A Calculation Framework for Twitter Messages based on Machine Learning and Information Extraction techniques

MASTER'S THESIS
submitted in partial fulfillment of the requirements for the academic degree
Master of Science
in the Master's Program
WEB SCIENCES – WEB ENGINEERING

Submitted by:
Johann Tuder, BSc

At the:
Institute for Application Oriented Knowledge Processing

Advisor:
a.Univ.-Prof. Dr. Birgit Pröll

Linz, November 2015
Sworn Declaration

I hereby declare under oath that the submitted Master’s degree thesis has been written solely by me without any third-party assistance, information other than provided sources or aids have not been used and those used have been fully documented. Sources for literal, paraphrased and cited quotes have been accurately credited. The submitted document here present is identical to the electronically submitted text document.
## Contents

1 Introduction 1

2 Scope 3
   2.1 Calculation framework 3
   2.2 Data 4
   2.3 Results 5

3 Concept 6
   3.1 Quality in Social Media 6
   3.2 Architecture 8
   3.3 Features 11
      3.3.1 Textual Features 12
      3.3.2 Additional Content Features 24
      3.3.3 Networking Features 27

4 System Documentation 33
   4.1 Technical documentation 33
      4.1.1 Development environment 33
      4.1.2 Libraries 33
      4.1.3 Implementation 37
   4.2 User documentation 55
      4.2.1 Installation 55
      4.2.2 Setting up the configuration file 57
      4.2.3 Initial view 61
      4.2.4 Selecting a configuartion 64
      4.2.5 Viewing results 67
      4.2.6 Adapions 72

5 Evaluation 73
   5.1 Quality 73
   5.2 Textual Features 73
      5.2.1 Domain related terms 73
      5.2.2 Word Count 75
      5.2.3 Spelling Errors 76
      5.2.4 Length Of Tweet 77
      5.2.5 Emoticons 78
      5.2.6 Repeated Characters 79
      5.2.7 Capitalized Letters 81
      5.2.8 Stop-word Ratio 82
      5.2.9 PoS 83
      5.2.10 Overall Textual Features 85
   5.3 Additional Content Features 88
      5.3.1 Location 88
      5.3.2 URLs 89
      5.3.3 Overall Additional Content Features 91
   5.4 Networking Features 91
      5.4.1 Hashtag Count 91
5.4.2 Mention Count .......................................................... 93
5.4.3 Retweet ................................................................. 94
5.4.4 Number Of Retweets & Favorite Count ...................... 95
5.4.5 Reply ................................................................. 95
5.4.6 Overall Networking Features ...................................... 96
5.5 Cross Scoring ............................................................ 97
  5.5.1 Textual Features & Additional Content Features ........ 97
  5.5.2 Textual Features & Networking Features .................. 99
  5.5.3 Additional Content Features & Networking Features .... 100
  5.5.4 Summing Up All Feature Categories ....................... 101
5.6 Interpretation Of The Results ......................................... 102

6 Conclusion and future work ........................................... 104
  6.1 Conclusion ............................................................ 104
    6.1.1 General .......................................................... 104
    6.1.2 Implementation specific ..................................... 105
  6.2 Other disciplines .................................................... 106
    6.2.1 Social ........................................................... 107
    6.2.2 Design .......................................................... 107
    6.2.3 Business ........................................................ 108
    6.2.4 Law ............................................................. 109
  6.3 Future work .......................................................... 109
# List of Figures

1. *Calculation framework* tasks .................................................. 2
2. Feature overview ................................................................. 3
3. Architecture of the *calculation framework* .............................. 8
4. Tweet in flood database .......................................................... 9
5. Class diagram ........................................................................... 37
6. FrontendBean class ................................................................. 38
7. User interaction ......................................................................... 38
8. Configuration class .................................................................... 39
9. GateHandler class ...................................................................... 40
10. GATE workflow ......................................................................... 41
11. Pipeline for annotating the learn set ........................................ 41
12. Pipeline for learning from the annotated files ........................... 44
13. Pipeline for categorizing the rest of the tweets .......................... 45
14. Helper class .............................................................................. 46
15. ConfigEntry class ...................................................................... 47
16. DBConnection class ................................................................... 47
17. Score class ................................................................................ 48
18. UserInfo class ........................................................................... 49
19. MyTweet class .......................................................................... 49
20. IRMImpact class ........................................................................ 51
21. SVCalculator class ..................................................................... 52
22. Learn set folder structure ........................................................ 58
23. Initial view of the system .......................................................... 63
24. After selecting a configuration .................................................. 64
25. After uploading a configuration ................................................ 65
26. Workflow diagram ..................................................................... 67
27. Results ....................................................................................... 68
28. Description of features .............................................................. 69
29. Default tooltip ........................................................................... 70
30. part-of-speech tooltip ............................................................... 70
31. Results after second run ............................................................ 71
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Overview of categories</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>Overview of features</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>Part-of-speech tags for <em>das Haus</em></td>
<td>23</td>
</tr>
<tr>
<td>4</td>
<td>User object of Twitter API</td>
<td>34</td>
</tr>
<tr>
<td>5</td>
<td>Tweet object of Twitter API</td>
<td>35</td>
</tr>
<tr>
<td>6</td>
<td>Entity object of Twitter API</td>
<td>35</td>
</tr>
<tr>
<td>7</td>
<td>Places object of Twitter API</td>
<td>36</td>
</tr>
<tr>
<td>8</td>
<td>Precision without JAPE &amp; Gazetteer</td>
<td>43</td>
</tr>
<tr>
<td>9</td>
<td>Precision with JAPE &amp; Gazetteer</td>
<td>44</td>
</tr>
<tr>
<td>10</td>
<td>Scoring of the domain related terms feature</td>
<td>74</td>
</tr>
<tr>
<td>11</td>
<td>Scoring of the word count feature</td>
<td>75</td>
</tr>
<tr>
<td>12</td>
<td>Scoring of the spelling errors feature</td>
<td>76</td>
</tr>
<tr>
<td>13</td>
<td>Scoring of the length of tweet feature</td>
<td>78</td>
</tr>
<tr>
<td>14</td>
<td>Scoring of the emoticons feature</td>
<td>79</td>
</tr>
<tr>
<td>15</td>
<td>Scoring of the repeated characters feature</td>
<td>80</td>
</tr>
<tr>
<td>16</td>
<td>Scoring of the repeated characters feature with adapted configuration</td>
<td>80</td>
</tr>
<tr>
<td>17</td>
<td>Scoring of the capitalized letters feature</td>
<td>82</td>
</tr>
<tr>
<td>18</td>
<td>Scoring of the stop-word ratio feature</td>
<td>83</td>
</tr>
<tr>
<td>19</td>
<td>Part-of-speech amounts in tweet #1</td>
<td>84</td>
</tr>
<tr>
<td>20</td>
<td>Part-of-speech amounts in tweet #2</td>
<td>84</td>
</tr>
<tr>
<td>21</td>
<td>Scoring of the PoS feature</td>
<td>85</td>
</tr>
<tr>
<td>22</td>
<td>Overall scoring of textual features</td>
<td>86</td>
</tr>
<tr>
<td>23</td>
<td>Comparison adapted configuration vs. adapted without important terms</td>
<td>87</td>
</tr>
<tr>
<td>24</td>
<td>Scoring of the location feature</td>
<td>89</td>
</tr>
<tr>
<td>25</td>
<td>Scoring of the URLs feature</td>
<td>90</td>
</tr>
<tr>
<td>26</td>
<td>Overall scoring of additional content features</td>
<td>91</td>
</tr>
<tr>
<td>27</td>
<td>Scoring of the hashtag count feature</td>
<td>92</td>
</tr>
<tr>
<td>28</td>
<td>Scoring of the hashtag count feature with adapted configuration</td>
<td>92</td>
</tr>
<tr>
<td>29</td>
<td>Scoring of the mention count feature</td>
<td>94</td>
</tr>
<tr>
<td>30</td>
<td>Scoring of the retweet feature</td>
<td>95</td>
</tr>
<tr>
<td>31</td>
<td>Scoring of the reply feature</td>
<td>96</td>
</tr>
<tr>
<td>32</td>
<td>Overall scoring of networking features</td>
<td>97</td>
</tr>
<tr>
<td>33</td>
<td>Overall scoring of textual features &amp; additional content features</td>
<td>98</td>
</tr>
<tr>
<td>34</td>
<td>Overall scoring of textual features &amp; networking features</td>
<td>100</td>
</tr>
<tr>
<td>35</td>
<td>Overall scoring of additional content features &amp; networking features</td>
<td>101</td>
</tr>
<tr>
<td>36</td>
<td>Overall scoring of all features</td>
<td>102</td>
</tr>
</tbody>
</table>
Abstract

Finding qualitative or relevant tweets or messages on a social media platform is a common task in the text engineering community and is useful for multiple branches.

In this work a prototype of a calculation framework for twitter messages was developed. The framework starts a machine learning component that learns categories of a learning data set and then assigns the categories automatically to the remaining data. Afterwards a score is calculated by using different approaches that are split up into three groups, namely Textual Features (e.g. domain related terms), Additional Content Features (e.g. URLs) and Networking Features (e.g. hashtag count).

The result shows that the part-of-speech and domain related terms features outperform all others. However the other features are very useful for a more detailed ranking of the tweets and that the best results can be achieved by combining all features for the calculation.
1 Introduction

Social media platforms like Facebook and Twitter became more and more popular in the recent years. They enable every user to post messages regardless of whether they are personal feelings, retweets of someone else’s messages or just links to newspaper articles. With those possibilities of expressing a message a problem arose, how is it possible to differ qualitative messages from those that are useless?

Finding qualitative messages in the internet is an ongoing research field and became more popular in the recent years because of the hype about such social media platforms. Often companies have the need to communicate messages to their followers in a way that they get attracted to new products or for acquiring new customers. Furthermore those companies want to find so called opinion leaders in social networks and they want to find out why they are so influential.

In 2012 it was estimated that on Twitter nearly 200 million tweets were generated and more than 1.6 billion search queries were issued every single day [1]. The paper states that there is a large proportion of low-quality and noisy messages but that Twitter is a good source of data. On Twitter's about page it is mentioned that about 500 million tweets are sent each day in 2015 [2]. It is known that a user posts messages of different types (e.g. personal feeling or newspaper article) and that only a small number of them is interesting to a broader audience [3]. It is also a well-known problem on Twitter that a user is overwhelmed by the number of tweets and it is hard to pick out those tweets that are interesting [1]. Tweets were analyzed in multiple projects, e.g. in [4] interesting Twitter content was searched by topical analysis. Other papers are measuring the influence between users, e.g. [5] created an InfluenceRank and also [6] addressed the problem of user influence in Twitter. Furthermore the Web Ecology Project released a publication about "The Influentals: New Approaches for Analyzing Influence on Twitter" [7] in 2009. [8] already combined the two approaches of social influence and content quality. The literature that this thesis is based on often focused on whether user specific (e.g. number of followers) or tweet specific (e.g. number of retweets) features and they used mechanical turks to rate tweets.

The calculation framework combines the features that were already investigated by other papers into one system. With this system a user should find qualitative tweets within a given context fast and easily. Therefore two tasks have to be accomplished:
The *classification* task divides a set of tweets into multiple categories (e.g. informational or private message), therefore some of the tweets already have to be categorized. Afterwards the *calculation* task is started and each tweet gets a specific score assigned. This score represents the quality and a higher score means better quality.

In this thesis two sets of Twitter messages are used. The first set was recorded during the floods in Austria and Germany in 2013. This is the main data-set on which the *calculation framework* is built on. To ensure the quality of the *framework* a second data-set from a different domain is used, this set is based on tweets with the hashtag #aufschrei. This hashtag was used for messages about sexist experiences.

The tweets that were captured while parts of Germany and Austria were flooded greatly vary in their quality. In this scenario a qualitative tweet consists of a location where the floods are happening and what the current flood level of a river is:

```
#Hochwasser #VOCKERODE @ #ELBE —&gt; #Pegel in 1 Stunde von 789 cm um 1 cm auf 790 cm gestiegen. [Stand: 08.06.2013 12:20]
```

Whereas other tweets that are e.g. messages between users are often of low quality:

```
@ViviLovesBTR doch aber es fahren keine Züge mehr dorthin wegen den Hochwasser und so /:
```

These two tweets already show that even when tweets are recorded referring to a specific topic or domain they vary in the information they transport and also their writing style. The *calculation framework* is needed to automatically sort out the less qualitative tweets and to present those tweets which are more qualitative more prominently which are more qualitative.
2 Scope

In this chapter the scope of the thesis is described. It will outline which features for the prototype of the calculation framework have to be implemented, which data it has to deal with and what kind of results are expected after a calculation.

2.1 Calculation framework

In this section a detailed look at the framework is taken, the score is described more precisely in the following section Results.

The typical usage of the calculator is to perform multiple computations on a given set of text messages. Therefore the user has to provide a configuration file and text messages for which the value has to be calculated. The calculator performs multiple computations where the result is an explicit score for each message and it also calculates an average value for each category (e.g. informational or private messages; see 4.2.2).

For calculating the score the framework uses different existing methods that are mentioned in the literature that this thesis is based on. Those approaches are combined into one system which is the calculation framework. Furthermore the user has the possibility to decide which methods should be used when generating the score for the messages.

The calculator handles every kind of text (not only Twitter messages, also e.g. comments in forums) and categorizes them. The categorization process is done using a machine learning approach where a defined set of messages is categorized by the user beforehand. In a first step the machine learning component learns the categorized messages and afterwards it sorts the rest of the messages which are not categorized. The calculation framework uses the following features, which are described in more detail in the section Features:

![Figure 2: Feature overview](image-url)

<table>
<thead>
<tr>
<th>Textual Features</th>
<th>Additional Content Features</th>
<th>Networking Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Domain related Terms</td>
<td>• Location</td>
<td>• Hashtag Count</td>
</tr>
<tr>
<td>• Word Count</td>
<td>• URLs</td>
<td>• Mention Count</td>
</tr>
<tr>
<td>• Spelling Errors</td>
<td></td>
<td>• Retweet</td>
</tr>
<tr>
<td>• Length of Tweet</td>
<td></td>
<td>• Number of Retweets</td>
</tr>
<tr>
<td>• Emotions</td>
<td></td>
<td>• Favorite Count</td>
</tr>
<tr>
<td>• Repeated Characters</td>
<td></td>
<td>• Reply</td>
</tr>
<tr>
<td>• Capitalized Letters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Stop-word ratio</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Part-of-Speech (PoS)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2.2 Data

In this section the data where the calculation framework is used on is described. The used data was collected in June 2013 where floods affected wide areas in Germany and Austria. The messages are tweets from various Twitter users that submitted a posting during this disaster. The data consists of 14,866 tweets and is stored in a MySQL database. Those tweets were exported into text files so that the system can simply load files and does not have to be connected to a database. The file name has the following pattern: TweetID.txt (e.g. 325674891.txt). The exported files contain the id of the tweet in the file name and the message of it in the file. The tweets are uncategorized, which means that they only have the same topic but they can belong to different types of messages. Therefore the tweets should be categorized. The categories for this topic are:

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>information</td>
<td>tweets transport information about floods (height etc.)</td>
</tr>
<tr>
<td>private communication</td>
<td>tweets that are directly addressed to another user</td>
</tr>
<tr>
<td>video</td>
<td>tweets that link to a video</td>
</tr>
<tr>
<td>other</td>
<td>all other tweets that do not fit in one of the above categories</td>
</tr>
</tbody>
</table>

Table 1: Overview of categories

The categorization process is done by the calculator. For the categorization of the messages the calculator needs a learning set, which consists of randomly chosen messages that were categorized by hand. Those messages have to be equally distributed over the categories to guarantee a successful learning process.

Comparison

Additionally to the already mentioned data set a second set is used to ensure the functionality of the framework when it is used with completely different data. The second set contains 28,200 tweets with the hashtag #aufschrei, which was very widespread in 2013. By using this hashtag tweets about sexist experiences were submitted. The results of the first calculation are used for examining if it is possible to find qualitative tweets in

---

1 Most of the calculations do not need database access, although for some calculations the calculator needs access to the database. This is described in the section Implementation.

2 More information can be found on the German Wikipedia page: https://de.wikipedia.org/wiki/Aufschrei
the second data set. Therefore a categorization step for the second set has not been done and all the tweets are in the unordered category.

2.3 Results

The results show a numerical score for each message. According to the specified values during the calculation process (e.g. tweet length more important than emoticons, ... see Features) and the chosen features these results may vary for a single tweet between different calculations. The calculator shows the score of a message and an average score for each category, furthermore it displays the scores of a prior calculation if the calculation process is triggered multiple times.

For the calculator it does not matter who created the tweet, because only the content is analyzed, this is a difference to the other approaches that are shown in the literature that this thesis is based on. The score describes the quality of a message from a user’s point of view and how s/he determines quality in the current context.

In this case the quality is determined on one side by the kind of transported information and on the other side by the level of detail. This means that messages that transport information about where something happens (e.g. Flood in Germany) achieve a higher score compared to those messages that do not provide any information (e.g. How’s the weather in Germany?). Furthermore tweets that describe the situation more precisely, such as the current height of the floods or estimated developments of the height, also get a higher score.
3 Concept

In this section the concept of the thesis will be described. First a short introduction into the area of quality in social media, especially for Twitter (3.1), is given. Afterwards an overview of the system architecture is provided (3.2) and the features will be discussed (3.3).

3.1 Quality in Social Media

In the early 1990s most of the online content resembled traditional published content in e.g. newspapers and web users were only consumers but this changed at the beginning of the 2000s, where user generated content became more popular [9]. The difference between those content types is the quality, another issue is the very high variance of quality in user generated content [9].

[9] focused on finding qualitative messages in question and answer forums and it was found out that for questions a topical category information is crucial, whereas the length of a message was the dominating feature for the answers.

In [5] it is mentioned that 50 % of the tweets on Twitter are coming from only 0.05 % of the users. This means that possible opinion leaders can spread information farther and faster than other users and they can also recommend their followers specific products they should buy [5].

Influential users often have a lot of followers which leads to a high initial spread coverage, but most of the users prefer to retweet messages that are useful [5] and therefore it is important to find out what a useful and qualitative tweet is. However, the number of followers is not so important than an active audience that spreads the messages [6]. This does not mean that a follower is interested in all the tweets of the followed person, because 57 % of all tweets are not of general interest except to e.g. close friends of the author [1].

In [10] a study is referenced that puts tweets into six categories and shows the distribution of the tweets over them: pointless babble (41 %), conversational (38 %), pass-along value (9 %), self-promotion (6 %), spam (4 %) and news (4 %).

