Reveal: Fine-grained Recommendations in a Social Ecosystem

Markos Aivazoglou
FORTH-ICS, Greece
markos@ics.forth.gr

Antonios O. Roussos
FORTH-ICS, Greece
roussos@ics.forth.gr

Sotiris Ioannidis
FORTH-ICS, Greece
sotiris@ics.forth.gr

Dimitris Spiliotopoulos
FORTH-ICS, Greece
dspiliot@ics.forth.gr

Iasonas Polakis
Columbia University, USA
polakis@cs.columbia.edu

ABSTRACT

Recommender systems have greatly evolved in recent years, and have become an integral part of the world wide web. From e-commerce sites to more specialized mobile apps, our everyday life evolves around a series of "small" decisions that are often influenced by such recommendations. In a similar manner, online social networks present (i.e., *recommend*) only a subset of the massive amount of content published by a user's contacts. However, the prevalent approach for the content selection process in such systems, is heavily influenced by the amount of interaction between the user and the contact who posted it. As a result, content of interest is often lost due to weak social ties.

In this paper we present `Reveal`, a fine-grained recommender system for online social networks, designed to create lists of recommended media content (e.g., music videos, movie trailers) mentioned or posted by the user's contacts. We conduct a qualitative user study to explore the value and requirements of a recommendation component within a social network, the findings of which drive our system's design. The core functionality of our system is broken down into several phases. First, `Reveal` determines whether any given post contains or refers to content relevant to a category of interest to the user. Next it must decide if the publisher has expressed a favorable or negative opinion about it. If the previous criteria is met, the system obtains a precise categorization of the content and, finally, compares it to the user's *interest profile*. `Reveal` leverages the abundance of pre-existing information in each user's account for creating the interest profiles, and calculating similarity scores at a fine-grained level for each contact. The intuition behind our system is to obtain information representations that can identify overlapping interests in very specific sub-categories (e.g., two users may only "agree" on hard rock music). While our system is intended as a component of the social networking service, we develop a proof-of-concept implementation for Facebook, and explore the effectiveness of our underlying mechanisms for the content analysis. Our experimental evaluation demonstrates the effectiveness of our approach at identifying content of interest which was not contained in the users' News Feed. We also conduct a user study for exploring the usability aspects of our prototype, and find that our system offers functionality that could significantly improve user experience in popular services.

1. INTRODUCTION

Online social networks are among the most popular web services, with reports indicating that they have also become the most-time consuming user activity [1]. A significant aspect of their popularity, which also increases user engagement, is the sharing of content among contacts. However, the massive amount of content published by contacts in social networks can result in users overlooking content that might be of interest, due to the overall noise. Furthermore, the content presented in Facebook's News Feed is influenced by the user's relationship to the contact that published it [7]; as a result, content that might be of great interest to the user, might be omitted due to a "weak" relationship to the content publisher.

In this paper we propose `Reveal`, a recommender system that exists within the social network, which categorizes content and offers suggestions based on the interest similarities to the user that publishes the content. Specifically, the system processes all content published by the user's contacts (e.g., posts, links), identifies content belonging to different categories (e.g., music, movies), and then collects information to assign that content to a more specific sub-category (e.g., music (sub)genres). Subsequently, the system analyzes any text accompanying that content, so as to identify the sentiment regarding that content and infer a positive or negative review by the content's publisher. Based on the user's interests profile, the similarity score with the content's publisher, the previous information is used to assign an *interest score* to the content that determines if the content should be suggested to the user or not.

Previous work has focused on `like` objects to recommend content to users. Similarly, Facebook has also implemented a recommender system that suggests content liked by contacts. Our approach follows a different approach; it aims to filter through the massive amount of content that is posted by a user's contacts, and select the most likely to be of interest to the user. Another key concept behind our system that differentiates it from other recommender systems, is that it enables users to leverage their existing knowledge regarding the overlapping interests they have with their online contacts, in a fine-grained manner. While certain users may have very similar tastes in a specific subcategory, they might have completely different interests in other categories. For example Alice and Bob may like the same arthouse films, but completely disagree when it comes to romantic comedies.

In a nutshell, our system works as follows. First, we collect all the interests found in the user's profile. Every element is processed for extracting information for its categorization (e.g., movie). Next, based on the category, various services (e.g., Wikipedia) and search engines are employed to collect more information that will allow further categorization (e.g., romantic comedy film). After all the elements are processed, an *interest profile* is created for the user, containing the various sub-categories that describe the user's interests, and a score reflecting its importance. Subsequently, the same process is repeated for each of the users contacts, and a *similarity profile* is created for each contact. This profile contains the overlapping interests of the two users, e.g., the categories and sub-categories that are of interest to both the users. Depending on the amount of overlapping elements (e.g., both like the films *Fight Club* and *The Matrix*) a similarity score is assigned to each sub-category. After the initial processing, the user can manually "tweak" the similarity score for each contact. Low value represents the difference in taste of the specific category. Furthermore, the system monitors which content the user clicks on to dynamically update the similarity scores for the specific content's publisher. A feedback rating function allows the user to rate a specific content which also updates the publisher's similarity score.

Our system is not intended as a replacement for existing content selection algorithms for users' news feeds; instead we aim to identify the most interesting items (e.g., Youtube videos) posted by a user's contacts, which may be otherwise lost amidst the massive amount of generated content. These selected posts are to be presented in a separate recommendations section, each dedicated to a specific category (e.g., music, movies).

The main contributions of this work are the following:

- We conduct a qualitative user study that explores the perception of Facebook users on the value of social recommendations within their friend network, and their expectations regarding their social circle's ability to provide movie/music recommendations. The rationale behind our study is to verify that users have overlapping tastes with certain members of their circle and, as a result, expectations arise regarding the expected interest of posted content. This study also collects specific requirements for the proposed recommender system design, aimed to provide validation on specific user expectations and identify the parameters that may lead to successful social recommendation for movie/music content within a network of friends.
- We develop `Reveal`, a proof-of-concept Facebook application that materializes our fine-grained recommender approach. Our system leverages sentiment analysis, named-entity recognition for identifying relevant posts, content ranking and similarity evaluation for providing socially-driven recommendations.
- We conducted a pilot study with real Facebook users in order to assess the value of the work proposition and explore the effectiveness of unobtrusive social recommendations in a practical setting. The proposed approach was evaluated over a period of one week using real-time Facebook account data. The user feedback provided valuable insight into the content selection of the recommender system. We also evaluated the user's experience with our prototype application.

2. USER STUDY AND MOTIVATION

The social aspect of recommendation continue can have a lasting impact, as the more advanced the features offered by the social networks are, the more uses the users will find, increasing the sharing preferences and posts relevant to several aspects of living [38]. Studies on user interaction in Facebook report that only 30% of Facebook user pairs interact consistently at least once every month. This shows that a high percentage of the pool of friends either lack interests or miss specific topics of interest due to the high speed and volume stream of posts that the casual user fails to follow upon [40]. Wilson et al. [41] reported that human interactions are limited by constraints such as time, and brings into question the practice of evaluating social networks in distributed systems directly using social connectivity graphs. The same study also reports that for the vast majority of Facebook users (âĹij 90%), 20% of their friends account for 70% of all interactions. Regarding video content, De Pessemier et al. [15] did a study on Netlog and mentioned the positive correlation of user-uploaded video popularity and number of friends.

