ABSTRACT
Recommendation systems leverage future internet services to predict personalized recommendations for products, services, media entities or other offerings. Based on the research and development of the FContent 2 initiative, we introduce an approach to compensate Cold Start and Sparsity Problems by analyzing semantics of external textual data, in terms of comments from social networks as well as item reviews from product and rating services. Thereby sentiment analysis and semantic keyword extraction approaches are explained and evaluated by using preliminary implementations. The mined data is transferred into, so called, preference ontologies describing the users interest in automatic analyzed topics and subsequently mapped to the properties of items in order to calculate the associated recommendation value.

Categories and Subject Descriptors
H.3 [Information Storage and Retrieval]: Miscellaneous; L.2.7 [Artificial Intelligence]: Natural Language Processing

Keywords
Recommendation Engine, Preference Ontology, Cold Start, Sparsity, Semantic Keyword Extraction, Sentiment Analysis

1. INTRODUCTION
One of the main challenges in recommendation systems is to solve the so called Cold Start Problem [1]. When new items or users are introduced, a typical recommender engine does not know anything about this element. Usually new registered users need to enter manually demographic data, preferences or at least give feedback to some offerings in order to let the engine predict personalized recommendations. For collaborative filtering new items need a set of feedback data to be recommended in order to avoid random recommendations or even skipping over this item.

In this paper we want to introduce an approach that analyses texts from third party service providers, such as social networks and review websites, in order to enrich the item and user data of recommender systems and thus allow an adequate personalization. The focus of this explanation is the textual analysis engine for German texts. Beyond direct user input from social networks like Facebook where users are able to like, share or rate items, we analyze opinions, in terms of sentiments, and extracted keywords. The underlying approach of generating preference ontologies is introduced in section 3. Section 4 looks at semantic keyword extraction and section 5 at the sentiment analysis as well as the used data sources and frameworks. Our application prototype for a preliminary analysis is a personal EPG which will be explained in more detail in section 6 and finally section 7 concludes with a summary and an outlook.

2. RELATED WORK
Besides the well-known Content-based and Collaborative Filtering, Recommendation Filtering is sometimes classified as new group representing a mixture of both Filtering approaches, the so called Preference Filtering. [2] Its goals are to find items that fit to the users’ preferences. These preferences can originate from manual user input or by learning them automatically. Therefore user interests (represented as attributes) are mapped to the item attributes, such as a sports interest of a user is mapped to the according category of an item (e.g. a sports program).

There are some well known problems almost every Recommendation Engine has to deal with. One of them is the Cold Start Problem. At the initial start of a recommender or even when a new item or user registers there is no meta data the engine can work with [3]. Furthermore a recommendation engine has to be capable of compensating the Sparsity Problem [4]. In most cases there is a vast quantity of items as well as many users who however lack any coherency. For example, a particular user of a video-on-demand-service only watches less than 1% of the offered items. Some users consume just a few items and even some items have never been watched. So the user-item matrix is really sparse. Jia Zhou and Tiejian Luo are grouping the methods solving this problem into two general classes: Dimensionality Reduction ignores less important rows and columns of the matrix. This implicates a loss of possible useful information. In contrast,
the second approach aims at finding and adding additional information, thus rows and columns are filled with new values.\[5\]

This paper focuses on the latter approach by automatically analyzing Lexical Semantics, where a sentence, no matter how complex, is treated as structured sequences of single words. The combination of these words results in an overall sense understandable first and foremost for humans. More and more machine learning algorithms try to analyze the meaning of the sentences in order to extract the intended semantic and to make it interpretable even for computers. A sentiment analysis is one approach for information retrieval from texts and describes the allocation of the author’s opinion of his writing to a numerical value.\[6\] The bi-polar polarities, called sentiments, are in most cases “positive” and “negative”.

3. PREFERENCE ONTOLOGIES

There are various approaches to compensate the data sparsity in Recommendation Engines. Some try to ask users about the users preferences.\[7\] Some try to offer a common set of items and ask the user to rate them.\[8\] Others try to add third party data such as feedback from external databases.

The blue box in picture 1 shows the well-know relationship between users and items for Collaborative Filtering in recommendation engines. The association between the two entities, User and Item, can consist of explicitly given feedback, such as likes, or consumption data – e.g. the watch time. Moreover the users feedback could be implicitly mapped to the properties of that item, as well. The connection of \( n \) users to \( n \) items is normally represented in the user-item-matrix. In case of the cold start problem this user-item-matrix is very sparse.