Twitters hashtags are used to group tweets together into specific topics but topic summarization is not easy. A problem is that tweets are expressed in an informal way and that not all of them meet the standard grammar requirements [8]. Therefore an algorithm for
building a topic summary has to be aware of a tweet’s content quality and in the work of [8] a salient tweet has to be published by an influential user and must have a good writing style. Furthermore it is mentioned in [11] that higher-value Twitter users retweet lower-value users less frequently and that each tweet has its own influential value. 

[12] argue that content relevance is harder to tamper for spammers and a ranking system based on content is therefore more difficult to influence compared to increasing retweets and the number of followers.

[13] writes that semantic quality is very hard to measure and that there is a correlation between spelling errors and content quality in the web. The results of the study, which was done over multiple internet domains including social networks, show that there is a high correlation between high lexical quality and the content of the page and that social media pages have a very low quality [13].

Because of these facts the following master thesis is considering those circumstances that arise from the usage of Twitter and the quality of tweets. With the calculation framework it is possible to rank tweets higher which are more interesting for the user. In this case quality stands for information, which is represented through the amount, level of detail and how well written the information is.
3.2 Architecture

In this section the architecture of the framework is described (figure 3). The system consists of five parts:

- machine learning component
- analyzing component
- database
- GUI
- configuration

As it was already mentioned earlier the main data for the calculation framework is a collection of tweets that were collected during floods. The messages are stored in text files and additional information (e.g. longitude and latitude) is stored in a database. An entry in the database looks like this:
If additional data is needed the Twitter API will be accessed. This is the case when recent content on Twitter or additional information about the users (e.g. number of followers) and tweets (e.g. number of retweets) have to be used.

The section in the middle of figure 3 consists of two components. The first one is the machine learning component which handles two different sets, one for learning the categories and a second which contains all other tweets that are not categorized yet. In the thesis the learning set of the flood tweets consists of four categories:

- **information**
  This cluster contains all tweets that provide information about the actual flood situation. This can be a tweet by a user or a tweet by a newspaper that refers to an article. Furthermore the tweet can contain information about the progress of the floods.

- **video**
  All tweets that probably contain a video should be collected in this cluster.

- **private communication**
  This category contains all tweets that can be seen as private communication. It does not matter if the message contains information about the floods.

- **other**
  All other tweets that can not be categorized into one of the above three clusters are assigned to the other category.
The machine learning component will learn which tweet belongs to which category by using the learn set. When the learning process is completed it will take the second set which is not categorized and will split those tweets into the categories that were learned in the prior step.

The analyzing component reads the clustered tweets in form of multiple text files and passes them to the calculators. The analyzer contains multiple calculators, in this case the calculation framework and a calculator for IRM (indegree, retweets, mentions) of the paper [6]. Each of the calculators is computing a score for all tweets in a different way. How the calculators are implemented and what they mean is mentioned in the following chapters. The analyzing component returns the ranked clusters and tweets to the results GUI.

The last section of the architecture is the GUI. The GUI consists of two pages where the first page is the settings page. On this page the user uploads a configuration file with the corresponding weights for calculation, the user can then activate or deactivate features. The values are then passed to the calculation framework and the analyzing process is started. After the analysis the results are displayed on the second page of the GUI. On the results page the score of each tweet for each feature is shown and explained, furthermore average scores for the categories are calculated and shown. The results page also shows the score of the prior calculation if it is available. In the next section it is focused on the features that were found in the literature that this thesis is based on.
3.3 Features

All the following features were already used in different papers and approaches, except the point [Favorite Count] which was not found in the literature this thesis is based on and is considered as new. The features are described with papers that used them and a short personal opinion about the usefulness and implementation is given. Furthermore it is mentioned how the calculation framework will make use of these features, i.e. how the configuration of each feature will look like. The use of a configuration file allows an easy and fast way to adapt the calculations for a different topic or domain.

The section is split into three parts where each part can be seen as a container of features that have something in common. These three containers are:

- **Textual Features**
  This container holds all features that can be extracted from the underlying tweet.

- **Additional Content Features**
  A tweet can also have an URL, image or location attached. These points are considered in this container.

- **Networking Features**
  Here are all features collected that symbolize a link between tweets or users.

In the following table all feature containers and the features they consist of are shown. The column *Type* shows how the feature is described in the configuration file. The three possible values are:

- **list**
  For this feature a list has to be configured with values that have to be looked up in a tweet.

- **range**
  Here the configuration consists of multiple entries where the values are ordered ascending for finding the right weight during calculation.

- **binary**
  These kind of features only need a single entry in the configuration file where a value is defined that should be met or not.
<table>
<thead>
<tr>
<th>Feature Container</th>
<th>Feature</th>
<th>Type</th>
<th>Implemented</th>
</tr>
</thead>
<tbody>
<tr>
<td>Textual Features</td>
<td>Domain related Terms</td>
<td>list</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Word Minimum</td>
<td>range</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Abbreviations</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Spelling Errors</td>
<td>binary</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Length of Tweet</td>
<td>range</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Emoticons</td>
<td>binary</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Exclamation &amp; Question Marks</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Repeated Characters</td>
<td>binary</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Capitalized Letters</td>
<td>range</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Word Length</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Numbers &amp; Measure Symbols</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Stop-word Ratio</td>
<td>range</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Part-Of-Speech (PoS)</td>
<td>binary</td>
<td>✓</td>
</tr>
<tr>
<td>Additional Content Features</td>
<td>Location</td>
<td>list</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>URLs</td>
<td>list</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Multimedia</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Networking Features</td>
<td>Hashtag Count</td>
<td>range</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Trending Topic</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Mention Count</td>
<td>range</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Retweet</td>
<td>binary</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Number of Retweets</td>
<td>range</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Number of Comments</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Favorite Count</td>
<td>range</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Reply</td>
<td>binary</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 2: Overview of features

3.3.1 Textual Features

Domain related terms

When a specific topic is discussed on Twitter very often the same words are used by multiple different users. However there are often words that describe a situation more
precisely or transport more information, these words usually signal a higher quality.

As mentioned in [8] a tweet is salient or a user influential if a tweet contains important terms. This means that a tweet with the words *regen* or *meter* is more valuable than a tweet without those words, because they possibly describe the situation during the floods. The important domain related terms should not be determined by iterating over all tweets and taking the most used ones as the most important. The list has to be generated by the user of the system. This has the additional advantage of easily using the system within other domains, because the list can be updated very quickly. Each term can be independently weighted, this enables the user to define the importance of specific terms more precisely. These terms and weights are stored in a configuration file and are defined in the following way:

```xml
<TermList weight="1" activated="true">
  <Term value="regen" weight="0.3"/>
  <Term value="meter" weight="1.0"/>
</TermList>
```

Listing 1: Example for term configuration

As shown in listing 1 on line 1 a feature can be weighted and deactivated which enables the user to distinguish more precisely between the chosen features. A tweet that contains the terms *regen* and *meter* would score 1.3 points, this means that all occurrences of the words in the configuration file are summed up and then multiplied by the weight that is specified in the first line. This mechanism is used for all other features, where the construct looks like this.

This feature is useful for finding qualitative or topic relevant tweets. However the user has to know in advance which words are used in qualitative tweets of the current domain. Another drawback of this feature is that a list of topic specific words is in most cases not practicable for other scenarios, because the important terms that occur during a disaster are not the same as those that are used when writing about a social problem or discussion.

**Word Count**

A tweet exists of at least one word, because it is not possible to submit an empty tweet. [8] stated that they ignore extremely short tweets with less than 3 words, because they can not really deliver any information. Also for [10] the length of a tweet is a sign for
quality. They say that longer tweets are more formal and informative. In a study done
by [14] those tweets were marked as interesting that were longer than the average of all
tweets in the data-set. Furthermore [4] observed that the length in characters and the
number of words in a tweet are very effective for estimating a tweets value. They also
say that this approach is very similar to the typical high-quality document classification
which assumes that the content of longer documents has a higher quality.
This can be demonstrated with the following two tweets. The first one is longer and
therefore can transport more information:

#de74x #740hn Live-Ticker Hochwasser Hochwasser: Schon 17 Tote, weitere vermisst
- BZ http://t.co/NZoW1zpcxV

Whereas the second one is not capable to do this:

@Gangnam_KID vielleicht wegen dem Hochwasser?

The following configuration is slightly different to the prior one, since in the value attribute
there are now numbers. Whenever the amount or a ratio should be used for weighing the
values have to be ordered ascending. The last value should always be a *-character since it
means that all values that are greater than the prior one should get the mentioned score.
In the comments of the configuration it is described how the scores will be generated for
a tweet. If a tweet e.g. has 10 words it would get 0.3 points and another tweet with 15
words would get 1.0 point.

```
<WordCount weight="1" activated="true"> <!-- values have to be ordered ascending -->
  <Count value="5" weight="0.0" /> <!-- up to 5 words no points -->
  <Count value="14" weight="0.3" /> <!-- 14 words 0.3 points -->
  <Count value="*" weight="1.0" /> <!-- more than 14 words 1 point -->
</WordCount>
```

Listing 2: Example for word minimum configuration

For finding the correct score the number of words that a tweet has is extracted first and
then the value is looked up in the configuration.
The use of a word minimum is recommended for removing spam tweets. Often a spam
tweet only has an URL without any text and sometimes a hashtag is attached to an URL.
With setting a minimum barrier for the amount of words such tweets are going to be
penalized and will be ranked lower than others. Of course this mechanism can be easily
manipulated by spammers, but the user has a higher influence on retrieving reasonable
results. Also it may have a problem when a tweet consists of long words, because the
maximum number of possible words in a tweet is decreased.

**Abbreviations**

There are a lot of different abbreviations that are used on Twitter, because a tweet is
limited to 140 characters. So if someone is using abbreviations then there are more
characters remaining for transporting other information. [14] and [8] determine that abbreviations negatively influence the readability of a tweet. The study done by [14] also points out that the tweets marked as interesting by the judges
were well formed and not abbreviated.

The following tweet used the abbreviation *u.* instead of the whole word *und*, which only
saves one character at the end:

```
#Hochwasser Leider kann Polizei/Feuerwehr/Helfer nicht zwischen Gaffer u. Infobeschaffer unterscheiden. Ihr alle macht Eueren Job ganz toll!
```

If one would implement such a penalization it could be done with lists that have common
abbreviations stored. Then each word can be compared against those list and each found
word would decrease the tweets score. However a tweeter is not tied to use specific
abbreviations. The user can add all forms and types of abbreviations that s/he wants and
so the list will not be complete and the penalization will not be as effective as intended.
[14] and [8] are right when they argue that an abbreviation decreases the readability, but
on the other hand it enables a user to transport more information. Also the user can
deliver more detailed information, e.g. during a disaster a user could shorten the name of
a city (e.g. VIE for Vienna) and save characters. A mechanism to penalize abbreviations
will not be implemented.

**Spelling Errors**

A spelling mistake means that a word is written wrongly or does not exist. This can hap-
pen in many different cases like changing two characters while typing, forget a character
to write or using a slang word. When such a mistake is made it is assumed as a spelling
error.
Again [14] and [8] mention that good readability is achieved when a tweet does not contain any spelling errors. [1] explicitly states that low-quality tweets contain many spelling errors, however they found out that spelling is not a strong indicator for quality. [1] also calculates a percentage of how many words of a tweet can be found in a dictionary and also [12] calculates a ratio of words out of a vocabulary to estimate the language quality of a tweet.

The tweet that is shown below has a spelling error in the first sentence:

```
@ennolenze aktuell ist diw Versorgung der Helfer nicht sichergestellt. Ich möchte nicht das jemand auf der Landebahn abklappt :( #hochwasser
```

The correct word that should appear instead of `diw` should be `die`. This mistake possibly happened, because the letters `w` and `e` are next to each other on a standard keyboard layout.

According to the papers that showed that spelling errors are a sign of low quality tweets, a mechanism that measures these errors should be implemented. Therefore the system has to contact a dictionary and lookup all the words that are used within the tweet. If a word is not found it probably isn’t found because of a spelling error. However, there can be multiple scenarios why a word could not be found in a dictionary:

- **slang**
  
  Often slang words are used within a tweet. Those words are maybe recognized by people from the area of the user, but often outsiders do not understand the word. This leads to an information loss and is going to be penalized.

- **location**
  
  During disasters or sport events also the names of cities are mentioned in tweets. While names of bigger cities like Vienna are frequently found in dictionaries, a reference to smaller cities is not so common. So a tweet that refers to a small city would be punished. To avoid this false decision the words that were not found in a dictionary have to be looked up on a map. If the word also is not found there then the tweet has to be penalized, because the word is slang or a spelling error.

```
<SpellingError value="98" weight="1" activated="true"/>
```

Listing 3: Example for spelling errors configuration
In this configuration a ratio is defined which means that if a tweet has more than 98% of correct words it will get one point. If the tweet does not reach this target it will not get this point.

The spelling errors feature usually finds those tweets that are of lower quality but it also has to be considered that a spelling error may occur because of the character limitation on Twitter. This means that a user will write a word in a shortened way (e.g. without vowels) to use those characters for other words. These words will result in a spelling error, whereas the tweet can possibly transport more information to other users.

**Length Of Tweet**

A tweet can have a length of maximal 140 characters and the importance of a tweet’s length is quite similar to the section [Word Count](#). The main difference is that not the number of words is used as measuring factor, but the length of the tweet. The assumption is that a longer tweet has a higher quality than a short tweet [10]. Also [12] argues that the length of a tweet positively affects the quality of a tweet, because it probably contains more information.

As one can see in the following example a tweet can transport information although it is very short:

> 718,00cm 18:50 #magdeburg #hochwasser

However a longer tweet usually is able to transport more detailed information:

> #Hochwasser #MÜHLBERG @ #ELBE —> #Pegel in 2 Stunden von 987 cm um 2 cm auf 985 cm gefallen. [Stand: 07.06.2013 17:30]

In the configuration a stepwise weighting like in section [Word Count](#) is used. Therefore the number of characters (also whitespace and punctuation characters) is extracted and divided by the maximum length (in this case 140) that a message can have to get the ratio of used characters. Afterwards the ratio is looked up in the configuration:

```xml
<TweetLength weight="1.0" activated="true">
  <Length value="25" weight="0.0"/> <!-- up to 25 characters get no points -->
  <Length value="50" weight="0.3"/> <!-- up to 50 0.3 points -->
  <Length value="84" weight="0.7"/> <!-- up to 84 0.7 points -->
  <Length value="*" weight="1.0"/> <!-- more than 75 get 1 point -->
</TweetLength>
```
The papers already give a good explanation why this measurement approach should be used. In traditional document retrieval approaches longer documents are also implied as more detailed and that they have a higher quality. Even though a tweet has a limitation of 140 characters a user may transport as much information with one tweet and therefore has to exhaust those 140 characters.

**Emoticons**

Emoticons are also known as smileys. Usually they are used to emphasize the feelings that should be transported with a message.

In [8] it is stated that the number of emoticons are useful for measuring content quality, but they do not state if more emoticons influence the quality positively or negatively.

The above tweet has an emoticon at the end of it to give the reader to understand that the message should be taken jokingly.

For the emoticons a list is used where a lot of emoticons are already mentioned\(^3\). Afterwards the words of the tweet are compared with the list and if one of the words matches an emoticon from the list the tweet gets penalized by using the following configuration:

```
Listing 5: Example for emoticons configuration

<Emoticons weight="-1.0" activated="true" />
```

Since the quality was not measured in comparison with emoticons it is assumed that emoticons negatively influence the quality of a tweet. This is hypothesized, because when the provided tweets about the floods were looked through, only very personal messages (e.g. an opinion) had emoticons in the text. This opinions did not provide any new or important information about the floods.

\(^3\)The emoticon list was generated using: [http://www.blifaloo.com/info/smiley_dictionary.php](http://www.blifaloo.com/info/smiley_dictionary.php) (accessed on August 08th, 2015)
Furthermore it has to be mentioned that the penalization of emoticons relies on the topic and the kind of messages that are assumed as useful. When it is important to find messages that express feelings for a topic than the emoticons feature has to be used with a positive value.

**Exclamation and Question Marks**

Exclamation (!) and question marks (?) are used to emphasize or ask something. They are abnormalities in punctuation and are considered in the work done by [1].

The following tweet uses an exclamation mark at the end to enforce the message of the tweet:

```
Allen, die nicht vom Hochwasser betroffen sind - ich wünsche uns spannende Stunden mit dem #me13 und 9 richtig grüne Regierungsrioritäten!
```

Those marks do not influence the quality of a tweet when they are used normally. During disasters tweets with exclamation marks will probably deliver situational information, e.g. that the water rose by a specific number of centimeters. Whereas question marks will not provide such an information and, such as emoticons, probably do not bring up further information. In the following section multiple occurrences of a character are considered, since both features use the same approach they share their configuration.

**Repeated Characters**

Repeated characters are defined as characters (also punctuation characters) that occur several times in a row. [1] also measured the number of repeated characters in a tweet.

In the tweet below are three dots at the end of the sentence:

```
Wie sinnvoll die GEZ Gebühren angelegt sind, sieht man daran, dass bei Hochwasser mehr Reporter vor Ort sind, als die Orte Einwohner haben...
```

To solve this problem the program has to iterate over each character in the tweet. The current character has to be compared to its predecessor and when they are equal a counter (that stores the value of how often the character is repeated) has to be increased. When the counter reaches the defined value of the configuration file the search can be aborted and the tweet has to be penalized.

The number of characters when a tweet should be penalized is defined in the configuration the following way:
Typical words that contain a great number of the same letter are exclamations like 'ahh-hhh' or 'ohhh'. Here the 'h' is the repeated character. Such words or exclamations usually occur in personal opinions. Therefore tweets with such characteristics rarely transport any useful information.

There exists one problem when e.g. the data set contains a lot of tweets with abbreviations of organizations. For those abbreviations often the starting letters of the words are put together and therefore can easily have repeated characters. This circumstance has to be considered when the value for the configuration is set.

**Capitalized Letters**

The capitalized letters feature is often strongly connected to the prior mentioned repeated characters. Exclamations like in the prior section are written all-capitalized which ends in words like 'AHHHHH' or 'OHHH'. This abnormality in a text was measured by [1].

The following tweet is completely written in uppercase letters:

RT @cs50737: DIE LEUTE DIE ZUM HOCHWASSER FAHREN NUR UM ZU KNIPSEN UND DANN SOGAR IM WEG RUMSTEHN, DIE SOLL AUCH DER BLITZ BEIM SCHEISSEN T...

