WebFeatures: A Web Tool to Extract Features from Collaborative Content

Aline C. Pinto
CEFET-MG
Belo Horizonte, Brazil
alinecristina3.14nto@gmail.com

Beatriz S. Silva
CEFET-MG
Belo Horizonte, Brazil
biasouza.inf@gmail.com

Priscilla R. M. Carmo
CEFET-MG
Belo Horizonte, Brazil
priscilla.rm.carmo@gmail.com

Raphael L. A. Lima
CEFET-MG
Belo Horizonte, Brazil
raphaluiz2017@gmail.com

Larisse S. P. Amorim
CEFET-MG
Belo Horizonte, Brazil
larisseamorim.mg@gmail.com

Rubio T. C. Viana
CEFET-MG
Belo Horizonte, Brazil
rubiotorres15@gmail.com

Daniel H. Dalip
CEFET-MG
Belo Horizonte, Brazil
hasan@cefetmg.br

Poliana A. C. Oliveira
CEFET-MG
Belo Horizonte, Brazil
polianacorrea@cefetmg.br

ABSTRACT

The production from collaborative web content has grown in recent years. Thus, exploring the quality of these data repositories has also become relevant. This work proposes to develop a tool called WebFeature. Such system allows one to manage, extract, and share quality related feature sets from text, graph and article review. To accomplish this, different types of metrics were implemented based on structure, style, and readability of the texts. In order to evaluate the WebFeature applicability, we presented a scenario with its main functionalities (creation of a feature set, extraction of features from a known dataset, and publishing the feature set). Our demonstration shows that this framework can be useful for extracting features automatically, supporting quality prediction of collaborative contents, analyzing text characterization, and improving research reproducibility.

KEYWORDS

Quality Assessment, Wikipedia, Machine Learning, Neural Networks

1 INTRODUCTION

Recently people are using the Web not just as readers of content but also as producers. Therefore, the Internet has become a big repository of information. Because of that, many researchers and the industry have grown interested in ways to analyze, predict and manage all these repositories. To accomplish this, they usually need to crawl Web pages and, after that, extract features to represent the documents in order to perform their studies or understand the collected data better.

For instance, in a social network in which having posts and their comments, it is important to rank the comments according to the importance or quality of the content. To perform this task, they can use some textual features such as the content readability [11], the written style such as use of short phrases [1] or graph representation of connection between articles [6].

Extracting quality related features from document is even more important when the document is bigger, as there is more information to extract regarding the structure (e.g. sections, paragraphs, links) of the content, their review and their the links between them. These kinds of features can be relevant in many contexts, such as predicting the quality of Wikipedia articles in which structural features have shown to be a good predictor of the content quality [6].

However, developing a tool to generate such features can be time consuming. Besides that, there are some features for which we need parameters such as how a big phrase needs to be in order to be considered a large phrase. Those parameters need to be well documented to make the reproducibility of the study easier.

Then, in this paper we present the system WebFeatures[1][2], its functionalities, and which features the system is able extract. In addition, we introduce a demonstrative scenario using this features to assess the quality of content of Wikipedia articles.

Therefore, here we propose a web system aiming at extracting features from content. To accomplish this, our system has three main functionalities: (1) to define a feature set, selecting all the features which the user intends to use, as well as choosing their parameters; (2) to upload the dataset in order to obtain the features from it; (3) to share the created feature set allowing other users to extract the same features (with the same parameters) in their datasets. Note that, during the upload, the system can receive as input HTML or text-plain format. Thus, its applicability is not restricted just to Web pages but also to other textual contents.

By using this system on the web, we hope that researchers and the industry can extract quality related features in an easier manner, also allowing them to share their feature set. Moreover, this tool allows people without a programing background to extract features

In: XIX Workshop de Ferramentas e Aplicações (WFA 2020), São Luís, Brasil. Anais Estendidos do Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia). Porto Alegre: Sociedade Brasileira de Computação, 2020.