To obtain insight on the aspects of social recommendations that we want to provide, we conduct a user study with 38 participants that were asked to reflect upon their interaction, expectations and needs from Facebook regarding movie and music recommendation from friends. The aim of this exploratory study was to establish facts on how Facebook users perceive their time spent on Facebook in terms of several factors that describe the use of the social network as a recommendation means for entertainment value, specifically for movies and music. Additionally, specific interaction relations were explored in order to gain insight into the potential of the social data in the context of movie and music recommendation. The latter was designed to collect information-specific requirements for the described approach.

2.1 Procedure. The users were asked to browse their Facebook timelines and respond to specific category-establishing questions, such as "Do you use Facebook to find posts about movies/music of interest?". The second group of information aimed to establish baseline user profiling, such as number of friends, etc. The final, and largest, group of information was designed to collect interaction-specific insights and the user expectations/requirements for the proposed fine-grained movie/music recommendation approach.

2.2 User profiling and general information. The initial finding was that the number of friends and time spent on the News Feed are not independent (p=0.033, chi-square). There is significant dependence between the percentage of active friends on Facebook and the belief that some of the friends would be great movie/music recommenders (p=0.022, chi-square) as well as commonality of movie/music interests with same gender friends (p=0.002, chi-square). It was also generally established that both the use of Facebook to find posts about movies and music and the self-proclamation about level of informedness about friendsínterests in movies/music are dependent on the percentage of movies/music related posts (p=0.039 and p=0.017 respectively). It was also reported that the users that consider their News Feed posts for music/movies as recommendations from friends, do use Facebook to find such posts (p=0.007, t-test), also expressing their interest in such posts (p=0.002, t-test). The interest itself was manifested in the perceived specific knowledge of actual interest overlap between the users and their Facebook friends (p=0.014, ANOVA). Additionally, it was evident that Facebook can be a premier source of interesting movie and music information, and that information is directly related to own friendsínterest (p=0.027, ANOVA).

2.3 Interaction. Users reported that their level of awareness of their friends' interest about movies and music are directly dependent to the frequency that they check their News Feed, which was also proportional to the time they spend everyday to perform that process (p=0.007, chi-square). There is significance in the fact that the same users proportionally stated that they thought that at least some of their friends would be great recommenders for movies and music (p=0.032, chi-square). As a remedy for the frequency of the required checking of the News Feed, the users reported that the most helpful requirements for a recommender system would be the ability to view ranked recommendations for movies/music (p=0.023, chi-square) and also to edit and fine-tune the recommendations for improved accuracy (p=0.033, chi-square). The later ability is also very significant to the time currently spent scanning the News Feed (p=0.007, chi-square). Additionally, a positive correlation exists between the importance of the requirement to view ranked recommendations within Facebook and the general knowledge of friends' interests in movies/music (p=0.002, correlation test), illustrating the expected impact of the particular requirement.

2.4 Expectations for the proposed solution. Users that do not currently consider or use Facebook for finding posts about music and/or movies, reported that a more fine-grained processing of friends' posts would result in more interesting recommendations while they consider the option to easily view a collection of recommendations as very useful (p=0.016, t-test). Interestingly enough, for all users, a positive correlation exists between strongly believing (assigning 5 in the 1-5 Likert scale) a more fine-grained processing of friends' posts would reveal the most interesting recommendations and strongly stressing (assigning 5 in the 1-5 Likert scale) the usefulness of easily viewing a recommendation collection, i.e. phasing out the noise from unrelated posts (p=0.015, correlation test). Finally, to verify the correlation of the findings, the expected dependences between the requested fine-grained processing of friends' posts and the number of times that users have to check their News Feed (p=0.004, chi-square) and the time in minutes spent on the News Feed (p=0.031, chi-square). Regarding specific expectations, the participants expected overlapping interests with a relative subgroup of Facebook friends and stressed the importance of choosing to extract the information from those specific sub-groups to result in accurate recommendations (p=0.020, ANOVA). The ability to view ranked recommendations from friends that are on the group of highly relevant interests was a major requirement, aligning with the expectation that common profile items between friends would lead to common interest posts, to be used by the recommendation engine (p=0.006, ANOVA). The motivation behind this work is to provide a well documented approach on fine-grained recommendation in a very large social ecosystem, such as Facebook. Collecting the requirements of real users validated by an extensive study on how they perceive the potential of such approach aims to build upon earlier findings on social interaction. Revealing hidden expectations about the potential of such an approach based on the user perceptions on life preferences overlapping and common interests, as perceived within circles of friends, is an open-ended exploration on social network relations. These relations are intrinsically the basis for enhancing traditional recommendation with social aspects for identifying interesting posts and ranking them accurately.

3. OVERVIEW OF SYSTEM DESIGN

In this section we present a high-level overview of our proposed system `Reveal`, and the components that comprise it, as shown in Figure 1. The functionality of our system is split into three main phases; the first one is triggered when a user deploys our application for the very first time, and consists of three one-time steps.

(i) Bootstrap phase, where every `like` under the music and movies categories in the user's and her friends' profiles are gathered and analyzed for extracting potential entities and conducting a fine-grained categorization for creating the *interest profile* of every user. Each like is grouped by its category and associated user.

(ii) Similarity Calculation and Score Tweaking, in which `Reveal` finds prominent similarity between a user and her friends, based on the *interest profile* created in the previous step, and assigning *similarity score* for each of the latter. At this stage, the active user can interactively tweak *similarity scores* to her liking, thus weighting friends that have been assigned scores differently than our system suggested.

(iii) Post Analysis and Lists Population, where heuristics such as Sentiment Analysis, Named Entity Recognition, and Post score calculation are used to extract information on whether a post will be of interest to the user and calculate its significance, to decide whether it should be recommended. The second component, is the interactive component of our application, where users can view the top-10 lists of recommended music and movie items, or tweak the weight of their friends to better reflect their actual similarity for specific sub-categories. Finally, the third component consists of 2 independent modules, one that refreshes the recommendation lists for each user, and another one that applies the aforementioned bootstrap phase, with both of them being executed periodically. Below, we give a more detailed description of the heuristics used in the `Reveal` architecture.

3.1 Sentiment Analyser. Another point of interest in recommendations, especially when natural language is involved, is figuring out the reviewer's point-of-view on the topics and subjects they post, for example the text that accompanies links shared on Facebook. Sentiment analysis defines the method of discerning the intrinsic attractiveness (positive valence) or aversiveness (negative valence) from the text. A large body of research has explored this subject, from detecting sentiment in literature texts [30], to classifying emotion and sentiment in social media and microblogs [18, 31], as well as the polarity in product reviews [27]. In our system it is essential, as we need to know the poster's opinion polarity in order to make correct recommendation decisions, using it as an initial binary rating method. We use a modified version of the SO-CAL approach introduced by Taboada et al. (2011) [37], along with its own dictionary set categorised as part of speech, coupled with extra dictionaries (noted below), and a self-assembled dictionary of emoticons.

3.2 Knowledge Base, Entity Extractor, Video ID to Topic Id Indexer. To process large collections of online data as fast as possible, we created an off-line Knowledge Base containing entities that are of interest. We leverage the Freebase [6] collection, and obtained every entity under musical artists/bands and movies coupled with detailed genre information, through the MQL API. Specifically, we obtained 293,506 unique entities, 221,091 movie entities and 72,415 musical artist/bands entities paired with genres, as `JSON` objects. Each entity in the Knowledge Base carries a unique `Topic ID` as provided by Freebase itself, that we use for entity recognition. We developed a dictionary-based entity extractor, that utilizes resources from our Knowledge-Base. In addition, to swiftly relate a video to its respective topic (entity) we made an indexer that mapped each `Topic ID` bound to its entity with every correspondent `Video ID` found in Youtube at that time. The latter heuristic, will be explained further in 4.4. This resource enables fast named entity recognition, without reliance on online APIs.

