile was better by 4.4%.

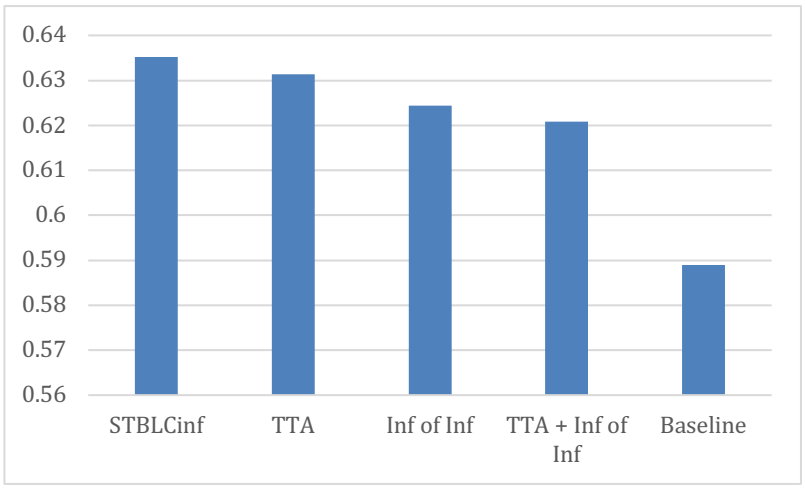

Figure 76. MAP values of producers group in indirect explicit relationships.

4.3.2.2.3. Friends count

In this category, the aim was to find connections between the performance of the profiles and the number of following friends. Therefore, the users were grouped into four groups based on their number of following friends, as explained previously. Then, the performance of the profiles of stage 2 was measured and compared against the stage 1 profile.

In the metric of AP@top-10 recommendations, as Figure 77 shows, all profiles achieved a small number equally in AP@1 as they achieved only 0.2 and they increased by more than 20% at AP@10. It can also be seen that the TTA and TTA + Inf of Inf profiles achieved better performance similarly among all profiles. They both outperformed the STBLCinf profile seven times, whereas the Inf of Inf profile outperformed it only twice. The STBLCinf profile outperformed all profiles one time when the k was set to 2, and outperformed the Inf of Inf profile four times.

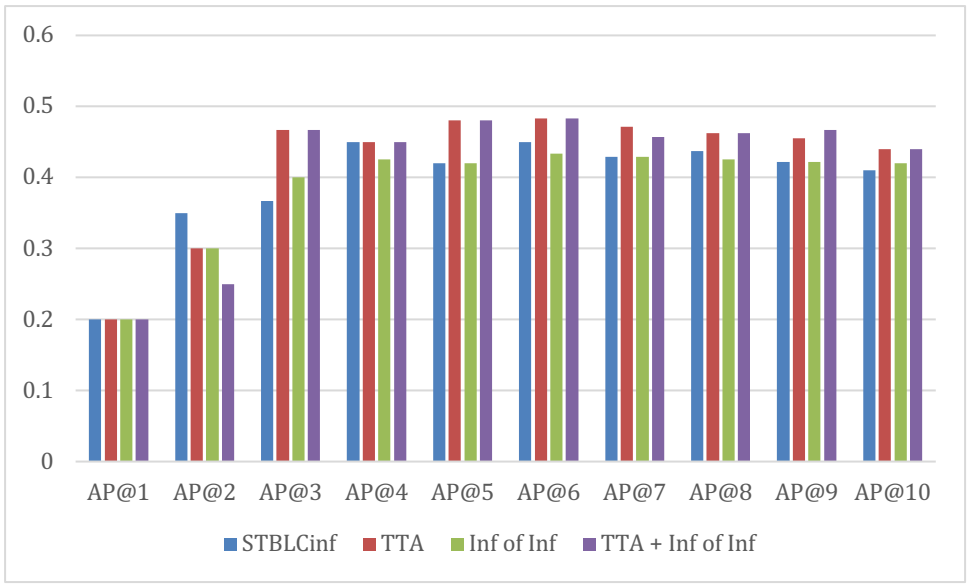

Figure 77. AP@top-10 values of group 1 in indirect explicit relationships.

In the previous figure, it is clear that TTA and TTA + Inf of Inf profiles were useful in recommending more related items when the top 10 recommendations were delivered to group 1. In order to find which profile is able to deliver all relative items better, the MAP metric was used, and the results are shown in Figure 78. The STBLCinf profile achieved the highest MAP value and the TTA profile had a slightly lower value. On the other hand, the Inf of Inf and the mixed profiles achieved better MAP than the baseline; the latter had a slightly higher value than the former and this is due to the high performance of the TTA.

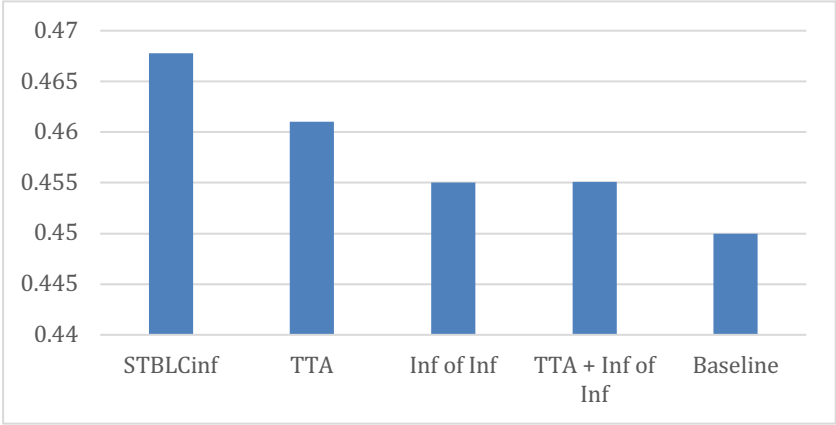

Figure 78. MAP values of group 1 in indirect explicit relationships.

In this group, the users follow a number of friends between 172 and 300. The results are shown in Figure 79, which clarifies that all profiles except the TTA profile performed the same in AP@1, as they had 0.80, which is a very high value. The Inf on Inf profile showed fantastic performance as it achieved the best AP values seven times out of 10. The TTA profile achieved the worst performance. The mixed profile did not show any better performance than the STBLCinf profile, and the latter outperformed all profiles twice.

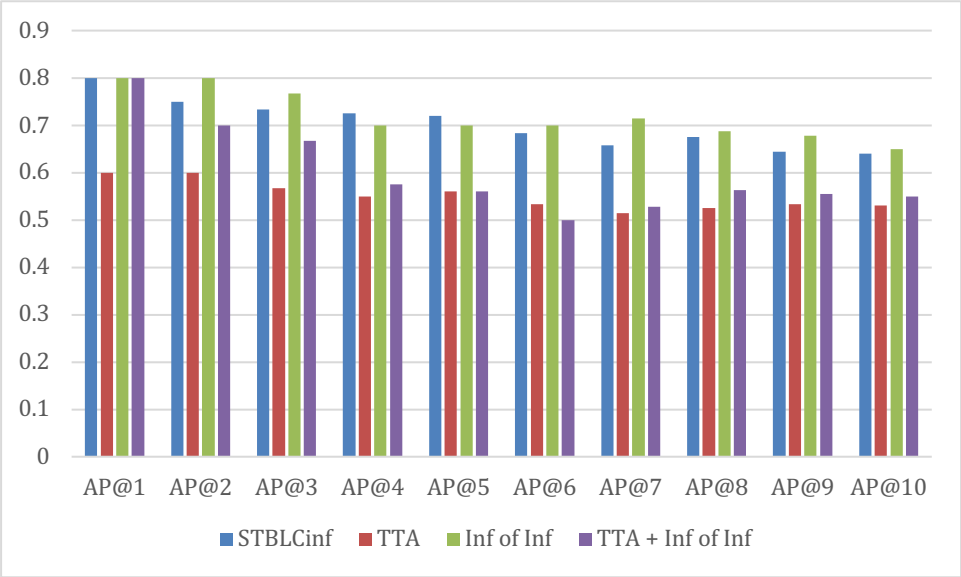

Figure 79. AP@top-10 values of group 2 in indirect explicit relationships.

It is clear that the Inf of Inf profile is the most suitable profile for this group to deliver more accurate items when the top 10 recommendations are recommended. The next step was to find the profiles' ability to retrieve all relevant recommendations, and this was calculated by the MAP metric, as shown in Figure 80. The Inf of Inf profile achieved better performance than the TTA and TTA + Inf of Inf profiles. However, it achieved a lower MAP value than the STBLCinf and baseline profiles by 1%.

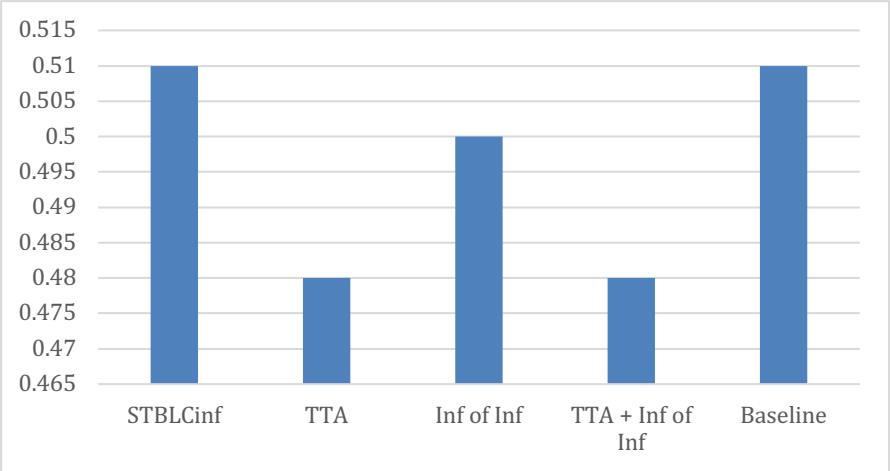

Figure 80. MAP values of group 2 in indirect explicit relationships.

Users in this group follow a number of friends between 300 and 440. The performance of the profiles when the top 10 tweets are recommended was measured by the AP@top-k; the results are presented in Figure 81. All profiles achieved the same high performance on AP@1 = 0.80 and AP@2 = 0.75. The competition in this group is between Inf of Inf and TTA + Inf of Inf profiles, as both outperformed the other profiles two times and both were higher than the STBLCinf profile four times. Comparing them, the Inf of Inf achieved higher AP values four times and the mixed profile achieved higher values three times.

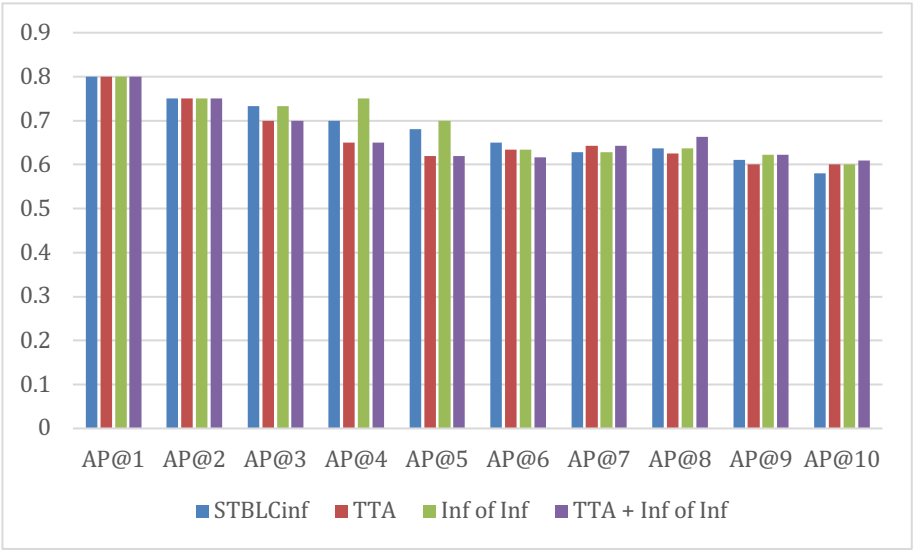

Figure 81. AP@top-10 values of group 3 in indirect explicit relationships.

It was seen previously in this group that the competition was between the Inf of Inf and TTA + Inf of Inf profiles. In order to find the overall performance in retrieving all relevant recommendations, the MAP metric was used in measurement and the results are presented in Figure 82. The mixed profile, TTA + Inf of Inf, outperformed all other profiles. The TTA profile gave the second-best performance and the Inf of Inf profile achieved a slightly lower MAP value than the STBLCinf profile. In comparison to the baseline, all profiles achieved better performance by a percentage between 4% and 5%.

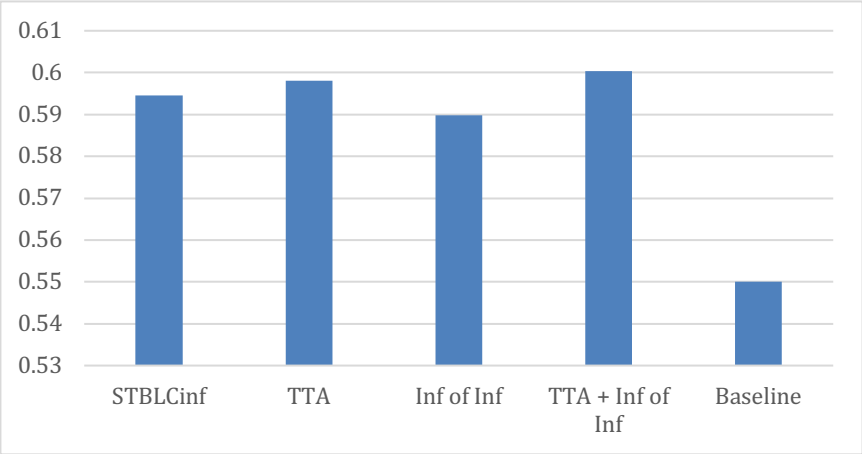

Figure 82. MAP values of group 3 in indirect explicit relationships.

In this group, users follow between 441 and 800 friends. The profiles at this stage were tested and their performance measured by the AP@top-10 and MAP metrics. In the former metric, the results show that all profiles in this group achieved the same value at AP@1 as they had 0.80, similar to the previous groups, 1 and 3. In comparison to the STBLCinf profile, each of the TTA and Inf of Inf profiles achieved better performance three times and the mixed profile was better only twice. The STBLCinf achieved the highest performance four times. See Figure 83.

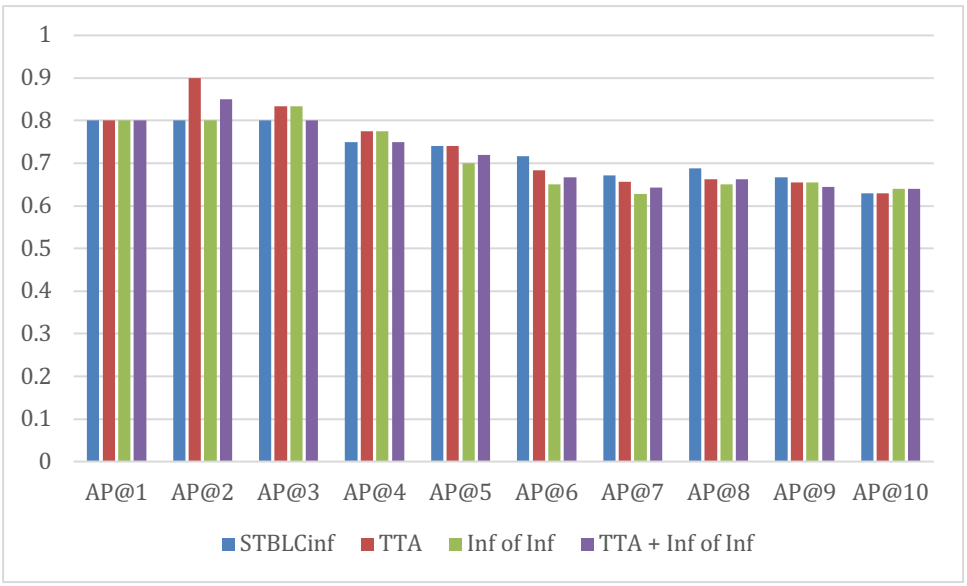

Figure 83. AP@top-10 values of group 4 in indirect explicit relationships.

In regards to the MAP metric, Figure 84 shows the results and it is clear that the STBLCinf profile achieved the best performance in ordering the relevant recommended items. In comparison to the STBLCinf, the TTA profile came second, as it achieved a 1% lower MAP value. The Inf of Inf profile was lower by 3% and the mixed profile was lowest among profiles at this stage by 3.4%. When the profiles were compared to the baseline, the results show that all profiles achieved better performance by between 4.6% and 7%.

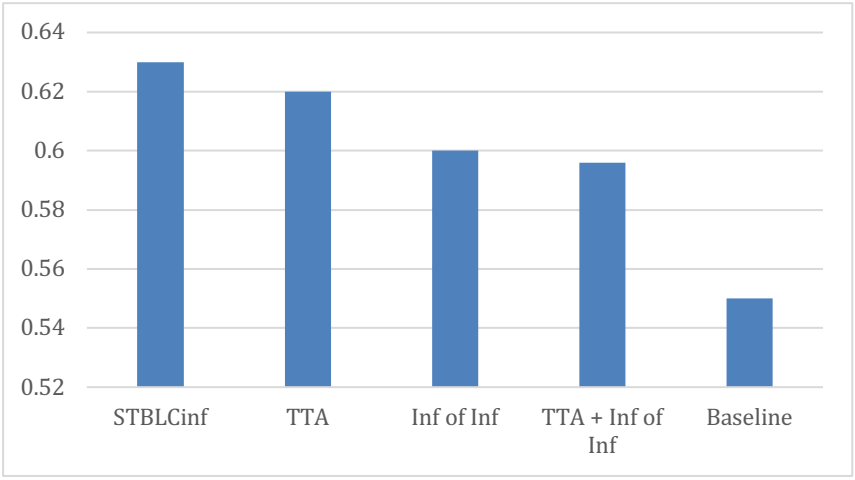

Figure 84. MAP values of group 4 in indirect explicit relationships.