ISSN 2596-1683

[1]https://www.youtube.com/watch?v=M70rCScft_Q
[2]https://github.com/daniel-hasan/webfeatures

from their text, so we expect that it will also be useful to people from distinct backgrounds.

This article is organized as follows: Section 2 describes the system architecture; Section 3 presents the implemented features up until this moment; Section 4 shows one possible application scenario as well as the features impact on it; and Section 5 highlights the main conclusions and future works.

2 SYSTEM ARCHITECTURE

The tool WebFeatures has been designed according to the Figure 1. This system consists in four modules: Web front-end, scheduler, database persistence and feature processing.

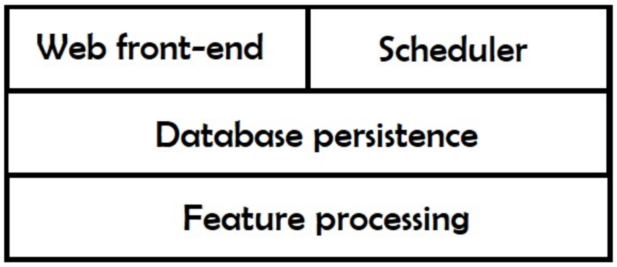


Figure 1: WebFeatures Modules

The Web front-end enables users to choose the features they want to use, to configure them (Figure 2) and to upload a dataset using a specified feature set (Figure 3).

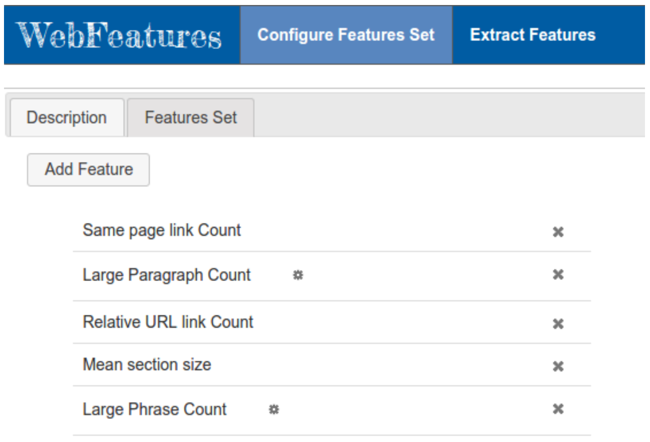
We decided to create a Web interface for this system in order to make it easier for anyone to interact with it. To accomplish this, we have studied the interfaces of some similar systems: IFeel and Weka [2, 8]. IFeel[3] is a web tool which enables the user to extract the sentiment polarity from text by using many sentiment analysis methods. Weka[4] is a data mining software which allows the user to perform machine learning and data analysis task.

The persistence module is responsible to save the uploaded datasets, the used features and its parameters. In order to implement this module, we have used the MySQL[5] database and Django Web Framework[6].

The feature processing module is where the feature computation properly happens. There, document texts are processed based on their format (HTML, text plain, graph or review history) in order to produce, the result considering all the selected features. Section 3 presents the features available in the proposed tool.

The scheduler chooses which dataset will be processed. We can have multiple schedulers running in different machines. Each scheduler always select the oldest not processed dataset. Then, in order to extract the feature set from the documents, the scheduler obtains the uploaded dataset and the feature set configuration by using the persistence and feature processing module.

