Georg-August-Universität
Göttingen
Zentrum für Informatik

ISSN 1612-6793
Nummer ZAI-MSC-2013-04

Masterarbeit

im Studiengang "Angewandte Informatik"

Analysis of Controversial Debates in Online Fora - A Showcase Analysis of the CCSVI Discussion in the DMSG Layperson Forum

Fabian Sudau

am Institut für Informatik
Gruppe Softwaretechnik für Verteilte Systeme

Bachelor- und Masterarbeiten
des Zentrums für Informatik
an der Georg-August-Universität Göttingen

26. April 2013

Georg-August-Universität Göttingen
Zentrum für Informatik

Goldschmidtstraße 7
37077 Göttingen
Germany

Tel. +49 (5 51) 39-17 20 00
Fax +49 (5 51) 39-1 44 15
Email office@informatik.uni-goettingen.de
WWW www.informatik.uni-goettingen.de

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Göttingen, den 26. April 2013

Master's Thesis

Analysis of Controversial Debates in Online Fora - A Showcase Analysis of the CCSVI Discussion in the DMSG Layperson Forum

Fabian Sudau

April 26, 2013

Supervised by Prof. Dr. Jens Grabowski
Software Engineering for Distributed Systems Group
Institute for Computer Science
Georg-August-University of Göttingen, Germany

Abstract

The nature of controversial debates in online fora is often hard to grasp due to the informal discussion style and the sheer number of contributions. Yet, important insights are buried in these openly accessible resources. We want to analyze a showcase of such a debate quantitatively in order to gain a deeper understanding of the underlying dynamics. The showcase stems from the medical field. It is about the controversial hypothesis of Chronic Cerebrospinal Venous Insufficiency (CCSVI) as a cause for Multiple Sclerosis (MS). The debate is observed in a forum provided by the Deutsche Multiple Sklerose Gesellschaft (Engl.: German MS Society) (DMSG) and targeted at laypersons. Our aim is to understand the roles of the forum users and their preferred references to sources of information better. In order to do so, we develop an Information Retrieval algorithm first, that is based on structural forum data, and is able to distinguish posts discussing CCSVI from irrelevant posts. We optimize the parameters of the algorithm by means of an Evolutionary Algorithm. We asses the referenced domains, then classify and visualize them. We identify references to scientific publications. We assign roles to users based on two distinct feature sets: One is the references posted and the other is a carefully selected feature set describing general user behavior. These roles are assigned by means of a kernelized version of the popular K-Means clustering algorithm. We also analyze the presence of homophily and determine the influence of users based on graphs known from the field of Social Network Analysis. Combining the results of these analyses, we can formulate a broad description of user behavior and relationships, community characteristics, and reference influence.

Keywords: online forum, discussion board, K-Means clustering, information retrieval, evolutionary algorithm, social network analysis, data mining, visualization

Contents

1 Introduction **1**

2 Foundations **3**
2.1 Information Retrieval . . . 3
2.1.1 Matthews Correlation Coefficient . . . 3
2.1.2 Evolutionary Algorithms . . . 4
2.1.3 K-Fold Cross Validation . . . 7
2.2 Pattern Discovery . . . 8
2.2.1 Kernelized K-Means Cluster Analysis . . . 8
2.2.2 Clustering Evaluation Metrics . . . 11
2.2.3 Metric Multidimensional Scaling . . . 13

3 Extracting the Corpus **17**
3.1 Problem: Making a Large Structured Web Resource Easy to Process . . . 17
3.2 Approach: Custom Crawling and XML . . . 18
3.3 Implementation: Choose Suitable Parser and Use a Data Access Layer . . . 20
3.4 Results: Birth of a Corpus . . . 22

4 Finding Relevant Content **23**
4.1 Problem: Examining the Corpus . . . 23
4.1.1 Finding Relevant Search Terms . . . 23
4.1.2 Examining the Partly Relevant Threads . . . 23
4.1.3 The Problem of Irrelevant Content . . . 25
4.2 Approach: A New Information Retrieval Algorithm . . . 26
4.2.1 Introducing a Forum Structure Based Algorithm . . . 26
4.2.2 Defining the Forum Structure Based Algorithm . . . 28
4.2.3 Finding Optimal Parameter Values . . . 29
4.3 Implementation: Efficient Use of an Evolutionary Algorithm . . . 30
4.4 Results: Optimal Parameters and Measures of Success . . . 32

5 Determining the Most Influential References **35**
5.1 Problem: Grasp the Information Hidden in Numerous References . . . 35
5.2 Approach: Use Visualization Methods and Exploratory Data Analysis . . . 36

5.2.1 Discarded: Cluster Referenced Web Pages . . . 36
5.2.2 Rank Domains . . . 37
5.2.3 Classify Domains Manually . . . 37
5.2.4 Cluster Users by Reference Use . . . 40
5.2.5 Find References Pointing to CCSVI Publications . . . 41
5.3 Implementation: Various Remarks . . . 42
5.3.1 Count References to Domains . . . 42
5.3.2 Load Classification and Generate Plots . . . 42
5.3.3 Kernelized K-Means with Pluggable Kernel . . . 43
5.3.4 Fetch URLs and Parse Content . . . 45
5.4 Results: Patterns and Trends in Reference Use . . . 45
5.4.1 Most Popular Domains . . . 46
5.4.2 Visual Representation of Reference Use Over Time . . . 47
5.4.3 User Patterns in Reference Use . . . 50
5.4.4 Delay in Use of Scientific Publications . . . 53

6 Describing User Behavior and Influence **55**
6.1 Problem: Describe User Behavior and Influence Based on Limited Information 55
6.2 Approach: Create Graphs and Define User Features . . . 56
6.2.1 Preface: Two Approaches to Graph Creation . . . 56
6.2.2 Assess User Communication . . . 57
6.2.3 Cluster Users by Behavior Features . . . 60
6.2.4 Compare Several User Influence Measures . . . 63
6.3 Implementation: Reuse Previous Implementation and Use Network Libraries 64
6.3.1 Create and Draw Graphs . . . 64
6.3.2 Calculate Features and Reuse K-Means . . . 64
6.3.3 Compare Measures and Rank Users . . . 65
6.4 Results: User Behavior and Influence . . . 65
6.4.1 Weak Homophily in Reference Use . . . 65
6.4.2 Six User Roles . . . 66
6.4.3 Measure Correlation and Roles of Most Influential Users . . . 70

7 Threats to Validity **73**

8 Conclusions and Outlook **75**

Abbreviations and Acronyms **77**

Bibliography **83**

1 Introduction

Online fora, also called discussion boards, are a very traditional type of medium in the social web. They are particularly well suited for topic centered discussion and information exchange. Some of these discussions are expected to be highly controversial. The aim of this thesis is to shed some light on an example of such a controversial debate from the medical sector.

The debate we want to investigate revolves around the hypothesis of Chronic Cerebrospinal Venous Insufficiency (CCSVI) as a cause for Multiple Sclerosis (MS). The hypothesis was originally proposed in [44] and suggests that obstructed venous blood flow in the neck is linked to MS. According to the hypothesis, treatment removing the obstruction could relieve symptoms of the disease. The claim is of high significance, because MS lacks other effective treatment. However, the hypothesis is fiercely debated in the scientific community. We want to examine, whether we can identify a similar debate in a community of patients. This is especially interesting due to the rise of the "expert patient". The term describes a responsible patient, who actively seeks all kinds of information from different sources (mostly through the Internet), makes intelligent use of them, and discusses them with other patients. Ideally, we can contribute findings to the question, whether this availability of a wide range of information leads to better decision-making in patients, or, on the contrary, patients are misguided by pseudoscientific sources. We expect to find such a discussion between patients in the online forum of the Deutsche Multiple Sklerose Gesellschaft (Engl.: German MS Society) (DMSG). The forum is open to anyone who wants to register. It is unstructured, unmoderated, and receives several contributions per day.

We want to use methods from the fields of Data Mining and Data Visualization in order to assess the roles of two important types of entities of the forum. The first type is references, by which we mean hyperlinks users include in their posts. They represent sources of information, that the users discuss and base their opinions on. We want to find out, what the most prominent websites are among the forum users, what types of information they provide and how the popularity of certain types of sources varies over time. We also want to know, when users posted links to scientific publications. Furthermore, we want to identify patterns in reference use, e.g. answer the questions whether different types of users prefer distinct types of references. The second entity type of interest are the users, or, strictly speaking, their virtual identities. It is of interest to analyze the behavior of the users. We want to examine, whether we can describe behavioral patterns by assigning user roles. We also want to show ways of determining the influence of individual users.

In order to achieve these goals, we structure this thesis as follows: First, we will summarize necessary foundations in Chapter 2. In Chapter 3, we show how we prepared our data by downloading the forum content from the web and converting it into a suitable persistent format. We then develop an Information Retrieval algorithm in Chapter 4, that can distinguish posts discussing CCSVI from posts discussing something else. Based on this distinction, we examine the role of references in Chapter 5. Here, we create a ranking of the most popular web domains in CCSVI discussions. We proceed to classify domains manually and show plots of the prominence of these domain classes over time. We then cluster users according to what domain classes the references they posted belong to. We identify discussed scientific publications by fetching hyperlinks and matching the retrieved content against a list of publications. In Chapter 6, we examine the roles of users and their relationships. Here, we analyze, whether users with similar reference use patterns show up in the same discussions more often than in a random scenario. We then assign roles to users by clustering them according to carefully selected behavior describing features. We conclude the user analysis with a comparison of influence measures and show, what patterns in reference use and behavior are exposed by the most influential users. In Chapter 7, we discuss threats to validity of these results and then give qualitative, summarized conclusions in Chapter 8.

2 Foundations

First of all, we want to introduce necessary foundations. These are divided into the two broad categories of Informational Retrieval and Pattern Discovery.

2.1 Information Retrieval

We develop an Information Retrieval algorithm in this work, that is able to distinguish posts discussing CCSVI from posts that discuss other topics. Thus, we first describe what constitutes a successful binary classification by introducing a measure of success in the form of the Matthews Correlation Coefficient. Because our algorithm has parameters, that need optimization, we then describe the use of Evolutionary Algorithms as a form of parameter optimization. While optimizing the parameters, we need to prevent overfitting and evaluate the obtained results, which is the reason why we show K-Fold Cross Validation in the last subsection.

2.1.1 Matthews Correlation Coefficient

When a predictive model is tested on a data set, the quality of the result must be quantified in order to obtain a conclusion not prone to subjectivity. In the case of binary classification, the Matthews Correlation Coefficient (MCC) is a measure that can provide such a quantification. It was first introduced by Matthews in [27]. The MCC's value always stems from the interval $[-1;1]$ and is regarded a binary-classification-equivalent of the popular Pearson product-moment correlation coefficient ρ. The value of the MCC can be interpreted in a similar manner as ρ[1]:

- A value of -1 means the classifier *always* predicts the opposite of what is actually true. This is an atypical outcome of a test and means, that the model can be fixed easily by inverting every prediction.
- A value of 0 means the classifier makes random predictions. Prediction and reality only match by chance, the classifier is of no worth.
- A value of 1 means the classifier predicts everything 100% correctly.

[1] ρ expresses the strength and direction of the linear relationship between two continuous variables. It is of no importance for this work, but serves as an analogy.

		Predicted Class	
		True	False
Actual Class	True	True Positive (TP)	False Negative (FN)
	False	False Positive (FP)	True Negative (TN)

Table 2.1: A confusion matrix showing all possible combinations of prediction and reality in binary classification.

In practice, values from $[0;1]$ are obtained. The closer the value is to 1, the better is the classifier. The MCC is calculated from the so called confusion matrix, which is a 2×2 matrix depicted in Table 2.1. Note that in the context of Information Retrieval, the positive class ("True") means a document is relevant with respect to a certain query. Once values for the four variables of the confusion matrix are obtained, the MCC can be calculated by:

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP) \cdot (TP + FN) \cdot (TN + FP) \cdot (TN + FN)}}.$$

The advantage of using the MCC is that it values both true positives and true negatives as equally important and is rather unaffected by biases in the sample [7].