3.3 Refresher and Re-Bootstrap script. Our system must periodically refresh the recommendation lists for each user. To that end, we have a backend daemon designed similarly to the application's list initialization process. For performance reasons, we also lookup online resources, for data not found in our knowledge base, only in this process. In addition, our refresh procedure polls the profiles of the users and their friends, to account for any changes made to their likes and interests. Therefore, the scores representing a users' friends are recalculated based on newly received information.

3.4 Similarity and Post score Calculator. A big challenge was to figure out the right similarity formula between users and friends and calculating the degree of interest for a specific post. In Smallworlds [19], the proposed system used an interactive graph, where item and friend weighting was done by dragging each node nearer or further away from the active user representation, thus requiring interaction from users even in the initial stage. Item weights in ones profile, were pre-assigned interactively. Finally, scores were in a capped scale (1 to 5 inclusive), thus not being as fine-grained as needed. Based on the Jaccard coefficient, we created a simple formula that is well suited for automatic similarity calculation in sets with weighted items and arbitrary size.

3.5 Slang Terminology handling effort. We made a web scrapper, that crawled Urban Dictionary [8] and constructed well-formed

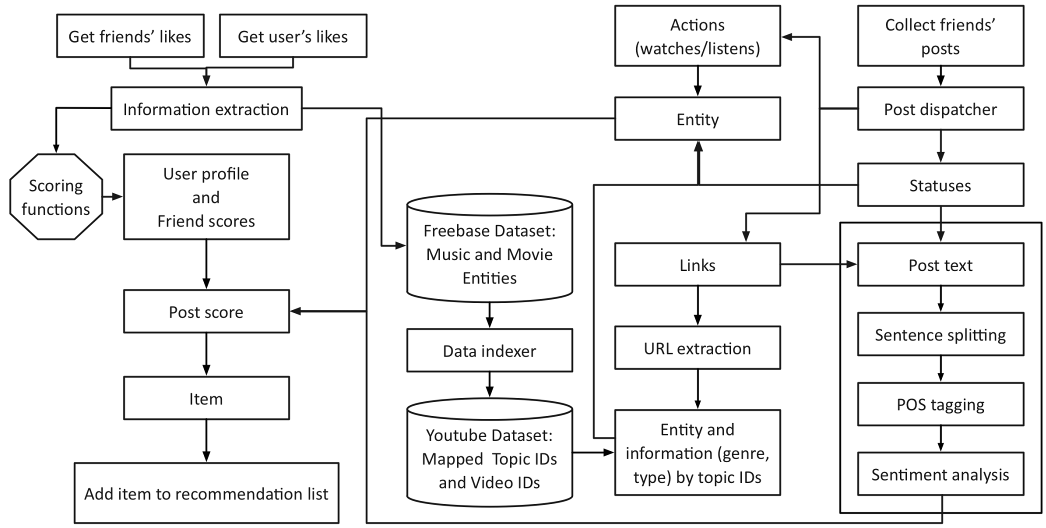

Figure 1: Overview of our system's components and processes.

XML documents for each letter, containing roughly 7 million entries along with their definitions. Definitions were essential, as they were used to decide whether a word had positive or negative semantic meaning, depending on how many times a definition was classified as either.

Low Accuracy Rationale and dismiss of Dictionaries. As there are no restrictions enforced in the urban dictionary, a large number of words we acquired contained semantic noise. There are entries like product brands, names and even common stop-words, that are classified (as positive or negative), hence the noise. The shortcomings of following an automated procedure to make dictionaries for such purpose as well as increasing the size of dictionaries, have been previously documented by others [37]. Both of the above, decrease the accuracy significantly. A suitable approach is to manually classify any word used for such a task, but this is too time-consuming to be applied to 7 million words. A manual selection of a significantly smaller subset could offer a solution. As SO-CAL works with part of speech tags, any new addition needs to be tagged. However, the size of our urban dictionaries, uncommon words (that are hard or impossible to be identified by taggers) as well as the fact that it contains phrases further complicated this task. Due to these obstacles, we ruled out the use of the dictionaries we made from urban dictionary as their use decreased our analyzer's performance in a limited pilot evaluation.

4. SYSTEM IMPLEMENTATION

Now that we described the overview of our system, we hereby present the implementation of our proof of concept application, any restrictions applied, and also the supplementary scripts used. Our proof of concept is implemented as a Facebook canvas app , which supports embedding websites. Due to newer v2.0 restrictions, we used Facebook's Graph API v1.0 as it is the only one allowing use of user's friends data.

4.1 Bootstrapping. This module is responsible for completing the first phase of the initial component of `Reveal`, the *bootstrapping* phase, where the required data about the active user and her friends, is gathered and processed. To successfully acquire the information about the user's preferences as well as their social circles', we create *interest profiles* based on their Likes found in their personal Facebook profile, under the Music and Movies Edges [3]. Furthermore, in order to achieve fine-grained filtering and provide correct recommendations, fine-grained genre-extraction from each Like is of the essence. Consulting our Knowledge Base, we manage to extract entities (see 4.4) and gather useful information about their genres and categories.

Genre Scoring and Like weighting. Each Like represents an entity which fits in genres specified by our Knowledge Base. For each user and each of her friends, we create a hash, with each genre as the key and its number of occurrences among their Likes, as the value. This is considered the *genre profile* $P(p,G)$, where p is a person (user or friend) and G represents a associated genre. For instance, Bob has 3 likes in his Facebook profile, under music. *The Rolling Stones* fits in *Pop music,Rock music,Country,Rhythm and blues* genres, *AC/DC* fits in *Hard rock, Heavy metal, Rock music*, and *P!nk* that fits in *Alternative rock, Pop punk, Dance-pop*. This will result in the hash shown in Table 4.1.

Genre	Score
Pop music	1
Country	1
Rythm and Blues	1
Rock music	2
Hard Rock	1
Heavy Metal	1
Alternative Rock	1
Pop Punk	1
Dance-pop	1

Table 1: Sample genre profile.

In order to calculate the genre similarity between the user and her friends at a later stage, we also calculate an overall score of genres for each of them as shown in Equation 1.

$$OverallGenreScore(p) = \sum_{i \in P(p,G)} Score(i), \text{ p= User | Friend} \quad (1)$$

where $Score(i)$ is a genre score in *genre profile*. Also, we need to assign a *Weight* to each person's Likes , as it is necessary to know the scale of preference. This is done based on genre scores of the previous step, calculating an average of the genres score each Like fits in. Using the previous example, and naming as L a Like of p, calculation of *Weight* $W(p,L)$ is presented in Table 4.1.

Liked Item	**Weight**
The Rolling Stones	1.25
AC/DC	1.33
P!nk	1

Table 2: Sample Like Weight.

We also create an overall weight of Likes

$$OverallLikeScore(p) = \sum_{j \in W(p,L)} Weight(j) \tag{2}$$

for calculating Like similarity, at a later stage. At this point, if either the active user or any of friends has a small set or no Likes at all in their Facebook profiles, we use their post history for creating an *interest profile* instead.

4.2 Similarity. We devise a simple formula based on the Jaccard coefficient, with a few slight adjustments, as described below. To calculate the similarity among a user an her friends in a fine-grained matter, we use the genre scores and Like weights calculated in the bootstrap phase (see Section 4.1). By leveraging the in-depth profiling, we achieve fine-grained tuning for scoring for friends and posts.