[3]http://blackbird.dcc.ufmg.br:1210
[4]https://www.cs.waikato.ac.nz/ml/weka
[5]https://www.mysql.com/
[6]https://www.djangoproject.com/



Same page link Count
Large Paragraph Count
Relative URL link Count
Mean section size
Large Phrase Count

Figure 2: Feature set configuration interface. Users can add features and also configure parameters of it (e.g. in large phrase count they can select the minimum number of words which a large phrase needs to have)

3 IMPLEMENTED FEATURES

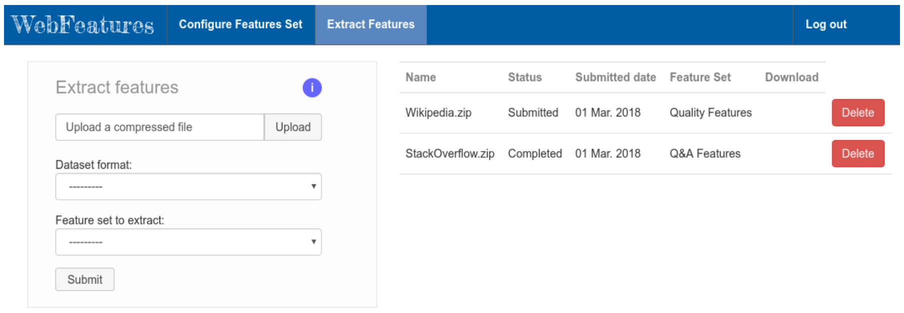
In this section, we explain the implemented features and present some examples of their applicability. We can divide the features into five groups: (1) structure; (2) style; (3) readability; (4) graph and (5) review features.

Structure features are those extracted from HTML tags. By exploring these tags, we are able to extract features such as section count, average section count, the number of images in the text, and the number of links. Style features are those which try to capture the written style through word usage. Examples of those features are the number of prepositions, pronoun count and the number of short phrases.

Readability features are metrics which tries to infer the text comprehensibility. We implemented some already proposed readability metrics such as Automated Readability Index (ARI) [12], Flesch reading ease [7] and Flesch-Kincaid [11]. The complete list of features can be found in our implemented tool when configuring the feature set. Note that, style and readability features are language dependent, because of this, we ask in which language the features are going to be used during the feature configuration.

Graph features are important metrics to find correlation between articles, for example, in-degree, out-degree and PageRank. Review features can describe reviews information about the articles, as number of reviews and reviewers and rate of reviews per time period. In collaborative digital libraries, features related to the reviewing process have been used with success to estimate the maturity level of the content. In general, a content that received many edits has likely improved over time. Table 1 describes some individual review features we have studied.

These features can be used in several domains. For example, to improve the ranking of answers in Question Answering websites [1]. Furthermore, most of these features were used to capture the content quality of collaborative documents on the web [6]. Automatically assessing the quality of content is important, for instance,



Name	Status	Submitted date	Feature Set	Download
Wikipedia.zip	Submitted	01 Mar. 2018	Quality Features	
StackOverflow.zip	Completed	01 Mar. 2018	Q&A Features	

Figure 3: Dataset upload interface.

Table 1: Examples of Review Features Implemented

Feature	Description
PercentReviewsPerDay	Diary percent of reviews of specific user related total diary reviews
ReviewCount	Total reviews of an article
AnonymousReviewCount	Number of reviews of an article done by anonymous users
RegisteredReviewCount	Number of reviews of an article done by registered users
ReviewsPerUser	Average's standart deviation reviews done by users
OccasionReview	Percent of reviews from occasional users
ReviewModSize	Percentage of modifications between the current version and a reference in the past

in collaborative encyclopedias in which the predicted quality can be used in order to indicate which documents need reviews. The quality of Wikipedia articles was also used in search query expansion task [4]. Readability features were used in order to infer influencer and detractors on Twitter [3]. Moreover, these features can be used in text characterization studies.

4 SYSTEM DEMONSTRATION

At the demonstration below are used textual features, containing style and structure features. In order to present the applicability of our tool, we introduce an example of how it can be used to predict the quality of collaborative content. To accomplish this, we here present an study on how important each proposed feature is in the quality assessment task.

By doing this, we are able to show three main functionalities of our tool: (1) creation of a feature set; (2) the feature extraction from a dataset; and (3) the ability to publish the feature set.