Another well-known way to quantify the success of binary classification is the F_β-measure. It has the advantage over the MCC, that the parameter β can be used to incorporate a certain preference: The measure values recall β times more important than precision. Precision is defined by $\frac{TP}{TP+FP}$ and describes the fraction of posts classified relevant, that are relevant in reality. Recall is defined by $\frac{TP}{TP+FN}$ and describes the fraction of the posts relevant in reality, that are classified as relevant. Thus, setting $\beta = 2$ implies that missing a relevant post is regarded twice as bad as receiving a non-relevant one [30]. The F_β-measure is computed by

$$F_\beta = \frac{(1 + \beta^2) \cdot TP}{(1 + \beta^2) \cdot TP + \beta^2 \cdot FN + FP}$$

and the values of the measure stem from the interval $[0;1]$. However, a specific weighting of precision and recall must be justified.

2.1.2 Evolutionary Algorithms

An Evolutionary Algorithm (EA) is a generic, stochastic, and metaheuristic optimization algorithm, that is inspired by the biological evolution of species. Good overviews are given in [1] and [21]. Valid solutions to a given problem play the role of individuals of a population. An EA simulates the process of natural selection, which produces individuals well-adapted to their environment, in an abstract way. This means that an EA attempts to find the best solution according to a fitness function, which describes the nature of the optimization

problem at hand. In the biological scenario, unfit individuals die while fit individuals survive and reproduce. The offspring might be different due to mutation and recombination of the genes of the parents. An EA mimics this process of gradual adaptation in an iterative way. There are numerous variants of EAs. The variant, that is of interest to this work, is best described as an Evolutionary Strategy by the definition of [1]. In this subtype of an EA, a solution is a tuple of floating point numbers and self-adaptive mutation and recombination is used. To get a better understanding of both, the biological analogy and the practical use, the algorithm and the required input are described in the following paragraph.

Firstly, the definition of a valid solution has to be given. For example, a 3-tuple with floating point values $\in [0;1]$ might be a solution to a specific problem. Possible solutions are said to be candidates and stem from a search space. Because each candidate performs differently, the aim of the EA is to search the search space for the optimal candidate. That the optimal candidate is found, however, cannot be guaranteed, because EA are metaheuristic. An EA always performs the following steps, where the initialization step is performed once and the other steps are repeated (in order) until a convergence criterion is reached.

Initialization: The population is created with a specified number of candidate solutions. The candidates are typically drawn randomly from the search space.

Evaluation: The fitness value for every candidate from the population is determined. This is achieved by the use of a problem-specific fitness function. A higher value of fitness means the candidate is a better solution to the problem at hand.

Selection: The candidates good enough for further processing are selected. This could be done by selecting the top x candidates deterministically, but often a random selection mechanism is used, that assigns higher probabilities to better candidates. The purpose is to retain some diversity in the population.

Reproduction: The selected candidates reproduce by means of mutation, recombination, or both. Mutation means a new candidate is created from a selected existing candidate by changing some of the features randomly. If solutions are 3-tuples of floating point numbers, the first and third element of the tuple might be changed by adding some value to it. Recombination means that an offspring candidate is produced from two parent candidates by mixing the features of the parents. The resulting tuple might draw the first and third value from one parent and the second value from the other parent.

Replacement: Because the population can not be allowed to grow uncontrollably, some members of the population from the previous iteration have to be replaced with those resulting from the selection step *and* those from the reproduction step. The replacement step is sometimes omitted in the literature, because in the trivial case the child generation resulting from the reproduction step replaces the parent generation as a

whole. However, more sophisticated mechanisms allow the fittest parents to survive instead of unfit children in order to prevent the loss of valuable solutions.

When the convergence criterion is reached, the algorithm stops and the candidate with the highest fitness value in the population is accepted as the best solution to the problem. If there is nothing known about the expected final fitness, a fixed number of iterations can be defined as the convergence criterion.

The advantage of an EA is that it is capable of optimizing a wide range of problems without in-depth knowledge of the nature of the problem. It is sufficient to define,

1. the search space that says what a valid solution looks like (a tuple of floats, a graph, a string, etc.);
2. a fitness function, applied to solutions, that provides values judging the quality of the solutions; and
3. the operators (and the corresponding required parameters) to use for selection, reproduction and replacement.

The search space and fitness function can be provided easily, because they are part of the problem definition. Selecting the right operators and their parameters is a complex task and not well understood. There is significant cost associated with tuning parameters and selecting operators and the importance of an individual modification greatly depends on the precise composition of the EA. Often, the tuning primarily increases the efficiency of the algorithm, which means, that fewer candidates need to be tested, before a local optimum is reached. These and other difficulties in tuning an EA are discussed in detail in [28].

The generic approach of an EA makes it particularly suited for non-linear and discontinuous problems. However, EAs also have their disadvantages. They require a lot of CPU time, because many candidates have to be evaluated. The required resources depend to a large extent on the fitness function and the amount of data it operates on. Also, it can not be guaranteed, that the global optimum is reached [21].

In this work, we consider the following selection operator and two reproduction operators:

Tournament Selection depends on the parameters t and n. The selection operator selects n candidates from the population by repeating a loop n times. Each time, t candidates are drawn randomly from the population (using a uniform distribution) and then the best candidate of each these "tournaments" is selected. This ensures a performance gain (only small lists of size t need to be sorted) and includes some nondeterministic behaviour.

Gaussian Mutation depends on the parameters r, μ and σ. Given a candidate, that is a tuple of floating point numbers, this mutation operator mutates every tuple element

individually with probability r. If a tuple element must be modified, a value drawn randomly from a Gaussian distribution $G(\mu, \sigma)$ is added to the element. This ensures that a child is likely very similar to the parent and change is becoming more gradual and adaptive.

Blend Crossover depends on the parameters r and α. It produces two children from two parent candidates. With a probability of r, the children are different from the parents, otherwise they are identical copies. If children different from the parents are needed, Blend Crossover performs a step that mixes the genetic information from the parents while including some random mutation. For every tuple index i of the parents, let max_i denote the larger value at index i and min_i the minimum value. The value for each of the children at tuple index i is then drawn separately from a uniform distribution with range $[min_i - \alpha; max_i + \alpha]$.

2.1.3 K-Fold Cross Validation

When training a model in the context of Machine Learning, the performance of the trained model needs to be evaluated. A predictive model like the binary classifier, that is supposed to be developed in this work, is not evaluated on how well it describes the data it was trained on. Instead, we want to know, how good the trained model is at classifying unknown data. Thus, we must assess, how well the model generalizes from what was learned during the training. Ideally, the model learns facts, that also hold true for unknown data. In order to evaluate the ability to generalize, the model must be trained and evaluated on different data sets. A systematic approach to do so is called K-Fold Cross Validation, which is explained in [20], [2] and others. Given a data set $X = \{x_1, x_2, \ldots, x_n\}$, the data set is divided randomly into k subsets of equal size. These subsets are denoted S_i and are part of the superset $S = \{S_1, S_2, \ldots, S_k\}$. Ideally, when $|X| \bmod k = 0$, each of the subsets S_i have the cardinality $|S_i| = \frac{|X|}{k}$.[2] The algorithm then performs k iterations. In each iteration, one subsets is used for evaluation and the union of all other sets is used for training. Formally, in the ith iteration we can state:

$$\text{training\_set} = S \setminus S_i \quad \text{evaluation\_set} = S_i.$$

This way of dividing the data ensures that each data object x_i is used for both training and evaluation. Interestingly, each object is only used once for evaluation and each iteration uses a large data set for training and a small one for evaluation, which is beneficial in Machine Learning. Because every iteration is expected to result in different model parameters as well as a different evaluation measure, it is common practice to report means and standard deviations. When interpreting the statistics, low standard deviations indicate a good generalization. High variances indicate that what is learned varies greatly depending on which

[2] Otherwise, a suitable implementation must decide on how to round.

data objects are picked. This may indicate that either the data set X is too small, or the model is not appropriate for the task. When a single model is wanted as a result of the k iterations, model parameter averages can be used. The parameter k has to be chosen in advance. By convention, 10 is often used, which is also done in this work.

2.2 Pattern Discovery

Based on the distinction between relevant and irrelevant posts, we want to reveal patterns in reference use and behavior of users. Because there is nothing known in advance about these patterns, we will use a method of exploratory data analysis. We thus describe a kernelized variant of the K-Means Cluster Analysis first, that is capable of grouping arbitrary objects together based on their similarity. We then proceed to describe evaluation metrics for such clusterings that help in finding the best value of k. Finally, we want to discuss a method for visualizing objects mapped into a feature space, namely Metric Multidimensional Scaling.

2.2.1 Kernelized K-Means Cluster Analysis

Cluster analysis describes the task of grouping objects together with the intention of having similar objects in the same group and dissimilar objects across groups. It is a task of unsupervised machine learning, because the only required input are the objects themselves. Labels or any sort of human feedback are not required, which is a very useful property when dealing with large data sets. Cluster analysis is a method of exploratory data analysis: There is expected to be some hidden meaning in the data, in this case, certain groups or implicit classes of objects. A cluster analysis can now find a mapping from objects, for example users, to groups. However, further human exploration and interpretation is required. The human analyst may then find out, what all users of a group have in common and how that contributes to answering the question.

More specifically, hard clustering is of interest in this work, which means that every object is assigned to exactly one group. Several algorithms are able to complete the required task. Throughout this work, the K-Means algorithm is used, which is efficient and known for producing quite good results despite the heuristic nature of the algorithm. It was first proposed in [26]. The algorithm in the most basic form works on a set of input objects $x_1, x_2, \ldots, x_n \in X$. These objects can be any type of entity, for example users, and they are not required to fulfill any specific mathematical properties. The objects are said to stem from a domain X. In addition to the input objects, a function $\phi(x) : X \to \mathbb{R}^d$ is required, which maps input objects to a real-valued vector space. There is no limitation on the dimensionality of the vector space ($1 \leq d \leq \infty$). The objects from the input domain X are said to be *embedded* in the vector space. Embedded objects $\phi(x) \in \mathbb{R}^d$ are called *data points* in this work. The parameter k, which stands for the number of desired clusters, has

to be chosen in advance. The algorithm then uses centroids as cluster-defining entities and proceeds as described in Algorithm 1.

Initialize: Place k initial centroids $c_1, c_2, \ldots, c_k$ somewhere in the vector space;
repeat
  Assign each data point $\phi(x_i)$ to the nearest centroid c;
  Update each centroid c_i to be the centroid of all data points assigned to it;
until *convergence criterion is reached*;

Algorithm 1: The K-Means algorithm on a high level of abstraction.

The abstract steps mentioned in Algorithm 1 describe the following:

Initialize Originally, the k initial centroids were placed randomly in the feature space. This, however, left the results open to pure chance and a bad placement of the original centroids could lead to an unwanted final result. Therefore, the deterministic initialization step proposed in [11] is used in this work. The proposed alternative is of heuristic nature and is expected to produce better results on average than the random initialization. It proceeds as follows:

- The two data points with the greatest distance in between them are selected as c_1 and c_2:

$$c_1 = \phi(x_j), c_2 = \phi(x_k) : \|\phi(x_j) - \phi(x_k)\| \geq \|\phi(x_l) - \phi(x_m)\| \forall x_l, x_m \in X.$$

  If more than one pair of data points have the largest distance, it is up to the implementation to choose a pair. The euclidean distance between two points p and q in a d-dimensional space is defined as

$$\|p - q\| = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \ldots + (p_d - q_d)^2}.$$

  Note that we calculate the euclidean distance of the input objects mapped to the hyperspace.