To calculate a score for similar genres between a user and her friend, we aggregate the sum of scores of each overlapping genre. The same applies for both categories(music,movies).

$$genreSimilarityScore(user, friend) = \sum_{i \in P(user,G) \cup P(friend,G)} user[score(i)] + friend[score(i)] \tag{3}$$

Then, the sum of overlapping likes' weight is aggregated (same principle as above with genres).

$$likeSimilarityScore(user, friend) = \sum_{i \in W(user,L) \cup W(friend,L)} user[weight(i)] + friend[weight(i)] \tag{4}$$

Using equations 3, 4, we calculate the final score that the (active) user has with a friend in either category, which represents the actual overlap between them and is used for post score calculation in a later stage. We find that a weighted average is suitable, giving genre overlap double the weight, as it proved to boost accuracy in a more fine-grained fashion.

$$friendScore_{\text{cat}} = \frac{2 * genreSimilarity_{\text{cat}} + likeSimilarity_{\text{cat}}}{3}, \text{cat=music,movies} \tag{5}$$

Finally, to calculate scores for posts and, thus, populate the recommendation lists we follow the process below. First, we gather every post found in every single friend's wall. To calculate a post's score, a simple multiplication between the product of all of the post's genre scores and the respective weight of the poster (for the active user's profile) is conducted (Equation 6).

$$postScore = \Big(\sum_{i \in PostGenres} User[Score(i)]\Big) \cdot friendScore_{\text{cat}}, \text{ cat=music,movies} \tag{6}$$

The resulting item score is not capped in any scale, as the initial values of genre scores and item weights cannot be predicted. This approach also provides a discreet ranking among items.

4.3 Sentiment Analysis. Sentiment analysis is conducted before any post is analyzed for Named entity recognition. If the message contained in the post has negative polarity, it is discarded, thus optimizing the process. Only when the post's message is classified as positive, is it maintained for further analysis. We assumed that the absence of text in a specific post implies a positive polarity.

Analysis with SO-CAL. To conduct the sentiment analysis, we base our implementation on the Semantic Orientation CALculator (SO-CAL) [37] and use the included dictionaries. SO-CAL is an application used for sentiment polarity and strength extraction from text. It consists of the proposed algorithm and a set of dictionaries categorized by part-of-speech. There are 6 different dictionaries in the set containing 1542 Nouns, 1142 Verbs, 2824 Adjectives, 876 adverbs, and 217 Valence shifters. Its main advantage is the ranked dictionaries, and also the scoring heuristics which perform better than other dictionary-based approaches. Additionally, we made some modifications, such as changing the SO value of some words, simplifying the negation lexicon, and adding an emoticon lexicon which we manually ranked using info from Wikipedia (contains 110 entries).

Text pre-processing. In order to correctly analyze text, any Facebook tags (with @), and non-English or non-printable characters, are filtered out. Also, it gets split into sentences on punctuation points and is assigned a part-of-speech tagging.

Semantic strength tagging using dictionaries. SO-CAL uses dictionaries with words grouped by part-of-speech, ranked with valence strength [33] (-5,5 inclusive, 0 is excluded). Tagging is necessary as a word may be defined with different parts of speech which results in different valence strength [37]. For example the word "best" has a SO value of 5 as an *adjective* while it has 0 (neutral) as a *verb*. Each word and emoticon in our defined sentence, gets assigned with a value depending on its part-of-speech tag.

Valence shifters. Valence shifters are words carrying different semantic values than the words described so far. They are called shifters as they change the strength or effect of a lexical item when they are nearby [33] in a sentence. Their area of effect is limited, which is defined by various grammatical aspects. Each valence shifter is assigned an SO value, though it is applied differently on the score calculation. Specifically, it works as an additive multiplier on the initial SO value of the lexical item it shifts. The default multiplier for every word is trivially 1. For example, *robust* has a SO value of 2, and *really* has +0.2 multiplier which consequently is an intensifier, thus *really robust* is assigned a score of (1 + 0.2) * 2 = 2.4 .

Negation. There are many approaches to apply semantic negation. One of them is *switch negation* [35], where the SO value of a lexical item is reversed; e.g.,*good* has an SO value of 1, *not good* gets a SO value of -1. Laurence R. Horn [21] proposes that there is no semantic symmetry between negative and affirmative sentences. Supporting that, Taboada et al. [37] states that applying *shift negation* is more realistic approach linguistics-wise. This way, instead of reversing polarity, the SO value of a term is shifted towards the opposite polarity by a fixed amount. When following SO-CAL's exact directives the amount is 4. Now as a result, a term with strong meaning like *best* having a SO value 5, in the phrase *not the best* has an SO value of $-4+5=1$ instead of -5 and the word *terrifying* (-3) in *not terrifying* gets an SO value of 1, which is more realistic. Negators used here, are basically of two types; the word *not* and any word containing the suffix *n't*, like *couldn't* or *shouldn't*.

Scoring. As explained above, valence shifters and negators apply as modifiers to the SO-CAL value of the lexical term they refer to. To calculate the final SO value of a lexical term, we recursively apply any modifier value found searching backwards, until a determiner (e.g., a *comma* or sentence connective) is found. For example, *expensive* having an SO value -1, the phrase *very expensive* has SO value of (1+0.2) * -1 = -1.2, and *not very expensive* applied recursively will be (-4)*(-1) + [(1+0.2)*-1] = 2.8. For every lexical term in a sentence this calculation is applied and, finally, the sum of every sentences' SO value is aggregated, producing the total SO value of given text. Scores are unbound, providing information about the sentiment strength, but are ignored in our case as we are interested only on the aspect of semantic orientation. What is more, there is a fault-tolerance threshold set at 0.7 (Section 5), that if the SO final score of a text is greater or equal than that threshold, it is then classified as positive, or negative otherwise.

Irrealis blocking. Irrealis blocking, as described in the SO-CAL documentation and other work [26, 37], is the grammatical term describing that something has not happened yet as the speaker talks and, thus, the result or action is uncertain. This contains subjunctive, conditional and imperative mood which can be detected by a pattern module. Though, some of them, such as the imperative mood, may be semantically significant in our case, as it can be used to express sentiment upon a subject. That means, words in the effective radius of an irrealis marker (such as modals and quoted sentences), have a nullified SO value. In addition, any sentence that is grammatically a question is ignored.

4.4 Entity Extraction. In order to gather the needed information about the friends' posts, we have to analyze their entire activity. Our purpose is to define all the entities in a friend's post and, subsequently, keep those that are music or movie related. `Reveal` utilizes 3 edges from the Facebook Graph API to obtain the needed data. *Links* which are posts containing a URL, *Statuses* with plain text, and *Actions* which are posts with user generated social stories [2]. We only use *listens* and *watches* from social stories. To get a hold of such a task, our generated Knowledge base (Section 3.2), plays a core role to the entity identification and linking.

4.5 Links. Facebook defines as *Links* any posts containing a link/URL posted alongside a message. In order to extract entities out of this type of posts, we take advantage of Facebook's embedded items in posts, which provide various information about the shared link. Our system looks specifically for Youtube URLs, using a targeted `Video ID`-to-`Topic ID` (See 3.2) pair matching. From every URL we extract the `Video ID` and find its correspondent pair of `Topic ID` in the Knowledge Base. Our entities in the Knowledge Base are indexed (See 3.2) with their unique `Topic IDs`, so we are able to directly extract any of them related to a given youtube video(Figure 2).