After analysing the performance of the stage 2 profiles in different groups based on number of friends, it was necessary to analyse the performance of the profiles across all groups. Figure 85 shows the MAP of all three profiles in each group. In general, the performance of all profiles improves when the number of friends increases. The performance of all profiles increases sharply between groups 2 and 3. The performance of the mixed profile dropped slightly in the group that has the highest number of followed friends, which is group 4.

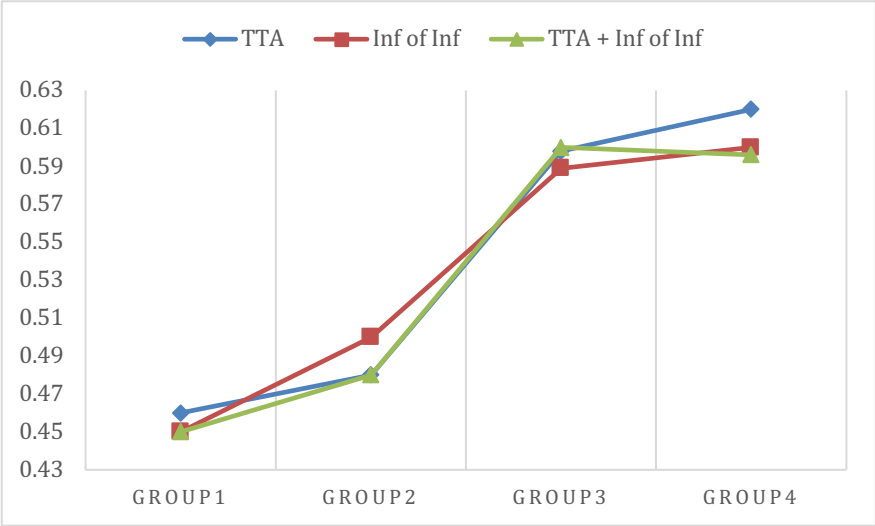

Figure 85. MAP values of indirect explicit relationships profiles in all groups.

Figure 86 shows the performance of the TTA profile across all groups when the top 10 recommendations are recommended. It shows that performance of the profile AP@k = 1 increases based on the increase in followed friends and decreases when the number of followed friends is small. It also shows that the performance drops when the k value increases in group 4, whereas it improves when the k value increases in group 1. In general, the performance of the TTA profile improves when the number of followed friends gets bigger.

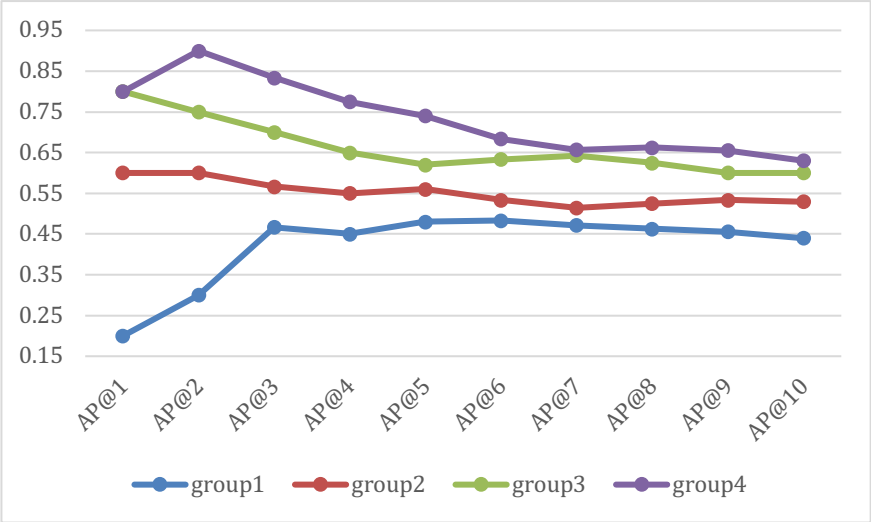

Figure 86. AP@top-10 values of TTA profile across all groups.

Figure 87 shows the performance of the Inf of Inf profile in delivering more relevant items when the top 10 recommendations are recommended across all groups. In AP@1, there is a big difference of 60% between group 1 and the other groups. In general, the performance descends when the k value increases in all groups except group 1. The profile's performance in group 1 was lower as it increases when the k value increases. Its performance increased by 22% when k = 10.

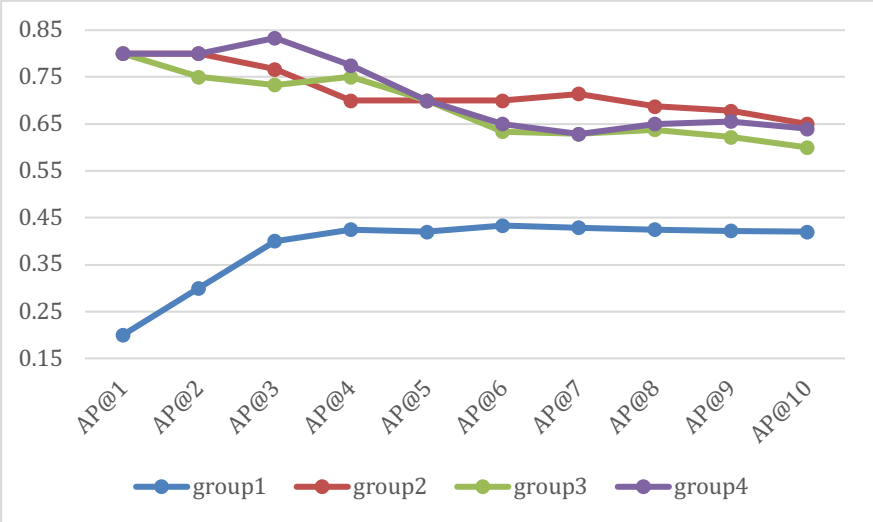

Figure 87. AP@top-10 values of Inf of Inf profile across all groups.

When the top 10 recommendations are recommended based on the mixed profile, which is TTA + Inf of Inf, the performance of this profile was measured across all groups in order to find any connection between its performance and the number of followed friends. Figure 88 shows the performance of the mixed profile in all the groups. Generally, the results show that the performance of the profile decreases when the k value increases in all groups, except group 1, where the performance improves when the k value increases. In group 4, which has the users who have the highest number of followed friends, the profile performed better than any other group in retrieving the most relevant recommendations. In a comparison between AP@1 and AP@10, group 2's performance dropped the most, by an AP value of 25%. The performance of group 3 dropped by 19%, and group 4 dropped by 16%. The performance of group 1 increased by 24%.

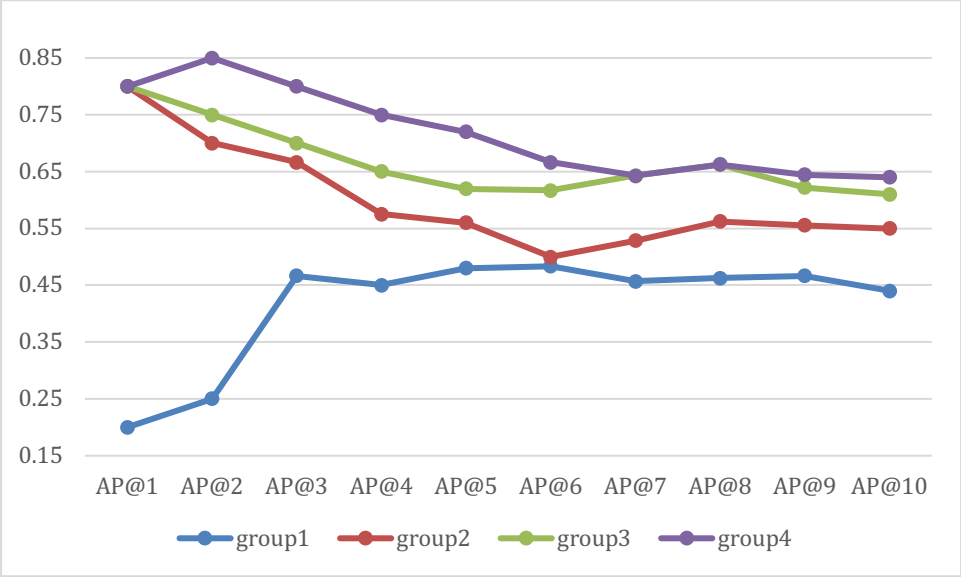

Figure 88. AP@top-10 values of the mixed profile across all groups.

4.3.3. Implicit relationships

4.3.3.1. General results

In this section, the experiment's aim was to see if the influence rule is capable of finding implicitly similar users better than the techniques from the literature, and then using those similar users' accounts to enrich the user profile with relevant tweets. The profiles built from different resources implicitly were compared against each other and against the highest performing profile from direct explicit relationships, which is the STBLCinf profile.

The average precision at top-k metric was used to measure and evaluate the performance of the candidate profiles, which are STBLCinf, Influencers' Followers' tweets (InfF), Follower Count Followers tweets (FCF) and the mixed profile of the last two profiles, InfF + FCF. When the top 10 recommendations are recommended and in terms of comparing only the two profiles InfF and FCF, as Figure 89 shows, the InfF profile had the highest AP valu, five times more than the FCF, whereas the latter outperformed the former only two times. The mixed profile did not achieve a higher AP value than both of the original profiles together. However, it achieved a higher performance three times more than the FCF when the k values were 2, 7 and 8. When the three profiles' performances were compared, the InfF profile outperformed the other profiles three times and the FCF profile outperformed the others only twice. All three profiles achieved the same performance two times, when the k values were 1 and 3. When the STBLCinf profile was compared to the stage 3 profiles, it outperformed them five times, and the InfF profile outperformed all other profiles only one time, when the k value was 4.

Figure 89. AP@top-10 values of the implicit profiles.

In regards to the MAP metric, the STBLCinf and the stage 3 profiles' performance were compared against each other and also against the baseline profile. The results show that the STBLCinf, InfF and FCF profiles achieved similar MAP values. The STBLCinf profile achieved a slightly higher value than the FCF, and the InfF achieved a lower value than both of them. The mixed profile achieved a lower MAP value than the other profiles, and this clarifies that combining the InfF and FCF profiles might affect performance. All profiles achieved better performance than the baseline profile. See Figure 90.

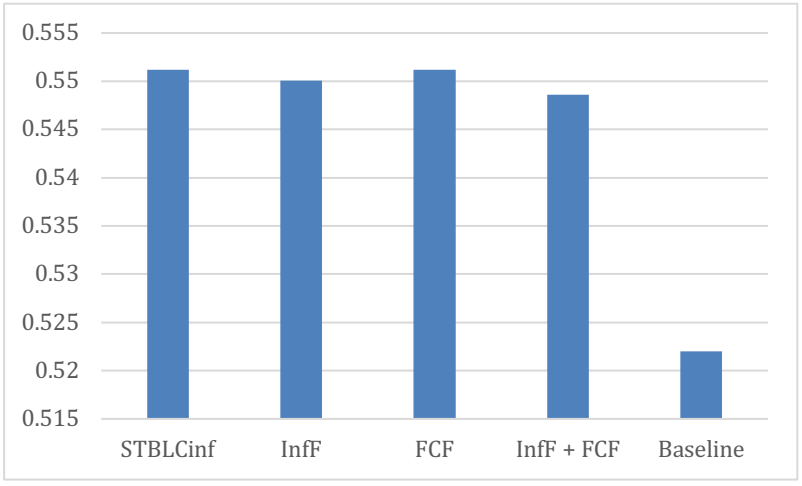

Figure 90. MAP values of the implicit profiles.

The results and figures provided above show that the STBLCinf profile is the best in delivering the top 10 recommendations to users, and the InfF profile is better than FCF. Additionally, it has been shown that the all the three mentioned profiles perform the same in retrieving all relevant recommendations. Therefore, there is a need to know the average number of tweets in each profile. All profiles are extended versions of the STBLCinf profile; the InfF, FCF and InfF + FCF profiles' tweets increase by 31.4%, 2.4% and 33.7%, respectively. Comparing the InfF and FCF profiles in recommending the top 10 recommendations, the extra 31.4% tweets in the former achieved better performance and the small extra 2.4% in the latter affected the performance, and this might mean that the quality of the tweets is low in comparison to the former. The extra 33.7% of tweets in the mixed profile (InfF + FCF) showed that these extra tweets affected its performance, as it became less accurate than the original profiles (InfF and FCF) in both recommending the top 10 items and in delivering all relevant recommended items. Despite the effects on performance of the extra number of tweets, all profiles achieved better performance than the baseline profile. See Table 15.

Table 14. Average number of tweets of STBLCinf and implicit relationships profiles.

Profiles	STBLCinf	InfF	FCF	InfF + FCF
Average number of tweets	3577	4701	3664	4781

It is believed that the machine learning classification excludes many tweets that come from followers of friends that have a large number of followers. As can be clearly seen in the average number of tweets, the FCF profile has fewer tweets that the InfF profile, and only 87 more tweets than the STBLCinf profile. This small number of tweets affected its performance, especially in recommending the top 10 items, and this might mean that the classification algorithms classify most tweets as not-representative (not-retweetable). In order to see whether the InfF profile provides more relevant information to the user, an extra experiment was conducted to discover how accounts from different sources are similar to the examined user. This experiment mainly used the cosine similarity metric to find the top similar accounts to the user. Cosine similarity

was measured between the user's timeline and other accounts' timelines. After that, the top 10 similar accounts were chosen, and the number of accounts that are relevant to either the InfF or FCF profiles was counted. Surprisingly, 61% of the top 10 similar accounts were followers in the InfF technique, whereas 39% of similar accounts come from the FCF technique. This similarity calculation proves that the InfF profile technique has more similarity than the FCF technique.

4.3.3.2. Dividing the dataset

As was done at all stages previously, the dataset was divided into the categories of activity, originality and friends count, to allow for deeper analysis and to understand the results in an advanced way. The profiles at this stage were compared to the STBLCinf profile in each category.

4.3.3.2.1. Activity (engagement)

In this category, the users were divided into two categories, less active and active users, and the same steps were applied as in previous stages. The profiles' performance (InfF, FCF and the mixed profile) was compared against the STBLCinf profile in both groups.

In the less active category, the performance of the profiles in delivering the top 10 ranked items was measured using the metric of AP@top-k. When the InfF and FCF profiles were compared, the results show that the former profile outperformed the latter six times out of the top 10 recommendations, whereas the latter outperformed the other only three times. Surprisingly, the mixed profile outperformed the InfF and FCF profiles once, when the k value was 9. It also outperformed the FCF profile more than the InfF profile. See Figure 91.

When the performance of all the profiles was compared to the STBLCinf profile, the results showed that the InfF, FCF and mixed profiles outperformed the STBLCinf profile five, three and four times, respectively. In an overall comparison, the InfF profile outperformed THE other profiles five times, while the FCF profile outperformed the others two times. Each of the STBLCinf and the mixed profiles outperformed the others only once.

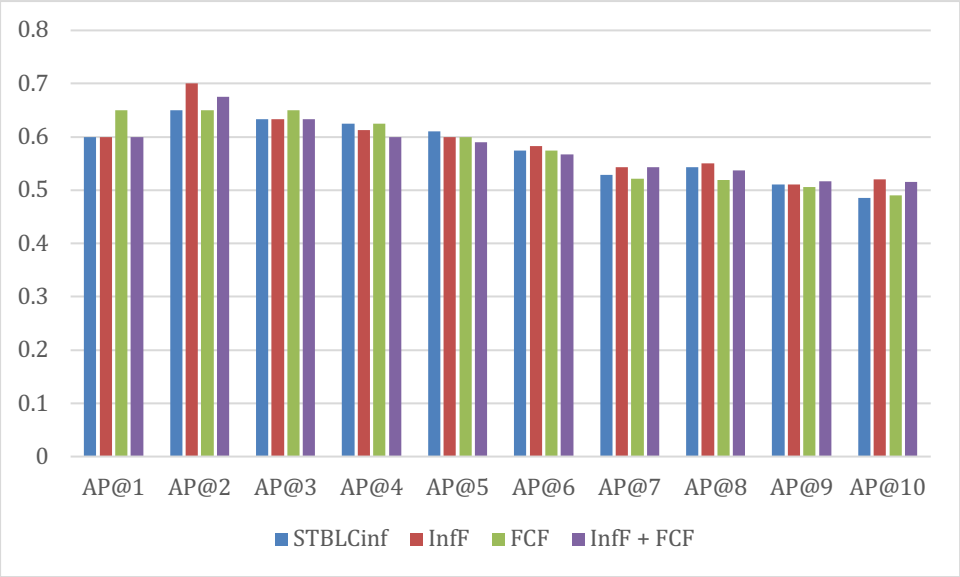

Figure 91. AP@top-10 values of implicit profiles of less active users.