- Every remaining centroid is placed in an iterative way one after another. When the *nth* centroid c_n needs to be placed in the vector space, the data point with the largest minimum distance to all previous centroids is chosen. If R denotes the set of every data point that has not been used as a centroid yet, the assignment of the *nth* centroid can be formulated mathematically as

$$c_n = \arg\max_{x \in R}(\min(\{\|x - c_1\|, \|x - c_2\|, \ldots, \|x - c_{n-1}\|\}))$$

$$R = \{\phi(x_1), \phi(x_2), \ldots, \phi(x_n)\} \setminus \{c_1, c_2, \ldots, c_{n-1}\}.$$

Assign Assigning a data point $\phi(x)$ to the closest centroid means finding the centroid with the minimum euclidean distance to the data point. The used distance measure is the same as the one used in the initialization step.

Update Each centroid is assigned the arithmetic mean of all the data points assigned to it. Let S_i denote the set of all data points assigned to c_i. The assignment can then be expressed as

$$c_i = \frac{1}{|S_i|} \cdot \sum_{\phi(x_i) \in S_i} \phi(x_i).$$

Convergence criterion In this case, total convergence is used as a criterion: The algorithm stops, when no data point was re-assigned to another centroid in the current iteration.

Finally, another significant improvement over the original K-Means algorithm is used in this work: The algorithm is said to be kernelized. As mentioned before, usually an embedding function $\phi : X \to \mathbb{R}^d$ is required, that maps input objects to the real-valued d-dimensional feature space. The kernel approach though makes it possible to compute the clustering in the feature space without having an explicit mapping to it. Instead, a kernel function $k : X \times X \to \mathbb{R}$ must be defined, which computes the dot product of two data points in the feature space:

$$k(x, y) = \langle \phi(x) \cdot \phi(y) \rangle.$$

The kernel function is thus said to contain an *implicit* mapping to the feature space as opposed to the *explicit* mapping of the ϕ function. This means that the coordinates of the data points in the hyperspace are not known. However, the values needed for the K-Means algorithm can be calculated from the scalar products alone. The euclidean distance of two points in the hyperspace, for example, can be calculated by

$$\begin{aligned} \|\phi(x) - \phi(y)\| &= \sqrt{\langle \phi(x) \cdot \phi(x) \rangle + \langle \phi(y) \cdot \phi(y) \rangle - 2 \cdot \langle \phi(x) \cdot \phi(y) \rangle} \\ &= \sqrt{k(x, x) + k(y, y) - 2 \cdot k(x, y)}. \end{aligned}$$

What the K-Means algorithm requires is the euclidean distance of a point $\phi(x)$ to each centroid $c_i = \frac{1}{|S_i|} \cdot \sum_{x_a \in S_i} x_a$ of a cluster, where the set S_i describes the members of the cluster. The coordinate value of the centroid can not be determined, but [38] have derived a formula, that does not require an explicit knowledge of c_i when calculating the distance to it. The formula stems from the definition of a centroid and the distance formula above and states

$$\|\phi(x) - c_i\| = \sqrt{A + B - C}$$

with

$$A = \|\phi(x)\|^2 = k(x,x)$$

$$B = \frac{1}{|S_i|^2} \cdot \sum_{x_a \in S_i} \sum_{x_b \in S_i} \langle \phi(x_a) \cdot \phi(x_b) \rangle = \frac{1}{|S_i|^2} \cdot \sum_{x_a \in S_i} \sum_{x_b \in S_i} k(x_a, x_b)$$

$$C = \frac{2}{|S_i|} \cdot \sum_{x_a \in S_i} \langle \phi(x_a) \cdot \phi(x) \rangle = \frac{2}{|S_i|} \cdot \sum_{x_a \in S_i} k(x_a, x).$$

The authors also noted that the term A is constant for each object and that the term B is constant for a cluster within one K-Means iteration.

The algorithms and formulas presented in this Section thus allow us to group a number of arbitrary objects $x_1, x_2, \ldots, x_n$ into k clusters based on the similarity of the objects. All that needs to be done is to define k and a kernel function that provides an appropriate measure of similarity between two objects and can be regarded a scalar product in a hyperspace. This provides great flexibility, because a wide range of different objects, such as users or websites, can be clustered.

2.2.2 Clustering Evaluation Metrics

We need a way to assess the quality of the output of a clustering algorithm on a specific data set. The reason is mainly, that there are different parameters to set and we want to know, which combination of parameter values produces the best result. In this work the K-Means algorithm is used and thus, the choice of the parameter k is the most important one.[3] Because k needs to be set up front, the only way to find an appropriate value is to perform repeated clusterings with different values of k, evaluate the results and chose the k which performed best. Manual analysis of the resulting clusters is highly subjective and requires a lot of effort, especially when the number of tried variants is high as well as the number of objects to cluster. Thus, we need metrics, that can be calculated from the output of the clustering algorithm and that describe the quality of the results. A detailed overview is given in [19].

These metrics fall into the category of external and internal evaluation metrics. The external metrics rely on the existence of a "ground truth". That means the grouping of the objects must be known in advance. If a set of class labels is already defined, measures can be defined that are based on the agreement between "actual" and "observed" situation. However, we want to use clustering to explore unknown data and reveal unknown patterns. Thus, the internal evaluation metrics are important, as they judge a clustering by the structure alone. This requires a definition of what a good clustering is and such a definition is subjective. A common approach is to state that a good clustering has a high intra-cluster similarity

[3] Note that the clustering algorithm itself is also open to choice as well as parameters like the used kernel function or feature normalization technique.

and a low inter-cluster similarity. Because these two properties are opposing trends, several features exist that aim to balance them. We want to show three of them. Let us define the obtained clusters as a set $S = \{S_1, S_2, \ldots, S_k\}$. Each of these clusters in turn contains a number of objects $S_i = \{x_1, x_2, \ldots\}$.

The Dunn index originally proposed in [14] is defined as the ratio of the distance of the two closest clusters to the largest cluster diameter:

$$dunn = \frac{\min\{d_{single\_linkage}(S_i, S_j) | 1 \leq i < j \leq k\}}{\max\{d_{complete\_linkage}(S_i) | 1 \leq i \leq k\}}.$$

Here $d_{single\_linkage}$ is defined as the distance of the two points from the two clusters, that are closest to each other:

$$d_{single\_linkage}(S_i, S_j) = \min_{a \in S_i, b \in S_j} \{\|\phi(a) - \phi(b)\|\}.$$

The other distance metric, $d_{complete\_linkage}(c)$, defines the cluster diameter as the distance of the two member objects furthest away from each other:

$$d_{complete\_linkage}(S_i) = \max_{a \in S_i, b \in S_i} \{\|\phi(a) - \phi(b)\|\}.$$

How the distance of two objects from another using a kernel function can be calculated, was already shown in Section 2.2.1.

Another variant of the Dunn index (named modified Dunn index in this work) is to use the $d_{average\_linkage}$ metric in both the enumerator and the denominator. The average linkage within a cluster is defined as the average distance to the centroid c_i:

$$d_{average\_linkage}(S_i) = \frac{1}{|S_i|} \cdot \sum_{a \in S_i} \|\phi(a) - c_i\|$$

with a kernelized variant discussed in Subsection 2.2.1. The average linkage of two clusters is defined as the distance of the centroids:

$$d_{average\_linkage}(S_i, S_j) = \|c_i - c_j\| = \sqrt{B_i + B_j - \frac{2}{|S_i| \cdot |S_j|} \cdot \sum_{a \in S_i, b \in S_j} k(a, b)}$$

with B as defined in Subsection 2.2.1. For both Dunn indices, a smaller value means a better clustering.

Another well known internal evaluation metric is the Davies Bouldin index originally proposed in [12]. It is defined as the average R_i value of the clusters:

$$davies\_bouldin = \frac{1}{k} \sum_{i=1}^{k} R_i.$$

The R_i value of a cluster describes, how well the cluster S_i is separated from all other clusters in the worst case. The separation of two clusters is described by the ratio of the two intra-cluster similarities to the inter-cluster similarity, thus

$$R_i = \max\{\frac{average\_linkage(S_i) + average\_linkage(S_j)}{average\_linkage(S_i, S_j)} | 1 \leq j \leq k, i \neq j\}.$$

This definition implies that a lower Davies Bouldin index value means a better clustering. In practice, these indices often behave differently depending on the separability of the cluster, the existence of outliers, and possibly other factors. Therefore, it is best to use all three indices simultaneously.

2.2.3 Metric Multidimensional Scaling

Given a set of objects $x_1, x_2, \ldots, x_n \in X$, for example users, that are implicitly mapped into a hyperspace, similarities and clusters can be calculated as mentioned before. However, to grasp the relationships among the objects intuitively, a graphical representation or visualization would be of great value. Plotting the points of the hyperspace directly though is not possible for two reasons. Firstly, the mapping is only implicit. This means, that we have a kernel function, that defines the scalar product of any two objects mapped to the hyperspace: $k(x, y) = \langle \phi(x) \cdot \phi(y) \rangle$. However, coordinates of the points in the hyperspace ($\phi(x)$) are not known. Secondly, only points of two or three dimensional space can be plotted in graphs. The hyperspace has more than three dimensions, possibly an infinite amount. Some of these dimensions can be expected to be less important than others in terms of describing the similarity between the objects of X. A solution to these problems would be to map the objects of X to a low dimensional space $\mathbb{R}^d$ with $d = 2$ or $d = 3$ while preserving the distances of the objects in the original hyperspace $\|\phi(x) - \phi(y)\|$ as well as possible. Metric Multidimensional Scaling (MDS) provides an optimal algebraic solution for the aforementioned problems. It was first introduced by Torgerson in [33], then discussed in greater detail in [34]. From a practical perspective, the following points are of interest:

- MDS is a method of projection and requires an $n \times n$ matrix $\mathbf{B}$ as input, which contains scalar products of the points in the original hyperspace. Each matrix entry b_{ij} contains the scalar product of object i and j in the original hyperspace, which can be written as $\langle \phi(x_i) \cdot \phi(x_j) \rangle$. The input matrix thus can be generated easily by populating it with the output of our kernel function.[4] Note that this layout of the scalar product matrix means it is symmetric.

[4] Note that MDS can also be used when only point distances d_{ij} are available. In fact, literature often assumes this is the case. A matrix of distances can be easily converted into a matrix of possible scalar products (the mapping is not bijective) by setting each element $b_{ij} = -0.5 \cdot {d_{ij}}^2$.

- MDS is now able to map every point $\phi(x_i)$ into a low dimensional space $\mathbb{R}^d$, where the points shall be denoted y_i. The output of MDS is an $n \times d$ matrix $\mathbf{Y}$, where each row denotes a point and each column a dimension in the target space. Because the points in the target space are constructed to yield the given scalar products of $\mathbf{B}$, the matrix $\mathbf{Y}$ must be found so that $\mathbf{B} = \mathbf{Y}\mathbf{Y}^\top$.

- The coordinates in the target space are obtained by laying out the points relative to each other. This is the case, because the scalar products only describe relationships among points and absolute locations have to be inferred. Thus, initially, a single point y_i needs to be set at the origin of the target space in order to have a starting point. This could be done arbitrarily, but a bad choice of this initial point can badly distort the obtained solution [24]. To prevent a possible distortion, the centroid of all the points y_i is set to the origin of the target space. This means that every column of $\mathbf{Y}$ must sum to zero $\sum_{i=1}^{n} y_{ij} = 0 \ \forall j$. To achieve this, the input matrix $\mathbf{B}$ has to be transformed into the matrix $\mathbf{B}^*$ first. The transformation is a form of normalization and is carried out by subtracting from every matrix element the row mean and the column mean, and than adding the overall mean of the matrix elements back.

  $$b^*_{ij} = b_{ij} - b_{i\cdot} - b_{\cdot j} + b_{\cdot\cdot}$$

  where

  $$b_{i\cdot} = \frac{1}{n}\sum_{j=1}^{n} b_{ij} \quad b_{\cdot j} = \frac{1}{n}\sum_{i=1}^{n} b_{ij} \quad b_{\cdot\cdot} = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij}$$

  The inference of the normalization step is not discussed in this work, but can be seen in [34] or [15].