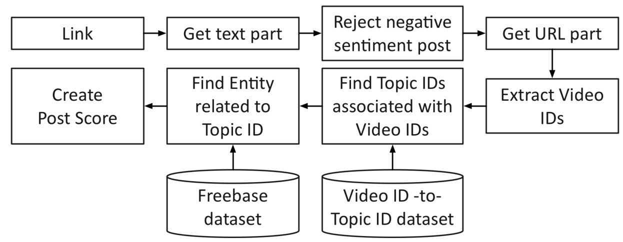

Figure 2: Overview of the link analysis process.

4.6 Statuses. Extracting an entity from such text, can potentially be accomplished with advanced pattern matching based on statistical learning [10], multi-heap based methods [25], or by using part-of-speech (POS) tagging [29]. Our experimental dictionary-based heuristic follows a more simplistic implementation of that proposed in [10], as it would required more complex algorithms, which is out of the scope of this work. Our approach takes plain text as an input ans is conducted in 3 steps. **(i) Filtering and POS-tagging**, **(ii) Production of candidate entities** and **(iii) Pattern matching and resulted entity**. We take advantage of the POS-tagging in order to set a limit on the candidates that can be produced. An action that limits the candidates is necessary as the resulting data from statuses is large. After the input is tagged, our algorithm creates candidate sequences based on certain grammatical criteria, such as steam of nouns,numbers and/or verbs, containing at most one connective and at most one *the* determiner. Leveraging the ranked text search of our database, and applying the Levenshtein distance to each candidate sequence, we extract entities that are lexicographically the most matched.

Statuses Handling inaccuracy Rationale. In social media, the flow of information is continuous and also presents many peculiarities. While a person shares information through social media, there are no rules to restrict the content, the context, the writing style, syntax, or the characters that will be used in order to present that piece of information. This lack of restrictions makes accurate entity extraction an even more difficult task. Having text formed as such, often leads to high false-positive rates. The noise is not only the result of the aforementioned, but also contributed by the names of the entities as well. Entities may come from different domains and have names or titles without any domain-specific relation. They can be part of every day phrases, expressions, or even single words. This problem, is exacerbated when trying to extract domain-specific entities i.e., music or movies, from text that can be related to any or more generic ones [16]. It could be more clear if we take real-life examples for both domains. For instance, consider the following popular bands: *Deep Purple*, *The Beatles*, *Blues Brothers*. None of these names follows a syntax rule, thus, it is pretty possible to find a statement like: "deep purple rock!" which compared to "I got a new deep purple dress with white stripes and it's awesome!" or "I love the beetles" and "Oh the beetles. Such magnificent creatures", may result in the extraction of the same entity. Regarding movie titles, entities can be found falling under the aforementioned cases. Another possible problem is that grammar mistakes may reduce the accuracy of extraction process. For example "Lock, stock and two smoking barrels" can be misspelled or with "2" instead of "two".

4.7 Actions. Entity recognition on these edges are almost trivial, as the name or title of listened or watched topic is bound to the post. However, there is a constrain enforced by Facebook as it is not possible to fetch any message posted with these actions; as a result we can't conduct sentiment analysis. Thus, we currently consider every item of this type, a candidate for recommendation. To find the corresponding entity of an *action* item, we consult our Knowledge base using the Facebook object's details, i.e. name. There is no need to identify their category as it is predefined by Graph API's edges. Though, we filter out and keep video items only designated by graph API,as movies.

4.8 Facebook Application. We implemented our application as a Facebook Canvas app, which supports `iframe` embedded websites. We leverage the Graph API v1.0 as it is the only version that supports

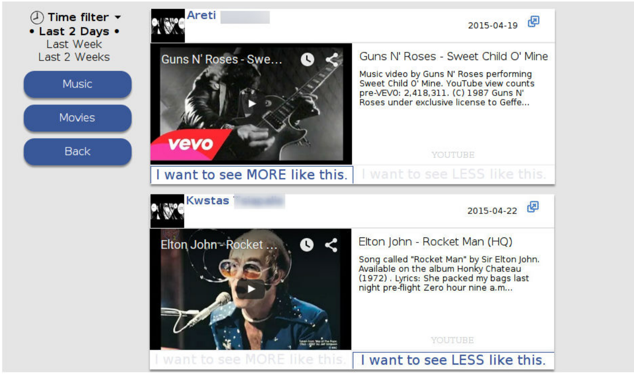

Figure 3: Screenshot of the recommendation page of our proof-of-concept implementation of `Reveal`.

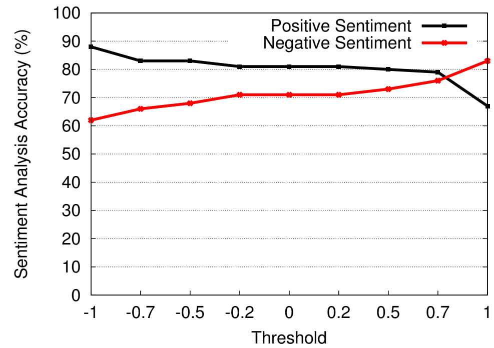

Figure 4: Threshold analysis.

friend lists, and was the officially supported version when we began our system implementation.

Initialization & score adjustment. `Reveal` creates the user's *interest profile* when used for the first time. Once the profile is created, the user is taken to the score adjustment page. It contains a list with friends and similarity scores for each category and genre. Contacts with no overlapping interests in any category are omitted. The user can tweak the scores for each user and genre on a scale of 0 to 100. The system will require an amount of time that depends on the number of friends and their posts, for populating the lists with relevant items. This part of our application is deployed only once by default, but can be accessed at any time by the user for further tweaking the scores.

Main Menu . In this simplified menu, users can navigate to different pages of the application. Specifically, they can choose which list they want to see, but can also adjust their friends' scores as mentioned. If any score adjustment takes place, there is no stall time anymore and changes are applied instantly.

Music & Movies Top-10 Lists (Figure 3). These two pages behave exactly the same, each one for its respective category. There is a navigation menu to the left, from which the user can switch between the two categories or go back to the main menu. Above the navigation menu, the "Time filtering menu" is located for the user to choose the time period she is interested in, which can extend up to two weeks in the past. In this page the recommended list is presented to the user. In our prototype, every list will contain up to 10 recommended items, but this can be easily changed for arbitrarily large numbers. If a list is empty, we leverage the most preferred genres the user has expressed on the current category, and recommend up to three random videos. Each item in the list, contains all the necessary information for the users to access; the contact that posted tge content originally, publishing dates, and the embedded video related to the entity mentioned in the friend's post, along with the video's description as provided by Youtube. Users can also view the original post by clicking a pop-up icon. Furthermore, users are able to express their opinion on the recommended items. Pressing one of two buttons attached to the lower boundary of each item, will tweak the importance of genres that the post fits, in the users' *interest profile*.

5. EXPERIMENTS

In this section, we describe and present the experiments we conducted for evaluating the effectiveness of our proof-of-concept implementation's heuristics and algorithms. For each experiment, we describe the preparation, the datasets used and, finally, present our findings.

Named entity recognition in Youtube links. As entity recognition from a **Links** edge provides the most valuable information, it is necessary to measure the performance of the heuristic that maps maps the unique video IDs to unique topic (and thus entity) IDs. TO obtain accurate results, it is necessary to filter out any post that does not contain a valid Youtube URL. The dataset was collected from music and movie related Facebook Groups and contained 5,310 links with a Youtube URL. Our heuristic managed to discern 4,743 valid titles-entities, reaching a coverage of 89.32% and demonstrating the robustness of our approach.