The above results show that the InfF profile achieved better performance than the other profiles when the top 10 recommendations were recommended. The next step was to find which profile performs better in recommending all relevant recommendations. The MAP metric was used in this step; Figure 92 shows the results. Surprisingly, the InfF and FCF profiles achieved better performance than the STBLCinf profile, and the former achieved a slightly higher MAP than the latter. The mixed profile achieved a lower MAP value than the STBLCinf, and this might clarify that mixing the best performing profiles can reduce the accuracy of the performance. All profiles performed much better than the baseline.

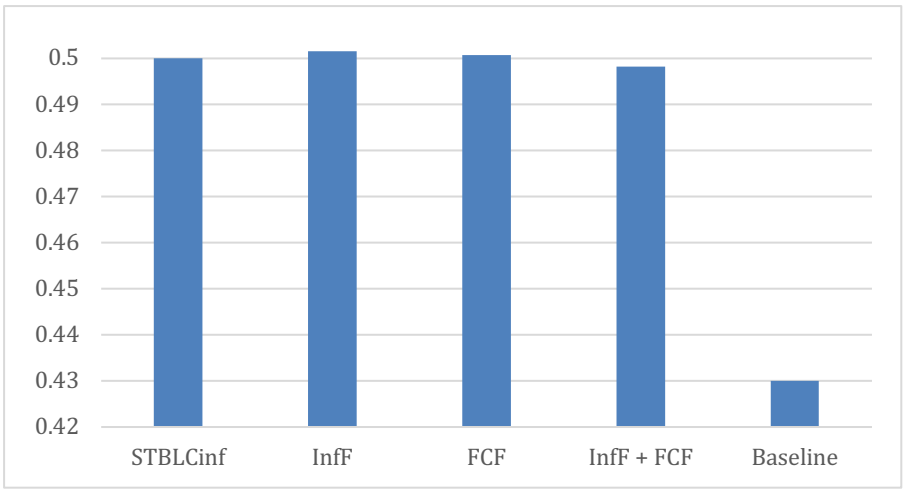

Figure 92. MAP values of implicit profiles of less active users.

In the group of active users, the performance was measured in terms of both delivering the top 10 recommendations and retrieving all relevant items. The AP@top-k metric was used to measure the performance of all profiles in recommending the top 10 recommendations (see Figure 93). A comparison of the InfF and FCF profiles shows that the latter outperformed the former six times, and this was double the InfF profile, which outperformed the FCF only three times. In comparison to the STBLCinf profile, the mixed profile surprisingly outperformed the STBLCinf four times, more than the original two profiles, while the InfF outperformed it three times and the FCF just once. Generally, the STBLCinf profile outperformed all other profiles four times, whereas the InfF outperformed others only once, when the k value was 4.

Figure 93. AP@top-10 values of the implicit profiles of active users.

In measuring the performance of the profiles in retrieving all relevant recommendation items, the MAP metric was used. The results show that generally the FCF profile achieved better performance than all other profiles. The STBLCinf came second and the baseline came third. The profiles InfF and InfF + FCF achieved lower MAP values than the baseline. The mixed profile achieved a better performance than the InfF profile. See Figure 94.

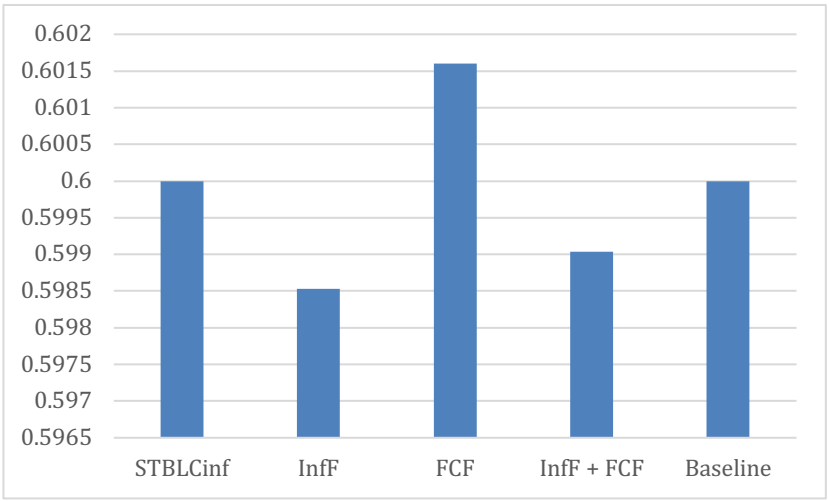

Figure 94. MAP values of implicit profiles of active users.

4.3.3.2.2. Originality

In this category, the users were divided based on their originality, i.e. whether they are receivers or producers. After that, the performance of stage 3 profiles was measured and then compared to the STBLCinf profile. The metrics used in this category were AP@top-k and MAP.

As explained previously, receivers are the users who retweet tweets more than posting tweets. Figure 95 shows the result when the stage 3 profiles were compared with each other and against the STBLCinf profile using the metric AP@top-10 recommendations. The general comparison results show that the InfF profile had better performance than the other profiles, even the STBLCinf profile. It achieved the highest AP@top-10 recommendations twice, which is better than the STBLCinf, which outperformed the others only once. When the InfF and FCF profiles were compared against each other, the former outperformed the latter five times, whereas the latter did not achieve any better performance. The mixed profile achieved better performance than the FCF only five times, and did not achieve any better performance when compared against the InfF. When all profiles' performance was compared individually against the STBLCinf, it was proved that the profiles InfF, FCF and InfF + FCF achieved higher AP values seven, four and five times, respectively. In recommending the top 10 recommendations, the results show that the profiles performed better than the STBLCinf profile for users who are receivers.

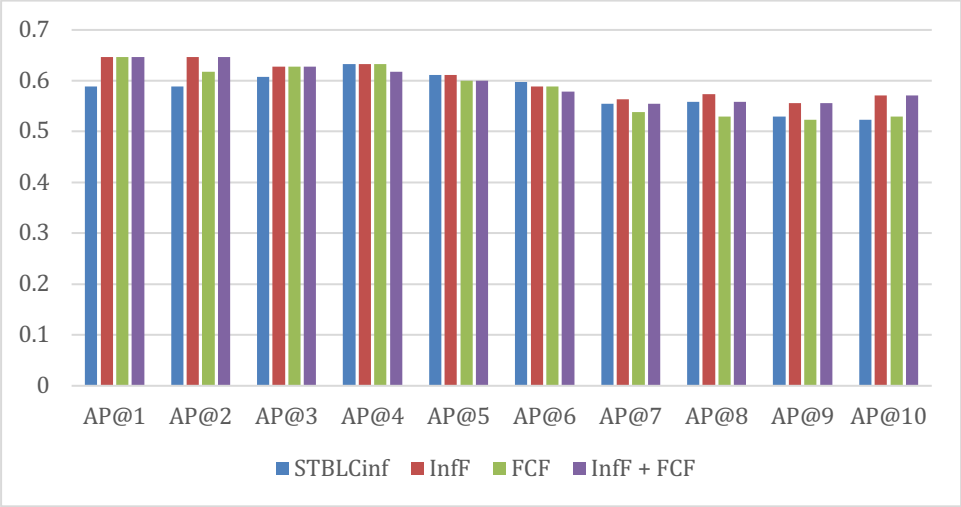

Figure 95. AP@top-10 values of implicit profiles of receivers group.

When the MAP metric was used, the aim was to test the profiles' ability to maintain their good performance in retrieving all relevant recommendation items. Figure 96 shows the results of the evaluation, and it is clear that all profiles outperformed the STBLCinf profile. The InfF profile achieved the best performance on the MAP metric. The FCF gave the second-best performance and the mixed profile came in third place. As a result, the profiles proved that they are capable of performing better than the STBLCinf profile in recommending the top 10 recommendations and retrieving all relevant recommendation items. The InfF profile was the best performing profile for users who are receivers.

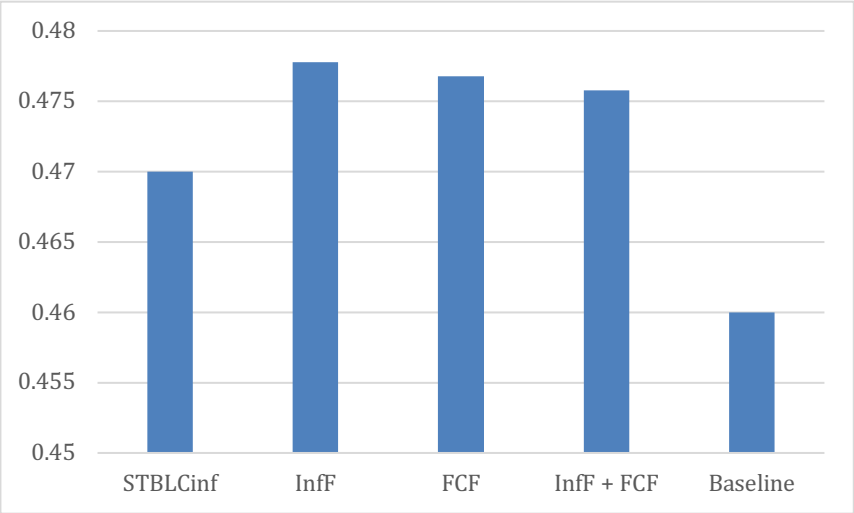

Figure 96. MAP values of implicit profiles of receivers group.

A producer user is a person who posts tweets more than they retweet. In this criterion, the performance of the stage 3 profiles was evaluated using the two metrics used on the receivers group, AP@top-10 and MAP. The profiles' performance was also compared against the STBLCinf, which is the original version of the profiles.

Figure 97 shows the evaluation results when the metric AP@top-10 recommendations was used. In a general comparison between all profiles, the results show that the FCF profile outperformed all other profiles twice, when the k values were 9 and 10. The results of a comparison of the InfF and FCF profiles shows that the latter outperformed the former seven times, whereas the former outperformed only once, when the k value was 4. The mixed profile

outperformed only the InfF profile, four times, and the latter outperformed the former only two times. When the profiles were compared to the STBLCinf individually, the InfF and InfF + FCF profiles were not able to achieve a higher AP value in any top-k recommendations. However, the FCF achieved higher performance than the STBLCinf two times.

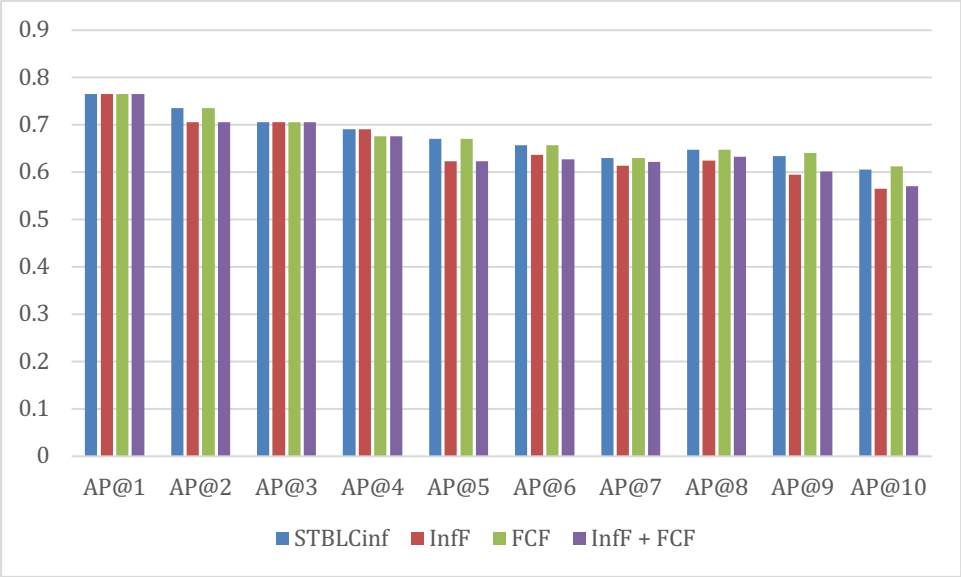

Figure 97. AP@top-10 values of implicit profiles of producers group.

Previous figures and results show that the FCF profile achieved a higher performance than the others when the top 10 recommendations were delivered. Therefore, its overall performance needed to be evaluated by the MAP metric, in order to find whether it was the best performing profile for the producers group. Figure 98 shows that the FCF profile achieved a higher MAP value, and therefore outperformed all other profiles; it was slightly higher than the STBLCinf profile.

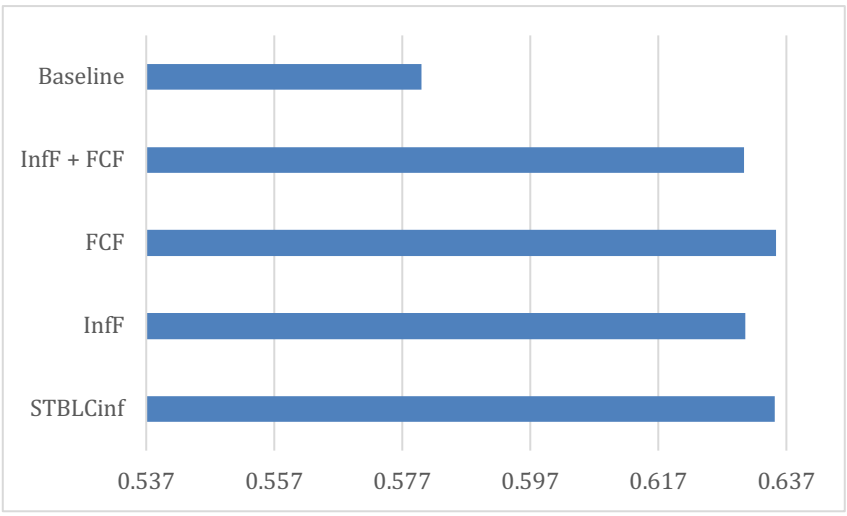

Figure 98. MAP values of implicit profiles of producers group.

4.3.3.2.3. Friend count

The users in the dataset were split into four groups in this criterion, based on the number of friends that the user follows. The range of the number of friends in each group was clarified previously. In each group, the performance of the stage 3 profiles was measured by the AP@top-10 and MAP metrics. The former measures performance in delivering more accurate top 10 recommendations and the latter measures the ability of such a profile in retrieving all relevant recommendation items.

In this group, the users follow a number of friends ranging from one to 171. Figure 99 shows the AP values of each profile's performance in this group when the top 10 recommendations were recommended. In a general comparison, the FCF profile outperformed the other profiles four times, while the InfF profile outperformed the others only one time, when the k value was 2. Surprisingly, the FCF profile was the only profile that had the highest AP value when the first item was recommended. In other words, it was more accurate in recommending the first item to users. When the FCF profile was compared to the InfF profile, the former was able to achieve higher AP values than the latter eight times. When all profiles were compared against the STBLCinf, which is the original version of all profiles, the FCF profile was able to outperform the others four times. The InfF outperformed the STBLCinf twice, whereas the latter

outperformed the former seven times. The STBLCinf outperformed the mixed profile seven times, it failed once and they were equal two times.

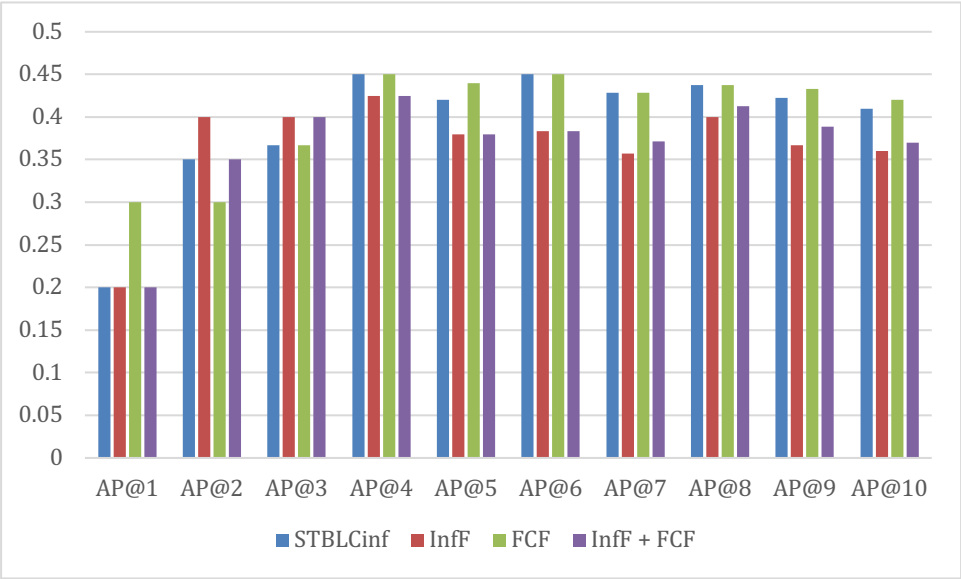

Figure 99. AP@top-10 values of implicit profiles of group 1.

The previous results show that the FCF profile was more accurate in recommending the top 10 items to the users of this group. Therefore, its performance needed to be evaluated using the MAP metric in order to have a clear view of its overall effectiveness. Figure 100 shows that the FCF continued its outstanding performance by achieving the highest MAP value compared to the other profiles. The STBLCinf profile achieved a lower value than the FCF but it achieved a slightly higher value than the InfF and the mixed profiles.