Within the program all capital letters of the tweet have to be summed up. Afterwards the sum is divided through the length of the tweet and the resulting ratio is looked up in the following configuration:

```
<CapitalizedLetters weight="1.0" activated=""/>
  <Count value="15" weight="1.0"/> <!-- up to 15% capitalized letters get 1 point -->
  <Count value="33" weight="0.3"/> <!-- up to 33% capitalized letters get 0.3 points -->
  <Count value="/" weight="-1"/> <!-- over 33% capitalized letters get -1 point -->
</CapitalizedLetters>
```
Also while looking on the flood tweets it was examined that mostly personal opinions and messages were written in all-capitalized letters. However, it has to be considered with abbreviations of institutions, e.g. UNO. Usually those abbreviations are written in capitalized letters but are not decreasing the quality or the information gain of a tweet. Since in the German language nouns start with capital letters, the ratio of capital letters in German tweets is usually higher than in English ones. The user has to remember these facts when other tweets are used.

**Word Length**

In this feature the length of every single word has to be examined, i.e. to look on the number of characters a word consists of.

In the studies done by [1] and [8] it is suggested that also the lengths of words in a tweet have an influence on the tweet’s quality. In the studies all words were analysed and the average word length was calculated. Since the time consumption of such a calculation is very high it will not be implemented in the framework, because the feature mentioned in section **Length Of Tweet** will be implemented and covers a very similar area.

In the tweet mentioned below the longest word *Homoeohenachrichten* consists of 18 characters:

| Da brennt es in einem AKW und in den Nachrichten kommen nur Hochwasser-, Drohnen- und Homoeohenachrichten? |

However, the feature could be implemented in different ways:

- **via configuration file**
  
  In the configuration file it has to be stated how many points a tweet would get for different word lengths. This would be done stepwise and can be compared to the configuration in the section **Length Of Tweet**. Afterwards the average word length of a tweet is calculated and compared against the configuration.

- **via calculation**
  
  Before a tweet gets rated the average word length of all tweets in the data-set has to be calculated. Therefore the system iterates over each tweet and calculates the
word length for one tweet. The results are summed up and divided through the number of tweets in the data-set. This returns the average word length of all tweets of a given set. Afterwards the word length of each tweet has to be compared to the overall average. In a configuration file it has to be determined how the deviation to the overall average has to be handled. For example if a tweet’s word length is higher than the overall average it gets one extra point, if it is lower the tweet will not get any points.

Numbers And Measure Symbols

A number or measure symbol like $ or % is mentioned as positive influence on the quality, because the tweet could provide monetary or statistical data [1].

After the examination of the flood tweets it is obvious that such symbols occur often during disasters. In the flood tweets were not $- or %-symbols, but ”cm” or ”m” declarations for centimeter and meter like in the following one:

```
#Hochwasser #SCHNACKENBURG @ #ELBE —&gt; #Pegel in 1 Stunde von 627 cm um 5 cm auf 632 cm gestiegen. [Stand: 07.06.2013 23:20]
```

These symbols do not get an extra implementation in the calculation framework, but if such symbols should be considered they have to be included into the list of Domain related terms that were mentioned earlier.

Stop-word Ratio

Stop-words are usually words that are not relevant for getting the semantics of a text even when they occur very often. In German such words are e.g. the articles (der, die, das) or und and in.

[8] and [1] argue that the stop-word ratio affects a tweet’s quality. In [15] it is mentioned that the presence of stopwords is positively correlated with the informativeness of a text and that documents with very few stopwords are more likely irrelevant.

To calculate this ratio a list of stopwords is needed and every word in a tweet has to be compared against all words on the list. A German stopword list is provided by the blog “solariz”. The list is provided under a creative commons attribution 3.0 unported license and is used by the framework.
Listing 8: Example for stop-word ratio configuration

The stop-word ratio has to be seen very critically when analyzing tweets. Since tweets have a limitation of 140 characters users will sometimes write tweets that do not have correct grammar or style and this will influence the mentioned score.

PoS

Part-of-speech means that a word or punctuation symbol is assigned a specific class. The term *das Haus* results in the following assignment:

<table>
<thead>
<tr>
<th>Term</th>
<th>Class</th>
<th>Abbreviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>das</td>
<td>article</td>
<td>ART</td>
</tr>
<tr>
<td>Haus</td>
<td>noun</td>
<td>NN</td>
</tr>
</tbody>
</table>

Table 3: part-of-speech tags for *das Haus*

In [1] part-of-speech is used for estimating the difficulty of a text. It is stated that longer pieces of text are normally harder to understand, but they can transport more information and are usually better written than short texts.

For the calculation they use an adaption of the *formality score*. In this work the same formula is used:

\[
F = \frac{(\text{noun frequency} + \text{adjective freq.} + \text{preposition freq.} + \text{article freq.} - \text{pronoun freq.} - \text{verb freq.} - \text{adverb freq.} - \text{interjection freq.} + \lambda)}{2}
\]

(1)
Where \( \lambda \) is set to 10, because of the restricted length in tweets according to [1]. The configuration for the part-of-speech feature looks like this:

```
<POS weight="1.0" activated="true"/>
```

Listing 9: Example for part-of-speech configuration

Since the value will always be calculated it is simply used and does not need any further adaption. In the configuration file the feature can be simply turned on or off. Furthermore it has to be stated like for the prior feature that often texts on Twitter are not written grammatically correct since they have a limitation of 140 characters.

3.3.2 Additional Content Features

Location

A user can add geo-information to a published tweet which is provided as longitude and latitude measure via the Twitter API or mentioned in the text.

In [16] the presence of location information is a sign of higher quality. Whereas [14] found out that the availability of a geo-location less often influences the salience to other users. [17] found out that during disasters a tweet containing a geo-location is more important to local individuals. Furthermore [17] found out that users who send tweets with geo-location and situational updates are carefully constructing tweets so that they can pack as much information as possible into it.

The following tweet has two locations in the text:

```
#Hochwasser #DESSAU @ #ELBE —&gt; #Pegel in 1 Stunde von 726 cm um 1 cm auf 727 cm gestiegen. [Stand: 07.06.2013 23:20]
```

Also within the calculation framework a given location information in a tweet should positively influence a tweet’s score. Therefore the longitude and latitude that is already stored in the provided floods database will be used. However, a tweet can also have a name of a city or country in it, without adding detailed geo-location information to the tweet. To examine those names a configuration file is needed, where all relevant cities and countries are listed. This feature can be compared to Domain related terms where it is also possible to weigh specific terms:

```
<Locations weight="1.0" activated="true"/>
```

Johann Tuder, BSc - 1255771
Listing 10: Example for location configuration

This feature is only useful in some contexts, e.g. in the floods context such information is essential, because the position of where something happens is useful to those who want to provide help. However in discussions about other events (e.g. politics) the geolocation information can be used to determine how people think in specific areas, but the information is not useful for finding qualitative tweets.

URLs

Tweets very often have an URL in their text so that the users can link to further information on blogs or other news sites. [10] and [16] stated that tweets that contain an URL have a higher quality, because they point to a web page that could provide more details about the topic. Also for [8] and [14] URLs have a positive influence. They say that tweets with URLs are usually linking to news pages, blog posts or other kind of shared information. Additionally to those facts [1] argues that it is also important to distinguish to what type of website the URL refers to, because not all links may be interesting. [14] further notices that tweets that only consist of an URL are marked as spam tweets in their study. In [12] it is stated that the measurement if a tweet contains an URL is the most effective one in terms of the tweet’s quality. They think that this is because of the 140 character limitation for a tweet and with a link more detailed information can be provided.

A tweet with an URL can look like this:

Dresden Hochwasser 2013 - Eine Stadt unter Wasser - Spendenvideo durch Werbung:
http://t.co/nO7R2TYX9h via @youtube

The flood tweets that were recorded often had an URL that linked to a web page that provided more detailed information about the current situation. In the configuration for the system will be an option, where URL-only tweets can be penalized. Furthermore a list of important URLs has to be provided and each URL can be weighted on its own.
It is very important to check where the URL is linking and not only if an URL is present, because a lot of links can refer to unreliable sources. Therefore also shortened URLs have to be followed until the final destination of the link is reached. At this point the URL has to be compared to the configuration. This list of URLs probably has to be updated for each domain where the calculator is used but it is also possible to have a standard set of URLs that are trustworthy or not.

**Multimedia**

Twitter messages can have two types of multimedia: videos and images. [16] lists in the set of "content quality features" multimedia, because a video or an image can be recognized more likely by other users.

Usually multimedia content can be extracted from the Twitter API very easily, because each tweet provides information about media elements that were uploaded with it. Since the usage of the Twitter API is restricted the data can not be fetched during calculation and also the data is not provided in the database. Therefore this feature is not implemented. If the data were in the database a configuration like in the part-of-speech section could be used, where a tweet gets a point if it has multimedia content.
3.3.3 Networking Features

Hashtag Count

On Twitter a hashtag is used for categorizing a tweet and make it easier to find for other users. A hashtag starts with an #-symbol and is directly followed by a character, a whitespace ends the hashtag definition.

[16] tackled the quality issue concerning hashtags with a boolean feature where a hashtag positively influences the content quality. Also [10] measured the hashtag count and determined it as one key factor for the rank of a tweet. The result of the study shows that the hashtag count has a low importance to their re-ranking model. This also accords to the definition of good readability by [8] which says that one factor for achieving good readability is not to have too many hashtags.

The following tweet has two hashtags:

@liesiundco Kieswerk Barleben (Barleber See) Anfahrt über Barleben - zum Sandsäcke füllen #Magdeburg #Hochwasser

In the framework hashtags will also be considered. The ranking is implemented like the above features where the user can assign different values to a specific hashtag count:

```
<Hashtags weight="1.0" activated="true">
  <Count value="0" weight="0.0" />
  <Count value="4" weight="1.0" /> <!-- 4 hashtags get 1 point -->
  <Count value="5" weight="0.3" /> <!-- 5 hashtags get 0.3 points -->
  <Count value="*" weight="0.0" /> <!-- more than 5 hashtags get no points -->
</Hashtags>
```

Listing 12: Example for hashtag count configuration

The background of this solution is mainly the definition of good readability by [8], furthermore the extensive use of hashtags may be a sign that a tweet is spam. This means that the user should define a maximum barrier for the hashtag count so that possible spam tweets score lower.

In some cases it has to be looked on the structure of the tweets that should be investigated, since the usage of hashtags may vary from one topic to another. In the floods data set the informative tweets excessively use hashtags whereas in the #aufschrei data set the usage is lower.
Trending Topic

The trending topic mechanism is a Twitter feature where hashtags that are currently used very often are merged into a list. [18] describes the functionality of trending topics as follows: The trending topics compose of top terms that were recently used on Twitter and are updated in real time. The comparison of the words within a tweet to Twitter’s trending topics will not be implemented. The reason therefore is that the underlying tweet base is old and the words will not appear in the trending topics.

If this feature were implemented the system has to remove all stop-words of a tweet. Afterwards the words have to be checked against the current trending topic list.

Mention Count

A tweet can not only consist of text or hashtags, it can also have a user mention, i.e. that the username is embedded into the text of a tweet. This is achieved by writing an @-symbol and directly afterwards the username without any whitespace.

According to the use of user-mentions there are two different positions. [16] lists mentions as a positive criteria for content quality, because the mentioned user will more likely see the tweet. Whereas in [8] it is again stated that those mentions negatively influence the readability of a tweet. A mention is similar to a reply, whereas a mention does not occur at the beginning of a tweet [7]. [6] says that a mention represents the name value of a user.

In the below shown tweet the user @DB_Bahn is mentioned:

@DB_Bahn wunderbar. Passt mir gut :) Hochwasser meint ihr oder die Gewitterfront ?

In the system the negative view of user mentions is supported. [16] handles it as a positive criteria, because maybe the mentioned person will see the tweet, but the tweet itself can still be useless. However if a tweet contains a lot of user mentions it has less space for qualitative information. The feature will be implemented with barriers:

```xml
<MentionCount weight="1.0" activated="true"/>
<Count value="0" weight="1.0"/>
<Count value="*" weight="0.0"/> <!-- if a user is mentioned it will get 0 points -->
```
The negative view of this feature depends on the topic that is investigated. In the context of the floods private messages (those start with a user mention) are not so important, because they often do not provide useful information. Whereas when a social topic is discussed it can be argued that only messages that are directed to a user are useful and therefore have to score higher.

**Retweet**

A retweet is like a share on Facebook where a user can re-send another user’s tweet, often retweets begin with ”RT”.

[1] examined retweets and found out that in terms of quality retweets are not a strong indicator and that they can also include low-quality and noisy messages. Also [19] says that it is not important for a tweet how many followers a user has, because the tweet reaches a broader audience as soon as it starts spreading through retweets. According to [17] during disasters tweets with geo-location are more likely to be retweeted than those without location information. [7] also mentions that a retweet occurs because a user is influenced to reproduce the content.

The following example is even a retweet of another retweet:

```
RT @bgebot: RT @genenano: @peteraltmaier @holst_alexander @AmChamGermany beweis für den erfolg der #energiewende ist #hochwasser. danke. #-
```

Since retweets hardly add new and useful information, they are going to be penalized by the system. However this setting will be made in the configuration file so that if a user thinks that retweets are important the value can be changed:

```
<IsRetweet value="-1.0" weight="1.0" activated="true"/>
```

Retweets never bring new information into a topic, since they are only copies of the original tweet. However they help to spread the original tweet so that it can reach more users. This means that retweets are good to spread information but not to find qualitative tweets. Furthermore it should be ensured that the original tweet is in the data set on
which the calculations should be performed otherwise information can get lost, because the retweet is penalized and scores lower than other tweets.

**Number Of Retweets**

In the works [16], [11], [14], [19] and [5] the number of retweets of a specific tweet is a sign of quality. [10] explains that the retweet count is a social voting for a tweet and argues that if more people share a tweet it will more likely have a higher quality. [5] says that the most accurate method for measuring the quality of a tweet is a semantic analysis. Despite the fact that this is almost impossible they measure the quality by using the ratio of the number of retweets and comments to the amount of tweets the user has published. Also [6] states that a retweet stands for a tweet’s content value. [12] argues that the number of retweets shows the popularity for a tweet, but the effect is not as high as expected and can be easily manipulated by spammers.

Due to the fact that the number of retweets is used in other studies, it is also implemented in the framework. It is assumed that a tweet is going to be retweeted when it is interesting and well written. This means that it is expected that every user on twitter has an internal requirement that has to be fulfilled, before retweeting a message. The configuration for this feature looks like the following example:

```xml
<RetweetCount weight="1.0" activated=""/>
  <Count value="100" weight="0.0"/>
      <!-- 100 retweets --> 0 points -->
  <Count value="1000" weight="0.3"/>
      <!-- up to 1000 retweets 0.3 points -->
  <Count value="10000" weight="0.7"/>
      <!-- up to 10000 retweets 0.7 points -->
  <Count value="*" weight="1.0"/>
      <!-- more than 10000 retweets 1.0 points -->
</RetweetCount>
```

**Listing 15: Example for number of retweets configuration**

**Number Of Comments**

Twitter is a bi-directional communication platform, which means that a user can write an answer to another user's tweet. It was already mentioned in the previous section that
uses a ratio of retweets and comments count in comparison to the numbers of a user’s
tweets.

The number of comments will not be implemented as a quality feature in the framework.
Comments can be made out of very different intentions, e.g. a comment is written when
a user agrees or disagrees with the opinion of the other user or it is also possible that the
answering user adds additional or more detailed information, but the number of comments
does not reflect the reasons for the comments.

The twitter API does not provide the information about how many times a tweet was
commented, i.e. that if this feature should be implemented it has to be iterated over all
answers and count them.

**Favorite Count**

In the literature this thesis is based on nobody used the favorite count of a tweet for
determining the quality. The favorite count can be accessed via the Twitter API and it
represents how often the current tweet was favored by other users. Twitter explains that
a favorite should be made when a user likes a tweet. With the favoring process a user
stores the tweet for a potential future use or only to let the other user know that this is
a good tweet[^4].

A favor is done when a user likes a tweet and like in section **Number Of Retweets** this is
probably done when a tweet is interesting and well written. The weighing is again done
with boundaries:

```
<FavoriteCount weight="1.0" activated=""/>

<Count value="0" weight="0" /> !-- 0 favors --> 0 points -->
<Count value="*" weight="1.0" /> !-- at least one favor 1 point -->
```

Listing 16: Example for number of favors configuration

**Reply**

Replies are answers or comments of a user, in form of a tweet, to another user’s tweet.
Both [14] and [1] found out that replies are less interesting and mostly have low quality

compared to normal tweets. [7] says that a reply occurs when a user is influenced to reply to the content.

The twitter API provides the field "in_reply_to" which holds information about the original tweet author, when the current tweet is a reply. It was already mentioned that replies hardly have good quality, so the framework will penalize replies unless it is configured otherwise by the user:

```
<IsReply value="-1.0" weight="1.0" activated="true"/>
```

Listing 17: Example for is reply configuration
4 System Documentation

In this section the calculation framework and its usage is described. It is started with the Technical documentation how it was developed and which libraries the system is using. Afterwards the usage is described in the User documentation by using screenshots and a guide for the calculation.

4.1 Technical documentation

The system documentation splits up in three parts. The first one is the Development environment, afterwards the used Libraries are mentioned and at the end the Implementation is described.

4.1.1 Development environment

Since the calculation framework is written in Java (version 8) the Eclipse Java EE IDE for Web Developers was used in version 4.4.0 (Luna Release) on a Windows 8.1 PC. The usage of Java is also owed to the usage of GATE which is also written in Java and therefore was easier to integrate. GATE was used in version 8.0. For the database MySQL was used in version 5.6.21. Since the calculation framework is a web based application the program was deployed onto an Apache Tomcat web server version 8.0.9.

4.1.2 Libraries

For the GUI two libraries were used, JSF in the version 2.2 and additionally to it PrimeFaces in the version 5.2.

JSF

JSF stands for JavaServer Faces and is used for building user interfaces for Java server applications [20]. The project page can be found under https://javaserverfaces.java.net/

PrimeFaces

PrimeFaces is an open source user interface library which consists of one single jar-file with no dependencies or configuration and is therefore considered as a lightweight library.
The usage is demonstrated on the project page under [http://www.primefaces.org/showcase/](http://www.primefaces.org/showcase/).

**twitter4j**

To access Twitter for crawling data twitter4j (version 4.0.2) was used. Twitter4j is a Java library for accessing the Twitter API, it does not require additional jar files, is lightweight and is also supported on the Android and Google App Engine platforms [22].

The Twitter API allows access to a lot of information that can be used in the thesis. The following short overview does not contain all provided information and was created using the information that is provided under [https://dev.twitter.com/overview/api/](https://dev.twitter.com/overview/api/) Under this link all available information is mentioned, in the following tables only information that is useful in the terms of the thesis are mentioned.

**Object: Users**

A user can be anyone or anything who e.g. tweets or follows another user [23].