Sentiment analysis threshold. Subsequently, we needed to evaluate our sentiment analysis module, and its effectiveness in providing accurate results regarding the sentiment of a given post's content. We obtained datasets released in prior work, that contained English Tweets and Facebook comments, which posed a reasonable match to the type of content we expect our system to handle. Specifically, we used:

- Sanders Analytics Twitter dataset [9] (5,513 tweets)
- Facebook Comments v1.0 [42]

First, it was necessary to ignore any text with a truly neutral sentiment i.e.,text with a sentiment score of 0, as it will not offer any valuable semantic information upon a subject. Then, to regulate the semantic noise i.e.,false positives/negatives, we experimented with sentiment score thresholds, that specified whether a post is classified as positive or negative in text with actual sentiments expressed (if a piece of text has a score different than 0, it carries a sentiment value). The results of our algorithm for each text were compared with the predefined polarity found in the aforementioned corpora, in order to calculate accuracy. Finally, the experiment was conducted using 31 discreet values, in the [-1.5,1.5] range. Nine present significant results, as any value smaller than -1 or larger than 1 expanded the chasm between positive and negative accuracies. Additionally, some values between -1 and 1 inclusive, offered no significant result which we could leverage for our system. The findings of this experiment are shown in Figure 4.

The experiment showed that for thresholds close to zero(-1 to 1 exclusive), the chasm between positive and negative accuracies was small and the overall accuracy was satisfactorily high, for both. Also, we saw that when the threshold was set either to -1 or 1, performance abruptly dropped significantly. The most efficient setting was at 0.7, where both positive and negative accuracy was high and close, which lead us to used that threshold in our system's sentiment analysis function.

Relevant Revealed Content. In our next experiment, we wanted to explore which types of posts from contacts should be handled by `Reveal`. In our initial experiments with 50,000 status updates, we found that our system would extract a very large number of entities, that were not relevant to our 2 categories of interest. This high level of false positives led us to the decision to ignore such posts and process all posts the belong to one of the following three Graph API edges; `Links` (any post with an embedded Youtube video), `Watches`, and `Listens`. Table 5, shows the relevant content found by our system's heuristics, extracted by Facebook posts with any of the aforementioned Edges, from 3,493 accounts that were connected to our pilot study's participants. For our prototype implementation we ignored `Links` with content from other domains apart from Youtube (e.g., Vimeo). Furthermore, we also identified cases of `Links` with invalid URLs; this included old Youtube links thawere no longer valid, or that had not been pasted in their entirety. As a result, 59.98% (94,626 of the 157,762 posts) were found with this inefficiency and, thus, removed. Of the remaining valid posts, approximately 14.5% contained relevant content under the music and movies categories.

Type	# of posts
Total (incl. status updates)	521685
Links	45438
Watches	28994
Listens	1262
Relevant	75694
Movies	28994
Music	46700

Table 3: Relevant Content.

Reveal vs Facebook. The goal of the next experiment was to confirm that users missed content of relevance due to Facebook's personalization algorithm for the users' News Feed. To achieve that, we gathered every News Feed post from 7 of our participants, without any data being filtered out, for the time period of the previous 2 weeks (maximum allowed by Graph API). We then compared with the content posted by all their contacts and processed the data with `Reveal`, for identifying the posts of interest to the users. We then compared the two datasets to identify the common items selected by the systems. As can be seen in Table 5, not only did our system select posts that Facebook already presented in the News Feed, but pinpointed more than twice as many additional posts that are suitable for recommendation, as they match the users' *interest profiles*.

User	News Feed	Common	Additional by Reveal
1	2,335	319	165
2	983	3	481
3	919	120	333
4	968	115	344
5	1,034	117	402
6	12,137	572	531
7	6,170	434	871
Total	24,546	1,680	3,127

Table 4: Number of recommendations selected by Reveal, compared to the content presented in the users' News Feeds by Facebook.

Running Time. As the system has to process a really big amount of data (always depending on the amount of friends and the available information in their profiles), it is necessary to measure the time needed to achieve that. As our application stands as a proof-of-concept, performance optimization was not a priority, though we needed to test if running times are reasonable. Furthermore, our system is designed as part of the social network, and not an external service. The results in Figure 5 show the time needed for 100 friends to be analyzed by `Reveal`, while running on a commodity desktop. The test was ran on 12 different users, for 100 randomly selected friends, with an average running time of 144 seconds. The times are dependant on the amount of information that each of the friends has posted. Furthermore, this step is conducted only once, when the user's profile is created. Given our results, we believe that these running times are within an acceptable range, and could easily be optimized when running on the social network's servers.

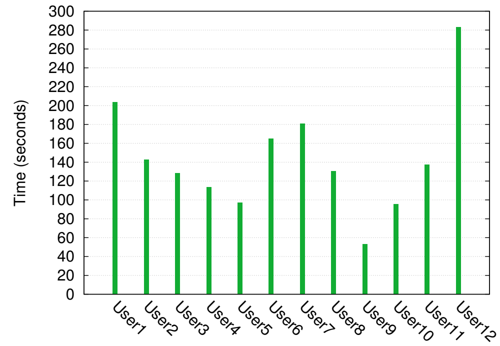

Figure 5: Profile creation running time for 100 sampled contacts.

6. USER EVALUATION

Our user evaluation was a pilot experiment that included 5 participants (not from the initial user study group). Our goal was to allow casual users to extensively interact with the prototype within Facebook, with their own accounts. We collected feedback (Likert scale, 1-5) on the fine-grained recommendations content as well as the user experience while interacting with the application. The users were asked to install the prototype application and use it for at least a few minutes, and at least 3 times within a week. They were asked to take notes of any new content the system presents, the level of interest for each finding, the accuracy of the scoring and to explain, if used, the need to adjust scores or ranking of the recommendations. At the end they were asked to provide usability feedback via online forms.

One of the major findings was that the users were given adequate number of recommendations, taking into account the number of Facebook friends they had. From those, on average, 32% were new posts with more than 50% classified as interesting. As seen in figure 6, users reported very high accuracy and interestingness of video and music posts, 82% and 84% on average, respectively. The new content found in `Reveal` was deemed very adequate and quite rich and diverse. Those qualities were attributed based on the use and revisiting of the application in the one week time span of the evaluation session. This is a direct improvement over the initial user study where casual users reported that it was disappointing to have to spend time and effort to scan their timelines in Facebook for interesting recommendation posts.

The feedback on the system usability was more or less expected, given the fact that the whole approach adopted a simple and clear design, as showcased in section 4. As can be seen in figure 7, the

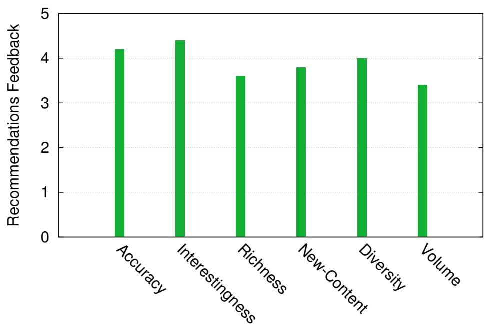

Figure 6: User Feedback on Recommendations (content).

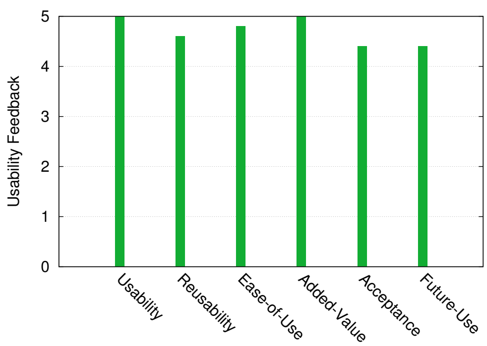

Figure 7: Usability Evaluation.

participants easily navigated and used the application for the whole duration of the evaluation. After a week-long use, the users reported that they were pretty familiar with the way the application works and easily integrated `Reveal` into their social activities' workflow. The general belief was that this recommendation approach would be welcomed for additional domains that they may require recommendation in the future, recognising a high potential in the added value of our proposed approach.