Thus, in order to see the impact of some features in the task of predicting the content quality, we here use the Information Gain metric. This metric is commonly used to measure the impact of features in a machine learning task [10]. It produce a relevance ranking of each feature according to its impact. To accomplish this,

Table 2: Top 10 textual features ranked with InfoGain

Ranking	Attributes
1	Flesch Reading Ease[7]
2	Relative URL link count per length
3	Complete URL link count per length
4	Same page link count per length
5	Images per length
6	Smog Grading[9]
7	Prepositions count
8	Articles count
9	Large phrase count
10	Coordinating conjunctions count

we need a dataset of documents already labeled according to its quality.

Therefore, we used a sample of 3.294 Wikipedia articles already used in [6]. The dataset is available for download[7].

This dataset contains HTML article texts from which we can extract the features and also the quality label. In Wikipedia, articles are classified, by the users, using the following classes[8]:

The classes are separated into FA, AC, GA, BC, ST, SB of which we can refer to FA as the best and SB as the worst.

The feature set was created using our tool. We also publish our feature set[9]. Through this link, people are able to see the features we used in this demonstration. Moreover, people can also create their own dataset using our configured feature set.

Then, in Table 2 we present the top 20 features in the task of predicting the quality of content. We can see in this rank features related to the structure (e.g. images per length), readability features (e.g. Flesch Reading Ease) and style (e.g. prepositions count) . As we could observe in the prediction of content quality is important to take several aspects of quality.

Structure features are important as its highlight how organized the content is. Style features tries infer a good writing style by, for example, counting the number of preposition, articles and large phrases in the text. Readability tries to infer whether the text is free of unnecessary complexities such as long phrases and words.

[7] https://github.com/daniel-hasan/wikipedia-html
[8]Note that, currently, there is also an intermediate class between ST and BC, the *C-Class* which did not exist at the time we performed the crawling (January 2008).
[9]http://www.webfeatures.com.br/p/hasan/vldb-feature-set

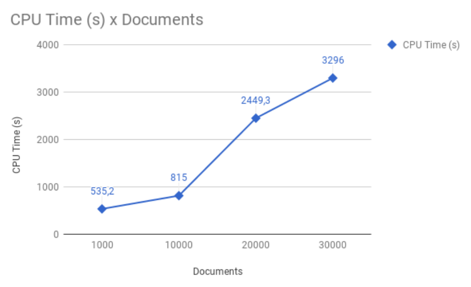


Figure 4: System performance.

4.1 System Performance

Finally, Figure 4 presents the CPU type (in seconds) when extracting all the used features from a determined number of documents. To accomplish this, we created datasets by randomly selecting (with replacement) the 3.294 documents from Wikipedia.

As we can see, our system takes from 535 seconds to 3.296 seconds to extract these features which is satisfactory performance. As a future work, we intend to improve the performance by using multi-threading as well as optimizing the code.

4.2 Performance Evaluation

- Primeiro paragrafo: Acabei escrevendo ele a grosso modo é o seguinte: Primeiro paragrafo: fale que nosso objetivo é entender o impacto das features implementadas na ferramenta no cenario de predição automatica de conteudo em documentos colaborativos online. As features implementadas da ferramenta podem ser vistas pela propria ferramenta em IREIDISPONIBILIZAROLINK. Note que, neste link, também é possível enviar um dataset para que seja extraido as features. Para entendermos melhor a performance, iremos comparar com o conjunto de features proposto em [5]. Este conjunto possui features de revisão de documentos colaborativos, grafo e textuais. Assim será possível perceber o impacto quando usa-se apenas features textuais. Para isso, foi usado o método e os resultados e disponibilizado pelos autores [6] [10].
- segundo paragrafo: fale da metrica utilizada e como é calculada (MSE)
- terceiro paragrafo: apresente o resultado (a) apresente a tabela falando o que significa cada numero linha. Useo formato da tabela igual tabela apresentada aqui como exemplo, para referenciar, use o comando do latex (ex. Tabela 3). Chame o feature set usado por [6] de "[6] feature set" e, o nosso, de "WebFeatures feature set". (b) descreva o que se viu no resultado.
- Caso fique muito dificil fazr a nova tabela, existem editores online para isso (https://www.tablesgenerator.com/).
- Conclusoes do experimento (fim do terceiro): caso o resultado seja proximo, fale que usando features com um custo computacional menor, conseguimos nos aproximar do conjunto de features proposto em [5].