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>followers_count</td>
<td>number of followers the user currently has</td>
</tr>
<tr>
<td>geo_enabled</td>
<td>geotagging of tweets enabled</td>
</tr>
<tr>
<td>id</td>
<td>unique identifier for the user</td>
</tr>
<tr>
<td>status</td>
<td>the user’s most recent tweet</td>
</tr>
<tr>
<td>statuses_count</td>
<td>number of user’s tweets</td>
</tr>
</tbody>
</table>

Table 4: User object of Twitter API

**Object: Tweets**

Tweets are the basic building blocks of Twitter and are also known as status updates [23].
### Field | Description
--- | ---
coordinates | geographic location of tweet as reported by the user or client application
entities | entities parsed out of text
favorite_count | how often tweet was favorited by users
id | unique identifier of the tweet
in_reply_to_status_id | identified of original tweet id if it is a reply
place | indicates association of tweet to place
retweet_count | how often tweet was retweeted
text | UTF-8 text formatted tweet
user | user who posted this tweet

Table 5: Tweet object of Twitter API

**Object: Entities**

Entities provide metadata about the posted content and are never divorced from the content they describe [23].

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hashtags</td>
<td>hashtags parsed out of tweet</td>
</tr>
<tr>
<td>media</td>
<td>media elements uploaded with tweet (has info about type of uploaded content, e.g. photo)</td>
</tr>
<tr>
<td>urls</td>
<td>URLs included in tweet</td>
</tr>
<tr>
<td>user_mentions</td>
<td>twitter users mentioned in tweet</td>
</tr>
</tbody>
</table>

Table 6: Entity object of Twitter API

**Object: Places**

Places are named locations with corresponding geo coordinates but they do not have to represent the location where the tweet was sent from [23].
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>country</td>
<td>country containing the place</td>
</tr>
<tr>
<td>name</td>
<td>name of the place (e.g. Vienna)</td>
</tr>
<tr>
<td>place_type</td>
<td>type of location (e.g. city)</td>
</tr>
<tr>
<td>attributes</td>
<td>e.g. the street address and postal code</td>
</tr>
</tbody>
</table>

Table 7: Places object of Twitter API

MySQL Connector for Java

Since a MySQL database has to be accessed to retrieve additional data the MySQL connector for Java in version 5.0.8 was used. The connector can be found on the MySQL developer page [https://dev.mysql.com/downloads/connector/j/5.1.html](https://dev.mysql.com/downloads/connector/j/5.1.html).

GATE

GATE stands for *general architecture for text engineering* and is used in version 8.0 build 4825. GATE is an open source software for solving text processing problems and a result of a R&D program which is already running since 1995 [24]. GATE can be downloaded from the project page and comes with a GUI: [https://gate.ac.uk/download/](https://gate.ac.uk/download/) For the usage within another program an embedded version is included in the download, a user guide for the usage of the embedded version can be found here [https://gate.ac.uk/releases/gate-8.1-build5169-ALL/doc/tao/splitch7.html#x11-1570007](https://gate.ac.uk/releases/gate-8.1-build5169-ALL/doc/tao/splitch7.html#x11-1570007).
4.1.3 Implementation

At first a look is taken on the class diagram and on the classes that exist in the calculation framework. Then each class is described and important implementation details are shown.

Classes

In figure 5 the class diagram of the framework is shown, properties and method names are not included.

![Class Diagram](image)

**Figure 5: Class diagram**

*FrontendBean*
The `FrontendBean` class is directly communicating with the xhtml pages (the settings and results page). It provides access to all the information and objects that are needed on the GUI or for calculation. The class is e.g. responsible for storing the uploaded configuration file directly on the server and starting the calculation process.

The bean holds objects for the `Configuration`, `Helper` and `GateHandler` classes. Furthermore it holds a list of `MyTweet` objects which are the currently calculated objects that get displayed on the results page. With the class is interacted in three ways:

```
public class FrontendBean {
    +tempFile: UploadedFile
    +configuration: Configuration
    +currentCalculations: List<MyTweet>
    +irmInfluence: IRMImpact
    +foundWords: HashSet<String>
    +relearnGate: boolean

    +FrontendBean()
    +onPageLoad(): void
    +getConfigPath(): String
    +startCalculation(event: ActionEvent): void
    -startSYCalculation(context: FacesContext): void
    -startIRMCalculation(): void
    +handleFileUpload(event: FileUploadEvent): void
    +showFeature(feature: String): boolean
    +AddFoundWords(foundWords: HashSet<String>): void
    -iterateOverTweetFolder(folders: List<File>): void
    -calculateNrofThreads(): void
    -startCalculationForTweets(): void
    -calculationInThread(lowerBound: final int, upperBound: final int)
    +getAverageScoreForCategory(category: String, feature: String, getOldScore: boolean): double
    +getTweetCountForCategory(category: String): int
    +getCategories(): List<String>
```

Figure 6: `FrontendBean` class

```
Figure 7: User interaction
```

Configuration
An object of the Configuration class will hold all those values that were specified in the uploaded configuration file and is created when the configuration file is uploaded. The configuration has to be an xml file and it will be passed to the constructor of the class. In the constructor the xml file will be parsed and the values are stored in properties of the class. Furthermore the class provides the following methods for extracting a configuration of a feature:

- List\(<\)ConfigEntry\(>\)getEntryTuples(String groupName, String entryName)

This method returns e.g. the configuration of the feature TweetLength. To get the values for this feature it has to be called like getEntryTuples("TweetLength", "Length"). The ConfigEntry class is described in a later part.
• `ConfigEntry getSingleEntry(String entryName)`
  This method is used when retrieving the configuration e.g. for the IsReply feature. The method call looks like `getSingleEntry("IsReply")`.

• `double getURLSpamPenalty()`
  This method simply returns the value that is stored in the spamPenalty attribute of the URLS tag in the configuration file.

• `double getOverallWeight(String groupName)`
  Like the prior method for the spam penalty this method returns the weight attribute for a given tag, e.g. when called for the WordCount feature `getOverallWeight("WordCount")`.

**GateHandler**

![GateHandler class](image)

The GateHandler does the whole interaction with GATE. When the handler is initialized the configuration object has to be passed so that it can extract the settings for GATE. During construction the local folders, where the files for learning and the other files are stored, are cleaned up. Afterwards the corpora are defined and the plugins that are needed for machine learning are loaded. When the GateHandler is fully initialized it can be started by calling its `startGate()` method. This method will prepare the pipelines for learning and categorizing and will also start them. When the whole process is finished the annotated files are saved as XML files in the local folders ”learning” and ”other”. The whole GATE process consists of three major steps:
These three steps also need three different pipelines. The first pipeline for annotating the learn set is set up as shown in figure 11.

![Figure 10: GATE workflow](image)

Since these processing resources are also used in the application pipeline they are described here in more detail.

The **gazetteer resource** contains two lists for the *calculation framework*, a measurement and a video list. The measurement list has some words in it that describe e.g. length-/height measurement whereas the video list contains words that are possibly referencing to a video e.g. the term *youtube*.

The **POS tagger** is initialized with the *german fast tagger* since the underlying tweets are
all in German. In 4.2 the configuration of GATE is demonstrated. In this configuration is also an entry for the POS tagger that enables to dynamically switch the POS tagger from one calculation to another. For correctly loading the needed tagger the whole path to the .tagger file has to be inserted, furthermore the tagger has to be in the plugins directory of GATE.

At the end of the pipeline are three JAPE rules that will annotate different parts in the text files. The class assignment rule will assign a category (info, private communication, video, other) to a specific file. The rule is implemented in the following way:

```
Phase: Tweet_Class_Assignment
Input: paragraph
Options: control = appelt

Rule: Classification_Assignment
{
    {paragraph}
} : par
------>
: par.CLASS = {CLASS}
```

Listing 18: class assignment jape rule

During runtime the #CLASS# placeholder will be replaced by e.g. kind=info. When this is done the current file that is processed will be annotated with the information that it belongs to the information category. Note that only during the annotation phase the placeholder will be replaced by such an information, in the application pipeline this will not be the case. The usage in the application pipeline is demonstrated afterwards.

The second jape rule mention assignment annotates every occurrence of a user name with the mention attribute. The rule looks like this:

```
Phase: Tweet_Mention_Assignment
Input: Token SpaceToken
Options: control = appelt

Rule: Mention_Assignment
{
    ({ Token.string==" ."})*({ Token.string=="@"}({ Token.string==" ."})*({ Token.
kind==number})*{Token.kind==word})
} : mention
```

Johann Tuder, BSc - 1255771
This rule annotates the occurrence of a text like @user with the attribute MENTION. The last jape rule is for annotating a retweet with RETWEET and this is done with the following code:

```plaintext
Phase: Tweet_Mention_Assignment
Input: Token SpaceToken
Options: control = appelt

Rule: Mention_Assignment

(  
  {Token.string=="RT"}  
): mention

->

:mention.RETWEET = {kind=RETWEET}
```

Listing 20: retweet assignment jape rule

**JAPE rules & Gazetteer list**

The jape rules and gazetteer lists were added after the first explorations with the machine learning component were done. Before those were added the precision of the categorization process was worse than expected:

<table>
<thead>
<tr>
<th>Category</th>
<th>Correctly categorized tweets</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>92.7%</td>
</tr>
<tr>
<td>private</td>
<td>0.0%</td>
</tr>
<tr>
<td>video</td>
<td>100.0%</td>
</tr>
<tr>
<td>info</td>
<td>80.6%</td>
</tr>
<tr>
<td>overall</td>
<td>83.3%</td>
</tr>
</tbody>
</table>

Table 8: Precision without JAPE & Gazetteer

The private category was not "recognized" by the machine learner, every tweet that should have been in this category was whether recognized as a video or as an other tweet. However there is a clear distinction from the private tweets to the other categories, because
A private tweet has to start with a username, e.g. @userName. After adding the rules and lists to the pipeline the precision increased to the following values:

<table>
<thead>
<tr>
<th>Category</th>
<th>Correctly categorized tweets</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>90.0%</td>
<td>-2.7%</td>
</tr>
<tr>
<td>private</td>
<td>96.3%</td>
<td>+96.3%</td>
</tr>
<tr>
<td>video</td>
<td>96.3%</td>
<td>-3.7%</td>
</tr>
<tr>
<td>info</td>
<td>100.0%</td>
<td>+19.4%</td>
</tr>
<tr>
<td>overall</td>
<td>95.7%</td>
<td>+12.4%</td>
</tr>
</tbody>
</table>

Table 9: Precision with JAPE & Gazetteer

This overall increase showed that the use of the jape rules and the gazetteer component is useful. After applying those rules on a file the annotating process is finished. In the next step the annotated files are used for learning, which is done with the learning pipeline that is shown in figure 12.

Figure 12: Pipeline for learning from the annotated files

Here the learning corpus has to be passed to the batch learning resource and the learningMode has to be set to LEARNING, when the pipeline is executed the component will start learning from the corpora. After learning the rest of the tweets can be categorized. Therefore an application pipeline has to be created and organized like it is shown in figure 13.
As one can see the application pipeline is a combination of the other two pipelines where at the beginning the file is annotated and afterwards the machine learning component is trying to categorize the text file.

The difference to the other two pipelines concerns the mention assignment jape rule and the batch learner. In the mention assignment the placeholder `#CLASS#` is replaced by an empty string and in the batch learner the `learningMode` is set to `APPLICATION` and the `outputASName` parameter is set to `Result`. This leads to an empty annotation of the text by the jape rule and so the batch learner is able to annotate it and the annotation will be stored in the `Result` annotation set.

The usage of the embedded GATE library is demonstrated in a user guide on the GATE project page that is already linked in the section Libraries.

**Helper**
The Helper class provides some methods that are used by multiple other objects. It provides a rounding method for double values, a ranking method for a `List<MyTweet>` object and a `getDocument(File xmlFile)` method that returns a document object from the given file.

Additionally it provides two methods for extracting information from xml files that were annotated by GATE.

The first method `getCategoryOfTweet(File xmlFile)` gets the category (of the given file) that was assigned by GATE. Since GATE generates XML files as output the information is stored in a construct like this:

```xml
<Annotation Id="50" Type="CLASS" StartNode="0" EndNode="138">  
  <Feature>
    <Name className="java.lang.String">kind</Name>
    <Value className="java.lang.String">other</Value>
  </Feature>
</Annotation>
```

To get the information it has to be iterated over all `<Annotation>` elements, because there are always multiple tags of this type in a file. The correct annotation is found when the `TYPE` attribute of the tag is `CLASS`. Then the `<Feature>` tag can be examined for the needed value that is then stored in the `<Value>` tag. For the example shown above the category is `other`.

The second important method is `getPlainTweetFile(File xmlFile)` that returns
the path to the original txt file, that only contains the message of the tweet. The information is stored within the following tag:

```xml
<GateDocumentFeatures>
  <Feature>
    <Name className="java.lang.String">gate.SourceURL</Name>
    <Value className="java.lang.String">file:/D:/gate%20workspace/learn%20tweets/other/34304544672977920.txt</Value>
  </Feature>
  ...
</GateDocumentFeatures>
```

Listing 22: Path to original tweet .txt file

The method iterates over the `<Feature>` tags in the `<GateDocumentFeatures>` node and stops when it found a node where the value of the `<Name>` node is `gate.SourceURL`. Then the string of the `<Value>` is taken. The preceding `file:` is removed and all occurrences of `%20` are replaced by a whitespace character. The adapted string is then returned.

**ConfigEntry**

![ConfigEntry](image)

Figure 15: ConfigEntry class

This class deals with the configuration elements from the configuration file. It holds the values and weights specified in the xml.

**DBConnection**

![DBConnection](image)

Figure 16: DBConnection class
An object from the DBConnection class can only be instantiated by passing a Connection object. The connection object is available as a property of the Configuration object and is created by a connection string that is passed through the configuration file.

The class provides the method

`ExecuteStatement(String query, List<String> parameters)` that prepares the passed query by setting the parameters. The following exemplary query needs one parameter:

```
SELECT Longitude
FROM tweet
WHERE TweetID = ?
  AND Longitude IS NOT NULL
```

Listing 23: SQL query with a parameter

A parameter is defined by the question mark character and has to be replaced by a concrete value before execution. This value is used from the parameters list that is passed to the method.

After replacing the parameters the query is executed and the result set returned.

**Score**

![Score class](image)

Figure 17: Score class

The Score class holds all values that where calculated for a tweet and it is part of a MyTweet object. All values are per default initialized with 0.0.

**UserInfo**
This class holds the information about the user that submitted the corresponding tweet.

**MyTweet**

This class stores all information of a single tweet. This includes the xml file, the message, the current and prior score and the configuration value for the feature. For the different calculations a tweet is needed in different representations.

To demonstrate how each of the single entries is represented an example tweet is used:
The originalMessage is the above mentioned text. The originalSplittedTweet is a List<String> object where the original message is split up on every whitespace character. This means that the list exists of chunks like: {#Hochwasser, #Schöna, @, ...} and so on.

The splittedNoLinkAndUserLowerCase property contains the original string where links and user mentions were removed. The removal of the links (URLs) is done by iterating over each chunk of a tweet and creating an URL object. If the creation of this object does not throw an exception an URL was found and is replaced by an empty string. The user mentions are removed by using the regular expression @[^\w]*. This only affects strings like @userName and not the @ alone like in the used example.

After the removal of those elements the string is converted to lower case, so that it can be compared better. The final list of words looks like this: {#hochwasser, #schöna, @, #elbe, —&, #pegel, in, 9, minuten, von, 979, cm, um, 5, cm, auf, 974, cm, gefallen., [Stand:, 07.06.2013, 18:30]}

The next property splittedNoLinkUserAndHashtagList uses the exact same representation from above and removes the hashtags. Therefore it is iterated over every element in the list and if the current element starts with a # character the element is removed. This results in the following list: {@, —&, in, 9, minuten, von, 979, cm, um, 5, cm, auf, 974, cm, gefallen., [Stand:, 07.06.2013, 18:30]}

The property splittedNoPunctuation contains the original tweet in chunks where all elements except number, letters and whitespaces are removed. For this adaption the regular expression [^a-zA-Z\söüöü0-9] is used. The resulting message looks like this:

Hochwasser SCHÖNA ELBE gt Pegel in 9 Minuten von 979 cm um 5 cm auf 974 cm gefallen Stand 07062013 1830

The message property consists of the original message in lower case where also the occurrences of &gt; were replaced by >. In this step this is done because this failure was already made during extraction of the tweets where such symbols got HTML encoded. Also all characters that are not letters, numbers or whitespaces and URLs are removed.
again. This is the result of the transformation:

| hochwasser schöna elbe pegel in 9 minuten von 979 cm um 5 cm auf 974 cm gefallen stand 07062013 1830 |

Afterwards the message is split up at the whitespaces for the `splittedTweet` property which looks like this: `{'hochwasser, schöna, elbe, pegel, in, 9, minuten, von, 979, cm, um, 5, cm, auf, 974, cm, gefallen, stand, 07062013, 1830'}`

At the end a stemmer is used to stem each single word in the tweet and to store it into the `stemmedSplittedTweet` property. This results in the following stemmed list: `{hochwass, schona, elb, pegel, in, 9, minut, von, 979, cm, um, 5, cm, auf, 974, cm, gefall, stand, 07062013, 1830}`

**IRMInfluence**

This is the class for the IRM calculation. The class is simply retrieving the user information from the database and is afterwards ranking the users.

The values are retrieved using the `userInfo` query that is specified in the configuration file. Afterwards it is iterated over the result set and for each entry a `UserInfo` object is generated.

After parsing the result set the entries are ranked. Therefore three comparators are used (for follower count, mention count and retweet count). A comparator looks like this:

```java
Collections.sort(userList, new Comparator<UserInfo>() {
    @Override
    public int compare(UserInfo ui1, UserInfo ui2) {
        if (ui1.getRetweetCount() > ui2.getRetweetCount()) {
            return -1;
        }
    }
});
```
Listing 24: IRMInfluence sorting comparator

When the sorting is finished the calculator has finished all its work.

**SVC calculator**

This is the class of the *calculation framework*. It has a method for each feature that can be calculated, e.g. `double calculateTermWeight(MyTweet tweet)`.

The majority of the features use the same mechanism where a specific value has to be found in the configuration file and the score for this entry read. This mechanism is explained by the `calculateTermWeight()` method. In this method it is iterated over each entry of the tweet’s `stemmedSplittedTweet` list that was described earlier. For each word it is checked if it occurs in the configuration that looks like this:

```
<TermList weight="1" activated="true">
  <Term value="regen" weight="0.3" />
  <Term value="pegel" weight="0.7" />
  <Term value="meter" weight="1.0" />
</TermList>
```
In this method a `<Term>` tag is a ConfigEntry object. If the current word from the
tweet matches an entries `value` attribute the weight is taken and added to a temporary
score. If a tweet has e.g. the word `pegel` twice in it, it is only counted once. Therefore
a list with the configuration entries is kept and updated when a tweet’s word is found.
After iterating over the words the temporary score is multiplied by the weight that is
specified in the `weight` attribute of the `<TermList>` tag.