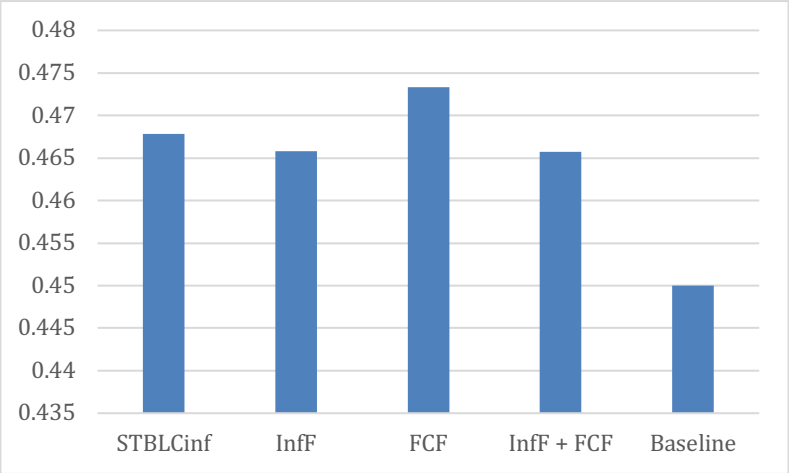

Figure 100. MAP values of implicit profiles of group 1.

In this group, the users follow a number of friends ranging from 172 to 300. The AP@top-10 metric was used to measure how accurate each profile is in delivering more relevant items within the top 10 recommendations. As Figure 101 shows, all profiles achieved the same high AP at k = 1 2. The InfF and InfF + FCF profiles outperformed the other profiles four times. The InfF profile outperformed the other profiles two times and the FCF profile achieved the highest AP value only once, when the k was 3. Comparing the InfF and FCF profiles, the former outperformed the latter six times and the latter achieved the highest AP value only once. In this group, the mixed profile performed better than the FCF, as it achieved a higher AP five times. The performance of the STBLCinf profile was not good enough for it to be more accurate than other profiles.

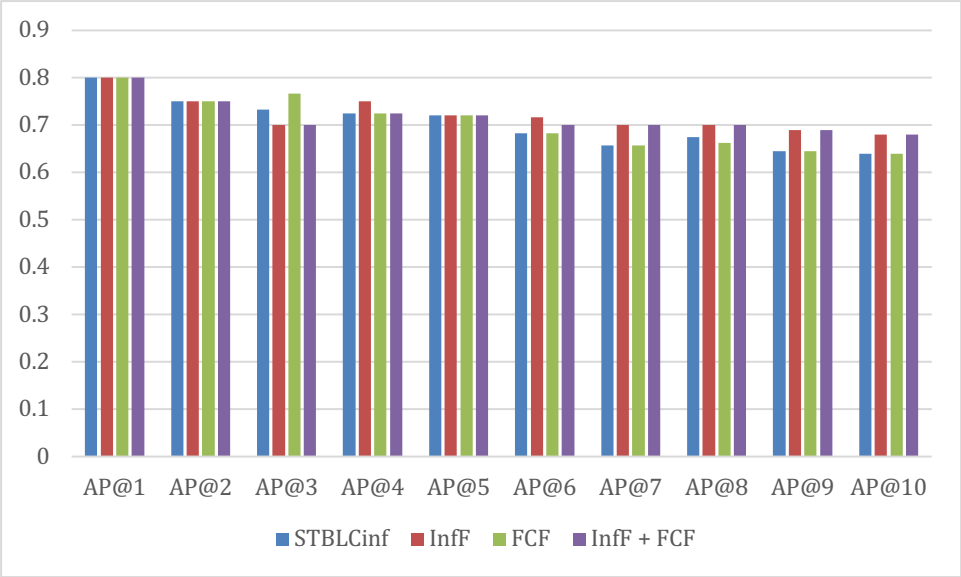

Figure 101. AP@top-10 values of implicit profiles of group 2.

The above figure showed that the InfF profile was slightly better than the other profiles in delivering more accurate items to the group members. In order to evaluate the overall performance, the MAP metric was used in this step, and the results are shown in Figure 102. Just like they had the best performance in the AP@top-k metric, the InfF and InfF + FCF profiles achieved higher performance than other profiles here too. The former was slightly better than the latter. However, the FCF profile achieved a lower MAP value than the STBLCinf and baseline profiles.

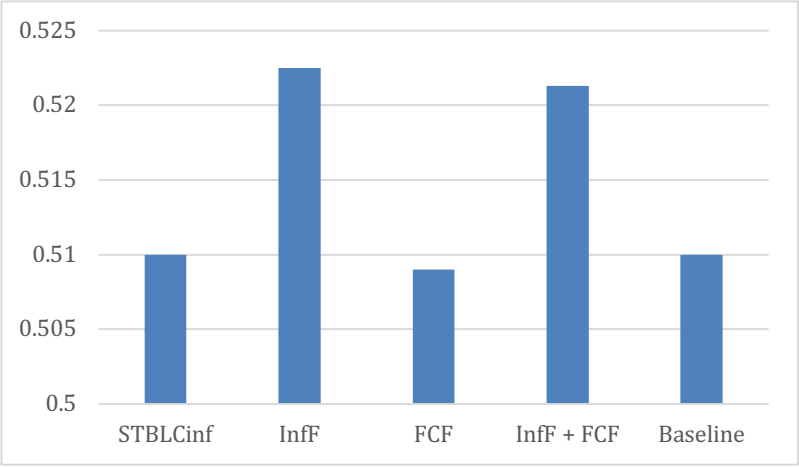

Figure 102. MAP values of implicit profiles of group 2.

As clarified before, the users in this group follow a number of friends ranging from 300 to 440. In the metric of AP@top-10 recommendations, as Figure 103 shows, all profiles achieved the same accuracy in recommending the first element of the top 10 recommendations, as they all had an AP of 0.80. Generally, the more accurate profiles in this group were InfF and STBLCinf, as they outperformed the others three and two times, respectively. Comparing them, as they were the best, each one outperformed the other three times and they were equal four times. When both InfF and FCF profiles were compared, the results show that the former achieved a higher AP value six times, whereas the latter achieved it only twice. The mixed profile was slightly better than the FCF profile and worse than the InfF profile.

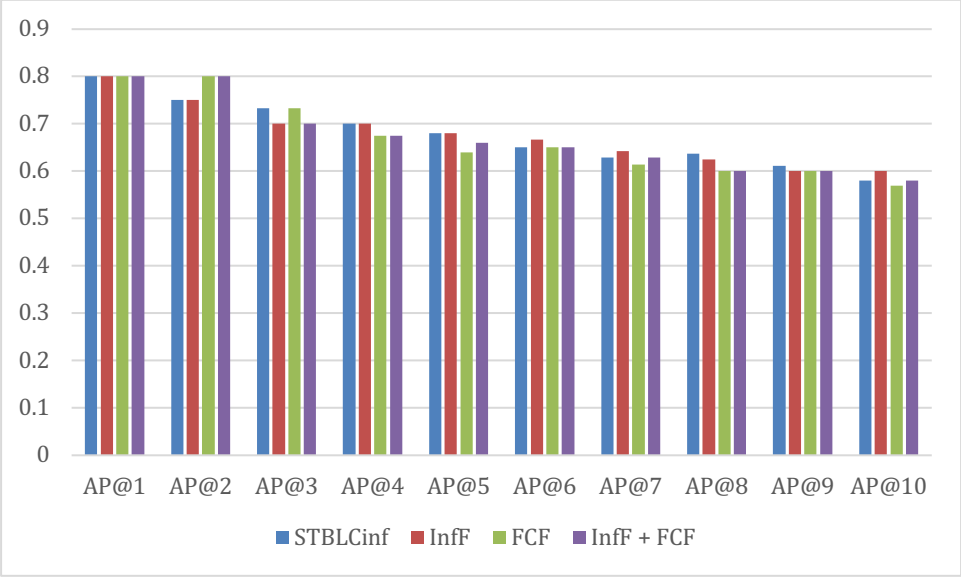

Figure 103. AP@top-10 values of implicit profiles of group 3.

As the above results show, both the InfF and STBLCinf profiles did well in recommending more accurate items to the user within the top 10 recommendations. It was necessary to measure their overall performance using the MAP metric. Figure 104 shows that both achieved the highest MAP values and their performance was similar. The STBLCinf exhibited a slightly better performance than the InfF profile. The FCF and InfF + FCF profiles exhibited similar performance, and both achieved better values than the baseline profile.

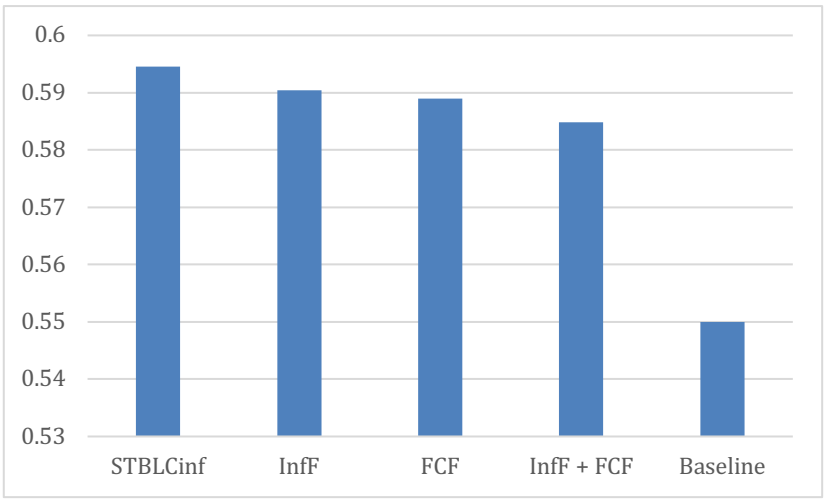

Figure 104. MAP values of implicit profiles of group 3.

The users in this group follow a number of friends ranging from 441 to 800. The performance of the profiles in this group was measured using the two metrics, AP@top-10 and MAP. In the AP@top-10 recommendations, the results in general show that all profiles had similar performance; the STBLCinf profile was the only profile that outperformed the others, doing so two times when the k value was 6 and 9. The InfF and InfF + FCF profiles achieved the most accurate performance in recommending the first element to the users, as they achieved 90% accuracy. When the performance of InfF and FCF profiles were measured and compared, the results show that the former outperformed the latter six times, whereas the latter achieved the highest AP value only once.

When each profile is compared individually to the STBLCinf profile, the InfF profile outperformed the former five times, and the former outperformed the latter only twice. The FCF profile outperformed the STBLCinf profile two times, and the latter achieved the highest AP value five times. The mixed profile achieved a higher performance five times, more than the STBLCinf, which outperformed it four times. See Figure 105.

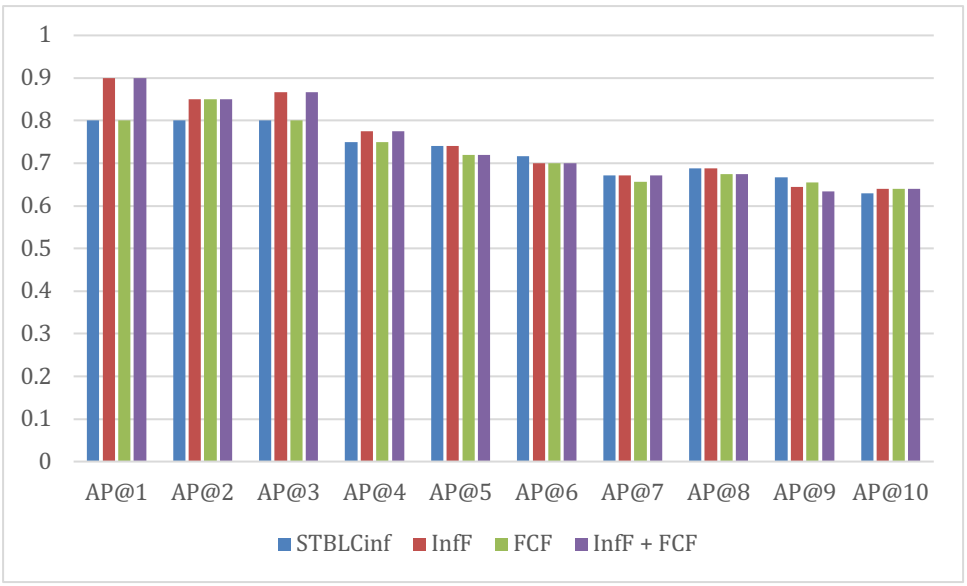

Figure 105. AP@top-10 values of implicit profiles of group 4.

The profiles' overall performance was measured using the MAP metric, and the results are shown in Figure 106. Surprisingly, the FCF, the performance of which in delivering the top 10 recommendations was not convincing, achieved the best performance according to the MAP metric. It also outperformed the STBLCinf profile, which was the only profile that outperformed the others in recommending the top 10 recommendations. The mixed profile achieved slightly better performance than the InfF profile, and this might be caused by the quality of the tweets that the FCF profile has.

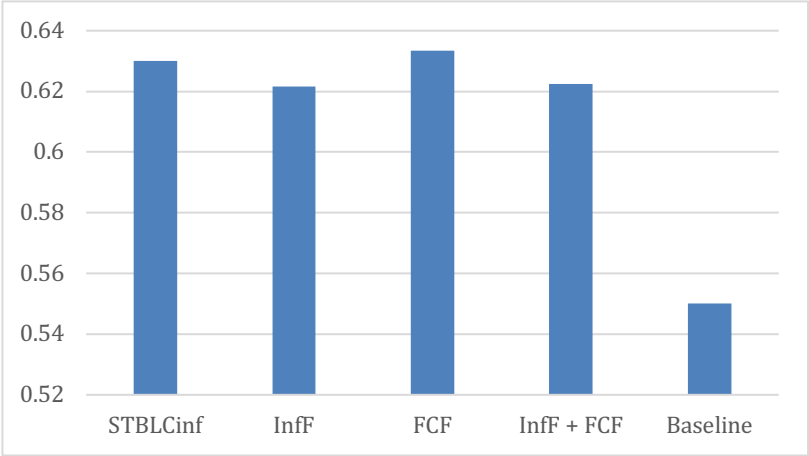

Figure 106. MAP values of implicit profiles of group 4.

4.3.4. Combining profiles from all stages based on influence

4.3.4.1. General results

The aim of this step was to discover whether the performance of the recommender system improves when all profiles are combined into one profile. The combined profiles are those which were built based on the proposed influence rule in all different stages. These profiles are: STBLCinf, Inf of Inf and InfF. The profile that contains all tweets from all mentioned profiles was named EXIM (explicit and implicit profiles). The performance of this profile was compared against the original profiles.

In the metric of AP@top-10, as Figure 107 shows, the EXIM profile did not achieve any better performance against the other profiles. However, the Inf of Inf profile outperformed all other profiles four times. The STBLCinf profile outperformed the others three times, while the InfF profile outperformed the others only twice. Generally, the Inf of Inf profile was better at delivering top 10 recommendations, whereas the EXIM was the worst.

Figure 107. AP@top-10 of all profiles based on the influence rule across all stages.

According to the MAP metric, and surprisingly, as Figure 108 shows, the Inf of Inf achieved the worst performance. The EXIM profile was slightly better than the Inf of Inf one. The STBLCinf profile achieved the highest MAP value. It was slightly better than the InfF profile in retrieving all relevant recommendation items.

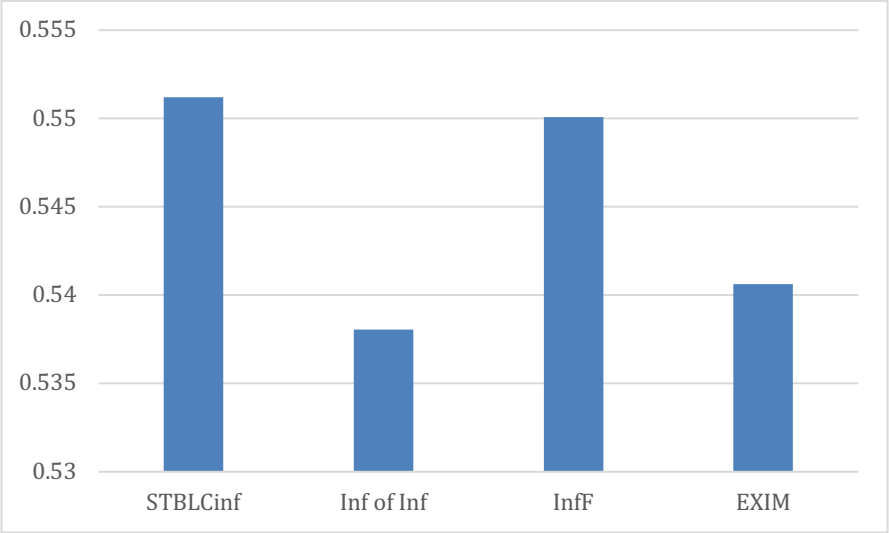

Figure 108. MAP values of all profiles of all stages.

4.3.4.2. Dividing the dataset

Similar steps in dividing the dataset were applied at this stage in order to discover which profile is stronger than others when the dataset is divided into the categories of activity and originality. This step was necessary in order to have a clear view of the results and the effectiveness of each profile in each category.

4.3.4.2.1. Activity (engagement)

In this category, the users were divided into two further categories, less active and active users, and the same steps were applied as at previous stages. Each profile's performance was then compared against each other in both groups.

In regards to the less active users, as Figure 109 shows, the InfF profile outperformed all other profiles four times in the metric of AP@top-10. The STBLCinf profile came in second place by outperforming others two times. The Inf of Inf profile was able to outperform the others

only one time, when the k was set to 3. The EXIM profile achieved the worst results in recommending the top 10 items.