7. RELATED WORK

Social Content filtering for rating. In multiple previous work, the use of social networks to obtain more accurate data regarding users' preferences and needs, was leveraged for overcoming problems that affect recommender systems, such as cold-start, item sparsity and explicit user ratings [22, 23].

Bellogin et al. [11] presented a Facebook app (MyMoviesHistory) that makes use of social context and the activity of users, either within or outside social groups and pages. That said, leveraging activities like *watching* a movie, *with* friends or groups and *when*, provides their system with useful information and disengages the users from the initial item rating. It renders feasible the use of collaborative filtering algorithms without explicit user ratings and reduced the cold-start and sparsity effects, due to the information gathered. However, this approach requires high social activity from users, either individually or inside the mentioned circles (to produce recommendations), and requires further user interaction for adding new items in her profile.

Similarly, Shapira et al. [36], suggested that apart from using social context manipulation to mine rating scores among users and social circles, one should also consider the number of clicks on links posted, as an improvement to the rating method. In addition, in another work focusing on social recommendations, Venkatesan et al. [39], worked mainly on solving the cold-start issue, and introduced the clustering of -not necessarily connected- user profiles with similar taste. Top-genre movies were recommended inside each cluster explicitly, with a minimal cold-start impact. Though this approach does not take any actual social connection between users into account, and may ignore content that is not very popular, it could still be of interest to users. Similar approaches were followed by Carmagnola et al. [14] who used information of related users in social networks, and how trending topics among their circles affect their likings. He et al. [20], introduced a recommender system with a common but more fine-grained rating system using collaborative filtering, and incorporated semantic filtering of social networks. What is more, Bonhard et al. [12] proposed the use of ratings to produce suggestions in controlled networks of familiarity.

Trust-Based Profiling. Another issue under discussion is how a system can determine trust between users, and leverage that information and assign weights among them to improve recommendations. By leveraging user activity and publicly accessible information, trust-based profiles are created, which can boost the recommendation process [28, 43].

Cakiroglu et al. [13], complements the use of social circles for similarity extraction, with the selection of trusted reviewers inside those circles, for boosting prediction accuracy. Furthermore, in websites such as FilmTrust [17], users are required to rate their friends for trust. This process follows the same guidelines as explicit item-rating. Accordingly, the higher rated contacts become the more reliable source of recommendations. Furthermore, Gretarssonet al. [19], developed an interactive Facebook app, where users can assign weights not only on their items of preference, but also on their friends, by dragging nodes in a graph projecting their profile. The Smallworlds application, presented to each user a graph with tiers, containing her likes, her friends and their respective likes. The distance on screen of each (interactively dragged) node, represented the weight of preference for each item on each of the three tiers. That way, their system managed to exploit social connections and content and provided recommendations, using simple coefficients as rating formulas. On the other hand, it relies explicitly on user interaction. Quijano-Sanchez et al. [34], developed a Facebook application that recommended movies, in specified facebok groups. Using them as trust groups, the system generated interest profiles and combined the trust information to make recommendations. Similar work was done by Jamali [22], proposing trust propagation i.e., trust among direct or indirect connected users in social networks, to further improve such systems.

Sentiment analysis as supplemental rating. The supplemental use of text reviews is also proposed, in order to tell if a user is expressing likeness or disaffection on a topic. Pak [32], mined opinions from a plethora of topics (politics, technology, multimedia) using tweets. Integrating opinion-mining in recommender systems could provide more accurate information about users' moods, which could be a significant improvement to topic rating. In a similar fashion, Cakiroglu [13] integrated opinion-mining in simple comments that users made after they were asked to. However, this is fairly simple approach for very targeted applications. Furthermore, Leung [24], incorporated the use of textual user reviews in collaborative filtering algorithms for movie recommendations.

Entity extraction in social networks. The ability to extract entities from text and topics, is a crucial aspect of the recommender system, especially in the context of social networks. Valuable data can be mined from such resources, however accurately applying this process is a complicated task. Miller et al. [29] used part-of-speech tagging in collaboration with statistical parsing to acquire accurate results. Gattani proposed the use of a Knowledge-Base with semantically interconnected topics, and sophisticated heuristics to extract entities related to multiple domains from tweets [16], and at the same time handle the processing of large real-time streams of content. It is clear, that in order to achieve high accuracy on extracted entities, the use of more sophisticated heuristics are demanded, such as statistical learning and advanced algorithms for natural language processing [10, 25].

8. LIMITATIONS AND FUTURE WORK

In this section we discuss certain limitations of our current work, as well as potential directions to follow as part of our future work.

Facebook API. While our system is meant to be integrated within a social network for extending its functionality, our proof-of-concept application allowed us to conduct a user study and obtain useful feedback regarding its usability and functionality. However, the result of Facebook updating to the Graph API v2.0 this year [4], was that apps can no longer obtain information about a user's contacts that haven't also installed the app. While this can significantly improve user privacy, it also prevents our prototype app from accessing information required for operating.

Language. Our system is designed to process content and produce accurate recommendations for posts in the English language. The part-of-speech taggers for other languages are crucial for achieving universal access, as both sentiment analysis and entity extraction are based on such tools. As our system is designed to analyze the sentiment of text, expanding support to include more languages will require the appropriate dictionaries and tools for each language. This lies out of the scope of our work, as our goal is the demonstrate the feasibility and practicality of our prototype system, and highlighting its usefulness within modern social networks.

Slang and typos. As discussed in Section 5, slang terminology can impact the accuracy of our content processing. Similarly, abbreviated versions or mis-typed words are common instances, and further exploration is required for effectively handling all such cases. However, major services such as Facebook are already focusing on processing and automatically understanding slang terminology [5].

Supported content categories. In this paper, we have focused on two specific types of content, namely movies and music. However, our approach is applicable to a number of topics, and could be extended for supporting various categories of content. Potential directions could include a wide range of topics including literature, sports, traveling, and politics. Our main source of datasets, Freebase, already contains data about the aforementioned topics which will streamline the implementation of the entity extraction process. However, further exploration is required for identifying cases where entities are assigned to multiple supported categories, and designing techniques for determining the likelihood of belonging to each category, and using collateral information from the post for further refining the extraction process. For example, when processing a post containing "In Bruges is fantastic!", our system would have to infer the reference to the movie titled "In Bruges" and not the city of Bruges. This will also require our system to expand its content analysis process to handle status updates, which can be problematic for the entity extraction step when no specific media content is present, as described in Section 5.

9. CONCLUSION

In this paper we explored the utility of a recommender system for social networks, designed to identify content published by the user's contacts that matches a fine-grained interest profile that is automatically generated from each user's disclosed interests (i.e., likes and groups). Our approach is designed as a complimentary mechanism to the main content selection algorithm already in place in social networks (e.g., News Feed in Facebook), which is influenced by the amount of interaction with each contact. While this is a intuitive and effective selection criteria for general content (e.g., general posts and life events), it is not optimal for "entertainment-based" content. Our driving motivation is that content that significantly matches a user's interests can be missed because it is published by contacts that seldom interact with the user. We first conducted a qualitative user study that allowed us to identify the expectations of participants regarding the ability of their contacts to publish content of interest for our supported categories (i.e., music and movies). We built a proof-of-concept application for Facebook, and explored the practical aspects and intricacies of processing and extracting information from user-published content. Our subsequent pilot study, highlighted the effectiveness of our approach, and the suitability of online social networks as a information-rich ecosystem for providing fine-grained recommendations to users.