- Once the normalized scalar product matrix $\mathbf{B}^*$ is obtained, it can be decomposed using Singular Value Decomposition (SVD) into

  $$\mathbf{B}^* = \mathbf{T}\mathbf{\Lambda}\mathbf{T}^\top,$$

  which works, because $\mathbf{B}^*$ is a symmetric real valued matrix. In such a decomposition, $\mathbf{\Lambda}$ is a diagonal matrix of eigenvalues of $\mathbf{B}^*$ and $\mathbf{T}$ is a matrix, where each column is a mutually orthogonal eigenvector of $\mathbf{B}^*$. Because the eigenvalues are all non-negative, we can rewrite the formula as

  $$\mathbf{B}^* = \mathbf{T}\sqrt{\mathbf{\Lambda}}\sqrt{\mathbf{\Lambda}}\mathbf{T}^\top = \mathbf{Y}\mathbf{Y}^\top \quad \text{with } \mathbf{Y} = \mathbf{T}\sqrt{\mathbf{\Lambda}}.$$

  The coordinates in the target space can thus be obtained by right multiplying the matrix of the eigenvectors with the square-rooted diagonal matrix containing eigenvalues.

- The resulting matrix $\mathbf{Y}$ is an $n \times (n-1)$ matrix, which represents the points in an $(n-1)$-dimensional space. Assuming the eigenvalues of $\mathbf{\Lambda}$ $(\lambda_1, \lambda_2, \ldots, \lambda_{n-1})$ are ranked in decreasing order, the ith column of $\mathbf{Y}$ shows the ith most important dimension. Thus, if a projection into a target space $\mathbb{R}^d$ is required, (for example with $d = 2$ in order to plot the points in a two dimensional coordinate system), the first d columns of $\mathbf{Y}$ provide the required values. The remaining columns of $\mathbf{Y}$ can be disregarded. Because the eigenvalues are ranked, it is ensured, that these dimensions are the best choice in explaining the similarities of the objects in the low dimensional space. If a measure of goodness of fit is required, the formula

$$fit = \frac{\sum_{i=1}^{d} \lambda_i}{\sum_{i=1}^{n-1} \lambda_i}$$

  represents a fraction of objects similarities / dissimilarities, that are still explained in the low dimensional space [23]. For example, $fit = 0.8$ means 80% of the information captured in the original (implicit) hyperspace is still present in the constructed low dimensional space.

3 Extracting the Corpus

The very first step is to download and extract the forum content in order to make it easily accessible for future use. This Chapter discusses the nature of the forum content and a suitable way of building a "corpus" from it.

3.1 Problem: Making a Large Structured Web Resource Easy to Process

The DMSG layman's forum[1] is unmoderated and very active. Because the original publication of Zamboni et al. [44] ranges back to 2009, forum content of about four years is of interest. Thus, a corpus of respectable size can be expected. In order to provide efficient machine-access, the forum content needs to be downloaded and transformed into a more suitable representation.

The structure of the forum can be described as flat (non-hierarchical). Wang et al. [40] have conducted extensive research on the mining of fora and also have defined some entities common in fora. In their terms, the DMSG forum has no *boards* (predefined thread categories), only *list-of-thread pages*, that present links to the available threads of discussion. Each such thread contains a number of posts presented on several consecutive *post-of-thread pages*. Each of these posts is considered the basic unit of information that is of interest. However, there is some meta-data (structural information) to consider, as shown on the screenshot in Figure 3.1. Every post is associated with an author and a timestamp. Note also the content of the post. It is not just plain text, but text with interwoven references (hyperlinks) and citations. The first post shows a link to the popular video sharing web site YouTube[2] that is embedded within the textual content of the post. The second post shows a citation of the previous one. A citation can also have a recursive structure (not shown in the screenshot): If a post cites content of another post, that does contain a citation itself, a citation-within-a-citation is obtained. This is represented by deeply nested Hypertext Markup Language (HTML) elements.

To sum this section up, forum posts have a document-like structure with recursive elements. Given that the corpus is quite large and in the form of presentation-centered HTML, a mechanism is needed that

[1] Available at `http://www.dmsg.de/multiple-sklerose-forum/index.php?w3pid=msforum`.
[2] Available at `http://www.youtube.com/`.

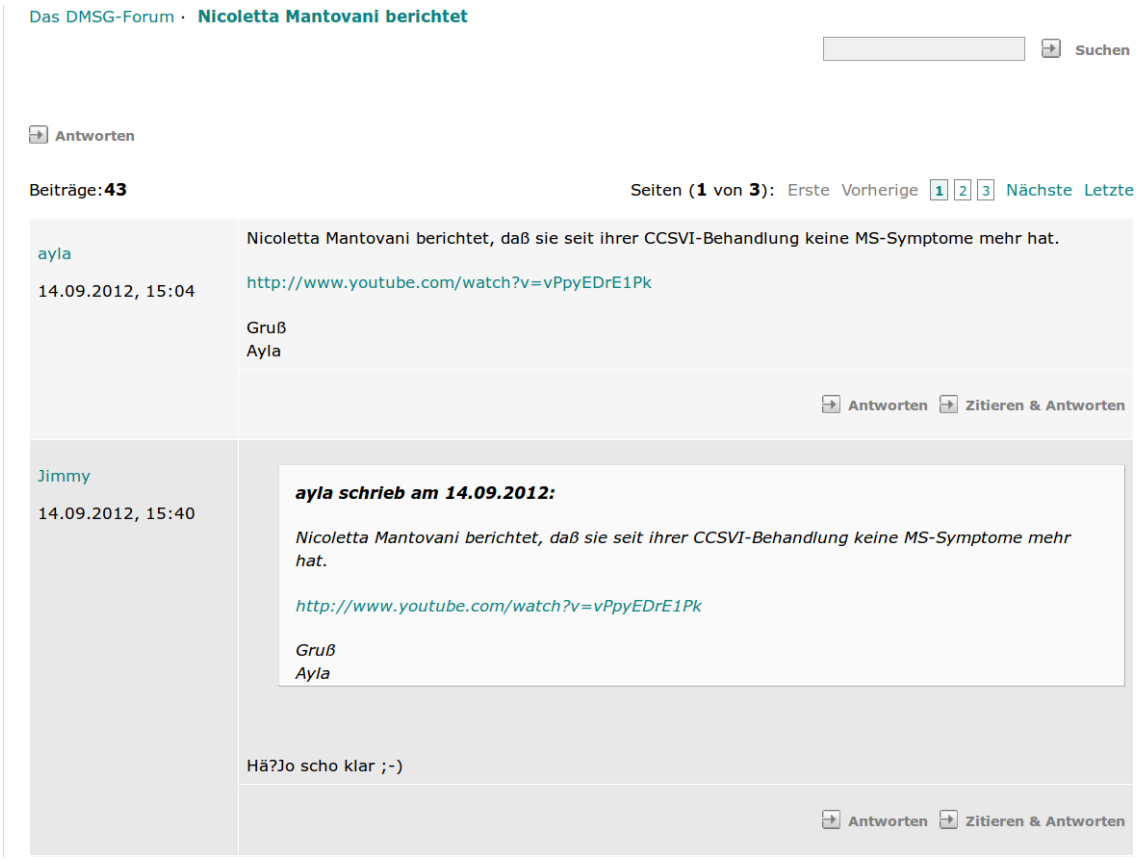


Figure 3.1: Screenshot of the top of a DMSG forum thread. The first post shows a reference to another website, the second post cites the first one.

- efficiently retrieves all relevant HTML content without missing anything,
- extracts every relevant piece of information from the HTML representation, and
- transforms the information into a suitable representation, that does not lose any of the semantics.

3.2 Approach: Custom Crawling and XML

How to find an effective crawling strategy has primarily been studied by Wang et al. [6][40]. In the referenced work, techniques of supervised and unsupervised learning are employed in order to find *skeleton links*, that point to valuable new information (that is threads), and *page-flipping links*, that can be used to traverse multiple *post-of-thread pages* in the

```
while another list-of-thread page exists do
    retrieve list-of-thread page;
    thread_list = parse thread links from list-of-thread page;
    foreach thread in thread_list do
        repeat
            visit post-of-thread page;
            parse posts from post-of-thread page;
        until no more post-of-thread pages exist;
    end
end
```

Algorithm 2: Crawling algorithm dedicated to the DMSG forum.

right order. The latter is also required when crawling the DMSG forum, as apparent in the top right corner of Figure 3.1. These approaches have the advantage of being generic, but do not always work perfectly. To overcome this limitation and because only a single forum is of interest here, a manual approach is chosen. In the approach used here, the location of *skeleton links* and *page-flipping links* is identified manually. The forum can then be traversed by simple iteration, as described in pseudocode in Algorithm 2, and no content is left out. Parsing of the content and extracting the relevant information is an issue that has also been addressed by generic approaches [43]. However, the same reasons as discussed in the crawling part motivate the use of a manual approach.

Extracted content must then be transformed into a persistent format. Because the content structure is document oriented, a document oriented Extensible Markup Language (XML) dialect seems to be the natural choice regarding the data format. Such a representation is better suited than a relational database mapping, because it can capture the interwoven text/references/citations mix of a post and the recursive structure of citations. The XML fragment shown in Listing 3.1 shows the design of the XML representation. The corpus root element contains the thread-post hierarchy, where posts keep their natural order. In the example fragment, the second post cites part of the first post. Note, that the citation node could easily contain other citations, which is a quite intuitive concept in this representation. The advantage of the XML based representation becomes clear when we think about accessing the corpus. The query "Return all posts that contain a certain substring in either post text, references, citations, or cited citations." would result in a lot of complex *join* operations in a relational environment. Using XML though, the query can be expressed quite intuitively using the XPath expression ``` ``//Post[contains(string(),'keyword')]'' ```.

```
<?xml version="1.0" encoding="utf-8"?>
<Corpus type="FlatForum" extracted="2012-09-20 18:42">
   <Thread title="Nicoletta Mantovani berichtet" id="187764">
      <Post user="ayla" timestamp="2012-09-14 15:04">
         Nicoletta Mantovani berichtet, daß sie seit ihrer CCSVI-Behandlung keine MS-
             Symptome mehr hat.

         <Reference>http://www.youtube.com/watch?v=vPpyEDrE1Pk</Reference>

         Gruß
         Ayla
      </Post>
      <Post user="Jimmy" timestamp="2012-09-14 15:40">
         <Citation user="ayla" timestamp="2012-09-14 00:00">
            Nicoletta Mantovani berichtet, daß sie seit ihrer CCSVI-Behandlung keine
                MS-Symptome mehr hat.

            <Reference>http://www.youtube.com/watch?v=vPpyEDrE1Pk</Reference>

            Gruß
            Ayla
         </Citation>
          Hä?Jo scho klar ;-)
      </Post>
      <!-- more posts here -->
    </Thread>
    <!-- more threads here -->
</Corpus>
```

Listing 3.1: The resulting XML representation of the two posts shown in Figure 3.1.

3.3 Implementation: Choose Suitable Parser and Use a Data Access Layer

When implementing such a crawler, the choice of the HTML parser to use is especially important. That is because HTML documents in practice often resemble "tag soup". The term describes HTML documents containing all kinds of errors like missing tags, improperly nested tags, missing DOCTYPE declarations, illegal characters, encoding errors and improperly escaped characters. A mature parser implementation based on heuristics and robust to these kinds of flaws is needed. The *jsoup* parser[3] implemented in Java is chosen as a suitable foundation providing such robustness, and accordingly, the programming language is chosen to be Java as well. The crawler could be implemented in other programming languages as well without disadvantages, as long as a suitable parser implementation exists[4]. That is, because:

[3] Available at `http://jsoup.org/`.