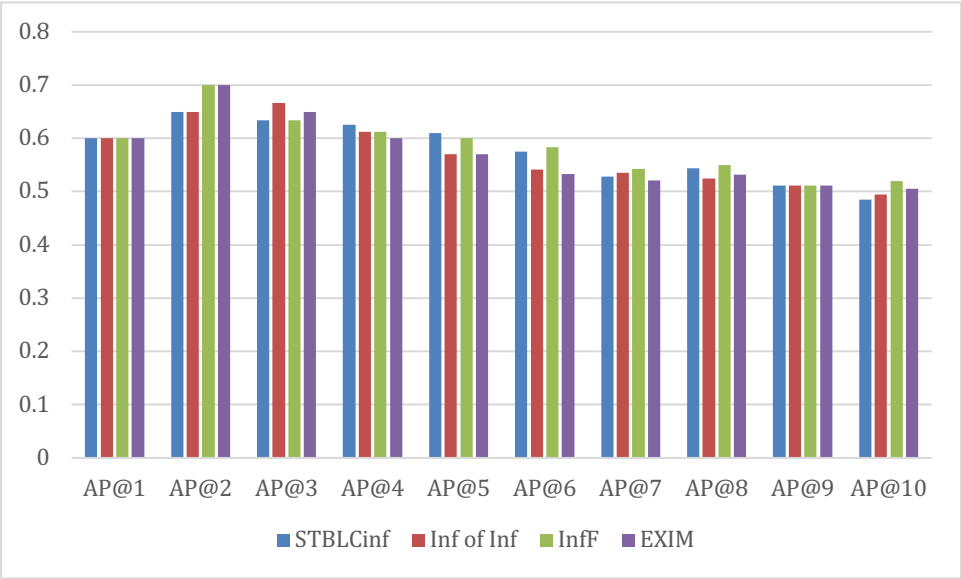

Figure 109. AP@top-10 of profiles of less active users.

In regards to the metric of MAP, as Figure 110 shows, the InfF profile continued to achieve the best performance by achieving the highest MAP value and retrieving all related items better than the others. The STBLCinf profile was the second-best performer. The Inf of Inf profile achieved the worst performance in retrieving relevant items, while the EXIM profile achieved a better MAP value than the former.

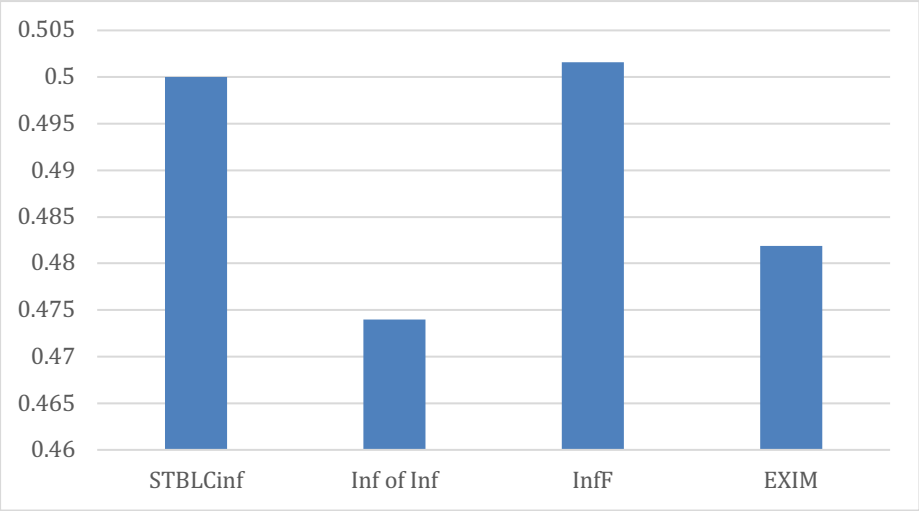

Figure 110. MAP values of profiles in the less active users group.

In regards to the active users, as Figure 111 shows, the Inf of Inf profile achieved the best performance in the AP@10 metric by outperforming the other profiles three times. The other profiles, i.e. STBLCinf, InfF and EXIM, achieved similar performance by outperforming the others only once. The InfF profile was able to achieve better performance in recommending the first recommendation item (AP@1).

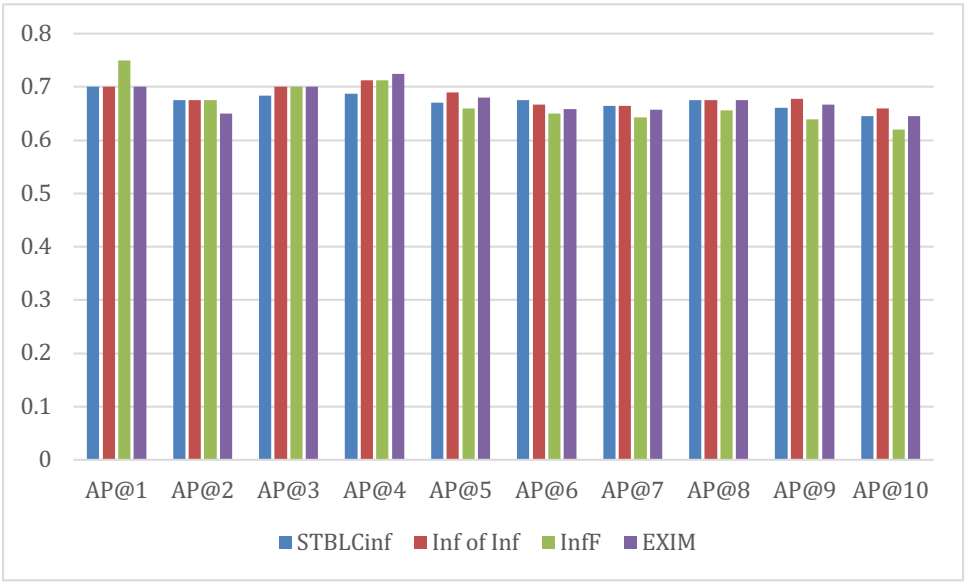

Figure 111. AP@top-10 of profiles in the active users group.

In regards to the MAP metric, the Inf of Inf profile achieved the best performance in retrieving related items in comparison to the others. The STBLCinf profile came in second place and the EXIM came in third place. The InfF profile achieved the worst performance in retrieving all related recommendation items. See Figure 112.

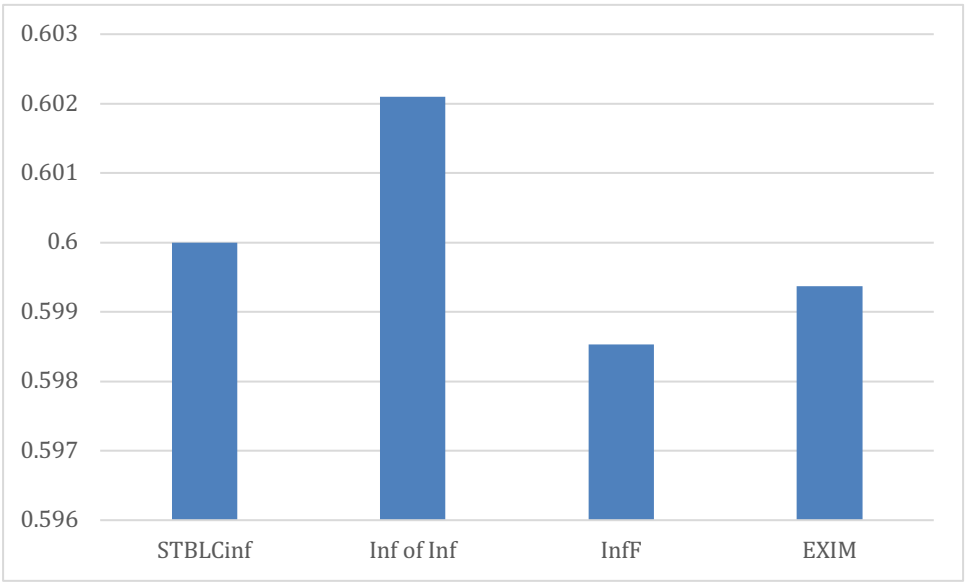

Figure 112. MAP values of profiles in the active users group.

4.3.4.2.2. Originality

In this category, the users were divided into two further categories, receivers and producers, and the same steps were applied as at previous stages. Each profile's performance was then compared against the others in both groups.

In the receivers group, as Figure 113 shows, the InfF profile achieved the best performance by outperforming others three times in the AP@top-10 metric. The EXIM profile, which combines all the other profiles, came in second place by outperforming the others twice. The STBLCinf and Inf of Inf profiles achieved similar performance by outperforming the others only one time each.

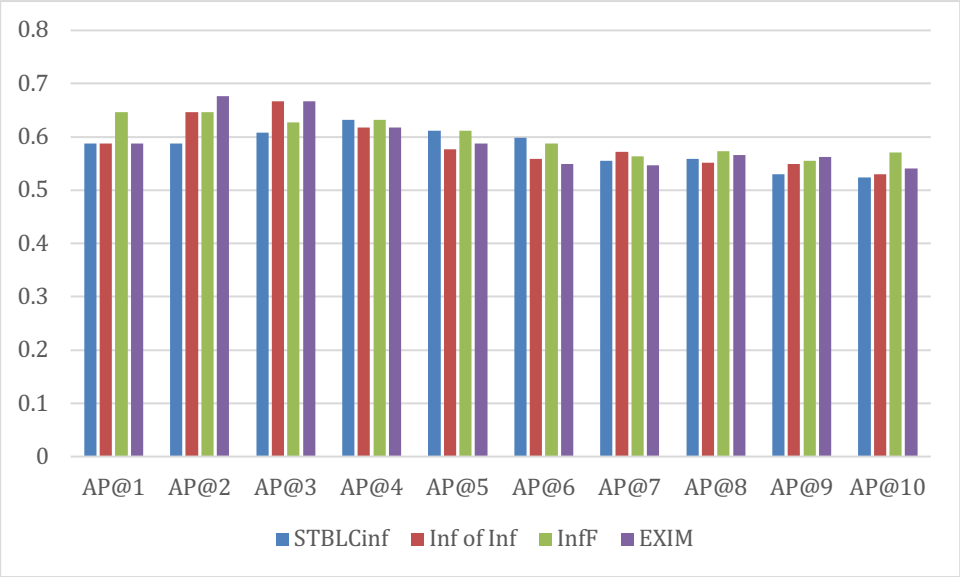

Figure 113. AP@top-10 values of profiles in the receivers group.

In regards to the MAP metric, as Figure 114 shows, the InfF profile continued achieving better performance in retrieving all related items. The STBLCinf profile achieved a slightly higher MAP value than the EXIM profile. The Inf of Inf profile's performance was the worst against all other profiles.

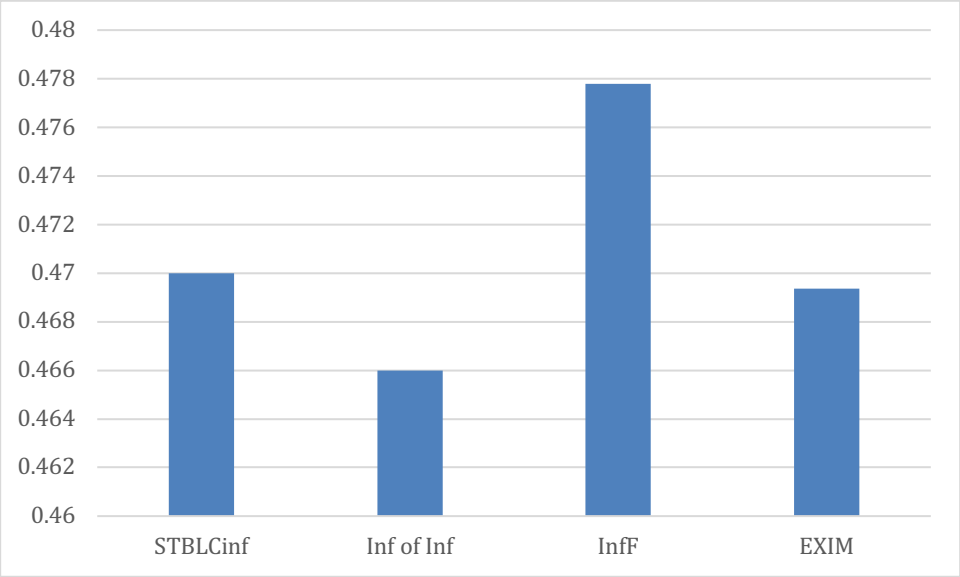

Figure 114. MAP values of profiles in the receivers group.

In the producers group, as Figure 115 shows, the STBLCinf and Inf of Inf profiles achieved the best performance in the AP@top-10 metric by outperforming the others two times each. The InfF and EXIM profiles were not able to show any better performance in recommending the top 10 recommendations.

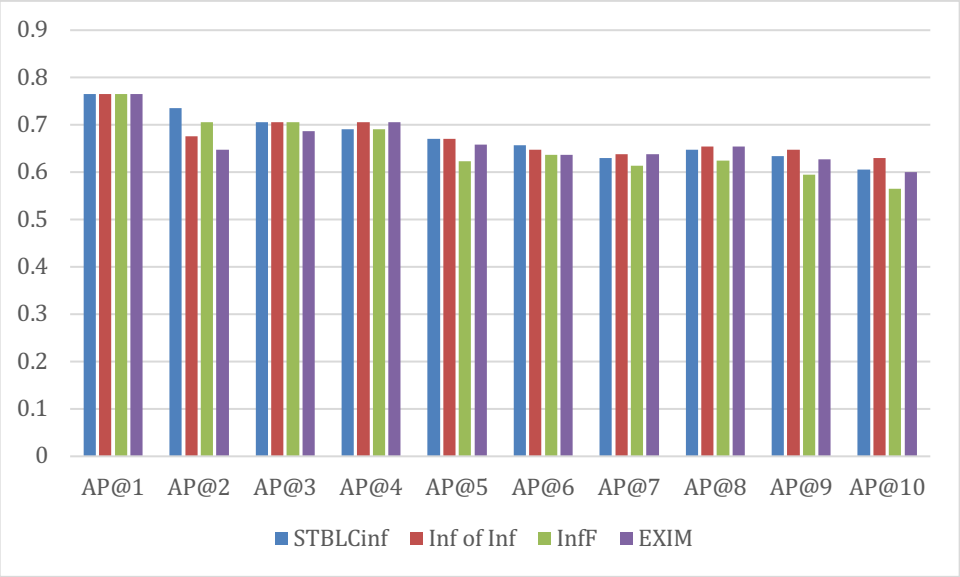

Figure 115. AP@top-10 values of profiles in the producers group.

In regards to the MAP metric, as Figure 116 shows, the STBLCinf profile outperformed all other profiles by achieving the highest MAP value. Surprisingly, the Inf of Inf profile, which was performing similarly to the STBLCinf, achieved the worst performance, and the InfF and EXIM profiles achieved higher values than the Inf of Inf.

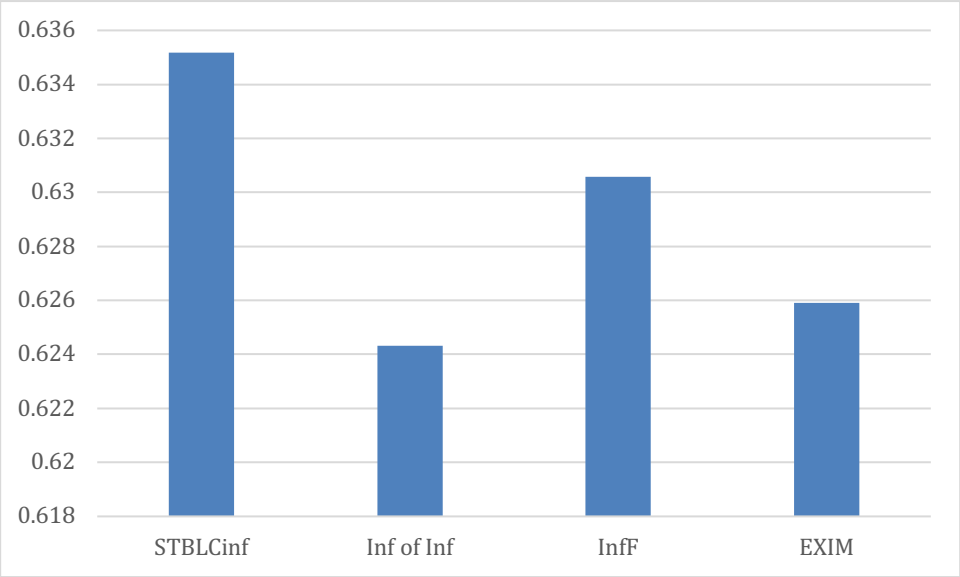

Figure 116. MAP values of profiles in the producers group.

4.4. Discussion

4.4.1. Direct explicit relationships

These result show that exploiting tweets of influential friends, with which the user has explicit direct relationships, within the short term, can form a strong profile that can deliver more accurate recommendations, and this performance was measured by using the AP@top-k and MAP metrics. The former was used to measure how good the profile is in recommending relevant items within the top-k recommendations, and the latter was used to measure how good the profile is at retrieving and ordering relevant recommendation items to the user. The importance of this research is in finding the influential members of the network around the user that are explicitly and directly connected to the user, and then using their tweets to build the user's profile from. The finding that recent activities can form a better user profile replicates the findings of modelling the user in a short-term profile in the study by Abel et al. (2013). In other words, two different profiles were built in our experiment in order to see which profile performs better, BLCinf or STBLCinf. The former contains all timeline tweets, tweets from influential users and tweets that were classified as representative (retweetable), while the latter differs in terms of timeline tweets as it only contains the short-term timeline tweets. The results show that profile with only recent activities performs better than the profile that contains old data by 1.4%% in terms of MAP and 70% in terms of AP@top-k. The other finding, that enriching a user profile with recent data can improve the quality of the profile, replicates the findings of using a decay function in long-term profile, which gives higher weight to recent tweets than old ones, being able to deliver better recommendations than the profile without the decay function, as in the study by Piao and Breslin (2016). Similarly, in our experiment, the BLCinf profile, which is the baseline profile that contains timeline tweets, enriched by short-term tweets from influential friends, performed better than the baseline profile in recommending top 10 items by 90% and in delivering all related items by 1.5%.