However one of the more complex methods is the `calculateSpellingError(MyTweet
tweet)` method. In this method it is iterated over each word that occurs in the tweet.
Then it has to be checked if the word is written wrong or not. This is done in two ways.
First the `SVCalculator` holds a HashSet with all the words that were found in prior runs.
If the word is in this list nothing has to be done and the next word can be investigated,
but if the word does not appear in this list the German wiktionary is called with the
word. From this call a json-object is returned and this object has the attribute `totalhits`.
If this value is greater than 0 then the word was found and will be added to the HashSet
so that when the word occurs again this call to the external service does not have to be
done again.

Another useful method is `getURLOfShortenedLink(String link)`. This method
accepts an url as parameter and tries to find the ”real” url if it was shortened by an
url shortener like [http://www.goo.gl](http://www.goo.gl). The approach for this method was taken
from [http://www.srccodes.com/p/article/37/expand-shortened-link-
using-java](http://www.srccodes.com/p/article/37/expand-shortened-link-
using-java).

For the final score all scores of the `calculateXXX()` methods are summed up and
divided by the number of features that are activated for the current calculation. The
calculated value is written to the `overallScore` property of the `MyTweet` object.
After the calculation also the `userID` of the user who wrote the tweet is added to the
`MyTweet` object.
Adding Features

Code
When a new feature should be added to the calculation framework the Configuration class has to be adapted. Here the developer has to add two properties for each feature, one for the state and one for rendering the checkbox. Afterwards in the constructor of the Configuration class the activated attribute of the new feature tag in the xml has to be read. Furthermore in the Configuration.featureGroupListener(String group) method the name of the new feature has to be added to a group, so that a click on the feature-group checkbox is handled correctly. After editing the Configuration class theSVCalculator class has to be adapted. In the calculate(MyTweet tweet) method the new feature name has to be added and also the method for calculating the score of this particular feature. Furthermore the score that the new feature will achieve has to be added as a property to the Score class.

GUI
In the index.xhtml a new <p:selectBooleanCheckbox> element has to be added. For the value and disabled attributes the corresponding properties of the Configuration class have to be used. On the results.xhtml page a new <p:column> element has to be added to the <p:dataTable id="dataTable">. It is also possible to show additional information to a tweet and not only the result, therefore the information of interest has to be added as a property to the MyTweet class.
4.2 User documentation

In this section the interaction of the user with the framework is described.

4.2.1 Installation

The framework can be used in two different ways, in the Eclipse IDE or on a Tomcat server. In this section it is shown which software has to be installed for running the framework in both ways.

Java

First of all Java has to be installed on the machine where the framework will run. It is recommended to use the 64 bit version of Java since all the data is loaded into the memory and therefore consumes a lot of space. It is recommended to download the JDK version from the following site: http://www.oracle.com/technetwork/java/javase/downloads/index.html.

MariaDB (MySQL) & Tomcat

MariaDB (a MySQL fork) and Tomcat can be installed with a single package called XAMPP which can be downloaded from the project site: https://www.apachefriends.org/de/index.html. The package also provides PHP and Perl but those are not used by the framework.

The database has two tables, one for the tweets and the other one for the users that wrote the tweets. In this case the name of the database is ”highwater” and the tables are named ”tweet” and ”userinfo”. To set up the database the following SQL statement has to be executed on the server:

```sql
CREATE DATABASE IF NOT EXISTS 'highwater' DEFAULT CHARACTER SET latin1 COLLATE latin1_swedish_ci;
USE 'highwater';

CREATE TABLE 'tweet' (
    'TweetID' bigint(20) unsigned NOT NULL,
    'UserID' bigint(20) unsigned NOT NULL,
    'CreatedAt' datetime NOT NULL,
    'CalculatedRelations' bit(1) NOT NULL DEFAULT b'0',
    'WordScore' float unsigned DEFAULT NULL,
);```
Tomcat has to be started via the XAMPP Control Panel and further configuration shouldn’t be required except if other services are using the ports where Tomcat should listen on. If this is the case the configuration file can be opened via the XAMPP Control Panel and updated with the new port numbers.

For running the framework on Tomcat the Web-Archive-File (war) has to be copied on the server, the path to the folder is: 

"%XAMPP_INSTALLATION_PATH%\tomcat\webapps\". The war file will be extracted
by the server during start-up or while it is running. After the file is deployed the framework can be opened with any browser by opening the URL \texttt{http://localhost:8081/SocialValue/faces/index.xhtml}. In the shown URL the Tomcat server is running on port 8081.

\textit{Eclipse IDE}

A development environment like the Eclipse IDE has to be used if the framework should be debugged or the code extended. It is recommended to download the 64 bit (Java and Eclipse have to be the same architecture) ”Eclipse IDE for Java EE Developers” version from the following site: \texttt{https://www.eclipse.org/downloads/}. The downloaded zip-file contains a single folder with the IDE and can be extracted anywhere and started.

Afterwards the framework has to be imported. This is done by clicking \textit{File - Import…} in the menu. In the following step an import source has to be selected. The ”WAR file” entry can be found in the ”Web” folder. In the following step the war-file has to be selected and a target runtime has to be chosen or created. The target runtime is a Tomcat server and in Eclipse at least version 8 of Tomcat has to be used. Since in the above mentioned XAMPP package only Tomcat server 7 is provided the newer version has to be downloaded from the following site: \texttt{https://tomcat.apache.org/download-80.cgi} After creating the runtime the war-file can finally be imported.

\subsection*{4.2.2 Setting up the configuration file}

The configuration is set up in multiple parts. In the first part environment variables for the calculator are defined:

\begin{quote}
\begin{lstlisting}[language=XML]
<?xml version="1.0" encoding="utf-8" ?>
<SocialValueConfig>
  <MaxMessageLength value="140" />
  <StopWordsList path="/resources/stopwords.txt" />
  <EmoticonList path="/resources/emoticons.txt" />
</SocialValueConfig>
\end{lstlisting}
\end{quote}

\textbf{Listing 26: Environment settings}

The root value of the configuration XML is the \texttt{SocialValueConfig} tag, below this tag are all settings defined. The first three child elements are self-explanatory since they only set the maximum value that a message can have and the paths to the stopword and emoticon
list. The next child element is the GATE configuration:

```
<GateConfig>
  <LearnSet path="G:\gate workspace\learn tweets" />
  <TestSet path="G:\gate workspace\unlearned tweets\tweets" />
  <MLConfig path="G:\gate workspace\ml-config-file.xml" />
  <JapeClassAssignment path="G:\gate workspace\Tweet_Class_Assignment.jape" />
  <JapeMentionAssignment path="G:\gate workspace\Tweet_Mention_Assignment.jape" />
  <JapeRTAssignment path="G:\gate workspace\Tweet_RT_Assignment.jape" />
  <GazetteerList path="G:\gate workspace\twitter_lists.def" />
  <POSTagger path="G:\Gate\plugins\Tagger_Stanford\resources\german-fast.tagger" />
  <PluginsHome path="G:\Gate\plugins\" />
</GateConfig>
```

Listing 27: GATE configuration

Here are also the paths set to the files that contain the necessary data for the processing resources that were already explained in section "Classes".

**Folder structure**

To build the correct categories during calculation, the tweets, of which the machine learning component should learn from, have to be organized the following way: Beneath the parent folder (in this case "learn tweets") exists at least one sub-folder. The names of the sub-folders will be the names of the categories ("info", "other", "private", "video") that are learned and the folders consist of the text files. In the case of the thesis the folder structure looks like in figure 22.

![Learn set folder structure](image)

Figure 22: Learn set folder structure

Afterwards the configuration for the database connection has to be setup:

```
<!-- Connection to database -->
```
In this configuration the used Java driver is specified. This enables to dynamically switch to another database, when it is needed. For switching to another database type the driver has to be placed into the \texttt{lib} folder in the project and the \texttt{connectionString} attribute has to be adapted. Within the \texttt{Database} node are the queries listed up that are needed for calculation. The \texttt{key} attribute of those queries should not be changed, but the \texttt{statement} can be changed. Furthermore it should be noted that the return sets of the queries should stay the same and the \texttt{?} character is a placeholder for a parameter that will be set by the calculator.

Below the configuration of the database the configuration of the features start. The detailed configuration of each feature is shown in the \textbf{Evaluation} section, whereas the following snippet provides an overview of how the elements are structured in the configuration file:

```xml
<Database driver="com.mysql.jdbc.Driver" connectionString="jdbc:mysql://localhost/highwater?user=root">
  <Query key="longitude" statement="Select Longitude From tweet Where TweetID = ? AND Longitude IS NOT NULL" />
  <Query key="userIDForTweet" statement="Select userID From tweet Where TweetID = ?" />
  <Query key="latitude" statement="Select Latitude From tweet Where TweetID = ? AND Latitude IS NOT NULL" />
  <Query key="tweetRetweetCount" statement="Select retweetCount From tweet Where TweetID = ?" />
  <Query key="favoriteCount" statement="Select favoriteCount From tweet Where TweetID = ?" />
  <Query key="userInfo" statement="Select * from userInfo" />
</Database>
```

```xml
<TextualFeatures activated="true">
  <TermList weight="1" activated="true">
    <Term value="regen" weight="0.3" />
    ...
  </TermList>
</TextualFeatures>
```
<WordCount weight="1" activated="true">
  <Count value="5" weight="0.0" />
  ...
</WordCount>

<SpellingError value="98" weight="1" activated="true" />

<TweetLength weight="1.0" activated="true">
  <Length value="25" weight="0.0" />
  ...
</TweetLength>

<Emoticons weight="-1.0" activated="true" />

<RepeatedCharacters value="3" weight="-1.0" activated="" />

<CapitalizedLetters weight="1.0" activated="" >
  <Count value="15" weight="1.0" />
  ...
</CapitalizedLetters>

<StopWords weight="1.0" activated="true" >
  <Ratio value="5" weight="0" />
  ...
</StopWords>

<POS weight="1.0" activated="true" />
</TextualFeatures>

<AdditionalContent activated="true">
  <Locations weight="1.0" activated="true">
    <Location value="magdeburg" weight="1.0" />
    ...
  </Locations>

  <URLs spamPenalty="-1.0" weight="1.0" activated="" >
    <URL value="wetterblogger.de" weight="1.0" />
    ...
  </URLs>
</AdditionalContent>
4.2.3 Initial view

When the system is accessed the first time it looks like shown in figure 23. On top of the page are two buttons for interacting with the configuration file. Below the two buttons are three groups of features:

- **Textual Features**
- **Additional Content Features**
• **Networking Features**

Each group has multiple features in it. Every group and also every feature has a checkbox beside it which are used for enabling or disabling features for the next calculation. Below the groups is an additional checkbox for resetting GATE and a button for starting the calculation.
Figure 23: Initial view of the system
4.2.4 Selecting a configuration

The first step after opening the calculation framework is to upload a configuration file. Therefore the Choose config... button has to be pressed. Afterwards a system dialog is opened where the user can select a file. In figure 24 it is shown how the interface looks directly after selecting a file and closing the system dialog.

With the Upload config button the selected configuration file is uploaded to the server. When the button is hit the configuration gets stored on the server and the xml file is parsed. After parsing the checkboxes of the GUI are set as shown in figure 25.
Figure 25: After uploading a configuration
Enabling/Disabling features

After the upload process of the configuration file is finished the user can enable and
disable features by (un-)checking the checkboxes beside the features. If a feature should
be disabled the checkbox has to be unchecked. It is also possible to disable all features
of a feature-group by deactivating the checkbox beside the group name. As one can
see in figure 25 there are three features where the checkbox is grayed-out, these are
RepeatedCharacters, CapitalizedLetters and URLs. Those features were already disabled
in the configuration file, e.g.:

```xml
<RepeatedCharacters value="3" weight="-1.0" activated=""/>
```

Listing 30: Deactivated repeated characters feature

If a feature is already disabled in the configuration file it can not be activated afterwards
on the GUI. When the feature should be used a new configuration file has to be uploaded
where the activated attribute is set to true.

Resetting GATE

GATE has to be reset when the underlying data has changed. This can happen e.g. when
new categories for the learn set were defined or when annotation files changed. When this
checkbox is activated the whole GateHandler will be reinitialized with the values that are
provided in the configuration file.

Start calculation

When all settings are set the user can click on the Start calculation button and the
calculation will be immediately started. The workflow of the calculation process is shown
in figure 26.
4.2.5 Viewing results

In figure 27 the result view for a calculation with all textual features enabled (except TermList, RepeatedCharacters and CapitalizedLetters) is shown.
At the top of the page is a small back link that brings the user back to the settings page. Below this link are multiple tables. The first table *OverallScore* shows the average values that were achieved by each category. The following tables have the same layout and show the average values of each category for the corresponding feature. After the feature tables a small description of the features can be displayed as shown in figure 28. When the checkbox for the papers is activated the corresponding papers that were used during this thesis are displayed.
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TermWeight</td>
<td>The paper &quot;Twitter topic summarization by ranking tweets using social influence and content quality&quot; mentions that a tweet or user is important if a tweet contains important terms.</td>
</tr>
<tr>
<td></td>
<td>In some studies short tweets were ignored, because they can not really deliver useful information. Furthermore it was found out that the length often is a sign for quality.</td>
</tr>
</tbody>
</table>
| WordCount     | - Twitter topic summarization by ranking tweets using social influence and content quality  
               | - Reorder User’s Tweets  
               | - Are Some Tweets More Interesting Than Others? #HardQuestion  
               | - Identifying interesting Twitter contents using topical analysis  
               | It is often mentioned that low-quality tweets contain many spelling errors and therefore they calculate a ratio of the words that where found in a dictionary. |
| SpellingError | - "Twitter topic summarization by ranking tweets using social influence and content quality"  
               | - Are Some Tweets More Interesting Than Others? #HardQuestion  
               | - Searching for Quality Microblog Posts; Filtering and Ranking Based on Content Analysis and Implicit Links  
               | - An Empirical Study on Learning to Rank of Tweets  
               | Here we also have the assumption that a longer tweet has a higher quality. This assumption also results from the fact that a longer tweet probably contains more information. |
| TweetLength   | - Reorder User’s Tweets  
               | - An Empirical Study on Learning to Rank of Tweets  
               | Emoticons                                               | In "Twitter topic summarization by ranking tweets using social influence and content quality" it is stated that the number of emoticons are useful for measuring content quality. It is assumed that emoticons negatively influence the quality. |
| RepeatedCharacters | Repeated characters like the 'r' in 'shhhhh' often occur in personal opinions and according to Searching for Quality Microblog Posts; Filtering and Ranking Based on Content Analysis and Implicit Links they do not transport any useful information. |
| CapitalizedLetters | The paper Searching for Quality Microblog Posts; Filtering and Ranking Based on Content Analysis and Implicit Links also found out that capitalized letters are often strongly connected to repeated characters. So mostly personal opinions and messages will contain such letters. |
| stopwords     | Multiple papers mention that the presence of stopwords is positively correlate with the informativeness of a text and that documents with few stopwords are more likely to be irrelevant. |
| POS           | STILL MISSING                                                                                                                                 |

**Additional Content Features**

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
</table>
| Locations | Multiple studies found out that the presence of location information, especially during hazard events, are usually more important and have a higher quality  
               | - Are Some Tweets More Interesting Than Others? #HardQuestion  
               |
Below the description is the table with the final scores. The default sorting is done via the *OverallScore*, but the user can also sort by other features. In the first column of the results table is the text of the tweet, followed by a column with the category it belongs to and the overall score. Afterwards are the columns for the activated features and at the end are the columns of the IRM calculator. For each value exists a dynamic description about how the tweet reached a particular score in a feature category. This description is shown as a tooltip that is activated by hovering a score in the results table. The default tooltip looks like in figure 29 where a counter followed by the corresponding configuration entry it fits in and the weighting value is shown.

![Figure 29: Default tooltip](image)

For part-of-speech tagging a special tooltip was implemented that shows the occurrences of parts-of-speech elements. This tooltip is shown in figure 30.

![Figure 30: part-of-speech tooltip](image)
Figure 31 shows the results page after a second run where only the part-of-speech tagging feature was enabled. Here it is demonstrated how the old scores are displayed besides the newly calculated scores.

<table>
<thead>
<tr>
<th>Category</th>
<th>Current Avg Score</th>
<th>Old Avg Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>7.43</td>
<td>7.43</td>
</tr>
<tr>
<td>private</td>
<td>6.72</td>
<td>6.72</td>
</tr>
<tr>
<td>info</td>
<td>9.29</td>
<td>9.20</td>
</tr>
</tbody>
</table>

Figure 31: Results after second run
4.2.6 Adaptions

The framework is independent of the underlying domain of data. This is demonstrated in the chapter [Evaluation] where the flood and the #aufschrei data sets are used and compared. When the framework is used with data of a different domain the configuration should be adapted to gain better and more reliable scores. The adaption is not necessary for all features, but for the following features an adaption will have a positive influence:

- Domain related terms
  Here it is useful to provide a list of terms that are frequently used in the new domain or which are significant for qualitative texts within it.

- Emoticons
  When it is necessary to find tweets that transport personal feelings the feature has to be changed so that a positive score is used for tweets containing emoticons.

- Location
  This feature has to be adapted according to the underlying source and if it is necessary to focus on such information.

In the configuration file of the framework the value for the maximum length of the message has to be adapted when the texts that are used are not tweets. This means that if e.g. the contributions to a message board should be analyzed and a contribution has a maximum length of 500 characters, this value has to be set in the MaxMessageLength node of the configuration file.

Furthermore it may be necessary to adapt the JAPE-rules and gazetteer-lists that are used for categorizing parts of a tweet. Currently the JAPE rules can classify parts of a tweet as retweet or user mention. If the investigated data are not twitter messages those rules will not be necessary and can be removed or replaced by other rules. The gazetteer-lists are labeling parts of tweets as measurements, e.g. ”cm” or ”m”, or as signal words for videos like ”Youtube”.

Johann Tuder, BSc - 1255771
5 Evaluation

In the first section of this chapter the quality measure for a tweet in the context of the thesis is defined. Afterwards a detailed look on each feature is taken:

- What is the feature about?
- How is it configured?
- How does an exemplary tweet for this feature look like?
- How high is the score?
- What is the difference compared to the #aufschrei data set?

Comparison

For comparison of the quality and for demonstrating the generic use of the framework a second data set from a different domain is used. The second data set is about the #aufschrei discussion that was already mentioned in the Scope chapter.

The result of the framework is a score for each tweet. The score shows if a tweet is qualitative (high score) or not (low score). To ensure that the framework works as expected the two above mentioned data sets (floods and #aufschrei) are used.

5.1 Quality

The focus in this thesis is on the tweets about floods that took place in summer 2013 in Austria and Germany. It is assumed that tweets that are mentioning the current situation at a specific place are the most useful ones. This is restricted to tweets that provide information like flood levels or progress of the floods. Whereas in the comparing data set about the #aufschrei discussion qualitative tweets are those that do not transport personal opinions but information and insight.

5.2 Textual Features

5.2.1 Domain related terms

In the section Domain related terms it was mentioned that a tweet is of higher quality if important terms are used. This also means that words that are domain dependent and
semantically rich for a specific topic are more interesting than others. For the floods-topic the following configuration is used:

```xml
<TermList weight="1" activated="true">
  <Term value="regen" weight="0.3" />
  <Term value="pegel" weight="0.7" />
  <Term value="meter" weight="1.0" />
  <Term value="cm" weight="1.0" />
  <Term value="liter" weight="1.0" />
</TermList>
```

In the configuration a list of important words and their weights is specified. Each word has its own weight and when it appears in a tweet, the tweet gets the specified weight. If multiple words of the list occur in a tweet they are aggregated, but if one term appears several times in the same tweet the score is only counted once. A tweet with the highest score of 1.7 points with this configuration containing the words *pegel* and *meter* is the following one:

```markdown
#Hochwasser Neuer Pegel für #Magdeburg-Strombrücke: 7.39 Meter (17 Uhr)
```

Table 10 shows the average scores of each category. With an average score of 1.46 the *info* category performs a lot better than the other categories. This shows that the tweets within the *info* category contain a lot of the important terms that were specified and therefore provide higher quality.

<table>
<thead>
<tr>
<th>Category</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>0.01</td>
</tr>
<tr>
<td>private</td>
<td>0.01</td>
</tr>
<tr>
<td>video</td>
<td>0.02</td>
</tr>
<tr>
<td>info</td>
<td>1.46</td>
</tr>
</tbody>
</table>

Table 10: Scoring of the domain related terms feature

**Comparison**

This feature was not used for the *aufschrei* data set, because it is a very content-related feature. To use this feature the content of some important or qualitative tweets should be known before the calculation.
5.2.2 Word Count

As stated in an earlier section a tweet that has a lot of words can provide more useful information. Therefore the configuration should be set up in a way that tweets with a lot of words get more points than those with fewer words. The configuration looks like this:

```xml
<WordCount weight="1" activated="true">
    <Count value="5" weight="0.0" />
    <Count value="14" weight="0.3" /> <!-- up to 14 words 0.3 points -->
    <Count value="*" weight="1.0" /> <!-- more than 15 words 1 point -->
</WordCount>
```

The following tweet got 1 point, because it has 20 words:

```
#Hochwasser #DAMNATZ @ #ELBE —&gt; #Pegel in 1 Stunde von 645 cm um 5 cm auf 650 cm gestiegen. [Stand: 08.06.2013 09:20]
```

As already mentioned in the implementation chapter every word and also every number or date is counted as one word by the algorithm. The shown tweet provides a very detailed information about what is going on at a specific place and therefore the rating is as expected. A tweet that did not get any point is the following:

```
HOCHWASSER IN DRESDEN 07.06.2013 http://t.co/JDzMlldQoY
```

The tweet only consists of 4 words, because the URL is filtered out and not considered as word. As it is shown in table [11] the info category performs better on average than the other three categories. Which demonstrates that tweets in the info category use more words and therefore have a higher quality.

<table>
<thead>
<tr>
<th>Category</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>0.58</td>
</tr>
<tr>
<td>private</td>
<td>0.57</td>
</tr>
<tr>
<td>video</td>
<td>0.39</td>
</tr>
<tr>
<td>info</td>
<td>0.89</td>
</tr>
</tbody>
</table>

Table 11: Scoring of the word count feature

**Comparison**

The average score for the data set is 0.65 in this feature category. Compared to the results
in table 11 the value is lower than for the info category, but it is better than those of the other categories. Tweets were present in all three ranges that were specified in the configuration. Which also shows that this feature alone is probably not sufficient enough for finding a qualitative tweet.

5.2.3 Spelling Errors

It is assumed that a tweet of high quality usually does not have any spelling error. Therefore the words are searched on wiktionary.org and if a word was not found there it is assumed as a spelling error. In the configuration it is specified that 98% of a tweet’s words have to be found so that it gets points:

\[
\text{<SpellingError value="98" weight="1" activated="true"/>}
\]

A tweet that did not get a point is the following one:

```
#Hochwasser Rekordhochwasser - Die Umwelt- und Gesundheitsrisiken nach der Flut
http://t.co/nw1K6zjrqs via @welt
```

This is because the term ”Rekordhochwasser” does not exist in the German language and was created by the user.

<table>
<thead>
<tr>
<th>Category</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>0.70</td>
</tr>
<tr>
<td>private</td>
<td>0.79</td>
</tr>
<tr>
<td>video</td>
<td>0.72</td>
</tr>
<tr>
<td>info</td>
<td>0.88</td>
</tr>
</tbody>
</table>

Table 12: Scoring of the spelling errors feature

As shown in table 12 tweets of the category info have the highest score in this feature. This is because most of the tweets in this category come from news agencies or television stations. They probably look more carefully on a text before tweeting it, whereas tweets by normal twitter users often contain individual creations like those in the example above. Those tweets occur more often in the other three categories and so it can be seen that those tweets are of lower quality.

Comparison
The average score in the spelling error feature is 0.76. This score is very similar to the scores of the flood data set. After analyzing some tweets it is unveiled that useful tweets sometimes mention a TV show or something similar which can not be found in a dictionary. This results in a punishment of the following tweet, because the word HörPunkt could not be found:

```
#aufschrei: HörPunkt zum Thema «Mann Macht Frau» am Samstag ab 10 Uhr live aus dem Museum der Kulturen in Basel: http://t.co/IESv24LT
```

Furthermore this means that the spelling error feature is not very significant for finding qualitative tweets.

### 5.2.4 Length Of Tweet

The length of a tweet is strongly related to the Word Count feature. The difference is that e.g. a more qualitative text like a newspaper article could probably use longer words, because they are more domain specific (technical terms etc.). So if a tweet would have a lot of such words it would not achieve a high score in the Word Count feature, but this feature compensates the problem. The configuration for the length of a tweet consists of a numeric value that indicates the ratio of used characters to maximum possible characters and an assigned weight:

```
<TweetLength weight="1.0" activated="true">
  <Length value="25" weight="0.0"/> <!-- up to 25% of used characters get no points -->
  <Length value="50" weight="0.3"/> <!-- up to 50% 0.3 points -->
  <Length value="84" weight="0.7"/> <!-- up to 84% 0.8 points -->
  <Length value="*" weight="1.0"/> <!-- more than 84% get 1 point -->
</TweetLength>
```

Also in this section a useful tweet scores with one point:

```
#Hochwasser #SCHÖNA @ #ELBE —&gt; #Pegel in 9 Minuten von 979 cm um 5 cm auf 974 cm gefallen. [Stand: 07.06.2013 18:30]
```

The tweet has a length of 121 characters and therefore reaches a 86.4% character usage ratio. One of the tweets that did not get a point is the following:
Again this tweet does not provide any further information, it only uses 18 characters which results in an usage ratio of about 12.9%. Also in table 13 it is shown that the info category achieves slightly higher scores than the other three categories. The comparison of the tables 13 and 11 demonstrates that the word count and length of tweet feature are very similar, but in the score they differ a lot, because in the word count feature the info category is much more superior than it is in the current feature.

<table>
<thead>
<tr>
<th>Category</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>0.87</td>
</tr>
<tr>
<td>private</td>
<td>0.81</td>
</tr>
<tr>
<td>video</td>
<td>0.83</td>
</tr>
<tr>
<td>info</td>
<td>0.96</td>
</tr>
</tbody>
</table>

Table 13: Scoring of the length of tweet feature

Comparison

For this feature the average value in the comparison data set is 0.8. This score is lower than those of the categories that are present in the floods data set.

5.2.5 Emoticons

Usually emoticons (e.g. :) ) can not provide any information, they only show the feelings or the mood of the user. So all tweets that have an emoticon will be penalized with the setting below:

<pre>
&lt;Emoticons weight="-1.0" activated="true" /&gt;
</pre>

The following tweet gets a penalization, because of the emoticon at the end of the text:

@Bejuel ja, bei mir geht’s erst morgen wieder los. War seit letzten Donnerstag frei wegen dem Hochwasser.... und ja, Baekho sieht süß aus =D

Table 14 shows that on average all four categories are not very different, but two points about the categories have to be addressed: First, the private category has the lowest score, it contains the most tweets that have emoticons which is expected and secondly
the 0 point score of the info category seems normal under the fact that the set mostly contains news tweets and they probably do not use emoticons very often to spread news.

<table>
<thead>
<tr>
<th>Category</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>-0.01</td>
</tr>
<tr>
<td>private</td>
<td>-0.05</td>
</tr>
<tr>
<td>video</td>
<td>-0.01</td>
</tr>
<tr>
<td>info</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Table 14: Scoring of the emoticons feature

Comparison
The average score using the emoticon feature is -0.2. This value is in the range of the results of the first data set. The results show that this feature may be an indicator for (less) qualitative tweets. One of those tweets that has an emoticon is the following:

Ich mach auch mal einen #aufschrei für alle #NBA Fans. #Iverson nicht in die D-Leage ? :( Wäre ja die Krönung, wie bei #Ailton die Oberliga.

The tweet is only about sports and has nothing to do with the topic. There are also some tweets that are addressing the topic but are transporting a subjective opinion and are not providing information or useful content like the following one, where the first part of the tweet is a citation and is followed by a subjective view:

"Die Sache mit der Selbstentscheidung der Frau ist ja vielschichtig." #aufschrei #jauchdas wird ein spaß morgen in den medien :)

5.2.6 Repeated Characters
As mentioned in the Repeated Characters and Exclamation and Question Marks sections the usage of multiple characters or punctuation elements (e.g. question marks) in a row are an evidence for the low quality of a text. The configuration of this feature is except for the xml-tag name completely identical to the configuration in the prior section:

```xml
<RepeatedCharacters value="3" weight="−1.0" activated="true"/>
```

Below is an exemplary tweet that gets penalized by the configuration:
In table 15 it is shown that the info is inferior to the other categories. At first this is unexpected, but if a closer look is taken at the corpora the problem of this category is revealed:

#Hochwasser #SCHÖNA @ #ELBE –&gt; #Pegel in 9 Minuten von 979 cm um 5 cm auf 974 cm gefallen. [Stand: 07.06.2013 18:30]

The problem is that nearly all tweets in the info category use an arrow that is written ”—>” (or html encoded —>). So the three dashes that are used for an arrow are causing the problem for this category. In a second step it was evaluated how the scores change if a tweet is penalized when it has characters that are repeated 4 times. In table 16 the results of this modification are shown. The info category now scores best with no penalization, but also the average values of the other categories are going down to a minimum where they can not be distinguished.

<table>
<thead>
<tr>
<th>Category</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>-0.16</td>
</tr>
<tr>
<td>private</td>
<td>-0.11</td>
</tr>
<tr>
<td>video</td>
<td>-0.17</td>
</tr>
<tr>
<td>info</td>
<td>-0.81</td>
</tr>
</tbody>
</table>

Table 15: Scoring of the repeated characters feature

<table>
<thead>
<tr>
<th>Category</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>-0.01</td>
</tr>
<tr>
<td>private</td>
<td>-0.02</td>
</tr>
<tr>
<td>video</td>
<td>-0.01</td>
</tr>
<tr>
<td>info</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Table 16: Scoring of the repeated characters feature with adapted configuration

Comparison

For the repeated characters feature the average value in the comparison data set is -0.11, which is significantly higher than the values of the other data set that are shown in
Most of the tweets that scored negatively in the comparison set had repeated punctuation characters like dots or question marks. However also here the less qualitative tweets got the penalization:

Über dumme Sprüche lachen oder nen blöden Spruch auf selben Niveau als Konter und ihr könntet zu 80% eure Probleme abhaken... #Aufschrei

The tweet has a very good grammar but it is lacking useful information and is subjective.

### 5.2.7 Capitalized Letters

Capitalized letters are often used for putting emphasis on a term, but not so often for transporting information. The problem with capitalized letters is that German sentences and also nouns start with them. A problem that more languages are facing is that abbreviations of organizations are written uppercase. The configuration for the capitalized letters feature looks like this:

```
<CapitalizedLetters weight="1.0" activated="true">
  <Count value="15" weight="1.0"/> <!-- up to 15% capitalized letters get 1 point -->
  <Count value="33" weight="0.3"/> <!-- up to 33% capitalized letters get 0.3 points -->
  <Count value="*" weight="-1"/>  <!-- over 33% capitalized letters get -1 point -->
</CapitalizedLetters>
```

Here is one of those tweets that achieve the highest score in this section:

719,00cm 19:00 #magdeburg #hochwasser

The tweet does not have any uppercase letter, but it is rather short. For comparison, here is a tweet with a negative score:

#Hochwasser #EISENHUETTENSTADT SCHL. UP @
#SPREE-ODER-WASSERSTRASSE —&gt; #Pegel in 0 Stunden von 521 cm um 10 cm auf 511 cm gefallen. [St

The main problem for this tweet is that also the hashtags are all written in uppercase. They provide useful information, because in this particular case the uppercase words are locations, but the configuration has to be made for all tweets in the data set. This
problem can also be seen in table 17 where the info category has a lower score than the other categories, which is because of the uppercase written hashtags. The other three categories are very similar and from the results it can be stated that this feature is not very expressive when it comes to determining the quality of a tweet.

<table>
<thead>
<tr>
<th>Category</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>0.95</td>
</tr>
<tr>
<td>private</td>
<td>0.95</td>
</tr>
<tr>
<td>video</td>
<td>0.91</td>
</tr>
<tr>
<td>info</td>
<td>0.72</td>
</tr>
</tbody>
</table>

Table 17: Scoring of the capitalized letters feature

Comparison
In the #aufschrei data set it is the same where tweets that do not provide any useful insight to a topic often have capitalized letters to emphasize the opinion of the user. This is demonstrated with the following tweet:

DIE MÄNNER SIND OPFER - so generell im durchschnitt. UND JETZT GLEICH BLAMING THE VICTIM!!!!!! AINEDTER! AHHHH!!!!!!! #zib2 #aufschrei

Here the message is an opinion of the user and does not mention further information. The average score for this feature is 0.97 for the whole data set which is above all the categories of the floods data set. This is probably the case, because the tweets were not separated into multiple categories and only a small number of tweets have multiple capitalized letters.

5.2.8 Stop-word Ratio

Like it was mentioned in the section Stop-word Ratio a text with only few stop-words is more likely to be irrelevant and therefore the configuration has to be organized in steps:

```xml
<StopWords weight="1.0" activated="true">
  <Ratio value="5" weight="0"/> <!-- ratio of up to 5% does not get a point -->
  <Ratio value="25" weight="1.0"/> <!-- up to 25% 1 point -->
  <Ratio value="40" weight="0.3"/> <!-- up to 40% 0.3 points -->
  <Ratio value="100" weight="0"/> <!-- up to 100% again 0 points -->
</StopWords>
```
The following tweet has a stop-word ratio of 10% and therefore hits a score of 1 point:

Pegel Strombrücke um 19 Uhr: 720 cm #Elbe #Hochwasser #Magdeburg

Whereas the following tweet does not get any point, because it does not have any stop-word:

RT @Intakt24: Remko ETF 320 Bautrockner Luftentfeuchter @Amazon #Hochwasser http://t.co/HIfWWp0NDK http://t.co/b29BXql0QN

In the table below it is depicted that for this feature the info category scores best of all the categories. The other and the private category have a quite equal score.

<table>
<thead>
<tr>
<th>Category</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>0.38</td>
</tr>
<tr>
<td>private</td>
<td>0.28</td>
</tr>
<tr>
<td>video</td>
<td>0.60</td>
</tr>
<tr>
<td>info</td>
<td>0.85</td>
</tr>
</tbody>
</table>

Table 18: Scoring of the stop-word ratio feature

Comparison

For the stop words feature the average score of 0.19 is lower than each score of the floods data set. This demonstrates that the stop words ratio is not a meaningful measurement in the Twitter universe where the texts are rather short, because often messages are kept short and are written with incorrect grammar. The following tweet scored with 1.0 in this feature and does not provide any information:

' #aufschrei ' HALT DIE FRESSE!!!

5.2.9 PoS

The formula for the calculation of this value was already mentioned in the section PoS. The configuration for this feature is very simple, because it only has to be decided if the feature should be used or not:
According to the formula higher scores are standing for a better quality. The highest score (14.5) achieved the following tweet:

```
#Hochwasser in #Aken: Starker Verkehr auf der B187a zwischen Aken und Köthen infolge der Evakuierung von Aken und der umliegenden Orte
```

From the tags that are relevant for the calculation the tweet has the following numbers:

<table>
<thead>
<tr>
<th>Part-of-speech</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>nouns</td>
<td>9</td>
</tr>
<tr>
<td>adjectives</td>
<td>2</td>
</tr>
<tr>
<td>prepositions</td>
<td>5</td>
</tr>
<tr>
<td>articles</td>
<td>3</td>
</tr>
<tr>
<td>pronouns</td>
<td>0</td>
</tr>
<tr>
<td>verbs</td>
<td>0</td>
</tr>
<tr>
<td>adverbs</td>
<td>0</td>
</tr>
<tr>
<td>interjections</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 19: Part-of-speech amounts in tweet #1

One of four tweets that scored with the lowest rating of -1.0 is the following:

```
Hat man eigentlich schon einmal nachgeforscht warum es dieses Hochwasser so wie es jetzt ist gibt?
```

The tags for this tweet are split up in the following manner:

<table>
<thead>
<tr>
<th>Part-of-speech</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>nouns</td>
<td>1</td>
</tr>
<tr>
<td>adjectives</td>
<td>1</td>
</tr>
<tr>
<td>prepositions</td>
<td>0</td>
</tr>
<tr>
<td>articles</td>
<td>0</td>
</tr>
<tr>
<td>pronouns</td>
<td>6</td>
</tr>
<tr>
<td>verbs</td>
<td>3</td>
</tr>
<tr>
<td>adverbs</td>
<td>5</td>
</tr>
<tr>
<td>interjections</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 20: Part-of-speech amounts in tweet #2
The category info is performing better than the other three categories and the category private has the lowest average score (table 21).

<table>
<thead>
<tr>
<th>Category</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>7.43</td>
</tr>
<tr>
<td>private</td>
<td>6.72</td>
</tr>
<tr>
<td>video</td>
<td>7.65</td>
</tr>
<tr>
<td>info</td>
<td>9.29</td>
</tr>
</tbody>
</table>

Table 21: Scoring of the PoS feature

Comparison

A tweet achieves higher points in the part-of-speech feature when it is well written. The average score of 6.13 shows that the tweets in the comparing data set are worse written than in the flood data set. The reason for this circumstance could be that more personal and subjective messages exist in the #aufschrei data set. When a closer look is taken on the results it is revealed that there are tweets that are scoring well and provide information:

#aufschrei: HörPunkt zum Thema «Mann Macht Frau» am Samstag ab 10 Uhr live aus dem Museum der Kulturen in Basel: http://t.co/IESv24LT

In this particular case it is mentioned that there is a live show taking place that is about the topic. However there are also tweets that are well written and do not provide useful information like the following one:

von wegen #Aufschrei ... Klassiker immernoch am praktikabelsten: Das Weib mit der Keule k.o. flirten und in die Höhle schleifen #Urschrei

Nevertheless there are more qualitative tweets that are scoring good in this section than others, which means that the part-of-speech tagging feature is a good indicator for qualitative tweets.

5.2.10 Overall Textual Features

In this section all features are combined and evaluated with the configurations that are mentioned in the previous sections. In the following table 22 the second column shows the
score over all features of the Textual Features section. With all features enabled the info category has the highest score. In the third column the following features were removed from the calculation:

- Repeated Characters
- Capitalized Letters

Those two features are removed, because they are not significant enough or are representing a false result in the domain of the thesis where qualitative and informative tweets should be found. A new calculation with the removed features shows that the info category achieves a higher score. All categories improved their score through this adaption, but the most important category had the highest increase.

<table>
<thead>
<tr>
<th>Category</th>
<th>Score (all features)</th>
<th>Score (adapted configuration)</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>1.19</td>
<td>1.42</td>
<td>+0.23</td>
</tr>
<tr>
<td>private</td>
<td>1.11</td>
<td>1.30</td>
<td>+0.19</td>
</tr>
<tr>
<td>video</td>
<td>1.21</td>
<td>1.46</td>
<td>+0.25</td>
</tr>
<tr>
<td>info</td>
<td>1.58</td>
<td>2.05</td>
<td>+0.47</td>
</tr>
</tbody>
</table>

Table 22: Overall scoring of textual features

The highest score (2.5) was achieved by the following tweet that belongs to the other category:

#Hochwasser in #Aken: Starker Verkehr auf der B187a zwischen Aken und Köthen infolge der Evakuierung von Aken und der umliegenden Orte

The best info tweet is on rank 5 and has a score of 2.34 points:

#Hochwasser #Boizenburg #Pegel #Elbe plus 80 cm innerhalb der letzten 12 Stunden - aktuell 11 Uhr: 591 cm

The tweet with the lowest score of 0.24 belongs to the category other:

@SilSte halle lohnt nicht. heute ist auch schon kaum noch was. wir fahren morgen vermutlich nach dessau #hochwasser
Afterwards a run on the data was made without using the domain related terms feature. Here the whole result set changed, since the part-of-speech feature seems now to outperform all other categories. This means that now more well-written tweets are listed as the best ones and those with useful information were kicked out of the top 50. The average overall score changed to the following values shown in table 23.

<table>
<thead>
<tr>
<th>Category</th>
<th>Score (adapted configuration)</th>
<th>Score (adapted wo. important terms)</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>1.42</td>
<td>1.66</td>
<td>+0.24</td>
</tr>
<tr>
<td>private</td>
<td>1.30</td>
<td>1.52</td>
<td>+0.22</td>
</tr>
<tr>
<td>video</td>
<td>1.46</td>
<td>1.69</td>
<td>+0.23</td>
</tr>
<tr>
<td>info</td>
<td>2.05</td>
<td>2.14</td>
<td>+0.09</td>
</tr>
</tbody>
</table>

Table 23: Comparison adapted configuration vs. adapted without important terms

As one can see all values improved a little bit and also on average the info category has the highest score. However when a detailed look on the tweets is taken nearly all of the tweets in the top 50 are from the other category. Nevertheless, the difference between the best tweet and the best tweet of the info category are only 0.45 points.

**Comparison**

The overall score for the textual features (excluding domain related terms) has an average of 1.17 which is lower than the results of the flood tweets. The tweet that scored highest with 2.31 points was the following one:

Nach #Brüderle neuer #Aufschrei in der #FDP wegen @TobiasHuch dem #Steuerhinterzieher mit #Porno Firmen im Ausland über #Strohmann

This tweet does not contain further information about the topic, since it only states that a new incident occurred that also refers to the topic. Whereas the following tweet which mentions that there is going to be a radio show that discusses the topic reached a score of 2.25:

#aufschrei: HörPunkt zum Thema «Mann Macht Frau» am Samstag ab 10 Uhr live aus dem Museum der Kulturen in Basel: http://t.co/IESv24LT

On the last place is the following message that has a score of -0.06 points:
Frauen kann man’s eh nicht rechtmachen. Je nach Stimmung wollen Sie anders behandelt werden... Das wird auch immer so bleiben #aufschrei

The tweet is a personal opinion and does not contribute to the discussion in a positive way, this can also be noticed for the other tweets that have a low score. This shows that it is possible to find qualitative tweets with textual features.

5.3 Additional Content Features

5.3.1 Location

As it was already mentioned in the section Location the presence of such information is very useful during disasters and also stands for a higher quality of the tweet. In the following configuration it is assumed that tweets about the city Magdeburg and the river Elbe are more important.

```xml
<Locations weight="1.0" activated="true">
  <Location value="magdeburg" weight="1.0" />
  <Location value="elbe" weight="0.7" />
</Locations>
```

A lot of tweets in every category have one of those words in it. One tweet that has both locations and therefore also has the highest score is the following:

```
#Hochwasser #MAGDEBURG-BUCKAU @ #ELBE —> #Pegel in 59 Minuten von 742 cm um 1 cm auf 743 cm gestiegen. [Stand: 08.06.2013 18:20]
```

Furthermore the average scores of each category in table 24 show that the category info has an advantage over the other categories, because every tweet in this category provides a location information.
<table>
<thead>
<tr>
<th>Category</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>0.28</td>
</tr>
<tr>
<td>private</td>
<td>0.20</td>
</tr>
<tr>
<td>video</td>
<td>0.11</td>
</tr>
<tr>
<td>info</td>
<td>0.59</td>
</tr>
</tbody>
</table>

Table 24: Scoring of the location feature

Comparison
The locations feature was not used with the second data set since they do not relate to a place.

5.3.2 URLs
Very often tweets are linking to external content on other websites. As mentioned in the previous chapters this is often done to provide additional information to the tweet’s message. In the context of the floods corpora tweets with links to webcams and helping sites (where a user can post if help is needed or can be provided) are defined as tweets with higher quality, because those tweets also often provide a location and show the current situation there. The configuration of this feature looks like this:

```
<URLS spamPenalty="-1.0" weight="1.0" activated="true">
  <!-- URL only tweets are penalized -->
  <URL value="wetterblogger.de" weight="1.0"/>
  <!-- links to wetterblogger.de will get 1 point -->
  <URL value="webcampool.de" weight="1.0"/>
  <!-- links to webcampool.de will get 1 point -->
  <URL value="flutspenden.de" weight="0.3"/>
  <!-- links to flutspenden.de will get 0.3 point -->
  <URL value="fluddhilfe.de" weight="0.3"/>
  <!-- links to fluddhilfe.de will get 0.3 point -->
</URLS>
```

A tweet with a link to a webcam and therefore the highest score is e.g. this one:
Webcam mit Blick auf die Marienbrücke in Dresden http://t.co/1ws6ECQoCb
#hochwasser #webcam

Whereas a tweet with a linked helping site looks like this:

Hilfsangebot: Schlafplatz (Johannstadt,Sobrigau) #FluDDHilfe #Hochwasser #Dresden http://t.co/a2lGcHvukS

In table 25 it is shown that the info and the private category do not get any points and that the video category has the highest score. That the video category performs best in this feature is obvious, because the tweets provide a link to a youtube video or a webcam. The tweets in the category info do not provide any links therefore the feature is not so relevant for the quality calculation on this data set.

<table>
<thead>
<tr>
<th>Category</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>0.01</td>
</tr>
<tr>
<td>private</td>
<td>0.00</td>
</tr>
<tr>
<td>video</td>
<td>0.15</td>
</tr>
<tr>
<td>info</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Table 25: Scoring of the URLs feature

Comparison

In this case the URLs feature is used to find tweets that only consist of a link. There are exactly six tweets that only provide an URL and a hashtag. One of those tweets is the following one:

http://t.co/NjCYWWCp#Aufschrei

Since there are only six out of 28,200 tweets that are scoring the average for this feature is 0.0 points. All the links are referring to external news sites which may have additional useful information on the topic of the tweet, but since the tweet does not mention anything about the content of the link it is rated as not qualitative. However it has to be stated that if only such a small number of tweets provide an URL this feature alone is not reliable for measuring the quality of a tweet.
5.3.3 Overall Additional Content Features

In table 26 the second column again shows the scores when both features are combined for the calculation. The third column is for comparison and are the results from the table 24 because the URLs feature was removed from this calculation.

<table>
<thead>
<tr>
<th>Category</th>
<th>Score (all features)</th>
<th>Score (adapted configuration)</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>0.14</td>
<td>0.28</td>
<td>+0.14</td>
</tr>
<tr>
<td>private</td>
<td>0.10</td>
<td>0.20</td>
<td>+0.10</td>
</tr>
<tr>
<td>video</td>
<td>0.13</td>
<td>0.11</td>
<td>-0.02</td>
</tr>
<tr>
<td>info</td>
<td>0.30</td>
<td>0.59</td>
<td>+0.29</td>
</tr>
</tbody>
</table>

Table 26: Overall scoring of additional content features

The positive progress of the info category in the above table shows that the adaption of the configuration was in favor of the quality calculation. One of those tweets with the highest score of 1.7 was the following tweet from the info category:

#Hochwasser #MAGDEBURG-STROMBRÜCKE @ #ELBE —&gt; #Pegel in 1 Stunde von 723 cm um 1 cm auf 724 cm gestiegen. [Stand: 07.06.2013 23:20]

Comparison

Since only the URLs feature was used for the comparing data set it is not possible to make an overall computation with the locations feature.

5.4 Networking Features

5.4.1 Hashtag Count

A hashtag is often used to assign a tweet to a specific topic. In the literature that were mentioned in the Hashtag Count section it is shown that when hashtags are rarely used they have a positive effect on the quality of a tweet. This is not the case when a tweet does not have any hashtags. Then the tweet is rated as less qualitative, because the tweet will be harder to find by other users and is not explicitly attached to a topic. The configuration for this feature is shown below:
The following tweet got the highest possible score for this feature, because it only provides two hashtags:

RT @SchulzStep: Fluthelfer an der Strombrücke. Aktueller Elbpegel 7.19 Meter. 7.41 Meter sollen es werden. #Hochwasser #Magdeburg http://t...
Comparison

For the hashtag count feature the average score in the second data set is 0.93 which is nearly as good as the score of the info category in the flood data set. There are a lot of tweets that provide more than the five hashtags that were defined in the above configuration, those tweets do not provide useful information:

Nach #Brüderle neuer #Aufschrei in der #FDP wegen @TobiasHuch dem #Steuerhinterzieher mit #Porno Firmen im Ausland über #Strohmann

The tweet gives little information about what is currently going on in the situation but it is using too many hashtags and does not provide further links to external pages. Whereas the following tweet has only one hashtag, an informative text and a useful link to an external resource:

Sorgerecht für ledige Väter. Ein erster Schritt in die richtige Richtung gegen die Diskriminierung der Männer #aufschrei http://t.co/QZRZQJg7

This shows that the hashtag feature can be used in combination with other features for finding qualitative tweets.

5.4.2 Mention Count

It was already mentioned that the existence of user mentions in a tweet is seen very critical. In this work a user mention is seen as a negative point of a tweet. This affects the configuration in the following way that only tweets without any user mention get a point:

```xml
<MentionCount weight="1.0" activated="true"/>
```

The following tweet is acknowledged a point, because it does not provide a user mention:

#Hochwasser #SCHÖNA @ #ELBE —> #Pegel in 9 Minuten von 979 cm um 5 cm auf 974 cm gefallen. [Stand: 07.06.2013 18:30]

Whereas the tweet below has several user mentions and does not score in this category:
In table 29 the info category performs best using this configuration, since the tweets are not directed to a particular user and therefore do not have a user mention. This is also the reason why the private category has the lowest score of the four categories, because all tweets in this category have a user mention.

<table>
<thead>
<tr>
<th>Category</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>0.46</td>
</tr>
<tr>
<td>private</td>
<td>0.00</td>
</tr>
<tr>
<td>video</td>
<td>0.40</td>
</tr>
<tr>
<td>info</td>
<td>0.74</td>
</tr>
</tbody>
</table>

Table 29: Scoring of the mention count feature

Comparison

In this feature category are very often tweets penalized that are retweets, because they automatically have a user mention but there are also other tweets with a user mention that are not retweets. The average score in the second data set is 0.43 which is like the other and video category of the flood tweets. The usage of this feature for determining qualitative tweets is useful since user mentions are often answers (which are mostly subjective) or retweets (that do not provide new information).

5.4.3 Retweet

In this work retweets are seen as not so qualitative than normal tweets, because they rarely provide any new or additional information. Therefore the configuration of the feature looks like this:

<IsRetweet value="-1.0" weight="1.0" activated="true"/>

The following example is a retweet that is penalized:

RT @tagesschau: Donau-Pegelstände bei Deggendorf sinken
http://t.co/WjJYRq5VQN #Hochwasser #Bayern
As one can see in table 30 the category other has the lowest score in this section, because most of the tweets were retweeted. The best score has the private category, which shows that tweets of private conversations are rarely retweeted. The info category is in the middle of those two values.

<table>
<thead>
<tr>
<th>Category</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>-0.54</td>
</tr>
<tr>
<td>private</td>
<td>-0.01</td>
</tr>
<tr>
<td>video</td>
<td>-0.27</td>
</tr>
<tr>
<td>info</td>
<td>-0.26</td>
</tr>
</tbody>
</table>

Table 30: Scoring of the retweet feature

Comparison

The retweet feature does not necessarily help in finding qualitative tweets since it only punishes retweets that provide information that was already submitted. It is more helpful in sorting out those tweets that do not have to be analyzed in detail, because the original tweet is possibly also in the data set. Nevertheless the average score for the feature is -0.36 which means that a lot of tweets in the data set are retweets.

5.4.4 Number Of Retweets & Favorite Count

The number of retweets and favorites of a tweet could not be calculated, because a lot of tweets were already deleted by the users and therefore a reliable measurement can not be made. However the feature is implemented and can be used for future tasks.

5.4.5 Reply

In the section it was already mentioned that answers to tweets are often not very interesting and are also of lower quality than normal tweets. Therefore the configuration was set to penalize replies:

\[<\text{IsReply} \text{ value}=-1.0 \text{ weight}=1.0 \text{ activated}=\text{true} />\]

Such a penalized tweet is the following:
In this tweet the user is directly responding to another user and does not provide any useful information. In table 31 the other and info category have no negative scores and it was expected that the private category has the lowest score, because private tweets are often replies to other tweets.

<table>
<thead>
<tr>
<th>Category</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>0.00</td>
</tr>
<tr>
<td>private</td>
<td>-0.47</td>
</tr>
<tr>
<td>video</td>
<td>-0.01</td>
</tr>
<tr>
<td>info</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Table 31: Scoring of the reply feature

Comparison

In this feature category the average value for the #aufschrei data set is -0.13. This shows that only a small number of tweets are answers from one user to another. However it helps in finding tweets that are of lower quality like the following:

@martthadear @BioMicha RT hin oder her, Dein Kommentar zum Bild spricht Bände. #aufschrei

Here the users are discussing why the tweets are not retweeted by other users and no information about the topic is provided.

5.4.6 Overall Networking Features

Like in the previous sections the second column of table 32 holds the average scores that were achieved during the calculation with all features of this section. With all the features enabled the info category has the highest score. The highest score that was achieved with this configuration is 0.5 points:

#Hochwasser #WITTENBERGE @ #ELBE —&gt; #Pegel in 1 Stunde von 754 cm um 2 cm auf 756 cm gestiegen. [Stand: 09.06.2013 01:20]

The lowest score was -0.5 points for the following tweet of the other category:
Comparison

When there are all linking features activated the \#aufschrei data set has an average score of 0.22 which is nearly as good as the info category of the floods data set. The highest score that was achieved is 0.5 points by multiple tweets. One of those tweets is:

\#aufschrei: HörPunkt zum Thema «Mann Macht Frau» am Samstag ab 10 Uhr live aus dem Museum der Kulturen in Basel: http://t.co/IESv24LT

Whereas the lowest score of -0.25 is also reached by multiple tweets like the following:

RT @4zido: So Ihr habt es geschafft: Ab sofort filtere ich "Sexismus" und "\#aufschrei" - ich kann euer pseudironisches Geseier dazu nicht mehr lesen.

The results show that the networking features group can be used to find qualitative tweets, however the difference between the scores is not very high and the distinction of them will be very hard.

5.5 Cross Scoring

In this section the configuration of the features across multiple categories are applied and analyzed. At the beginning two categories are combined, afterwards are all three combined and evaluated.

5.5.1 Textual Features & Additional Content Features

Here all features of the Textual Features and Additional Content Features groups were used to calculate the scores for the tweets. As it is shown in the second column of table
the info category scored significantly better than all the other categories. However the tweet with the highest overall score of 1.68 is from the other category:

#Hochwasser in #Aken: Starker Verkehr auf der B187a zwischen Aken und Köthen infolge der Evakuierung von Aken und der umliegenden Orte

The highest score achieved by a tweet out of the info category was 1.65:

#Hochwasser #Boizenburg #Pegel #Elbe plus 80 cm innerhalb der letzten 12 Stunden - aktuell 11 Uhr: 591 cm

Furthermore it has to be stated that the user who posted the tweet with the best score does not have the most followers of the dataset. With 8757 followers the user achieves rank 88 of 5051 in the dataset. Also the user is not mentioned in the tweets of the dataset that often. The mention count lies at 114 mentions which is rank 307.

In a second run the configuration was a little bit adapted. As also supposed in the Textual Features section the features Repeated Characters and Capitalized Letters were removed. Additionally as it was done in the section Additional Content Features the URLs feature was disabled. In this additional run the average score of the info category is again the highest and also improved the most compared to the first run. Nevertheless the tweet with the highest score of 1.94 is again the tweet from above out of the other category. However in the second run another tweet scored 1.94 points. This tweet is from the info category:

#Hochwasser: Neue Prognose: Elbe-Pegel in #Magdeburg laut Wasser- und Schiffsfahrtsverwaltung des Bundes am Sonntag-Abend 7.43 Meter

Furthermore there are only two from the other category in the top 35 whereas the rest is from the info category.

<table>
<thead>
<tr>
<th>Category</th>
<th>Score (all features)</th>
<th>Score (adapted configuration)</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>1.00</td>
<td>1.14</td>
<td>+0.14</td>
</tr>
<tr>
<td>private</td>
<td>0.92</td>
<td>1.04</td>
<td>+0.12</td>
</tr>
<tr>
<td>video</td>
<td>1.02</td>
<td>1.16</td>
<td>+0.14</td>
</tr>
<tr>
<td>info</td>
<td>1.35</td>
<td>1.66</td>
<td>+0.31</td>
</tr>
</tbody>
</table>

Table 33: Overall scoring of textual features & additional content features
Comparison

When the two feature sets of textual and additional content features are combined (excluding the important terms and location features) the average score of all tweets is 1.04, which is like the lowest score (in the category *private*) of the flood tweets. The highest score that was reached is 2.06 by the following tweet:

```
Nach #Brüderle neuer #Aufschrei in der #FDP wegen @TobiasHuch dem #Steuerhinterzieher mit #Porno Firmen im Ausland über #Strohmann
```

The first tweet that is seen as qualitative and useful has a score of 2.0 and is ranked on the third place:

```
#aufschrei: HörPunkt zum Thema «Mann Macht Frau» am Samstag ab 10 Uhr live aus dem Museum der Kulturen in Basel: http://t.co/IESv24LT
```

The lowest score is -0.06 and is achieved by the tweet that is also ranked on the last place in other feature sets:

```
Frauen kann mans eh nicht rechtmachen. Je nach stimmung wollen Sie anders behandelt werden... Das wird auch immer so bleiben #aufschrei
```

The results are very similar to the overall textual features score, which is easy to explain, because the combination of textual and additional content features only adds the URLs feature to the textual features. This means that in this case the additional content features do not provide enough evidence for finding qualitative tweets.

5.5.2 Textual Features & Networking Features

The second cross scoring is done for the feature categories *Textual Features* and *Networking Features*. Also for this combination the *info* category has the highest score on average after the initial run with all features activated (see table 34). The highest score achieved was 1.58 by the tweet of the *other* category that also performed best in the previous cross scoring. Also the tweet with the highest score out of the *info* category is the same from before with a score of 1.49 points.

In the second run only the *Repeated Characters* and *Capitalized Letters* of the *Textural Features* category has been removed. Also with the modification the average score of the *info* category is higher and increased more than those of the others. However the tweet
of the other category has again the highest score with 1.77 and on second place with 1.67 points is the same tweet from the info category as in the previous cross scoring section. Textual Features & Additional Content Features. Also the ratio of the 35 highest ranked tweets stays the same.

<table>
<thead>
<tr>
<th>Category</th>
<th>Score (all features)</th>
<th>Score (adapted configuration)</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>0.87</td>
<td>0.96</td>
<td>+0.09</td>
</tr>
<tr>
<td>private</td>
<td>0.77</td>
<td>0.84</td>
<td>+0.07</td>
</tr>
<tr>
<td>video</td>
<td>0.88</td>
<td>0.98</td>
<td>+0.10</td>
</tr>
<tr>
<td>info</td>
<td>1.20</td>
<td>1.43</td>
<td>+0.23</td>
</tr>
</tbody>
</table>

Table 34: Overall scoring of textual features & networking features

Comparison

The average score for this cross calculation for the #aufschrei data set is 0.79 points which is significantly lower than the scores of the floods data set. However with this calculation now the tweet with the highest score of 1.54 is also an informative one:

#aufschrei: HörPunkt zum Thema «Mann Macht Frau» am Samstag ab 10 Uhr live aus dem Museum der Kulturen in Basel: http://t.co/IESv24LT

Now the more qualitative tweets are ranked better even when they have a lower score. This demonstrates that the linking features are useful instruments to more precisely filter out less qualitative tweets. On the last position, with a score of 0.0 points, is a retweet of a subjective opinion:

RT @vonhorst: (es gruselt mich, was da sonst noch gewesen sein könnte, an das ich mich nur nicht erinnern kann.) #aufschrei

The results also show that at the end of the ranking are mostly retweets and messages that are part of private conversations which are directly addressed to a specific user.

5.5.3 Additional Content Features & Networking Features

In the last cross scoring analysis where two feature categories are combined the Additional Content Features and Networking Features groups are used. Also in this combination the info category has the highest score on average (see table 35). The tweet with the highest score of 0.62 belongs to the info category:
A second run with a slightly modified configuration was also done in this case, where only the URLs feature was excluded. Also with this adaption the info category achieves the highest score, but the categories only increased with a small amount and the video category even had a slight decrease. The tweet with the highest score is the same like before, but now with 0.74 points.

<table>
<thead>
<tr>
<th>Category</th>
<th>Score (all features)</th>
<th>Score (adapted configuration)</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>0.15</td>
<td>0.18</td>
<td>+0.03</td>
</tr>
<tr>
<td>private</td>
<td>0.05</td>
<td>0.06</td>
<td>+0.01</td>
</tr>
<tr>
<td>video</td>
<td>0.14</td>
<td>0.13</td>
<td>-0.01</td>
</tr>
<tr>
<td>info</td>
<td>0.34</td>
<td>0.41</td>
<td>+0.07</td>
</tr>
</tbody>
</table>

Table 35: Overall scoring of additional content features & networking features

Comparison

The average value for the comparing data set is 0.14 and is within the range of the floods data set. Here again a useful tweet is positioned on the first place with a score of 0.33:

#aufschrei: HörPunkt zum Thema «Mann Macht Frau» am Samstag ab 10 Uhr live aus dem Museum der Kulturen in Basel: http://t.co/IESv24LT

A look on the result shows that with this feature combination a lot of useful tweets are ranked high. Which implies that the networking features are also very efficient for finding useful tweets, since the prior combination of textual and networking features showed very similar results. On the last places, with a score of -0.17 points, are several tweets that are all retweets or messages to other users using a user mention, e.g.:

@spektrallinie: Wo? RT: "Sieht so aus als gäbe es einen Dissens darüber, wer #aufschrei-en darf. Ich dachte da waren wir schon längst drüb..."

5.5.4 Summing Up All Feature Categories

In this last calculations all available features were used. As it is shown in table the info category scores best over all features. In the second run the configuration was again
adapted. The features *Repeated Characters*, *Capitalized Letters* and *URLs* were removed. It is shown that this adaption has a positive effect on all categories, but again the category that benefits most is *info*. Also the gap between the *info* category and the others increased to +0.45 and +0.57. Now the two tweets that were already performing quite well in the sections Textual Features & Additional Content Features and Textual Features & Networking Features are equally on the first place with a score of 1.63. However the ratio between *info* and *other* tweets in the top 35 changed, because now 7 tweets are from the *other* category. The tweet with the lowest score of 0.13 belongs to the *private* category:

@manni_best ja hochwasser hama fast keins mehr, also überschwemmt is zwar noch imma viel aber wenigstens ists nimma so arg :-)

<table>
<thead>
<tr>
<th>Category</th>
<th>Score (all features)</th>
<th>Score (adapted configuration)</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>other</td>
<td>0.78</td>
<td>0.91</td>
<td>+0.13</td>
</tr>
<tr>
<td>private</td>
<td>0.68</td>
<td>0.79</td>
<td>+0.11</td>
</tr>
<tr>
<td>video</td>
<td>0.78</td>
<td>0.90</td>
<td>+0.12</td>
</tr>
<tr>
<td>info</td>
<td>1.08</td>
<td>1.36</td>
<td>+0.28</td>
</tr>
</tbody>
</table>

Table 36: Overall scoring of all features

*Comparison*

The average score over all feature categories is 0.79, which is equal to the lowest score of the flood data set. However the results show that the tweets are ordered like in the cross scoring of textual and additional content features. This implies that the textual features are those that are the most relevant for finding qualitative tweets. Whereas the last two cross-scores showed that also the networking features can determine a qualitative tweet. Nevertheless the project shows that one configuration will probably not fit two different topics and therefore the configuration has to be adapted. However the current configuration is a good starting point to get a first look on the tweets.

### 5.6 Interpretation Of The Results

As it is shown in the section Cross Scoring the combination of all features, with a light adaption, leads to a tweet of the *info* category on the first place. However this tweet and also the other very well performing tweet of the *other* section were always top-ranked in
those cross scoring evaluations where the *Textual Features* were used. This leads to the assumption that those textual features are more useful in determining a qualitative tweet than the other features.

This can also be verified with the results of the overall ranking of the *Textual Features* that took place in section 5.2. Also there those two tweets had a very high score, the other features are influencing the ranking to a smaller degree. Furthermore it was shown that also some features are not suitable for the underlying data. This is a problem that has to be examined for each dataset on its own.
6 Conclusion and future work

In this section a short recap of the thesis is made in section Conclusion. In the following section views from other disciplines on the thesis’ topic are taken and in the last section the future work is mentioned.

6.1 Conclusion

6.1.1 General

In the first section a Scope was defined for the thesis. The goal was to implement a prototype of a framework, that calculates a score for a text message using different features. The data for testing the framework are 14,866 tweets that were saved in 2013 during floods in Germany and Austria.

In the System Documentation section first the system was documented, what was the development environment, used libraries and finally the implementation. Most of the effort went into the machine learning component, since it does a very important job for the categorization of the messages. The calculator itself only does multiple calculations on one tweet and sums up the results of these calculations. Afterwards the documentation shows how the user interacts with the system. The configuration file is described in detail since it is the most important part for the calculation of the tweets scores. After the configuration file the GUI is described by using screenshots of the system.

In the Evaluation section multiple calculations were made using the calculation framework. At first every feature in the thesis was calculated on its own. Here it showed that the features RepeatedCharacters, CapitalizedLetters and URLs were not suitable for the underlying data since most of the informational tweets had a lot of capitalized letters (hashtags were written in upper-case) and URLs were used in all categories. Afterwards calculations were made for the three feature groups Textual Features, Additional Content Features and Networking Features. Where in a first run all features of the group were used and in other runs some of the features were deactivated.

Already in the calculation of the Textual Features it was shown that on average the info category has the highest score, but when a look is taken only on the tweets and not on the category averages other tweets get slightly better scores than the best info tweet. In an additional run the Domain related terms feature was removed from the Textual Features
calculation, which resulted in higher average scores for all categories and the info category still being the one with the highest score. However the best info tweet was now even out of the top 50 tweets, which demonstrates that domain related terms are useful for finding some kind of tweets.

For the Additional Content Features group the info category was also the best. In the Networking Features group it was shown that the right adaption of the Hashtag count feature resulted in better results that will more likely show those tweets that are wanted. It was also shown that the informational tweets were not replies to other tweets and that the info category performed best in every feature group. Afterwards the feature groups were combined and multiple calculations were made. The results show that the info category had higher average scores than the other categories. However the tweet that scored highest in the calculation with all feature groups activated belonged to the other category. This relies to the fact that it has a very good score in the part-of-speech tagging feature.

6.1.2 Implementation specific

PrimeFaces

PrimeFaces is easy to use and has all the components that are needed for a standard project. However there were some problems when using it.

There was a problem with the basic file upload that is demonstrated here: http://www.primefaces.org/showcase/ui/file/upload/basic.xhtml. The problem was that the choose config... button was not rendered like all the other buttons in the configured bootstrap style. Instead it was rendered as an operating system specific button and the caption of the button was automatically replaced by the standard caption in the OS defined language. The solution was to update PrimeFaces from version 5.0 to 5.2.

GATE embedded

The integration of GATE embedded is straight forward when the user guide and examples on the project page are used. Additionally it is recommended to firstly create the wanted application in the GATE developer GUI, if the experience with GATE is not that high.
twitter4j

There are multiple libraries to access Twitter, a list is presented under the link https://dev.twitter.com/overview/api/twitter-libraries. Twitter also developed a java library but only for accessing the Streaming API. The twitter4j library enables everything to do with the Twitter API and is the only Java library that is listed on the page. However the documentation of the library is not very good. All the information for accessing the API and using the library are somewhere on the project page but it is hard to find the needed information.

For accessing the Twitter API an AccessToken is needed. To get such a token the program has to be registered under https://apps.twitter.com/ with an active Twitter user account. Even when the information is outdated on the twitter4j page it demonstrates the usage of the token. The tutorial can be found here: http://twitter4j.org/en/code-examples.html#oauth.

The twitter4j library and the Twitter API are currently not used by the calculation framework because of multiple reasons, one reason is that the API has very restricted rate limits. The API allows to make 15 requests per rate limit window (15 minute windows) and access token [25]. The search itself is rate limited to 180 queries per window [25]. This makes it cumbersome to load additional information (e.g. retweet count of a tweet) while calculating a score, because when the limit is reached the program has to wait until the new rate limit window starts.

Furthermore the Search API does not retrieve complete information. On [26] it is stated that the API is focusing on relevance and not completeness, i.e. that it is possible that users or tweets are not in the result set.

Considering the above mentioned points the use of the Twitter API for research purposes is not useful. To get useful information a crawler should be implemented that traverses the Twitter page itself. This probably leads to faster results, because the access of the web page itself is not rate limited.

6.2 Other disciplines

In this section the thesis is viewed from other perspectives. This is done because the Web Sciences master study is an interdisciplinary degree program with multiple branches.
6.2.1 Social

In terms of the social branch there is also a lot of information that can be retrieved using the framework. Some exemplary projects would be to analyze if/how the tweets between genders vary. Furthermore it could be analyzed if there are differences between the tweets of older and younger users or in which field the users are working or where they come from. Such information could be relevant for people in politics or marketing, because then they could adapt their texts to fit more to the language of the users they want to address. It could also be analyzed how the people are communicating with each other. Maybe they are using different words or another writing style when they write with different persons. In [27] some language features that are gender specific were mentioned. It stated that women use more emoticons, ellipses, repeated characters and complex punctuation on Twitter. Whereas numbers, technology and swear words are mainly used by men. Furthermore it is mentioned that women more likely write their texts in form of a diary and men are linking to external content.

In the study of [28] a more detailed look is given on the automatic guessing of a user’s gender. The accuracy of the determination when using a user’s full name is 89.1 % which seems pretty clear, however the accuracy that is reached by only using the text is also 75.5 %. At the end they combined the tweet’s text, profile description, screen name and full name of the user and were able to reach an accuracy of 92 % of determining the correct gender of the user.

Furthermore it was stated in [27] that models for predicting the gender of a text’s author achieve an accuracy of 80.5 % and outperform human raters. Also in [28] it is mentioned that the human performance is very low in these tasks, in their study they were able to label 60.4 % correctly.

6.2.2 Design

From the design the calculation framework was kept very simple as it was shown in the User documentation section. The buttons all have the same design and a distinct naming. On the index.xhtml page, where the features of the configuration can be activated/deactivated, the feature groups were separated by using a gray background. Between the groups is a white space so that the distinction is easy. Furthermore the names of the feature groups were written bold for better orientation.
This was done because of the proximity principle that is mentioned in [29], because elements that are near to each other are perceived as a group. Therefore single features that belong to a feature container are next to each other and within a grey box. The next feature group is clearly separated.

On the results.xhtml page the grouping mechanism with the gray background color was also used for the tables that show the average scores for each feature and the overall score. Here again the names of the features were written bold for an easier identification. At the end of the page is the results table that holds all tweets. This table is also in the same design of the buttons so that those two pages fit better together.

6.2.3 Business

In terms of business and economy the framework offers options for finding relevant tweets. In some of the literature that this thesis is based on, it was mentioned that it is important for companies to find opinion leaders in social networks. The reason for this interest is that a company can negotiate with this user about writing messages or tweets where e.g. a new product of the company is promoted.

Furthermore a company could analyze their own tweets and compare them to those from a competitor. Another usage example could be that advertising tweets are maybe better recognized or retweeted by users when they fulfill certain criteria. Then those tweets can again be taken and analyzed with the tool and the result will show which criteria must be met by future tweets so that they reach a higher acceptance. It has to be mentioned that the framework is not only restricted to tweets, every kind of text can be used with the tool, this means that e.g. advertising letters, advertisements in news papers or web page text can be analyzed.

In [28] it is mentioned that an accurate prediction of a user’s gender and age can be very useful for marketing. Furthermore very interesting facts were found by the work of [30]. They say that online relationship development is needed for the survival of a business, since Facebook and Twitter users also influenced major political elections around the world. From the top 100 Fortune companies at least 65 use at least one of the four (Facebook, Twitter, YouTube, LinkedIn) social networks, 54 of them two or more and 16 all four [30]. The companies also responded to the followers/fans eight times more often than they wrote their own wall posts [30].
6.2.4 Law

From the law perspective it has to be considered what someone can do with the data and that everyone can access Twitter and save tweets locally. This also means that if someone has bad intentions s/he can make use of this. Once the data is stored locally it can be easily manipulated and spread. Furthermore it is stated in the terms and conditions of the Twitter API that if a user makes a dataset with Twitter content downloadable, it is only allowed to contain the IDs of tweets or users, but not any additional information.

In Austria there were already some problems concerning postings on social network sites. There was a decision by the Austrian Supreme Court of Justice that an employee can be instantly dismissed when the employee posts an business secret \[31\]. It has to be remembered that the Internet never forgets and even when a secret is deleted after a short time the possibility is very high that it was already captured (e.g. by a bot) and stored.

In July 2015 an apprentice was instantly dismissed, because he posted a loathing message against asylum seeking people on Facebook

\[\text{http://derstandard.at/2000019660633/Porsche-entlaesst-Lehrling-nach-Hassposting-auf-Facebook}.\]

6.3 Future work

There are some issues that can be addressed in the future. One of these adapations would be to find a better formula for calculating the score. Currently the tool is simply summing up the values that are achieved for each category and a weighting has to be made in the configuration file. This leads to the fact that when all features are treated the same the part-of-speech feature is outperforming the other values, because it can achieve higher scores. Furthermore additional features could be implemented like the time when a tweet was posted, since this is also used in other papers.

The GUI could be better used to display more detailed information during the processing, now after clicking the start button everything is logged to the system console and this means that a development environment is mandatory for using the tool. Also a exporting feature for the results table should be added so that they can be saved on the local disk without re-running the whole tool with the data sets.

\[5\text{retrieved on July 15, 2015}\]
It is also possible to improve the domain related terms feature. Currently the list of terms has to be created by the user. The next step would be to use an ontology where the words do not have to be specified explicitly by the user and the messages are compared to the words in the ontology.

Another improvement would be to use more information that is provided on Twitter. This can be done in two ways. The first way is to access the REST API of Twitter. Since the API is rate limited queries have to be written very well to get data frequently, because when the limit is exceeded someone has to wait for 15 minutes to submit a new query. The second, and probably better, approach would be to get the data with a web crawler. The access of the twitter web page is not rate limited and so a crawler could start with a root page and then crawl e.g. a user profile for tweets and all additional information.
References


Johann Tuder, BSc - 1255771