[4] For example BeautifulSoup for Python available at `http://www.crummy.com/software/BeautifulSoup/`.

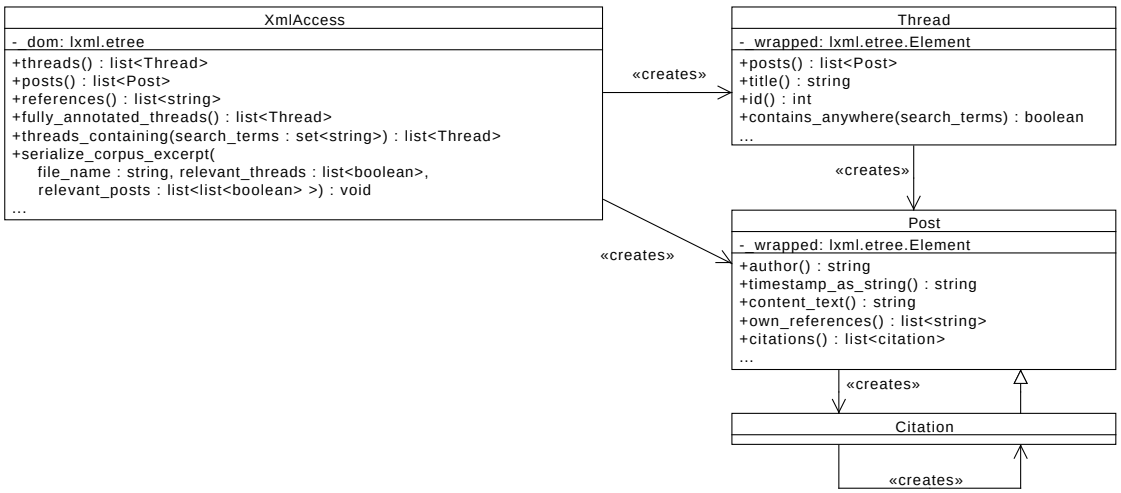


Figure 3.2: UML class diagram of the data access layer providing an abstraction from the underlying XML. Note that several methods are left out, because they are not required to understand the design principle.

1. The crawling code is in no way connected with any further analysis code. The requirements are just to extract the forum content and transform it into XML.

2. The crawling implementation is very specific to the DMSG forum and can not be reused for other fora.

The implementation itself is about finding the right HTML elements and extracting and transforming the relevant content.

The crawler generates an XML document as shown in the previous section. Subsequent analysis of the corpus (implemented in Python) requires a way to access the extracted corpus. In order to obtain an abstraction from the format the data is in (in this case XML), and in order to have a consistent, clean access to it, a data access layer is designed and implemented in Python as well. The corresponding module is named `xml_access` and a class diagram is shown in Figure 3.2. The implementation uses the fast XML parser *lxml*[5] based on native code. The objects representing threads, posts, and citations wrap around a Document Object Model (DOM) element and provide methods to access their content without showing the underlying XML.

[5]Available at `http://lxml.de/`.

3.4 Results: Birth of a Corpus

It took 5:40 hours to download and extract everything from the DMSG layperson forum, which includes a politeness delay of 100 ms between every two requests. The size of the resulting XML document is 85,2 MB. Table 3.1 shows some other volume-related statistics.

Timespan	01.01.2008 to 17.08.2012
# of threads	11.997
# of posts	139.912
# of users	13.072

Table 3.1: Size of the extracted corpus.

4 Finding Relevant Content

This work is intended to shed some light on the CCSVI-related discussion in the DMSG layperson online forum. After the extraction step, the corpus is now easily accessible, but CCSVI is by far not the only topic discussed in the forum. What is now needed is a way to differentiate relevant and irrelevant content.

4.1 Problem: Examining the Corpus

In order to illustrate the need to find a way to determine the relevance of content, this section explores the corpus by looking at CCSVI-related keywords, analyzes where they occur and what the connection to the relevance of content might be.

4.1.1 Finding Relevant Search Terms

In order to find out, which keywords are suitable for finding relevant threads, the corpus was searched. The keyword list includes German and English versions of the term CCSVI, including abbreviations and some variants in spelling. The name of the author of the original CCSVI research paper was also included. To be precise, the text content of the posts was searched in a case-insensitive way. References to external sources (hyperlinks in the original HTML) and citations (paragraphs with special markup showing the source) were regarded part of the post content and thus included in the search. Table 4.1 shows the results.

As it can be seen, just two search terms already suffice to identify most of the relevant content. The full name of the syndrome though is rarely spelled out. The terms with zero matches can be dropped in order to enhance performance while the others will be used from now on to identify relevant threads and posts. With this first criterion in mind, a lot of threads can already be classified as not relevant at all. Before a more fine-grained concept of relevant content can be developed, the threads already identified as partly relevant have to be examined.

4.1.2 Examining the Partly Relevant Threads

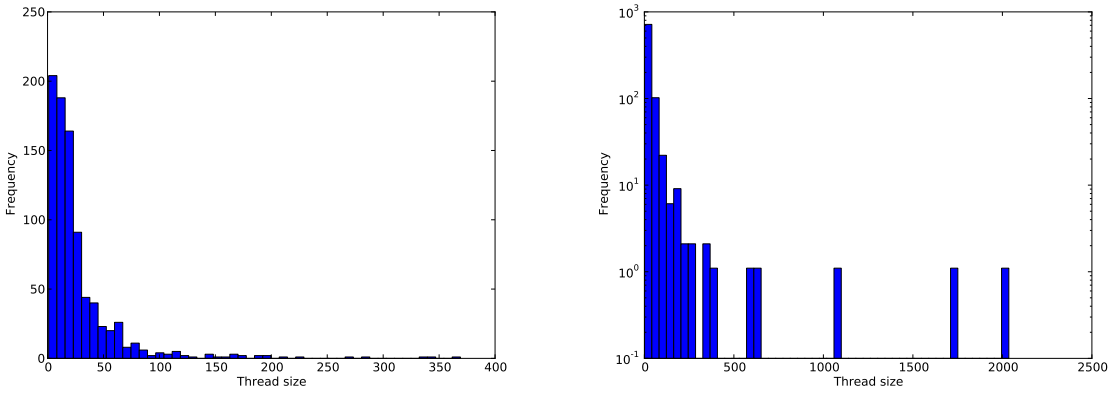
Every thread, that does contain at least one keyword (in its title, posts, references or citations) is examined in this section. The distribution of thread sizes (measured in number

Search Term	# of matching Threads
CCSVI	797
Chronische Cerebro-Spinale Venöse Insuffizienz	7
Chronische Cerebro Spinale Venöse Insuffizienz	0
Chronische Cerebrospinale Venöse Insuffizienz	10
CCVI	11
Chronic cerebrospinal venous insufficiency	33
Chronic cerebro spinal venous insufficiency	0
Chronic cerebro-spinal venous insufficiency	1
Zamboni	374

Table 4.1: Occurrences of manually defined search terms in the corpus.

of contained posts) is plotted in two different histograms, as shown in Figure 4.1. The histogram on the left suggests, that the distribution follows the power-law, which is consistent with similar observations, as, for example, in [13]. However, there are outliers in the data set. To emphasize this, a histogram with logarithmic scale is shown on the right side in Figure 4.1. Using the logarithmic scale, three bins with more than 1000 posts are visible, with a frequency of $10^0 = 1$. That means, there are exactly three threads with more than 1000 contained posts. Keeping in mind that there is no rigid mathematical definition of an outlier, two more threads with more than 500 posts could also be described as outliers. The existence of these outliers raises the question of what the structure of the threads (in general, not just the outliers) is. In order to get a better understanding of what the discussion of these presumably relevant threads was about, we take a look at the histogram in Figure 4.2. It shows the fraction of keyword-containing posts per thread. The distribution looks similar to the thread size distribution shown in Figure 4.1. The distribution indicates that the majority of the threads contains less than 20% posts with keywords. This quite low number suggests that the discussion within such a thread might not always be relevant. Figure 4.3 provides some insight into where content likely to be relevant could be located within a thread. It shows a histogram of keyword-containing posts within their containing threads. The distribution resembles a uniform distribution. Roughly speaking, this means that keyword-containing posts are equally likely to appear anywhere in a thread.

The examination of the partly relevant threads yielded two important findings. Firstly, most of the threads contain only a small number of keyword-containing posts, which raises the question, whether there might be irrelevant parts within these threads. Secondly, the position of keyword-containing posts gives no hints about possible discussion structures, as keyword-containing posts can be found anywhere with roughly equal likelihood. The next section gives an explanation of these observations.



(a) Histogram with linear scale. *(b) Histogram with logarithmic scale*

Figure 4.1: Thread size (measured in # of posts) distribution of threads containing at least one keyword.

# Posts	German Title	Original Content Description
596	Erklärung der Durchführung des Vit D ...	Supplementary Vit D3 intake
616	Es ist richtig deprimierend HIER	A joke-telling thread
1083	was ich von euch will? mucke, die euch ...	Tastes in music
1734	Vitamin D & MS - das sollte man wissen ...	Supplementary Vit D3 intake
2036	Neuer Thread	Completely arbitrary talk

Table 4.2: Intended content of the four largest threads.

4.1.3 The Problem of Irrelevant Content

It is important to point out, that the assumption "One thread discusses one topic" does not hold in practice. As already mentioned in [18] and [46], users tend to deviate from the original topic as time progresses. It is possible, that unrelated topics are discussed at different points in time within the same thread. The tone of the discussions occasionally resembles small talk, as also mentioned in [46], and topics can be mentioned casually.

To illustrate these issues, we take a look at the five threads, that have been identified as outliers in size in the previous step. Intuitively, these very large threads are likely to be related to other topics than CCSVI. Table 4.2 shows their initial topic as intended by the thread starter, identified manually. As it can be seen, none of the threads were started with the intention of discussing CCSVI. Yet, all of them contain CCSVI-keywords at some point in time, even those threads that are not about MS treatments at all. Thus, filtering out threads with no posts containing any keywords was a reasonable first step. Because keywords form the basis of more sophisticated approaches, this preliminary filtering step

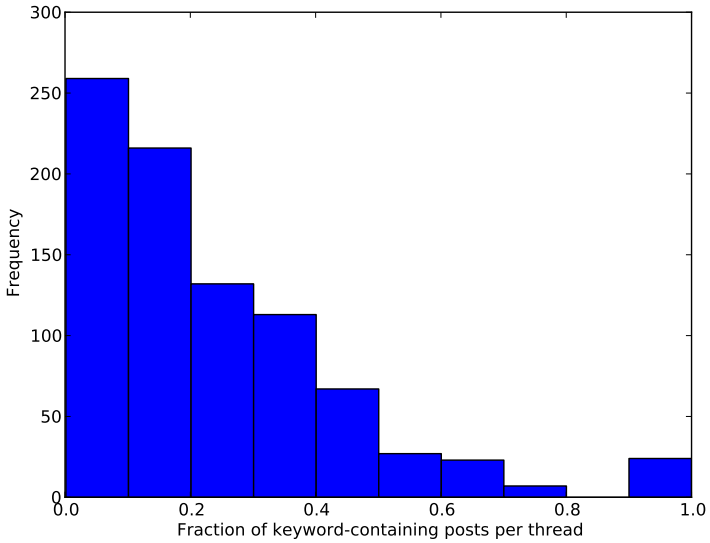


Figure 4.2: Fraction of keyword-containing posts per thread. If, for example, a thread has 4 posts and 1 of them contains a keyword, the fraction of keyword-containing posts for this thread is 0.25. This histogram shows the mentioned fraction for every thread in the corpus.

already removed completely irrelevant threads, but it is not sufficient. The corpus still contains lots of off-topic posts, which have to be removed in order to provide a meaningful data source for further analysis. As thread-membership of posts is not a good indicator, a mechanism to determine the relevance of individual posts is required.