REFERENCES

[1] Emarketer - social, digital video drive further growth in time spent online. http://www.emarketer.com/Article/Social-Digital-Video-Drive-Further-Growth-Time-Spent-Online/1009872.

[2] Facebook - using actions, . https://developers.facebook.com/docs/sharing/opengraph/using-actions.

[3] Facebook - graph api edges, . https://developers.facebook.com/docs/graph-api/overview/#basics.

[4] Facebook - api version v2.0, . https://developers.facebook.com/docs/apps/changelog#v2_0_login.

[5] Wired - why facebook is teaching its machines to think like humans, . http://www.wired.com/2013/10/sentiment-analysis-deep-learning/.

[6] Freebase - a community-curated database of well-known people, places, and things. https://www.freebase.com/.

[7] Facebook - how does news feed decide which stories to show? https://www.facebook.com/help/166738576721085.

[8] Urban dictionary. http://www.urbandictionary.com/.

[9] Twitter sentiment corpus, 2011. URL http://www.sananalytics.com/lab/twitter-sentiment/.

[10] S. Ashwini and J. D. Choi. Targetable named entity recognition in social media. *arXiv preprint arXiv:1408.0782*, 2014.

[11] A. Bellogín, I. Cantador, P. Castells, and F. Díez. Exploiting social networks in recommendation: a multi-domain comparison. In *DIR*, pages 40–41. Citeseer, 2013.

[12] P. Bonhard and M. A. Sasse. 'knowing me, knowing you' – using profiles and social networking to improve recommender systems. *BT Technology Journal*, 24(3), July 2006.

[13] S. Cakiroglu and A. Birturk. Suggest me a movie: A multi-client movie recommendation application on facebook. In *ISCIS*, volume 62, 2010.

[14] F. Carmagnola, F. Vernero, and P. Grillo. Sonars: A social networks-based algorithm for social recommender systems. In *User Modeling, Adaptation, and Personalization*, pages 223–234. Springer, 2009.

[15] T. De Pessemier, S. Dooms, J. Roelandts, and L. Martens. Analysis of the information value of user connections for video recommendations in a social network. In *2nd International Workshop on Future of Television held at the 9th European Conference on Interactive TV and Video (FutureTV-2011)*, pages 1–7, 2011.

[16] A. Gattani, D. S. Lamba, N. Garera, M. Tiwari, X. Chai, S. Das, S. Subramaniam, A. Rajaraman, V. Harinarayan, and A. Doan. Entity extraction, linking, classification, and tagging for social media: A wikipedia-based approach. *Proc. VLDB Endow.*, 6 (11), Aug. 2013.

[17] J. Golbeck, J. Hendler, et al. Filmtrust: Movie recommendations using trust in web-based social networks. In *Proceedings of the IEEE Consumer communications and networking conference*, volume 96, pages 282–286, 2006.

[18] S. Gouws, D. Metzler, C. Cai, and E. Hovy. Contextual bearing on linguistic variation in social media. In *Proceedings of the Workshop on Languages in Social Media*, LSM '11, 2011.

[19] B. Gretarsson, J. O'Donovan, S. Bostandjiev, C. Hall, and T. Holerer. Smallworlds: Visualizing social recommendations. *Comput. Graph. Forum*, 29(3):833–842, 2010.

[20] J. He. *A Social Network-based Recommender System*. PhD thesis, Los Angeles, CA, USA, 2010. AAI3437557.

[21] L. Horn. *A Natural History of Negation*. University of Chicago Press, 1989.

[22] M. Jamali and M. Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 135–142. ACM, 2010.

[23] I. Konstas, V. Stathopoulos, and J. M. Jose. On social networks and collaborative recommendation. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 195–202. ACM, 2009.

[24] C. W. Leung, S. C. Chan, and F.-l. Chung. Integrating collaborative filtering and sentiment analysis: A rating inference approach. In *Proceedings of the ECAI 2006 workshop on recommender systems*, pages 62–66. Citeseer, 2006.

[25] G. Li, D. Deng, and J. Feng. Extending dictionary-based entity extraction to tolerate errors. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, 2010.

[26] B. Liu. *Sentiment Analysis: Mining Opinions, Sentiments, and Emotions*. Cambridge University Press, 2015.

[27] B. Liu, M. Hu, and J. Cheng. Opinion observer: Analyzing and comparing opinions on the web. In *Proceedings of the 14th International Conference on World Wide Web*, WWW '05, 2005.

[28] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King. Recommender systems with social regularization. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 287–296. ACM, 2011.

[29] S. Miller, H. Fox, L. Ramshaw, and R. Weischedel. A novel use of statistical parsing to extract information from text. In *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference*, NAACL 2000, pages 226–233, 2000.

[30] S. Mohammad. From once upon a time to happily ever after: Tracking emotions in novels and fairy tales. In *Proceedings of the 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 105–114. Association for Computational Linguistics, 2011.

[31] F. Å. Nielsen. A new ANEW: evaluation of a word list for sentiment analysis in microblogs. *CoRR*, abs/1103.2903, 2011.

[32] A. Pak and P. Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In *LREC*, 2010.

[33] L. Polanyi and A. Zaenen. Contextual valence shifters. In *Computing attitude and affect in text: Theory and applications*, pages 1–10. 2006.

[34] L. Quijano-Sánchez, J. A. Recio-García, B. Díaz-Agudo, and G. Jiménez-Díaz. Happy movie: A group recommender application in facebook. In *24th International Florida Artificial Intelligence Research Society Conference, FLAIRS*, pages 127–134, 2011.

[35] R. Sauri. *A Factuality Profiler for Eventualities in Text*. PhD thesis, Waltham, MA, USA, 2008. AAI3304029.

[36] B. Shapira, L. Rokach, and S. Freilikhman. Facebook single and cross domain data for recommendation systems. *User Modeling and User-Adapted Interaction*, 23(2-3), 2013.

[37] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede. Lexicon-based methods for sentiment analysis. *Comput. Linguist.*, 37(2), jun 2011.

[38] J. Tang, X. Hu, and H. Liu. Social recommendation: a review. *Social Network Analysis and Mining*, 3(4):1113âĂŞ1133, 2013.

[39] M. Venkatesan and A. Mai. Recommendation of tv shows and movies based on facebook data. 2012.

[40] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi. On the evolution of user interaction in facebook. In *Proceedings of the 2nd ACM workshop on Online social networks*, pages 37–42, 2009.

[41] C. Wilson, A. Sala, K. P. Puttaswamy, and B. Y. Zhao. Beyond social graphs: User interactions in online social networks and their implications. *ACM Transactions on the Web (TWEB)*, 6 (4):17, 2012.

[42] K. Zhang, Y. Cheng, Y. Xie, D. Honbo, A. Agrawal, D. Palsetia, K. Lee, W.-k. Liao, and A. Choudhary. Ses: Sentiment elicitation system for social media data. In *Proceedings of the 2011 IEEE 11th International Conference on Data Mining Workshops*, ICDMW '11, 2011.

[43] X. Zhou, Y. Xu, Y. Li, A. Josang, and C. Cox. The state-of-the-art in personalized recommender systems for social networking. *Artificial Intelligence Review*, 37(2):119–132, 2012.