[10]https://github.com/daniel-hasan/multiview-method

Table 3: Top 10 textual features ranked with InfoGain

Ranking	Attributes
1	Flesch Reading Ease Readability Feature
2	Relative URL link Count per length
3	Complete URL link Count per length
4	Same page link count per length
5	Images per length
6	Smog Grading Readability Feature
7	Prepositions Count
8	Articles Count
9	Large Phrase Count
10	Largest Phrase Count

5 CONCLUSION AND FUTURE WORK

In this paper we present WebFeature. A web based framework which extracts quality related features from the content. This software goal is to manage, extract and share feature sets. By doing this, we hope to help researchers and the industry in the feature extraction task. Furthermore, this software is a good example of how we can improve the reproducibility of research.

As presented here, this tool has a good applicability in research such as question and answering ranking, content quality prediction and text characterization analysis. We could also show a system demonstration where we demonstrated the relative importance of the extracted features in the context of automatically quality assessment of content.

REFERENCES

[1] Eugene Agichtein, Carlos Castillo, Debora Donato, Aristides Gionis, and Gilad Mishne. 2008. Finding high-quality content in social media. In *Proceedings of the 2008 international conference on web search and data mining*. ACM, 183–194.

[2] Matheus Araujo, Joao P Diniz, Lucas Bastos, Elias Soares, Miller Ferreira, Filipe Ribeiro, and Fabrıcio Benevenuto. 2016. iFeel 2.0: A Multilingual Benchmarking System for Sentence-Level Sentiment Analysis. In *Tenth International AAAI Conference on Web and Social Media*.

[3] C. Bigonha, T. N. Cardoso, M. M. Moro, V. Almeida, and Marcos André Gonçalves. 2010. Detecting Evangelists and Detractors on Twitter. In *Anais do Simpósio Brasileiro de Sistemas Multimídia e Web*. Belo Horizonte, Minas Gerais, Brazil, 107–114.

[4] Wladmir C Brandão, Rodrygo LT Santos, Nivio Ziviani, Edleno S Moura, and Altigran S Silva. 2014. Learning to Expand Queries using Entities. *Journal of the Association for Information Science and Technology* 65, 9 (2014), 1870–1883.

[5] Daniel H. Dalip, Marcos A. Gonçalves, Marco Cristo, and Pável Calado. 2011. Automatic assessment of document quality in web collaborative digital libraries. *Journal of Data and Information Quality (JDIQ)* 2, 3 (2011), 14.

[6] Daniel H. Dalip, Marcos André Gonçalves, Marco Cristo, and Pável Calado. 2017. A general multiview framework for assessing the quality of collaboratively created content on web 2.0. *Journal of the Association for Information Science and Technology* 68, 2 (2017), 286–308. https://doi.org/10.1002/asi.23650

[7] R. Flesch. 1948. A New Readability Yardstick. *Journal of Applied Psychology* (1948), 221–235.

[8] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. 2009. The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter* 11, 1 (2009), 10–18.

[9] G. Harry McLaughlin. 1969. SMOG grading: A new readability formula. *Journal of Reading* (1969), 639–646.

[10] Thomas M. Mitchell. 1997. *Machine Learning*. McGraw-Hill Higher Education.

[11] Sandy Ressler. 1993. *Perspectives on Electronic Publishing: Standards, Solutions, and More*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

[12] E. A. Smith and R. J. Senter. 1967. Automated Readability Index. *Aerospace Medical Division* (1967).