4.2 Approach: A New Information Retrieval Algorithm

The following section discusses a solution to the problem of only partly relevant threads.

4.2.1 Introducing a Forum Structure Based Algorithm

Because threads are only partly relevant, an algorithm is needed, that classifies individual posts. This is a binary classification task in the form of Information Retrieval. The positive category is 'relevant' (meaning part of the discussion about CCSVI), the negative one is 'irrelevant' (meaning part of some other discussion). Approaches based on linguistic features alone can not be expected to perform well. This is because:

- casual / informal language is used

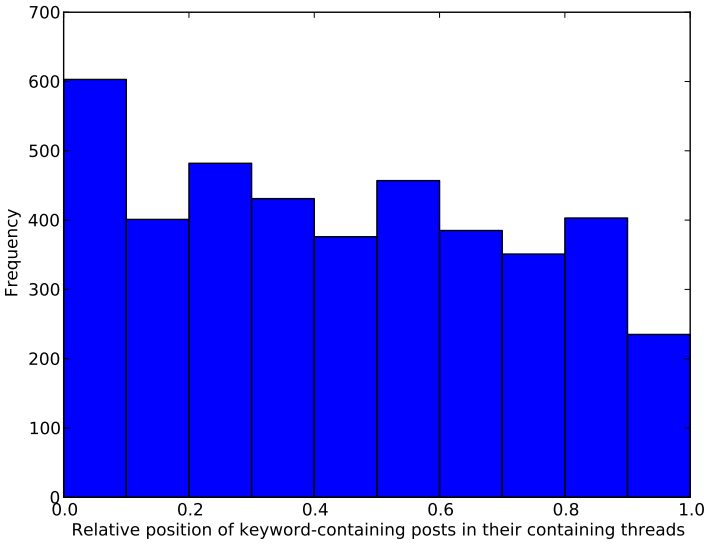


Figure 4.3: Relative position of keyword-containing posts in their containing threads. Relative position is defined by post_index/thread_length. A value of 0 means the post is the first one in the thread, a value of 1 means it is the last one.

- the language used varies from person to person
- posts can consist of only a few words
- from a the textual content of a comment alone, it is not always possible to say, what topic it refers to. For example, a generic comment like "I do not agree" can occur in different kinds of discussions.

The naive approach of classifying only keyword-containing posts as relevant is expected to result in a lot of false negatives. This is because comments / responses do not always repeat the discussed terms. To address these challenges, a new algorithm is proposed, that utilizes the structural features of a forum instead. The algorithm makes use of post content, post order, post author, and quotations. It relies on the following assumptions:

- the relevance of a post depends on context information from within the containing thread
- keyword occurrence is an indicator of relevance
- a post, that follows a relevant post, is likely to be relevant

- if a post cites or quotes other relevant posts, it is likely to be relevant
- a user, that often writes keyword-containing posts, is more likely to write relevant posts in general

Note that temporal distances between posts are assumed to be irrelevant features with respect to the classification task. If a post follows the previous post with a large time delay, it is assumed to make no difference to what the posts are about. This assumption can be disputed and we encourage future work to improve the performance of the algorithm by taking temporal distances into account. The next subsection defines the algorithm formally.

4.2.2 Defining the Forum Structure Based Algorithm

A thread is modeled as a sequence of posts denoted by $p_0, p_1, \ldots, p_n$. The relevance of an individual post p_i is determined by calculating a real-valued relevance score using function $s()$ and comparing it to a post-independent threshold t. Thus, the classification function $f()$ can be written as:

$$f(p) = \begin{cases} \text{relevant} & \text{if } s(p) \geq t \\ \text{irrelevant} & \text{if } s(p) < t \end{cases}$$

We define $s \rightarrow [0;1]$ and $t \in [0;1]$. The score function $s()$ is a bounded linear combination of the factors influencing the relevance of the post as assumed above. Every feature is represented by a function ϕ, which is weighted with a constant model parameter. We define four features which results in the following score function:

$$s(p_i) = \min(k \cdot \phi_k(p_i) + c \cdot \phi_c(p_i) + f \cdot \phi_f(p_i) + u \cdot \phi_u(p_i), 1)$$

Function ϕ_k is defined to reflect keyword occurrence in a binary way:

$$\phi_k(p_i) = \begin{cases} 1 & \text{if } p_i \text{ contains at least one keyword} \\ 0 & \text{otherwise} \end{cases}$$

Function ϕ_c reflects citations. It is the sum of all scores of the cited posts:

$$\phi_c(p_i) = \sum_{x \in C_i} s(x) \qquad C_i = \{\text{posts cited by } p_i\}$$

Function ϕ_f reflects following of the previous post:

$$\phi_f(p_i) = s(p_{i-1})$$

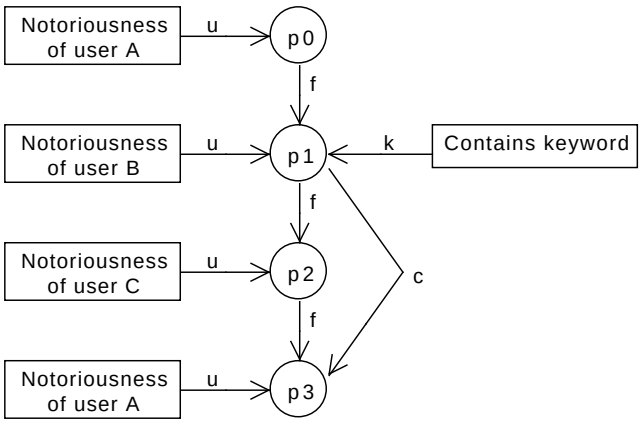


Figure 4.4: Intuitive graph-based visualization of an example thread. The circular nodes denote posts, the squared nodes denote features contributing relevance to the posts. Posts also inherit fractions of relevance scores from other posts by following them or citing them.

Function ϕ_u reflects the user behavior or "notoriousness" of a user. It is defined as the fraction of keyword-containing posts of that user, calculated from the whole corpus.[1]

$$\phi_u(p_i) = \frac{\#\text{ of keyword-containing posts of the author of } p_i}{\#\text{ of posts of the author of } p_i}$$

Figure 4.4 gives a visual example of the model and how the relevance scores are determined. Post 0 only draws relevance from the user behaviour edge. Post 1 has multiple sources of relevance, because it follows Post 0 and also contains a keyword. Post 3 cites Post 1 and thus gains additional relevance, which is proportional to the relevance score of Post 1. Thus, relevance propagates from node to node in a top-down way.

4.2.3 Finding Optimal Parameter Values

The regression model requires five parameters: The threshold t and the feature weights $k, c, f, u \in [0;1]$. Because there is no appropriate way to set those parameters a priori, a Machine Learning technique has to be employed. More specifically, in a supervised learning setting, labeled posts shall be used to train the model (that is to find appropriate values for

[1] This approach to modeling user behavior is rather simplistic and does not take into account that user behaviour might change over time. Future work may try to introduce an adaptive modeling of user behavior.

the aforementioned parameters). To do so, a sufficient number of partly relevant threads have to be randomly selected and every post has to be labeled manually as either "relevant" or "irrelevant". A relevant post in this sense is one, that (at least partially) refers to the topic of CCSVI or comments on a statement about CCSVI. Post exclusively containing topics other than CCSVI or personal attacks are regarded irrelevant.

The non-standard regression model requires a method to train it. Because relevance propagates through the graph, the relevance score of an individual posts depends on the scores of other nodes. Developing a mathematically exhaustive theory on the optimization of this model is non-trivial. Here, an alternative simplistic approach is chosen: A metaheuristic Evolutionary Algorithm as discussed in Section 2.1.2. This family of optimization algorithms has the great advantage of requiring only a very little understanding of the problem. Because it is defined what a possible solution is and how the fitness of such a solution can be determined, an Evolutionary Algorithm can be employed to optimize the parameters. We accept the disadvantage that it is only heuristic and an optimal solution can not be guaranteed.

As a measure of fitness, the Matthews Correlation Coefficient (MCC) as discussed in Section 2.1.1 is chosen. The (main) reasons are that the MCC is a common measure of success in binary classifications and that it is robust to biased samples. The proposed model does not only need to be trained, but the trained model also needs to be evaluated. Training and evaluating, however, can not be performed on the same data set, because this would result in overfitting. Therefore, the annotated data set needs to be divided into separate subsamples for training and testing. To achieve this, the well-established method of K-Fold Cross Validation (see Section 2.1.3) is employed and the parameter k is set to 10, which is a decision based on convention.

In order to justify the model complexity and prove the validity of the assumptions, the complex model needs to be compared against a baseline. To do so, two reduced models are inferred by removing some of the aforementioned features and parameters. The simplest baseline model utilizes keyword occurrence only. A more advanced baseline model utilizes keyword occurrence, citations, and post following as features, but no user notoriousness.

4.3 Implementation: Efficient Use of an Evolutionary Algorithm

The proposed algorithm, the training thereof, and all analyses, that will follow in this work, are implemented in Python. We chose the Python programming language due to its rapid development, good maintainability, excellent data manipulation capabilities (due to functional aspects and a powerful standard library), and excellent scientific libraries.

All functions and classes mentioned in this section can be found in the module `relevance_analysis`. To provide means of annotation, a console-based user interface is implemented, that randomly selects a thread from the corpus on each run and asks the user to

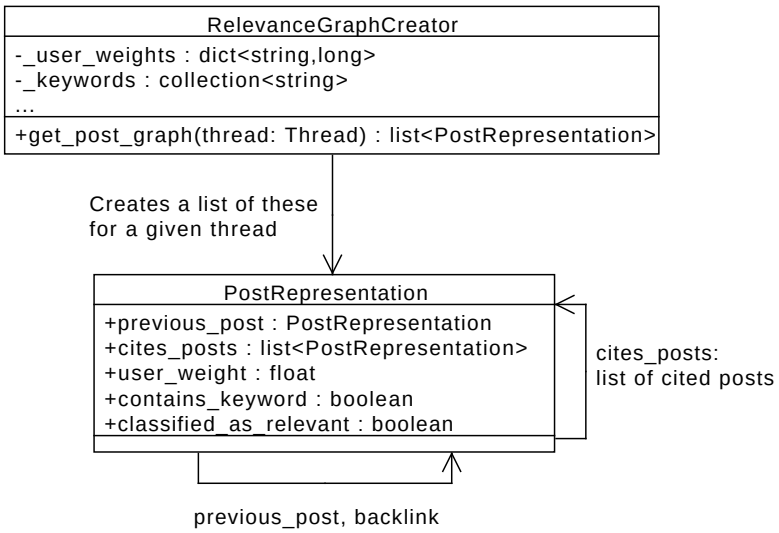


Figure 4.5: UML class diagram showing the transformation of threads into a more efficient representation with respect to the Evolutionary Algorithm.

classify each post of the thread. The classification is then stored in the XML document itself by setting the attribute `relevant='true/false'` of the `Post` element.

Because the evolutionary algorithm requires a re-computation of the relevance scores for every candidate in every round, an efficient representation of posts is required. This is achieved via a preprocessing step: The class `RelevanceGraphCreator` creates a list of `PostRepresentation` objects for every thread as seen in Figure 4.5. The created objects represent posts with respect to the evolutionary algorithm and contain every required piece of information. Because these objects maintain references to other `PostRepresentation` objects, the resulting structure is called a graph as illustrated in Section 4.2.2.

The requirements of the Approach Section are then met by a number of different classes, that share the responsibilities. An object of class `KFoldCrossValidation` (inheriting from `ValidationMethod`) prepares the data using the aforementioned `RelevanceGraphCreator` and separates it into test and training sets. It then performs multiple iterations of training and evaluation, then summarizes and averages results. The training and evaluation itself, however, are carried out by an object of class `EvolutionaryParameterOptimizer`. This is where the actual evolutionary algorithm is employed. The object uses an instance of a `BaseRelevanceStrategy` subclass to carry out the classification of posts based on `PostRepresentation` objects and model parameters. Also, a fitness function has to be provided to the `EvolutionaryParameterOptimizer` object.