When the friends of a user are divided into three groups, namely influential, less influential and non-influential friends, the aim was to test whether this classification plays an important role in the quality of the profile, and to see if irrelevant data affects the profile and thus the

recommender system's performance. One of our findings as that the profile that only has carefully chosen tweets from such sources is better than the profile that contains all tweets from the same sources. In contrast, when the classification is applied to tweets from less influential friends, this step can raise the profile quality and then deliver more accurate recommendations than the profile that has all tweets from the same sources without considering the classification. Two profiles were built similarly in our experiment, STBLCinf and STBLinf. The difference between them is that the former applies classification to tweets of less influential friends and stores only the tweets that are classified as retweetable, and the latter stores all tweets without any consideration of classification. The results show that STBLCinf profile outperformed the other by 100% in the AP@top-k metric and had a better performance by 4.5% in the MAP metric. Surprisingly, the STBLinf profile had 22% more tweets and achieved the worst performance against all other profiles, even worse than the baseline. This extra number of unclassified tweets can affect the quality of the profile and make it sparse, and therefore affect its performance. The other two profiles in our experiment were built similarly but the difference was in taking the classification into account, and these profiles are BLCinf and BLinf. Both profiles are an extended version of the baseline and both contains tweets by influential friends. The difference is in classifying the tweets of less influential friends. The results show that the BLCinf profile, which applied classification to less influential friends, achieved better performance by 80% in the AP@top-10 metric and 1.8% in MAP. In other words, the unclassified tweets reduced the quality of the profile. This idea might be applied to testing other findings from the literature, as most of them found that using external sources, for instance URLs in tweets, to enrich a user profile can achieve better performance than a profile built from Twitter's timeline (Alonso et al., 2010; Garcia Esparza, O'Mahony and Smyth, 2013). Enriching the profile from external sources can cause a reduction in quality of the profile, as the results show for the STBLinf and BLinf profiles. Karidi, Stavrakas and Vassiliou (2016) argue that URLs might link to useless content that might affect the performance of a system.

Our proposed algorithm for calculating the influence score of each one of the user's friends proved its quality in identifying influencers from the user's perspective. These influencers then become a primary source for building the user's profile, as their tweets play an important role

in achieving better recommendations. To evaluate the quality of our algorithm and those taken from the literature, other influence profiles were built similar to the STBLCinf profile, but the difference is that each profile was built based on its influence metric. The friends list was divided into three groups, influential, less influential and non-influential, based on the score on the influence metric of the algorithm. For example, in the follower count profile, friends were divided based on their number of followers. The candidate profiles were: follower count, follower rank, signal strength and retweet impact. The STBLCinf profile, which is based on our proposed influence algorithm, outperformed all other influence metrics from the literature. Furthermore, it was better by 100% in the AP@top-10 metric and 5.6% better than the closest profile, which was follower count.

Other popular similarity and distance measurements were used in order to test our influence algorithm. Various profiles were built in the same way as the STBLCinf profile, but the difference was in dividing friends based on the score of each measurement. The metrics used were cosine, Jaccard, Euclidean and Manhattan. The result show that our algorithm was better by 40% in the AP@top-10 metric and 1.6% better than the closest profile, which was the Euclidean.

In order to test our proposed method of building the user profile and also to understand the key strengths of our technique, the dataset was divided into different categories: activity, originality and number of friends. This step aimed to discover which category plays an important role in our strong profile. In each category, the performance of the proposed profile was evaluated against the baseline profile, influence metrics from the literature and the similarity metrics. In the activity category, users were classified into active and less active. In the less active group, the results show that the STBLCinf profile works well against the influence metrics from the literature and the similarity metrics, as it achieved better performance than them in the metric of AP@top-10 by 100% and 60%, respectively. It also outperformed all other profiles in the MAP metric. In the active users group, the proposed method achieved better performance in the AP@top-10 metric than the baseline and the literature profiles by 60%. However, it did not achieve the best performance against the similarity and distance profiles. In

the MAP metric, it achieved the same value as the baseline and outperformed other similarity and distance profiles.

In comparison of the performance of the proposed profile in the active and less active groups, the results show that the performance in the MAP metric was better by 10% in the active users group. The same comparison was done to the baseline as results showed that it achieved a better performance in the active group by 17%. Moreover, it increased more than the proposed profile by 7%. In the active users group, the proposed profile might have achieved a higher value in performance, but this does not make it the best among the others. Less activity with a lower performance value might make it the best performing profile among others. Also, this might mean that the baseline profile in the active users category has only tweets from within a short-term time, and therefore they help it to achieve similar performance to the proposed profile. It is known that Twitter allows developers to collect only the last 3200 tweets. For active users, this might be just a couple of weeks in the past. To see the difference between the least active user and the most active user in the dataset, the account age of each user was calculated by subtracting the date of tweet number 3200 (the oldest retrieved tweet) from the date of tweet number 1 (the latest retrieved tweet). The account age of the most active user is 17 days, and 2738 days for the least active user. This explains why the baseline profile's performance was better with active users.

The dataset was also divided into receivers and producers, based on the originality score of the users. This step was to find if there was any correlation between originality and the outstanding performance of the proposed profile based on our proposed influence score algorithm. In the receivers group, the proposed profile was better in achieving the highest AP@k values in 60% of the cases in comparison to the baseline and the profiles from the literature. It also outperformed all literature profiles and the similarity profiles in the MAP metric. When the profile was tested in the producers group, it achieved the highest AP@top-10 values in 80% of cases in comparison to the baseline and the literature profiles. It also outperformed the baseline, literature profiles and similarity profiles in the metric of MAP. The profile in both groups achieved better performance and it seems that originality has no effect on the performance of the proposed profile. However, the distribution of active and less active users in each group has

to be taken in consideration. The analysis shows that a majority of the receivers group are the less active users by 71%, whereas the active ones form only 29%. On the other hand, the producers group has 76% active users and 24% less active ones. The receivers group was cleaned from active users in order to make it 100% receivers who are less active. The less active users were also removed from the producers group. After that, the evaluation was applied to both groups. In the MAP metric, the proposed profile was better than the baseline by 1% before cleaning the receivers group of active users. The profile achieved better performance than the baseline by 2% after removing active users. The proposed profile was better than the baseline by 4.7% before removing less active users from the producers group. The proposed profile achieved equal performance with the baseline after removing less active users. This step proves that activity is significant in terms of improving the proposed profile's performance, and that originality likely has little effect. In contrast, the results show that the profile achieved better metric values in the producers group, which has active and less active users, than in the receivers group. It was better in the producers group by 20% in AP@top-10 and 3.7% in MAP.

The tested users were then divided into four groups based on their number of friends (following list members). This step aimed to discover any relationships between the performance of the proposed profile and number of friends. One of our findings is that the performance of the proposed profile increases when the number of friends increases. This increase could be caused by the activity and originality distribution in each group. Another thing to notice in our dataset is that activity decreases when the number of friends increases. The percentage of active users in group 1 is 60%; this decreases to 50% in groups 2 and 3, and to 40% in group 4. The proposed profile was able to outperform the baseline, literature and similarity profiles in all groups except group 2 in the MAP metric, as it achieved the same performance as the baseline and the Euclidean profile. Therefore, there might be a connection between the performance of group 2 and activity and originality. In terms of activity distribution, groups 2 and 3 share the same percentage of active and less active users, as both have 50% active users and 50% less active users. The distribution of originality varies between groups; group 2 has 33% producers and 67% receivers, while group 3 has 63% users who are producers and 37% users who are receivers. This explains why the performance of the proposed profile

was similar to the performance of the baseline and Euclidean profiles in group 2, whereas it was better in group 3. As highlighted before, a group with more producers shows improved performance. In addition, and in this case, the active users and less active users share the same percentage in the group, and the important factor now is the distribution of originality in the group.

Group 1 and group 4 saw similar situations when the dataset was divided into receivers and producers based on originality. The majority of users in group 1 are active and producers, and this is similar to the producers group previously, whereas the majority of group 4 are less active and receivers, and this is similar to the receivers group.

4.4.2. Indirect explicit relationships

One of our findings from this stage is that in general the proposed profile can be enriched by other sources from the indirect explicit relationships around the user, and our proposed influence algorithm can help in finding the appropriate accounts in order to use their tweets in building the profile and then developing its performance. The results show that finding influential friends of influential friends can help in improving the performance of the profile in recommending more relevant items than the original profile, which is STBLCinf. In contrast, the Inf of Inf profile, which is an extended version of the STBLCinf profile that contains extra tweets from influencers of influential friends, was able to outperform the STBLCinf profile in 20% of cases in the AP@top-10 metric. In other words, it was found that the Inf of Inf profile can retrieve more relevant recommendation items within the top 10 recommendations. The TTA and Inf of Inf + TTA profiles were not good enough to achieve better performance in recommending top 10 items, and the reason might be that some of the extra added tweets had less relevance to the user than in the STBLCinf and Inf of Inf profiles. In other words, the sources of Inf of Inf are more relevant to the user than in the TTA profile. Also, if one of the TTA accounts is interesting to the user, then the user will follow it, and then it may become one of their influential friends. In the MAP metric that measures how good the profile is at retrieving all related recommendation items, the STBLCinf profile was better than the Inf of Inf profile by 1.1%.

In order to find how useful the profiles are at this stage and to determine the factors that may affect their performance, the dataset was divided here into the same categories as in the previous stage: activity, originality and number of friends. In the activity category, one of the findings in our experiment was that enriching the profile with tweets by influential friends of influential friends can achieve better performance when the user is active. Furthermore, the Inf of Inf profile was able to achieve better performance in recommending more related recommendations within the top 10 than others by 30% in the group of active users. It was also able to achieve slightly better performance than the STBLCinf profile in the MAP metric by 0.2%. However, the STBLCinf profile is more suitable for the group of less active users, as it was better at recommending the top 10 recommendations by 10% than the Inf of Inf profile. It was also better by 2.6% than the Inf of Inf profile in retrieving and ordering the related recommendations items. Despite coming behind the STBLCinf profile by 10% in recommending the top 10 items, the Inf of Inf profile achieved a lower MAP value than TTA by 1% in the less active group.

One of the findings in the category of originality is that the Inf of Inf profile achieved better performance than the STBLCinf profile in recommending the top 10 items, as it was better in 30% of cases in the producers group and 10% in the receivers group. This outstanding performance might be caused by the activity distribution inside each group, and, as seen previously, because the Inf of Inf profile works better with active users. The majority of the producers group are active users, forming 76% of the group members. As a result, the producers group was cleared of less active users and active users were removed from the receivers group. This step aimed to find whether performance was affected by the activity factor. The results show that the Inf of Inf profile achieved outstanding performance by achieving the highest AP values in recommending the top 10 items, as it was better in 80% of cases in the producers group. It was better by 20% in the receivers group. In the MAP metric, it was discovered that the Inf of Inf profile outperformed the STBLCinf profile in the producers group by 0.3%. However, the performance of the profile in the receivers group decreased by 0.9%.

4.4.3. Implicit relationships

One of the findings from this stage is that enriching the user profile with tweets from implicit sources, such as followers of influential friends or friends who are followed by a high number of followers, reduces the quality of the profile. The STBLCinf profile was able to outperform other profiles in the AP@top-10 metric, scoring 30% higher than the InfF profile, and also in the MAP metric. The results show that the InfF profile was better than the FCF profile by 30% in recommending more relevant items within the top 10 recommendations. However, the FCF was able to achieve a slightly better MAP value than the InfF. This good performance of the FCF profile is quite logical, as it might be that the machine learning classification excluded many tweets that are not relevant to the user, and this was clear in the average number of tweets in the profile. The results of cosine similarity between the user's timeline and other accounts' timelines show that most of the top 10 similar accounts come from InfF technique. In the Twitter API, the most recent followers are retrieved first. One of the limitations of this experiment is that developers cannot retrieve the oldest followers of an account that has a large number of followers. Twitter allows developers to retrieve only 200 followers, and after that it forces the developer to be on hold for 15 minutes, only then letting them continue to retrieve another 200 followers. This strategy from Twitter is time consuming for researchers. For example, it would take the researcher many years to collect all followers if a user follows the account of Liverpool football club, which has 11 million followers. Another limitation is that the Twitter API does not provide the date and time when the user started following an account, which could help in understanding the results clearly.

Other findings at this stage are that the implicit relationships profiles achieved better performance than the STBLCinf profile in some cases, when the dataset was divided into different categories. In the activity category and in the MAP metric, the InfF profile outperformed other profiles by achieving a higher value in the group of less active users. The FCF profile was able to outperform other profiles in the active users group. Both profiles were higher than the STBLCinf profile by 0.1%. In recommending more related items in the top 10 recommendations, the InfF profile was better than the FCF by 20% and better than the

STBLCinf by 30% in the less active users group. However, the STBLCinf was better than the InfF profile by 30% in the active users group.

In the originality category, it was discovered that the InfF profile works better in recommending more related items within the top 10 recommendations by 10% in the receivers group. It also outperformed all other profiles in the MAP metric, as it was better than the STBLCinf by 0.7%. This is because less active users form a majority of the receivers group, and this might mean that activity has an effect on performance. In the producers group, the FCF profile was able to deliver more related items within the top 10 recommendations by 20%. Moreover, it outperformed the STBLCinf profile by achieving a slightly higher MAP value by 0.02%. This also might be caused by the distribution of activity inside the group, as 76% of the members are active users. As seen previously, the FCF profile was able to outperform the others in the active users group. Surprisingly, the STBLCinf profile did not achieve any better performance in both groups in both metrics.

When the dataset was divided based on number of friends, the FCF profile was found to be more suitable for users who follow a small number of friends (group 1) and users who follow the highest number of friends (group 4) in the MAP metric only. The performance of the STBLCinf and InfF profiles was similar when users follow an average number of friends, as in groups 2 and 3. One of the findings is that the performance of stage 3 profiles in each group is affected by the distribution of activity and originality. For example, the FCF profile outperformed the InfF profile in 60% of cases in AP@top-10 and also had a higher MAP value. The group is formed primarily by active users and producers. When the group was shared equally between active and less active users, as in group 2, the InfF profile was better than the FCF profile by 50% in AP@top-10 metric and also outperformed others in the MAP metric. This is due to the formation of the group, as 67% of users are receivers and it was proved before that the InfF profile works better with receivers.

4.4.4. Combined profile against all profiles based on influence

When the combined profile from all stages (EXIM) was compared against all other profiles based on influence score, one of the findings in general is that combining all methods into a single profile does not perform better than the original profiles. In fact, it performed slightly less accurately than others in delivering top recommendations. It is also less accurate than others in retrieving all related recommendation items, except the Inf of Inf profile, which was less accurate by 0.02%. However, the Inf of Inf profile achieved better performance than the others in delivering the top 10 recommendations to users. The STBLCinf profile came second best in recommending the top 10 recommendations, and achieved the best performance in retrieving all relative recommendation items. It was clear from the results that the STBLCinf had a more balanced performance than the Inf of Inf profile as the former came in second place in recommending the top 10 items, and in the first place in retrieving and ordering all related recommendation items. The latter achieved first place in the AP@top-10 metric and the worst performance in the MAP metric.

When the dataset was divided based on activity, in the less active users group, the EXIM profile was the worst profile in recommending the top 10 recommendations, and the second worse profile in retrieving and ordering the related recommendation items. The InfF profile had the best performance against all other profiles in both AP@top-10 and MAP metrics. The STBLCinf profile was less accurate by 20% in the AP@top-10 metric and 0.01% in the MAP metric.

In the active users group, the EXIM profile achieved similar performance to the STBLCinf and InfF profiles, as they achieved 10% in recommending the top 10 recommendations. It achieved a lower MAP value than the STBLCinf profile by 0.01% and a higher value than the InfF profile by 0.01%. The Inf of Inf profile achieved the best performance in both recommending the top 10 recommendations and retrieving all related recommendation items. It was better than other profiles by 20% in the AP@top-10 metric and 0.002% in the MAP metric.

When the dataset was divided into receivers and producers based on originality, the EXIM profile came in second place in recommending the top recommendations in the receivers group. It was behind the InfF profile by 10% in the AP@top-10 metric and better than the STBLCinf

and Inf of Inf by 10%. On the other hand, and in the MAP metric, the InfF profile proved its ability to retrieve and order all related recommendation items better than the others. The STBLCinf profile came second, slightly better than the EXIM profile by 0.01%. The Inf of Inf profile was the least accurate profile in retrieving the related items.

In the producers group, the STBLCinf profile had a more balanced performance as it achieved the same AP@top-10 value as the Inf of Inf profile, and it achieved the highest MAP value against all profiles. The latter achieved the lowest MAP value in comparison to all profiles. In other words, the STBLCinf profile is more suitable for producer users in both recommending the top 10 recommendations and retrieving and ordering the related recommendations items.

5. Conclusion and Future Work

This approach has presented a novel way to improve the performance of recommender systems that are based on short-text data (tweets) by exploiting the network around the user. The data (tweets) of members of this network can help to enrich the user profile and raise the accuracy of the recommended items. Alongside the machine learning technique, our approach redefined the influence role in order to select the most important accounts to the user and then collect their data to use them in building the user profile. This approach has proved its strength in improving the performance of short-text-based recommender systems and in solving the lack of data in previous researches.