The red and green boxes in picture 1 mark the novel approach. First the ontology building step in the red box: A preference ontology consists of the actual user, a topic (issue), and an opinion on this topic. External data sources are analyzed to retrieve all important topics the user is writing about. The bi-polar polarities, called sentiments, are in most cases “positive” and “negative”.

The green box indicates the mapping of the preference ontology to properties of items to be predicted in the recommendation engine. This mapping value is represented by a normalized numerical distance, where 0 means the preference is equal to the property – e.g. an analysis of Facebook estimates a high interest in sports and the genre of the current item is sport, as well. The higher the values, the more diverse the entities. Thereby it does not matter if it is a semantic distance or a distance based on common usage transactions.

The main aspect of this paper is the ontology building process using a self-designed semantic analyzer which is developed to extract meaningful data out of unstructured texts from existing web services. It consists of two core components. The first one tries to identify semantic keywords from texts summarizing the topic by searching for the most meaningful terms that represent this text. The second approach tries to allocate an implicit feedback, in terms of the authors opinion to the identified topics (sentiment analysis).

4. SEMANTIC KEYWORD EXTRACTION

The semantic keyword analysis aims at identifying important words in a semantic network that summarizes the core aspects of a text. We assume a manual chunking and shallow parsing process before using the system in order to process only reasonable text corpora. Movie review texts, for instance, normally consist of a short synopsis and the actual critic. While the synopsis is valuable for the keyword extraction, it causes an offset in the authors sentiment as most movies are about to solve a problem. The review section in turn is more appropriate for a sentiment analysis than for a semantic keyword extraction.

At the beginning the TreeTagger [9] (a Part of Speech Tagger with stemming algorithms of the University of Stuttgart) identifies word stems of the selected text. A term extraction algorithm identifies the most important words by analyzing the POS Tags (e.g. nouns, proper names, adjectives) and removes unimportant and falsified stop words. For instance, these stop words are "sein", "haben", "können", "dürfen", "müssen", the German words for "be", "have", "can", "may", "must". We discovered that these words are too general to be used for the identification of semantic keywords.

The next step is to match the important words with a self made graph database containing all the words and relations from OpenThesaurus [10], a German word net with synonyms, hyperonyms and hyponyms, as well as with GermanNet [11], a lexical semantic net for German similar to its English equivalent WordNet [12]. The implemented processing algorithms of the graph database retrieve a set of terms representing the hyperonyms of all words by recursively analyzing the respective superordinate, its superordinate and so forth.

A simple example is shown in figure 2 and portraits a text containing the following words marked green: two times the word "car", one time "bicycle" and one time "train". The grey terms show their hyperonyms. The strength of the grey color indicates the keyword importance. The darker, the more important. Actually this approach is designed for German texts, but should also work when using word nets of other languages like English.

The following formula shows the calculation of the "nearest common hyperonym" of two words and thus, is the relevance value for a keyword in the given context:

\[
value_{\text{keyword}}(h) = \sum_{n=0}^{N} \left(1 + \frac{1}{1 + e^{w_{n,h}}}ight)
\]
$h$ is the keyword to be determined. $W_h$ is the set of important words $w$ in the analyzed text having $h$ as a hyperonym and thus $W_{h,w}$ is its amount. $e_{w,h}$ is the number of edges between the word $w$ and its hyperonym $h$. The weight of a single edge is 1. The higher the resulting $\text{value}_{\text{keyword}}$, the more important the keyword $h$.

According to this, the term "vehicle" has the highest predicted $\text{value}_{\text{keyword}}$ with 5.25, followed by "means of transport" with 4.95 and "wheeled vehicle" with 4.33. In contrast to the original words "car" (value of 4.00) as well as "bicycle" and "train" (each 2.00), the terms "two-wheeler" and "railroad" only retrieve a value of 1.50.

This approach is used to predict the keywords of a corpus automatically, that represent the topics of the given texts. The amount of relevant keywords depends on the text size and the calculated keyword value. Therefore future work is needed to identify reasonable thresholds.