To carry out the computation of the Evolutionary Algorithm, the python library inspyred[2] is chosen. The library provides an excellent API that separates problem-specific components from algorithm-specific components and thus allows for a quick adaptation to the problem at hand. The library requires the choice / definition of several functions that are derived from the steps described in Section 2.1.2:

Generator Performs the initialization. We create 100 n-tuples as candidates, where n denotes the number of parameters a concrete relevance strategy requires. Each tuple value is drawn from a uniform distribution limited by $[0; 1]$.

Selector We use Tournament Selection as described in in Section 2.1.2 with $t = 2$ and $n = 100$ to select 100 parents from a population of 100 candidates. The selection operator ensures that fitter candidates are likely to be selected multiple times for parenthood.

Variator Produces offspring from parents. We use a combination of Gaussian Mutation $r = 0.3, \mu = 0, \sigma = 1$ and Blend Crossover $r = 1, \alpha = 0.1$. This particular combination of mutation and recombination is regarded well-suited for problems with a fixed number of bounded real values by the authors of the library[3].

Replacer Determines which candidates survive into the next generation. We use Generational Replacement, which means the children replace their parents. We include weak elitism: The fittest candidate of the parent generation survives instead of the unfittest child, if the parent is fitter. The elitism ensures that a good candidate is not accidentally discarded.

Terminator We terminate after 20000 iterations. The value is chosen because it is good enough (more iterations do not improve the results). This does not exclude, however, that less iterations could be used requiring less CPU time.

The library provides reference implementations for common variants of these functions.

4.4 Results: Optimal Parameters and Measures of Success

The manual classification of randomly selected threads yielded a corpus of 51 partly relevant threads containing 1348 annotated posts. While performing the manual classification, two more common keywords ("Stent" and "Dilatation") were discovered and added to the keyword list. The extended keyword list was then used for the evolutionary training, but the original keyword list was used for the identification of partly relevant threads. This distinction is made, because an introduction of new keywords in the middle of the annotation

[2] Available at `http://inspyred.github.com`.

[3] See for example `http://inspyred.github.com/tutorial.html#lunar-explorer`.

process would have biased the annotation sample. In the following paragraphs, the results of the three models being trained and evaluated using 10-Fold Cross Validation will be discussed. To do so, mean, standard deviation, minimum, and maximum of each parameter and measure of success are shown because the concrete values vary within the 10 rounds.

The first model is called the $t-k-model$, because it has a threshold t, but the only feature of a post is keyword occurrence (attributing a value of k). The model can be described as the naive approach and is of value, because the evaluation results provide a baseline for the more complex models to be compared to. Table 4.3 shows that on average, parameter t has a value of 0.397 and parameter k has a value of 0.837. This is a plausible result of the heuristic training, because it means, that every post containing a keyword gets assigned a relevance score larger than the threshold and will thus be labeled relevant. The MCC value is not very high on average. The same holds true for the F_2-Measure, which is an alternative way of measuring the success of the binary classification.

The second model is called the $t-k-c-f-model$, because the features citations c and post following f are included additionally. The inclusion of these features raises the MCC on average by 0.099, as seen in Table 4.4. Parameter f has a value of 0.709 on average, which indicates that post following plays an important role in determining whether a post is relevant. The citation parameter c though has a much smaller value. This indicates that the citing of other posts seems to be a less important structural feature.

The third model is called the $t-k-c-f-u-model$, because it includes user notoriousness as an additional feature. Thus, all features discussed in Section 4.2.2 are included in this model. Table 4.5 shows another slight increase in MCC and F_2-Measure values. Interestingly, the user notoriousness parameter u has a value of 0.556, which shows, that some kind of repetitive behavior of users does exist and that this information can be useful in determining relevance. It also shows, that all discussed features are of value when determining relevance. Thus, the most complex model is used for finding relevant posts, with averaged parameter values from Table 4.5. By the use of the model, the partly relevant corpus containing 30116 posts can be reduced even further to a corpus of 10416 posts. The latter posts are from now on referred to as “relevant corpus parts” and are the input for most of the analyses that follow.

Variable	Value			
	Mean	Std. Deviation	Minimum	Maximum
MCC	**0.558**	0.097	0.430	0.760
F_2-Measure	**0.523**	0.077	0.390	0.682
t	0.397	0.184	0.010	0.646
k	0.837	0.188	0.432	1.000

Table 4.3: Results of the 10-Fold Cross Validation using the tk-model.

Variable	Value			
	Mean	Std. Deviation	Minimum	Maximum
MCC	**0.657**	0.120	0.492	0.860
F_2-Measure	**0.822**	0.073	0.702	0.916
t	0.417	0.082	0.237	0.561
k	0.752	0.113	0.566	0.858
c	0.089	0.041	0.017	0.148
f	0.709	0.065	0.599	0.828

Table 4.4: Results of the 10-Fold Cross Validation using the tkcf-model.

Variable	Value			
	Mean	Std. Deviation	Minimum	Maximum
MCC	**0.699**	0.102	0.578	0.893
F_2-Measure	**0.844**	0.061	0.746	0.928
t	0.435	0.114	0.152	0.591
k	0.927	0.070	0.760	1.000
c	**0.067**	0.019	0.034	0.089
f	0.612	0.092	0.358	0.693
u	**0.556**	0.128	0.395	0.860

Table 4.5: Results of the 10-Fold Cross Validation using the tkcfu-model.

5 Determining the Most Influential References

References, by which we mean hyperlinks users include in their posts, represent sources of information that the users discuss and base their opinions on. As we want to find out more about them, we begin by discussing the specific questions we aim to answer.

5.1 Problem: Grasp the Information Hidden in Numerous References

Several thousand references / hyperlinks are used in the posts classified as relevant. In order to infer any meaningful analysis from this amount of references, the information has to be compacted in a way. That is why it is useful for many analysis types to reduce URLs to their domains. For example, `www.example.com/a` and `www.example.com/b` both stem from the same domain (or "host part") `www.example.com`.

The main question is what kind of material had the most influence on the controversial discussion. Starting from that, several more specific questions can be derived:

1. Which are the most popular domains of the relevant posts and how do these compare to those in the whole corpus? How can we interpret the results?

2. What kinds of references are used? How intensively are these kinds of references used over time? Are there tendencies or time-based patterns visible? How does the use of different kinds of references in relevant posts compare to that of the whole corpus?

3. Are there preferences with respect to reference types among groups of users? If so, how can we find and describe certain types of reference users?

4. Are scientific papers cited in the layperson forum? If so, what kind of delay is evident from the time of publication to the time the paper is mentioned in the forum?

To answer these questions, different analysis types and mechanisms have to be employed. The following section discusses approaches to them.

5.2 Approach: Use Visualization Methods and Exploratory Data Analysis

In this section, we first present an attempt to quickly obtain an overview about the kinds of discussed references which relies on the clustering of referenced websites based on keywords collected from page content and metadata. This approach was ultimately unsuccessful and in the rest of this section we present four successful approaches to the four problems stated in the previous section.

5.2.1 Discarded: Cluster Referenced Web Pages

To get an overview about what kind of references play the most important roles, a good start would be to to group referenced web pages by similarity. Doing this automatically by the use of a clustering algorithm (see Section 2.2.1) could result in some first insights on the "kinds" of references. Found clusters could be described by a human annotator by looking at a few examples of each cluster. However, a clustering algorithm relies on descriptive features as input and finding relevant features of a web page is difficult, because there is lots of irrelevant navigational content in them and also non-textual parts, dynamic content and so on. It became evident, that using the text visible on the page as the feature source by applying tokenization does not provide good descriptive results. Therefore, an attempt was made to utilize features in the form of keywords derived from the following sources:

1. The commercial Alchemy API[1], that provides a web service interface able to provide some keywords for a given URL. The exact algorithm remains the secret of the company.

2. The HTML `<meta name='keywords'>` element, which is supposed to contain keywords from the web page creator, but in practice is often left out.

When keywords were used as features, a keywords kernel was employed that returned the number of shared keywords between two web pages. If two web pages are each represented by a set of keywords, the keywords kernel of two such sets, x and y, is defined by $k(x,y) = |\{x \cap y\}|$. This means the web pages are implicitly mapped into a hyperspace, where each keyword represents a dimension. The vectors of the points in the hyperspace then only hold 1s and 0s depending on whether the keyword is contained or not.

Several clustering attempts were made. At first, only YouTube videos were clustered, as there are 124 different and still accessible videos in the relevant corpus parts. Interestingly, the `<meta name='keywords'>` element provides the keywords given by the uploader of the video. However, there are usually just 5-10 keywords associated with a video and these

[1] Available at `http://www.alchemyapi.com/`.

keywords overlap rarely. The clustering thus produced no meaningful results and no differentiation between, for example, patient experiences and professional talks became evident. When general web pages were clustered, the different URLs were grouped together according to what domain they belong to. Then it was attempted to cluster "domains" by merging the keywords of all corresponding URLs. Keywords from both sources as mentioned above were used. This lead to a degenerate clustering in which all domains appeared in the same cluster and very low intra-cluster similarity was obtained. This is partly because the web pages were in different languages, but also because sitemap and page content were too different to perform a successful feature-based clustering. Because these attempts were not successful, neither implementation nor results will be discussed any further.

5.2.2 Rank Domains

The most straightforward approach to grasp the importance of a referenced domain is simply to count the posts, that contain a reference to the domain. When counting the posts, `www.example.com` and `example.com` are regarded the same domain, because it is common for websites to forward from one domain to the other. When the occurrence counts are determined, the domains can be ranked and the results of the relevant corpus parts and the whole corpus can be compared directly using a table. For comparison purposes, it is useful to display the rank a domain has in the other corpus.

5.2.3 Classify Domains Manually

Because the automated clustering of web pages or web sites based on keywords alone is regarded unfeasible (see Section 5.2.1), the only method left to get an overview of the various kinds of references in use is to classify them manually. However, the relevant corpus parts contain 2829 different URLs. To reduce the amount of required manual work, only the domains of the URLs are classified, as mentioned in Section 5.1. In order to provide two different levels of abstraction, a classification scheme with primary and secondary classes is defined. The scheme is composed of 8 primary and 45 secondary classes, which are the following (in alphabetical order):

Association Meant in a broader sense, including foundations, organizations and unions. These are sometimes professional and often promote some kind of agenda.

- **CCSVI Association** Explicitly and exclusively promoting CCSVI as the cause of MS and advertising corresponding treatment.
- **MS Association** Generic associations providing various information and services.
- **Other Association** Associations not directly connected to MS.

Commerce Private business selling products or services that do not include treatment.

Health Insurance Health insurance companies.

Marketing / PR Company Companies offering marketing or public relations related services.

Medical Equipment Vendors of medical equipment (devices, tools).

Pharma Pharmaceutical companies.

Pharmaceutics Price Check Search engines comparing the prices of pharmaceutics from different sources.

Supplement Producer / Vendor Producers or vendors of dietary supplements.

News Commercial news providers.

Financial Information about markets, prices, stock.

Health Newspaper Magazines or newspapers focusing on health and medical issues.

MS Newspaper Magazines or newspapers addressing novelties regarding MS.

News Portal Providers of general news in different formats including articles and videos.

Newspaper Online edition of classic daily newspapers but also magazines and online-only newspapers.

TV Streaming Providers of live broadcasts, tv shows, and news in video form.

Other Various content not fitting into the other classes.

Alternative Medicine Information and sometimes commercial offers regarding complementary (belief-based) practices.

Health Portal Portals providing a wide range of health related information, FAQ, etc.

Religion / Esoterism Belief-centered content.

Search Engine / Translator Search engines and automated translator tools.