The influence rule in our approach was redefined based on the perspective of the user, and this algorithm was able to identify and rank reliable sources, and the user profile is built from their tweets. Its performance was compared against other algorithms from the literature such as influence, distance and similarity algorithms. The proposed influence algorithm was able to outperform them, showing outstanding performance. Additionally, it proved that the influence rule can vary between users; a celebrity for example can be an influencer for one user but not an influencer for another. Normal users can have more influence on a user than celebrities, for instance.

The proposed approach has several advantages. Firstly, it improves the performance of short-text-based recommender systems by exploiting the relationships between users within a network. Secondly, it solves the problem of lack of data in user activities. Many users do not provide reliable or enough tweets to help a recommender system in delivering interesting recommendations; our approach solves this problem. Thirdly, it enriches the user profile with reliable data from guaranteed sources such as influential accounts, whereas other approaches use external sources such as Wikipedia and URLs in tweets, meaning irrelevant data will be used in building the user profile. Our approach clarified how irrelevant data affects the accuracy of the recommender system, such as including tweets by non-influential users. Finally, our approach uses Twitter itself as a resource, and the user's recent activity, which was proved to improve the recommender system's accuracy.

In order to discover which factors our proposed work achieves the best performance with, the dataset was divided into categories: activity, originality and number of friends. In direct explicit relationships, the STBLCinf profile was able to achieve better performance than other algorithms from the literature in all categories. The activity factor has a very strong connection with our approach as it works better with less active users. In indirect explicit relationships, the profile of influential friends of influential friends achieved better performance with active users in all evaluation metrics. In implicit relationships, the profile built from the tweets of influential friends' followers achieved better performance with less active users. Activity distribution in the originality and friends categories plays an important role in the performance of the proposed profiles.

The profiles that are built based on the proposed influence rule were combined into one profile (EXIM), and this was then compared to other profiles, which are STBLCinf, Inf of Inf and InfF. The general results show that the Inf of Inf profile is more accurate in recommending the top 10 items, and the STBLCinf is more accurate in retrieving and ordering all relative items. When the performance of the profiles was tested on active and less active users, the InfF profile proved its ability to deliver more accurate recommendations and retrieve more related items for less active users. The Inf of Inf profile proved itself more accurate than the others in recommending top 10 items and retrieving related items for active users. When the users were divided into receivers and producers, the InF profile was able to prove more accurate recommendations and to retrieve all related items in the receivers group. In the producers group, the STBLCinf profile had more balanced performance in both the AP@top-10 and MAP metrics.

This novel work has contributed to current knowledge by proposing a system that provides recent users' profiles with more rich data that reflects their interests, and has proved its strength in raising the accuracy of recommendations. This rich data was collected through different resources using the advantages of the properties of explicit and implicit relationships.

5.1. Contributions to knowledge

Building effective profiles for recommender systems based on short-text activities, such as Twitter posts for example, face certain challenges in delivering more accurate recommendations to users. The main contribution of this thesis is the ability to build an efficient profile by exploiting the network around the user and then to recommend more accurate items and solve some limitations that are found in the literature. It was proved that recent Twitter users' activity reflects their current interests. However, many users do not provide enough data in their recent activity. This research was able to find an innovative technique to solve this problem by exploiting relationships between users within a network and then using its members' recent activity in building the profiles. Furthermore, explicit relationships among social media users were extracted and utilised, and their usefulness in improving the recommender system's performance was clear in the results. Moreover, the approach was extended to exploit the implicit relationships properties of a wider network around its users. This extension collaborated in improving the performance of the recommender system.

Another contribution is redefining the influence rule to be based on users' perspectives instead of influencers' perspectives. In the literature, naming a user as an influencer is a debatable topic, and there is no agreement as to how influencers should be defined. In this research, the influence rule is redefined based on the activities between the user and their friends (following list members), and therefore the influence rule is flexible, and it proved its strength when it was compared to other measurements found in the literature. This redefined influence rule helped in raising the accuracy of the recommender system as it was able to identify the best sources for the user and then use their data in building the profiles.

The proposed influence rule was used in this research to find different sources from different networks around the user, especially the hidden networks (implicit relationships) in order to use its members' recent activity to build the user profile. Therefore, more accurate items were recommended to the users.

Another contribution of this thesis is to model users in different ways based on the recent activity of networks around them (explicit and implicit networks) by building various profiles

in order to find the profile that best reflects user interests. The dataset was divided into different categories: activity, originality and number of friends. Each category has a suitable profile, and this may help future researchers build on this contribution.

5.2. Study limitations

The study showed that the proposed approach has advantages in enriching users' profiles from different sources that were identified by the proposed influence algorithm. It would be good to test the approach on a higher number of real Twitter users. However, the main limitation of collecting data through the Twitter API is the limited number of times that each type of request that can be performed. These numbers are restricted by tokens which are different depending on the type of request. Furthermore, a working window of 15 minutes is given to each developer, and if the tokens are consumed, the developer has to wait to finish the current window. After that, the API restarts the number of tokens for another 15 minutes, and so on. For example, when we want to collect the friends list of a user who follows 1000 people, we need five windows of 15 minutes, as the number of friends that can be collected through each window's tokens is limited to 200. As a result, we need one hour and 15 minutes. When we need to collect friends of friends, for example, much more time is needed, when some of the accounts follow thousands of friends. When the follow-up relationships depth level increases, even more time is needed to test the approach. The influence rule was tested at the first and second depth levels, and it would be good to test more depth levels in order to discover how far the influence effect can reach the tested user. These kind of depth levels are time consuming because of Twitter API restrictions. The number of collected tweets from users and also from their friends is limited, as Twitter only allows developers to retrieve the last 3200 tweets. When collecting the timeline of an active user, these tweets can only form that user's activity and interest within a short period of time, and this will raise the performance of the baseline profile as it only includes very recent tweets. Also, other accounts might not be considered influencers because the Twitter API cannot retrieve more tweets from the past, and the set of available tweets only identifies the recent accounts that the user has interacted with.

One of the limitations is testing our proposed work against more influence algorithms from the literature. This limitation is mainly caused by the restrictions of the Twitter API. Also, some researchers have not published their technical work, thus not allowing other researchers to use their work in their own research.

In the implicit relationships experiment, only recent followers were collected in each technique of building the profiles, because of the API restrictions. Based on the average number of tweets used in each profile, the experiment showed that the machine learning classification excluded many tweets from the FCF profile. It has 87 tweets more than the STBLCinf profile, and this small number of tweets affected its performance against the STBLCinf and InfF profiles in recommending the top 10 items. We tried to overcome this problem by calculating the cosine similarity between the examined user and the followers of both InfF and FCF profiles in order to discover which profile is more relevant to the user. In other words, cosine similarity was calculated between the timelines. As a result, this research was able to prove that the most similar accounts come from the followers of influential friends.

5.3. Future work

Future research into this topic would benefit from developing the influence score from the typical user's perspective rather than from the influencer's perspective. The proposed influence algorithm could be modified and used in other social media platforms based on the follow-up relationships. The proposed influence algorithm could also be used in different fields in regards to recommender systems, such as recommending users to follow and things to read.

The findings of this approach could be used to understand why users follow others and do not show any computable interactions. This kind of relationship needs to be examined, and this approach might be effective in modelling users, as we think users follow these accounts for a reason. Additionally, they might affect the user's interests, even if the user does not show any interaction with the followed users' activities.

By overcoming Twitter's API restrictions with very high programming skills, deeper levels of explicit follow-up relationships could be discovered and tested in order to measure the influence effect level. We tested only two depth levels; more levels could be tested and then

discovered to see at which levels the influence rule can be useful and at which levels it cannot. Deeper levels of implicit relationship sources could also be discovered and then used in building the user profile, such as the followers of influential friends of influential friends. If there was a way of collecting more than 3200 tweets by a user, the proposed influence algorithm could be tested in identifying the user's all-time influencers, especially the active users.

References

Abel, F., Gao, Q., Houben, G. J., & Tao, K. (2011). Analyzing temporal dynamics in Twitter profiles for personalized recommendations in the social web. In *Proceedings of the 3rd International Web Science Conference* (p. 2). ACM.

Abel, F., Gao, Q., Houben, G. J., & Tao, K. (2013). Twitter-based user modeling for news recommendations. *IJCAI*, *13*, pp. 2962-2966.

Abel, F., Hauff, C., Houben, G. J., Stronkman, R., & Tao, K. (2012, April). Twitcident: Fighting fire with information from social web streams. In *Proceedings of the 21st International Conference on World Wide Web* (pp. 305-308). ACM.

Adomavicius, G., & Tuzhilin, A. (2005). 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), pp. 734-749.

Aggarwal, C. C., & Zhai, C. (2012). A survey of text classification algorithms. In *Mining Text Data* (pp. 163-222). Springer, Boston, MA.

Agichtein, E., Brill, E., & Dumais, S. (2006, August). Improving web search ranking by incorporating user behavior information. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 19-26). ACM.

Alonso, O., Carson, C., Gerster, D., Ji, X., & Nabar, S. U. (2010, July). Detecting uninteresting content in text streams. In *SIGIR Crowdsourcing for Search Evaluation Workshop*.

Alshammari, A., Kapetanakis, S., Evans, R., Polatidis, N., & Alshammari, G. (2018). User modeling on Twitter with exploiting explicit relationships for personalized recommendations. Paper presented to the 18th International Conference on Hybrid Intelligent Systems. Porto. 13-15 December.

Alshammari, A., Polatidis, N., Kapetanakis, S., Evans, R., & Alshammari, G. (2019). Personalized recommendations on Twitter based on explicit user relationships modelling. Paper presented to the 24th UK Academy for Information Systems International Conference. Oxford. 9-10 April.

Amatriain, X., Pujol, J. M., & Oliver, N. (2009). I like it... I like it not: Evaluating user ratings noise in recommender systems. In *User Modeling, Adaptation, and Personalization* (pp. 247-258). Springer, Berlin, Heidelberg.

Anand, S. S., Kearney, P., & Shapcott, M. (2007). Generating semantically enriched user profiles for web personalization. *ACM Transactions on Internet Technology (TOIT)*, *7*(4), p. 22.

Anger, I., & Kittl, C. (2011). Measuring influence on Twitter. In *Proceedings of the 11th International Conference on Knowledge Management and Knowledge Technologies* (p. 31). ACM.

Armentano, M. G., Godoy, D., & Amandi, A. (2012). Topology-based recommendation of users in micro-blogging communities. *Journal of Computer Science and Technology*, *27*(3), pp. 624-634.

Asur, S., & Huberman, B. A. (2010, August). Predicting the future with social media. In *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology-Volume 01* (pp. 492-499). IEEE Computer Society.

Barla, M. (2011). Towards social-based user modeling and personalization. *Information Sciences and Technologies Bulletin of the ACM Slovakia*, *3*(1), pp. 52-60.

Beel, J., & Langer, S. (2015, September). A comparison of offline evaluations, online evaluations, and user studies in the context of research-paper recommender systems. In *International Conference on Theory and Practice of Digital Libraries* (pp. 153-168). Springer, Cham.

Bouguessa, M., & Romdhane, L. B. (2015). Identifying authorities in online communities. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *6*(3), p. 30.

Boyd, D., Golder, S., & Lotan, G. (2010, January). Tweet, tweet, retweet: Conversational aspects of retweeting on Twitter. In *43rd Hawaii International Conference on System Sciences (HICSS)* (pp. 1-10). IEEE.

Breuss, M. (2013). *Content Recommendation in Social Media* (Doctoral dissertation, University of Amsterdam).

Broder, A. Z., Glassman, S. C., Manasse, M. S., & Zweig, G. (1997). Syntactic clustering of the web. *Computer Networks and ISDN Systems*, *29*(8-13), pp. 1157-1166.

Brusilovsky, P., & Millán, E. (2007). User models for adaptive hypermedia and adaptive educational systems. In *The Adaptive Web* (pp. 3-53). Springer, Berlin, Heidelberg.

Buckley, C., & Voorhees, E. M. (2017, August). Evaluating evaluation measure stability. In *ACM SIGIR Forum*, *51*(2), pp. 235-242.

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, *12*(4), pp. 331-370.

Celebi, M. E., Kingravi, H. A., & Vela, P. A. (2013). A comparative study of efficient initialization methods for the k-means clustering algorithm. *Expert Systems with Applications*, *40*(1), pp. 200-210.

Chai, W., Xu, W., Zuo, M., & Wen, X. (2013). ACQR: A novel framework to identify and predict influential users in micro-blogging. In *Pacis* (p. 20).

Chen, C., Gao, D., Li, W., & Hou, Y. (2014). Inferring topic-dependent influence roles of Twitter users. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval* (pp. 1203-1206). ACM.

Chen, J., Nairn, R., & Chi, E. (2011, May). Speak little and well: Recommending conversations in online social streams. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 217-226). ACM.

Chen, K., Chen, T., Zheng, G., Jin, O., Yao, E., & Yu, Y. (2012). Collaborative personalized tweet recommendation. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 661-670). ACM.

Chien, W. S. (2000). *Learning Query Behavior in the Haystack System* (Doctoral dissertation, Massachusetts Institute of Technology).

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., & Singer, Y. (2006, March). Online passive-aggressive algorithms. *Journal of Machine Learning Research*, *7*, pp. 551-585.

Cunningham, P., & Delany, S. J. (2007). k-Nearest neighbour classifiers. *Multiple Classifier Systems*, *34*(8), pp. 1-17.

Dai, W., Xue, G. R., Yang, Q., & Yu, Y. (2007, July). Transferring naive Bayes classifiers for text classification. *AAAI*, 7, pp. 540-545.

Davenport, S. W., Bergman, S. M., Bergman, J. Z., & Fearrington, M. E. (2014). Twitter versus Facebook: Exploring the role of narcissism in the motives and usage of different social media platforms. *Computers in Human Behavior*, *32*, pp. 212-220.

Díaz-Agudo, B., Jimenez-Diaz, G., & Recio-García, J. A. (2018, November). SocialFan: Integrating Social Networks Into Recommender Systems. In *2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI)* (pp. 171-176). IEEE.

Dong, A., Zhang, R., Kolari, P., Bai, J., Diaz, F., Chang, Y., ... & Zha, H. (2010, April). Time is of the essence: Improving recency ranking using Twitter data. In *Proceedings of the 19th International Conference on the World Wide Web* (pp. 331-340). ACM.

Duan, Y., Jiang, L., Qin, T., Zhou, M., & Shum, H. Y. (2010, August). An empirical study on learning to rank of tweets. In *Proceedings of the 23rd International Conference on Computational Linguistics* (pp. 295-303). Association for Computational Linguistics.

Dumais, S., Cutrell, E., Cadiz, J. J., Jancke, G., Sarin, R., & Robbins, D. C. (2003, July). Stuff I've seen: A system for personal information retrieval and re-use. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 72-79). ACM.

Elmongui, H. G., Mansour, R., Morsy, H., Khater, S., El-Sharkasy, A., & Ibrahim, R. (2015, April). TRUPI: Twitter recommendation based on users' personal interests. In *International Conference on Intelligent Text Processing and Computational Linguistics* (pp. 272-284). Springer, Cham.

Ellison, N. B. (2007). Social network sites: Definition, history, and scholarship. *Journal of Computer-Mediated Communication*, *13*(1), pp. 210-230.

del Fresno Garcia, M., Daly, A. J., & Segado Sanchez-Cabezudo, S. (2016). Identifying the new influences in the Internet era: Social media and social network analysis. *Revista Española de Investigaciones Sociológicas*, (153).

Garcia Esparza, S., O'Mahony, M. P., & Smyth, B. (2013). Catstream: Categorising tweets for user profiling and stream filtering. In *Proceedings of the 2013 International Conference on Intelligent User Interfaces* (pp. 25-36). ACM.

Gasparetti, F., & Micarelli, A. (2007, January). Exploiting web browsing histories to identify user needs. In *Proceedings of the 12th International Conference on Intelligent User Interfaces* (pp. 325-328). ACM.

Gates, K. F., Lawhead, P. B., & Wilkins, D. E. (1998). Toward an adaptive WWW: A case study in customized hypermedia. *New Review of Hypermedia and Multimedia*, *4*(1), pp. 89-113.

Gauch, S., Speretta, M., Chandramouli, A., & Micarelli, A. (2007). User profiles for personalized information access. In *The Adaptive Web*, pp. 54-89.

Gershenson, C. (2003). Artificial neural networks for beginners. Retrieved on 5 February 2018 from: http://arxiv.org/ftp/cs/papers/0308/0308031.pdf.

Ghorab, M. R., Zhou, D., O'Connor, A., & Wade, V. (2013). Personalised information retrieval: Survey and classification. *User Modeling and User-Adapted Interaction*, *23*(4), pp. 381-443.

Gomaa, W. H., & Fahmy, A. A. (2013). A survey of text similarity approaches. *International Journal of Computer Applications*, *68*(13), pp. 13-18.

Gunawardana, A., & Shani, G. (2009, December). A survey of accuracy evaluation metrics of recommendation tasks. *Journal of Machine Learning Research*, *10*, pp. 2935-2962.

Hajian, B., & White, T. (2011, October). Modelling influence in a social network: Metrics and evaluation. In *Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third International Conference on Social Computing (SocialCom)* (pp. 497-500). IEEE.