5. SENTIMENT ANALYSIS

The approach of analyzing the sentiment tries to map the author’s opinion on his written text (implicitly on the analyzed topics) to a numeric scale of $[-1, 1]$, where $-1$ is the worst opinion and $1$ is the best. The generated feedback data is allocated to the identified keywords and thus, represents the preference ontology.

Similar to the Semantic Keyword Engine, we assume a manually text corpus selection. The first step is to get sentiment values for all word stems. SentiWS [13] is a sentiment database offering 31,000 lemmas, polarities, in terms of sentiment. In addition we apply a self designed grammar and a positive verb result in a negative combination of the sentiment. A simple example is the combination of a negative adjective and a positive verb result in a negative combination of the sentiment. In addition we apply a self designed grammar for recursively combining phrases with allocated polarities called Alexis. It takes shift words (e.g. "not" turns the sentiment) as well as intensifier (e.g. "very") and reducer (e.g. "little") into account. After composing sentiment values of single words to a combined sentiment, this step is repeated with combined sentiments until there is only one sentiment left for the whole sentence.

As a starting point, we have analyzed 948 randomly selected German texts, where we manually labeled an expected sentiment value in range of $[-1,1]$. These texts are taken from different sources that can be categorized in comments of social media (Facebook.de, Twitter.de), reviews of products (Amazon.de, Chip.de, Connect.de, Douglas.de, eBay.de, Testberichte.de), services (Lieferheld.de) and media items (Filmstarts.de, MoviePilot.de) as well as news (Sportbild.de, Welt.de).

The results of the case study can be seen in the table 1. Even though the Mean Absolute Error (MAE) is small for each algorithm and setting, the results are very scattered, as you can see by the deviation to the Root Mean Squared Error (RMSE). A weighted combination (averaged by algorithm) and by used word net for relations) indicate better performance when using OpenThesaurus for searching for more synonyms and hyperonyms. The mean average of all algorithms indicate a better performance then single ones, as it minimizes outliers. An optimized algorithm, in contrast, minimize the error values the best by weighting the single algorithms.

The most service providers which offer product reviews have a good accuracy: Connect (accuracy of 90.0%), Chip (89.9%) and Testberichte (87.6%), but the lowest values originate from other product sellers, as well: Amazon (77.7%), Douglas (73.4%) and eBay (76.4%). Social Networks show the highest average value (Facebook: 86.6%; Twitter: 86.8%) and thus, seem to allow an accurate prediction of the users interest in a specific topic.

Clustered by the amount of words in a text, the system is more accurate when processing less words (at the best between 3 and 50), but in average the amount of words does not have a huge impact on the accuracy. However, more important is the labeled or expected sentiment. While neutral (and positive) sentiments can be predicted the best, negative sentiments seem to be very unpredictable. At all, accuracy based on MAE when using a hybrid algorithm is about 84.123% of right allocated sentiments.