Hannon, J., Bennett, M., & Smyth, B. (2010, September). Recommending Twitter users to follow using content and collaborative filtering approaches. In *Proceedings of the Fourth ACM Conference on Recommender Systems* (pp. 199-206). ACM.

Hong, L., Dan, O., & Davison, B. D. (2011, March). Predicting popular messages in Twitter. In *Proceedings of the 20th International Conference Companion on the World Wide Web* (pp. 57-58). ACM.

Hotho, A., Nürnberger, A., & Paaß, G. (2005, May). A brief survey of text mining. In *LDV Forum*, *20*(1), pp. 19-62.

Huang, A. (2008, April). Similarity measures for text document clustering. In *Proceedings of the Sixth New Zealand Computer Science Research Student Conference (NZCSRSC2008), Christchurch, New Zealand* (Vol. 4, pp. 9-56).

Huang, M., Yang, Y., & Zhu, X. (2011). Quality-biased ranking of short texts in microblogging services. In *Proceedings of the 5th International Joint Conference on Natural Language Processing* (pp. 373-382).

Huberman, B. A., Romero, D. M., & Wu, F. (2008). Social networks that matter: Twitter under the microscope.

Hughes, D. J., Rowe, M., Batey, M., & Lee, A. (2012). A tale of two sites: Twitter vs. Facebook and the personality predictors of social media usage. *Computers in Human Behavior*, *28*(2), pp. 561-569.

Hu, Y., Koren, Y., & Volinsky, C. (2008, December). Collaborative filtering for implicit feedback datasets. In *Eighth IEEE International Conference on Data Mining* (pp. 263-272). IEEE.

Jabeur, L. B., Tamine, L., & Boughanem, M. (2012, October). Active microbloggers: Identifying influencers, leaders and discussers in microblogging networks. In *International Symposium on String Processing and Information Retrieval* (pp. 111-117). Springer, Berlin, Heidelberg.

Java, A., Song, X., Finin, T., & Tseng, B. (2007, August). 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* (pp. 56-65). ACM.

Jawaheer, G., Szomszor, M., & Kostkova, P. (2010, September). Comparison of implicit and explicit feedback from an online music recommendation service. In *Proceedings of the 1st International Workshop on Information Heterogeneity and Fusion in Recommender Systems* (pp. 47-51). ACM.

Jawaheer, G., Weller, P., & Kostkova, P. (2014). Modeling user preferences in recommender systems: A classification framework for explicit and implicit user feedback. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, *4*(2), p. 8.

Joachims, T., Freitag, D., & Mitchell, T. (1997, August). Webwatcher: A tour guide for the world wide web. *IJCAI*, *1*, pp. 770-777.

Jones, K. S., Walker, S., & Robertson, S. E. (2000). A probabilistic model of information retrieval: Development and comparative experiments: Part 2. *Information Processing & Management*, *36*(6), pp. 809-840.

Jung, J. J. (2012). Online named entity recognition method for microtexts in social networking services: A case study of Twitter. *Expert Systems with Applications*, *39*(9), pp. 8066-8070.

Kaplan, C., Fenwick, J., & Chen, J. (1993). Adaptive hypertext navigation based on user goals and context. *User modeling and user-adapted interaction*, 3(3), 193-220.

Karidi, D. P., Stavrakas, Y., & Vassiliou, Y. (2016, July). A personalized tweet recommendation approach based on concept graphs. In *Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress, 2016 Intl IEEE Conferences* (pp. 253-260). IEEE.

Kawamae, N. (2011, February). Trend analysis model: Trend consists of temporal words, topics, and timestamps. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining* (pp. 317-326). ACM.

Kazienko, P., & Kolodziejski, P. (2006). Personalized integration of recommendation methods for e-commerce. *IJCSA*, *3*(3), pp. 12-26.

Kelly, D., & Teevan, J. (2003, September). Implicit feedback for inferring user preference: A bibliography. In *ACM SIGIR Forum*, *37*(2), pp. 18-28.

Kim, S. B., Han, K. S., Rim, H. C., & Myaeng, S. H. (2006). Some effective techniques for naive Bayes text classification. *IEEE Transactions on Knowledge and Data Engineering*, *18*(11), pp. 1457-1466.

Kriger, C. (2007). Prediction of the influent wastewater variables using neural network theory.

Kwak, H., Lee, C., Park, H., & Moon, S. (2010, April). What is Twitter, a social network or a news media?. In *Proceedings of the 19th International Conference on the World Wide Web* (pp. 591-600). ACM.

Lee, W. J., Oh, K. J., Lim, C. G., & Choi, H. J. (2014). User profile extraction from Twitter for personalized news recommendation. In *16th International Conference on Advanced Communication Technology* (pp. 779- 783). IEEE.

Li, W. J., Wang, K., Stolfo, S. J., & Herzog, B. (2005, June). Fileprints: Identifying file types by n-gram analysis. In *Proceedings from the Sixth Annual IEEE SMC Information Assurance Workshop* (pp. 64-71). IEEE.

Liben-Nowell, D., & Kleinberg, J. (2007). The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, *58*(7), pp. 1019-1031.

Lieberman, H. (1995). Letizia: An agent that assists web browsing. *IJCAI*, *1*, pp. 924-929.

Lops, P., De Gemmis, M., & Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. In *Recommender Systems Handbook* (pp. 73-105). Springer US.

Lu, J., Wu, D., Mao, M., Wang, W., & Zhang, G. (2015). Recommender system application developments: A survey. *Decision Support Systems*, *74*, pp. 12-32.

Lü, L., Medo, M., Yeung, C. H., Zhang, Y. C., Zhang, Z. K., & Zhou, T. (2012). Recommender systems. *Physics Reports*, *519*(1), pp. 1-49.

Maynard, D., Peters, W., & Li, Y. (2006, May). Metrics for evaluation of ontology-based information extraction. In *International World Wide Web Conference* (pp. 1-8).

McDonald, D. W. (2003). Ubiquitous recommendation systems. Computer, 36(10), 111-112.

Meij, E., Weerkamp, W., & De Rijke, M. (2012, February). Adding semantics to microblog posts. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining* (pp. 563-572). ACM.

Micarelli, A., & Sciarrone, F. (2004). Anatomy and empirical evaluation of an adaptive web-based information filtering system. *User Modeling and User-Adapted Interaction*, *14*(2-3), pp. 159-200.

Micarelli, A., Sciarrone, F., & Marinilli, M. (2007). Web document modeling. In *The Adaptive Web* (pp. 155-192). Springer, Berlin, Heidelberg.

Michelson, M., & Macskassy, S. A. (2010, October). Discovering users' topics of interest on Twitter: A first look. In *Proceedings of the Fourth Workshop on Analytics for Noisy Unstructured Text Data* (pp. 73-80). ACM.

Mobasher, B. (2007). Data mining for web personalization. In *The Adaptive Web* (pp. 90-135). Springer, Berlin, Heidelberg.

Morone, F., & Makse, H. A. (2015). Influence maximization in complex networks through optimal percolation. Nature, 524(7563), 65.

Myers, S. A., & Leskovec, J. (2012, December). Clash of the contagions: Cooperation and competition in information diffusion. In *IEEE 12th International Conference on Data Mining* (pp. 539-548). IEEE.

Nagarajan, M., Purohit, H., & Sheth, A. P. (2010). A qualitative examination of topical tweet and retweet practices. *ICWSM*, *2*(10), pp. 295-298.

Nagmoti, R., Teredesai, A., & De Cock, M. (2010, August). Ranking approaches for microblog search. In *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology-Volume 01* (pp. 153-157). IEEE Computer Society.

Naveed, N., Gottron, T., Kunegis, J., & Alhadi, A. C. (2011, June). Bad news travel fast: A content-based analysis of interestingness on Twitter. In *Proceedings of the 3rd International Web Science Conference* (p. 8). ACM.

Nichols, D. (1998). Implicit rating and filtering. In *Proceedings of the Fifth DELOS Workshop on Filtering and Collaborative Filtering*. ERCIM.

Oghina, A., Breuss, M., Tsagkias, M., & de Rijke, M. (2012, April). Predicting IMDB movie ratings using social media. In *European Conference on Information Retrieval* (pp. 503-507). Springer, Berlin, Heidelberg.

Oard, D. W., & Kim, J. (2001). Modeling information content using observable behavior. In *Proceedings of the 64th Annual Conference of the American Society for Information Science and Technology*.

Pal, A., & Counts, S. (2011, February). Identifying topical authorities in microblogs. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining* (pp. 45-54). ACM.

Pal, M. (2005). Random forest classifier for remote sensing classification. *International Journal of Remote Sensing*, *26*(1), pp. 217-222.

Pazzani, M. J., Muramatsu, J., & Billsus, D. (1996, August). Syskill & Webert: Identifying interesting web sites. In *AAAI/IAAI*, *1* (pp. 54-61).

Pazzani, M. J., & Billsus, D. (2007). Content-based recommendation systems. In *The Adaptive Web* (pp. 325-341). Springer, Berlin, Heidelberg.

Peng, H. K., Zhu, J., Piao, D., Yan, R., & Zhang, Y. (2011, December). Retweet modeling using conditional random fields. In *2011 11th IEEE International Conference on Data Mining Workshops* (pp. 336-343). IEEE.

Petrovic, S., Osborne, M., & Lavrenko, V. (2011). RT to win! Predicting message propagation in Twitter. *ICWSM*, *11*, pp. 586-589.

Piao, G., & Breslin, J. G. (2016, September). Exploring dynamics and semantics of user interests for user modeling on Twitter for link recommendations. In *Proceedings of the 12th International Conference on Semantic Systems* (pp. 81-88). ACM.

Polat, K., & Güneş, S. (2007). Classification of epileptiform EEG using a hybrid system based on decision tree classifier and fast Fourier transform. *Applied Mathematics and Computation*, *187*(2), pp. 1017-1026.

Qiu, F., & Cho, J. (2006, May). Automatic identification of user interest for personalized search. In *Proceedings of the 15th International Conference on the World Wide Web* (pp. 727-736). ACM.

Quercia, D., Ellis, J., Capra, L., & Crowcroft, J. (2011, October). In the mood for being influential on Twitter. In *Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third International Conference on Social Computing (SocialCom)* (pp. 307-314). IEEE.

Ramage, D., Dumais, S. T., & Liebling, D. J. (2010). Characterizing microblogs with topic models. In *Fourth International AAAI Conference on Weblogs and Social Media* (pp. 130-137).

Ramage, D., Hall, D., Nallapati, R., & Manning, C. D. (2009, August). 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* (pp. 248-256). Association for Computational Linguistics.

Razis, G., & Anagnostopoulos, I. (2016). Discovering similar Twitter accounts using semantics. *Engineering Applications of Artificial Intelligence*, *51*, pp. 37-49.

Resnick, P., & Varian, H. R. (1997). Recommender systems. *Communications of the ACM*, *40*(3), pp. 56-58.

Ricci, F., Rokach, L., & Shapira, B. (2015). Recommender systems: introduction and challenges. In *Recommender systems handbook*. Springer, Boston, MA.

Riquelme, F., & González-Cantergiani, P. (2016). Measuring user influence on Twitter: A survey. *Information Processing & Management*, *52*(5), pp. 949-975.

Ritter, A., Cherry, C., & Dolan, B. (2010, June). Unsupervised modeling of Twitter conversations. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (pp. 172-180). Association for Computational Linguistics.

Ritter, A., Clark, S., & Etzioni, O. (2011, July). Named entity recognition in tweets: An experimental study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing* (pp. 1524-1534). Association for Computational Linguistics.

Romero, D. M., Galuba, W., Asur, S., & Huberman, B. A. (2011, September). Influence and passivity in social media. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (pp. 18-33). Springer, Berlin, Heidelberg.

Sakaki, T., Okazaki, M., & Matsuo, Y. (2010, April). Earthquake shakes Twitter users: Real-time event detection by social sensors. In *Proceedings of the 19th International Conference on the World Wide Web* (pp. 851-860). ACM.

Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, *24*(5), pp. 513-523.

Sankaranarayanan, J., Samet, H., Teitler, B. E., Lieberman, M. D., & Sperling, J. (2009, November). Twitterstand: News in tweets. In *Proceedings of the 17th ACM Sigspatial International Conference on Advances in Geographic Information systems* (pp. 42-51). ACM.

Schafer, J. B., Frankowski, D., Herlocker, J., & Sen, S. (2007). Collaborative filtering recommender systems. In *The Adaptive Web* (pp. 291-324). Springer, Berlin, Heidelberg.

Shani, G., & Gunawardana, A. (2011). Evaluating recommendation systems. In *Recommender Systems Handbook* (pp. 257-297). Springer, Boston.

Sharma, V., Rai, S., & Dev, A. (2012). A comprehensive study of artificial neural networks. *International Journal of Advanced Research in Computer Science and Software Engineering* (pp. 278-284), *2*(10).

Shavlik, J., & Eliassi-Rad, T. (1998, July). Intelligent agents for web-based tasks: An advice-taking approach. In *AAAI/ICML Workshop on Learning for Text Categorization* (pp. 63-70).

Shavlik, J., Calcari, S., Eliassi-Rad, T., & Solock, J. (1998, December). An instructable, adaptive interface for discovering and monitoring information on the World-Wide Web. In *Proceedings of the 4th International Conference on Intelligent User Interfaces* (pp. 157-160). ACM.

Smith, A. N., Fischer, E., & Yongjian, C. (2012). How does brand-related user-generated content differ across YouTube, Facebook, and Twitter?. *Journal of Interactive Marketing*, *26*(2), pp. 102-113.

Spina, D., Meij, E., De Rijke, M., Oghina, A., Bui, M. T., & Breuss, M. (2012, August). Identifying entity aspects in microblog posts. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 1089-1090). ACM.

Stamou, S., & Ntoulas, A. (2009). Search personalization through query and page topical analysis. *User Modeling and User-Adapted Interaction*, *19*(1-2), pp. 5-33.

Stash, N. V., Cristea, A. I., & De Bra, P. M. (2004, May). Authoring of learning styles in adaptive hypermedia: Problems and solutions. In *Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers & Posters* (pp. 114-123). ACM.

Sugiyama, K., Hatano, K., & Yoshikawa, M. (2004, May). Adaptive web search based on user profile constructed without any effort from users. In *Proceedings of the 13th International Conference on World Wide Web* (pp. 675-684). ACM.

Suh, B., Hong, L., Pirolli, P., & Chi, E. H. (2010, August). Want to be retweeted? Large scale analytics on factors impacting retweet in Twitter network. In *2010 IEEE Second International Conference on Social Computing* (pp. 177-184). IEEE.

Thorat, P. B., Goudar, R. M., & Barve, S. (2015). Survey on collaborative filtering, content-based filtering and hybrid recommendation system. *International Journal of Computer Applications* (pp. 31-36), *110*(4).

Tintarev, N. (2007, July). Explaining recommendations. In *International Conference on User Modeling* (pp. 470-474). Springer, Berlin, Heidelberg.

Tumasjan, A., Sprenger, T. O., Sandner, P. G., & Welpe, I. M. (2010). Predicting elections with Twitter: What 140 characters reveal about political sentiment. *ICWSM*, *10*(1), pp. 178-185.

Uysal, I., & Croft, W. B. (2011). User oriented tweet ranking: A filtering approach to microblogs. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management* (pp. 2261-2264). ACM.

Vallet, D., Cantador, I., & Jose, J. M. (2010). Personalizing web search with folksonomy-based user and document profiles. In *Advances in Information Retrieval* (pp. 420-431). Springer, Berlin, Heidelberg.

Vassileva, J. (1996). A task-centered approach for user modeling in a hypermedia office documentation system. *User Modeling and User-Adapted Interaction*, *6*(2-3), pp. 185-223.

Vosoughi, S. (2015). *Automatic Detection and Verification of Rumors on Twitter* (Doctoral dissertation, Massachusetts Institute of Technology).

Weng, J., Lim, E. P., Jiang, J., & He, Q. (2010, February). Twitterrank: Finding topic-sensitive influential Twitterers. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining* (pp. 261-270). ACM.

Xiao, B., & Benbasat, I. (2007). E-commerce product recommendation agents: Use, characteristics, and impact. *MIS quarterly*, *31*(1), pp. 137-209.

Xu, R., & Wunsch, D. (2005). Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, *16*(3), pp. 645-678.

Xue, H., Yang, Q., & Chen, S. (2009). SVM: Support vector machines. *The Top Ten Algorithms in Data Mining*, *6*(3), pp. 37-60.

Yang, J., & Counts, S. (2010). Predicting the speed, scale, and range of information diffusion in Twitter. *ICWSM*, *10*(2010), pp. 355-358.

Yang, Z., Guo, J., Cai, K., Tang, J., Li, J., Zhang, L., & Su, Z. (2010, October). Understanding retweeting behaviors in social networks. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management* (pp. 1633-1636). ACM.

Yin, Z., & Zhang, Y. (2012, September). Measuring pair-wise social influence in microblog. In *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on Social Computing (SocialCom)* (pp. 502-507). IEEE.

Yue, Y., Finley, T., Radlinski, F., & Joachims, T. (2007, July). A support vector method for optimizing average precision. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 271-278). ACM.

Zaman, T. R., Herbrich, R., Van Gael, J., & Stern, D. (2010, December). Predicting information spreading in Twitter. In *Workshop on Computational Social Science and the Wisdom of Crowds, Nips*, *104*(45), pp. 17599-17601.

Zhou, D., Lawless, S., & Wade, V. (2012). Improving search via personalized query expansion using social media. *Information Retrieval*, *15*(3-4), pp. 218-242.