### Table 1: Accuracy (accy) of averaged (avg.) and weighted optimized results over all 948 texts

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>avg. sent.</th>
<th>MAE (MAE)</th>
<th>accy (MAE) in %</th>
<th>RMSE (RMSE)</th>
<th>accy (RMSE) in %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Labeled</td>
<td>0.196</td>
<td>0.596</td>
<td>70.200</td>
<td>0.742</td>
<td>61.399</td>
</tr>
<tr>
<td>Statistic</td>
<td>0.746</td>
<td>0.596</td>
<td>70.200</td>
<td>0.742</td>
<td>61.399</td>
</tr>
<tr>
<td>StatisticAll</td>
<td>0.103</td>
<td>0.633</td>
<td>68.375</td>
<td>0.795</td>
<td>60.234</td>
</tr>
<tr>
<td>Klenner</td>
<td>0.075</td>
<td>0.503</td>
<td>74.846</td>
<td>0.709</td>
<td>64.536</td>
</tr>
<tr>
<td>Alexis</td>
<td>-0.157</td>
<td>0.520</td>
<td>73.980</td>
<td>0.721</td>
<td>63.930</td>
</tr>
<tr>
<td>3 to 20 words</td>
<td>0.100</td>
<td>0.293</td>
<td>85.311</td>
<td>0.542</td>
<td>72.899</td>
</tr>
<tr>
<td>21 to 50 words</td>
<td>0.149</td>
<td>0.288</td>
<td>85.555</td>
<td>0.537</td>
<td>73.125</td>
</tr>
<tr>
<td>51 to 100 words</td>
<td>0.219</td>
<td>0.326</td>
<td>83.666</td>
<td>0.571</td>
<td>71.422</td>
</tr>
<tr>
<td>101 to 760 words</td>
<td>0.340</td>
<td>0.376</td>
<td>81.199</td>
<td>0.613</td>
<td>69.340</td>
</tr>
<tr>
<td>Negative [-1,-0.5]</td>
<td>-0.614</td>
<td>0.673</td>
<td>66.362</td>
<td>0.820</td>
<td>58.952</td>
</tr>
<tr>
<td>Neutral [-0.5,0.5]</td>
<td>0.044</td>
<td>0.230</td>
<td>88.455</td>
<td>0.480</td>
<td>75.974</td>
</tr>
<tr>
<td>Positive [0.5,1]</td>
<td>0.068</td>
<td>0.371</td>
<td>81.437</td>
<td>0.609</td>
<td>69.534</td>
</tr>
<tr>
<td>Mean avg.</td>
<td>0.191</td>
<td>0.358</td>
<td>82.082</td>
<td>0.598</td>
<td>70.068</td>
</tr>
<tr>
<td>Optimized</td>
<td>0.237</td>
<td>0.318</td>
<td>84.123</td>
<td>0.564</td>
<td>71.825</td>
</tr>
</tbody>
</table>
6. PREFERENCE MAPPING/FILTERING

The introduced analysis tools are implemented in a first application in order to validate the system using a real-world use case. Our application prototype is a web based personal Electronic Program Guide (EPG) which is able to offer TV recommendations just seconds after users log in for the first time with their Facebook accounts by parsing their personal data. The program data is being delivered by a recommendation engine called TV Predictor [8]. Moreover the novel recommendations based on the users Facebook profile are highlighted and it is even possible to see what the user’s friends could like. This might be useful if the user has a rather empty profile with only few comments or likes.

Basically we want to find out how many of the determined criteria/attributes are matching a certain program. Since meta data like title or genre might not always be written the same way throughout Facebook movie pages and various EPG data provider, we use the Levenshtein distance to improve robustness while matching these strings that is “The smallest number of insertions, deletions, and substitutions required to change one string or tree into another” [16]. We calculate a similarity degree which is 1 minus the Levenshtein distance divided by the length (maxLength) of the longest of the two strings S1 and S2:

\[
sim(S1, S2) = \frac{1 \text{ distance}}{\text{maxLength(S1, S2)}}
\]

Samples from a limited user base show that even with a low degree of activity (likes, posts etc.) users will get a sufficient amount of recommendations. While these recommendations have only few matches they are considered to be weak recommendations. The more likes and posts users have given to TV related items, the more strong recommendations they get. The help of users’ friends becomes less noticeable the more the likes the user has given since there are more and more overlappings. As a test case we used 70 channels with about 2500 programs. The user with the fewest amount of interactivity received 0.8% (from the 2500 programs) of weak, 0.5% of medium and 0.5% strong recommendations. The user with the highest activity level (i.e. many likes, posts, comments etc.) received 1.1% of strong, 8% of medium and 6% of weak recommendations. The prototype indicates that using textual analysis of Social Networks could compensate the cold start and sparsity problems predicting automatically the interest in otherwise unknown items by 1.8% up to 15.1%. As a result users can start with a personalized service without giving explicit feedback and by only logging into a Social Network.

7. OUTLOOK AND FUTURE WORK

We have explained an approach to compensate the cold start problem by using preference ontologies. A way to identify semantic keywords was introduced with the formula of ”closest common hyperonyms”. In turn our sentiment engine allows to allocate a numerical feedback, in terms of the users opinion in range [-1,1] to the detected topics. A first implementation of the preference filtering approach is implemented in a social Electronic Program Guide and proofs the possible advantages of compensating the cold start problem.

First case studies indicate the strengths and weaknesses of such a system and its components, but larger evaluations analyzing the whole preference ontology system need to follow. Moreover we are planning to implement methods to identify relationships between apparently incoherent issues and properties, such as “sports”, "iPhone" and "How I Met Your Mother". Association Rules or other machine learning algorithms shall solve this issue.

8. REFERENCES