Unreachable HTTP 404 error codes and other forms of inaccessibility.

Unrelated Content not fitting in any other category and off-topic.

Personal Static content from a single person.

E-Book A book in electronic form.

Personal Homepage Classic homepages known from the early era of the web.

Scientific Sources of scientific work and knowledge.

Government Institute Institutes of various functions.

Library Access to catalogues and online content of libraries.

Science Journal Printed peer-reviewed journals.

Science Newspaper Newspapers or magazines addressing advance in science.

Science Portal Information services around science.

Social Encyclopedia Collaboratively edited online encyclopedias.

Statistics Portal Access to various offical statistics and data.

University Universities.

Social Social media web sites revolving around communication and user-generated content.

Blog Blogs by individual persons or sometimes small groups of authors.

Layperson Forum Classic online fora open for patients and laypersons.

Petitioning Presentation of petitions and signature collection.

Social Auctioning Auction houses for private persons.

Social Networking Social networking sites with profiles and friendships.

Social Newspaper News based on user-generated content and collaborative editing.

Social Photo Sharing Sharing of photos.

User Generated Content Hoster Offering upload of files, images, etc.

Video Sharing Sharing of videos.

Treatment Content and offers of treatment not limited to MS.

CCSVI Clinic A clinic offering dilatation based on the belief of CCSVI causing MS.

Clinic Generic clinics (not offering CCSVI related treatment).

Doctor's Office Presence of a doctor's office.

Health Q&A Questions and answers regarding treatment and medication.

Medical Laboratory Laboratories offering examination of medical samples.

The results of the classification can then be used to generate a compact visualization of which domain classes are used how much and at what point in time. To do so, posted references are mapped to their domain classes and aggregated for each month. A stacked area chart is then generated with a timeline as the horizontal axis and domain class occurrence per month as the vertical axis. Each area is assigned a different color and the classes are ordered descendingly by total occurrence over the whole time period. Using "stacked" areas means the first class fills the area between $y = 0$ and c_1, where c_1 denotes the occurrence of class 1 per month. The nth most important class then fills the area between $\sum_{i=1}^{n-1} c_i$ and c_n. This form of visualization can be used for both levels of detail, primary classes and secondary classes, and can also be generated for the whole corpus for comparison.

5.2.4 Cluster Users by Reference Use

When the approach of the previous section is implemented, a mapping from every relevant URL to the corresponding domain class exists. Because it is also known which users posted which references, we can construct a mapping from users to a map showing how many times they posted references from certain domain classes. An example of such a mapping is shown in Listing 5.1.

```
'John':{
        'social.blog' : 18,
        'social.video sharing' : 8,
        'scientific.science journal' : 3
}
```

Listing 5.1: Example of a mapping from a user to used domain classes.

This information already characterizes each user regarding their reference use. What is needed is a method of exploratory data analysis that can determine "types" of users from this raw data. Cluster analysis does exactly that: An algorithm groups the users by similarity in their reference use and then a human analyst can look at the clusters and describe the nature of the user types. The kernelized K-Means algorithm described in Section 2.2.1 is well suited for the task. The generic implementation can be adapted for this specific form of input by providing an appropriate kernel function. Such a kernel shall be denoted "dimension-value-map" kernel. It takes two maps like the inner map in Listing 5.1 as input and calculates the sum of the products of each value-pair. By value-pair we mean the two values retrieved from the two maps when using the same key. When x and y denote two such maps and C denotes the set of all 45 domain classes, the kernel can be stated as

$$k_{dvm}(x,y) = \sum_{c \in C} x[c] \cdot y[c].$$

The kernel thus implicitly maps into a hyperspace where every domain class is one dimension. Because different users differ in the total number of referenced domain classes, we need a mechanism to compensate for this effect. The standard approach is to compute the cosine similarity of the vectors in the implicitly defined hyperspace. For any two vectors v_1 and v_2 defined in a vector space, the cosine similarity is defined as

$$cosim(v_1,v_2) = \frac{v_1 \cdot v_2}{|v_1| \cdot |v_2|}.$$

From this definition we can derive a cosine similarity kernel that uses only implicitly defined points in the hyperspace by "wrapping around" an existing kernel. The cosine similarity kernel k_{cosim} thus has an original kernel k_{orig} as an additional argument and is defined as

$$k_{cosim}(x,y,k_{orig}) = \frac{k_{orig}(x,y)}{\sqrt{k_{orig}(x,x) \cdot k_{orig}(y,y)}}.$$

Wrapping the cosine similarity kernel around the original dimension-value-map kernel thus enables the K-Means algorithm to cluster users by their similarity in a reliable way. When the clustering is done, the human analyst can name the clusters by looking at maps as shown in Listing 5.1 but defined on a per cluster basis.

5.2.5 Find References Pointing to CCSVI Publications

Philipp provided a list of 67 scientific publications dealing with CCSVI in a non-published technical report ([29]). The list was generated by automatically constructing a dynamic citation network (see [4]) combined with some manual inspection of the obtained results. More specifically, the algorithm started with the original publication by Zamboni ([44]), determined what publications cited the paper, then determined which publications cited these, etc. The necessary publication data was retrieved from the scientific search engine CiteXplore[2]. The obtained citation network then consisted of 120 publications. The number of relevant publications was then reduced to 67 by the manual post processing. Now that it is known which publications dealt with CCSVI and when they were published, it is of interest to see, which of these were referenced by the forum users and with how much delay. A compact visual representation of the information of interest is to use a separate timeline for every publication that was referenced by the forum users. On these timelines, the publication date shall be indicated with one distinct symbol and the dates when the publication was referenced with another symbol.

However, before the visual representation can be generated, it must be determined which hyperlinks point to which publications. Title and publication id of the relevant publications are known. It is then assumed that at least one of these will be contained in a resource that gives access to a certain publication. It is also assumed that these resources can be either (X)HTML documents or Portable Document Format (PDF) documents. The structure of these resources is not known in advance. Preliminary experiments also showed that, in the case of scientific search engines[3], many different publications show up on a single web page, because there is a suggestion-of-similar-items mechanism in place. Due to these rather practical challenges, the following human-assisted algorithm was employed for every hyperlink in the relevant corpus:

1. Retrieve the resource from the web. If it is a PDF document, parse all the text from the document and include the meta data field “title”. If it is an (X)HTML document, concatenate all text from the text nodes of the document. This approach includes all kinds of content, including javascript code.

2. If title or publication ID of any of the publications is contained in the extracted text, present a list of all matched publications to the user. The user then visits the URL

[2]Available at `www.ebi.ac.uk/citexplore`.
[3]For example `http://www.ncbi.nlm.nih.gov`.

with a web browser and chooses a single publication that the web resource is about, or possibly "none of the above". The user also enters whether the URL pointed directly to the identified publication or rather to a resource that exclusively discusses the publication. Other publications or works that only contain CCSVI publications in their bibliography are not regarded a match.

The generated mapping from URLs to publications is then stored in a file. The mapping can then be used to generate the timelines. The implementation must ensure that the timelines have the same scale for the sake of easier comparison. They should also be ordered descendingly by the number of publication references on them in order to have a visual ranking of the popularity of the different publications.

5.3 Implementation: Various Remarks

The following four subsections discuss implementations of the four aforementioned approaches (not counting the discarded one). The discussion includes the used libraries, class and function structure, as well as remarks on algorithms when necessary. Unless denoted otherwise, all mentioned functions and classes reside in the module `reference_analysis`.

5.3.1 Count References to Domains

The implementation of the domain ranking is, just like the approach itself, also straightforward. The function `compare_references(original_xml, reduced_xml, ranking_list_size=15)` takes the two corpora to compare and the desired size of the ranking. The implementation just iterates over posts and counts how often each domain is referenced. References are mapped to their domains by means of the `urlparse` module, which is part of the Python standard library. References in citations are ignored and so are multiple mentions of the same reference within the same post. The function then prints a ranking table of the two corpora.

5.3.2 Load Classification and Generate Plots

Because the classification has to be done manually, the simplest and most straightforward approach is to write the mapping from domains to domain classes into a Character Separated Values (CSV) file using a text editor. The CSV file containing this mapping is located in `in/reference_classifications.csv`. It is loaded by the class `ReferenceClassesProvider` upon object instantiation. Such an object can then assign primary, secondary, or both domain classes for a given URL and assigns a default value for "not found". The input for the plot generation is created by the function `get_month_domain_use_map(...)`, which generates a map as shown in Listing 5.2.

```
'2009-02':{
        'social.blog' : 18,
        'social.video sharing' : 8,
        'scientific.science journal' : 3
}
'2009-03':{
        'social.blog' : 4
}
```

Listing 5.2: Example of a mapping from months to used domain classes.

The class `ReferenceUsePlotCollection` of module `plots` is then responsible for generating the following plots:

1. Relevant corpus parts, only primary classes over time
2. Relevant corpus parts, secondary classes over time
3. Whole corpus, only primary classes over time
4. Pie chart of domain classes in the top 15 domains.

The plots are generated using `pylab / matplotlib` and especially the `fill_between(...)` function[4]. Because of the high information density in the graphs, the color layout is important. Therefore, the `plots` module has `ColorProvider` classes, that provide lists of carefully selected colors. The purpose is to have easily distinguishable colors on adjacent primary classes. In addition to the first requirement, all secondary classes belonging to one primary class should look similar.

5.3.3 Kernelized K-Means with Pluggable Kernel

The required kernelized K-Means clustering is implemented in the `kernelized_k_means` module. An implementation in pure Python is not the most efficient one, but it has the advantages of fast development, easy maintenance, and seamless integration with the other parts of the system. The implementation is generic, because the kernel function to use can be passed-in as an argument and there are no restrictions on the data structure representing the features of an object. As long as the kernel function can produce a valid value from two feature collections, the requirements of the implementation are met. Therefore, the implementation will be reused in later parts of the work.

The implementation is based directly on the formulas discussed in Section 2.2.1. Figure 5.1 shows a UML class diagram of the three classes of the implementation. The clusterer object

[4]See `http://matplotlib.org/examples/pylab_examples/fill_between_demo.html`.

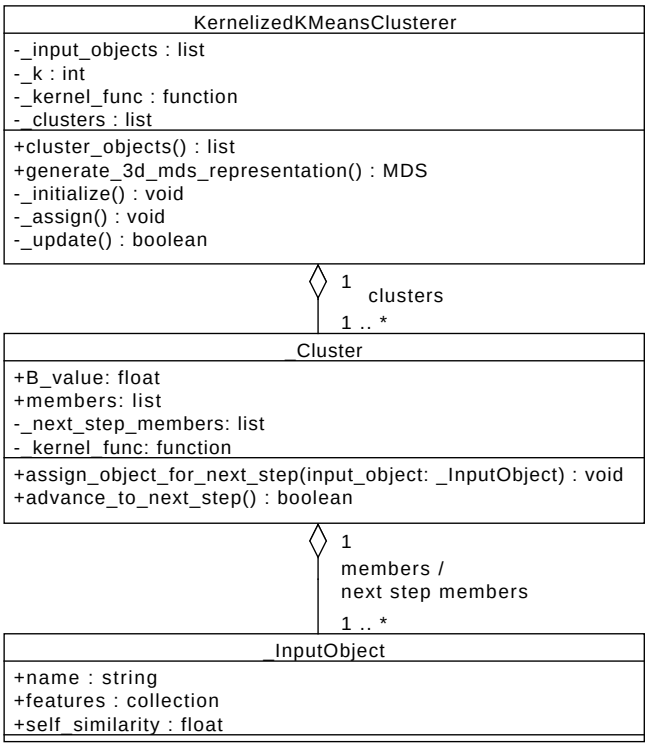


Figure 5.1: Class diagram of the kernelized K-Means implementation. The plus (+) symbol indicates public attributes and the minus (-) sign private ones. The structure is a simple three level aggregation that is intended to cache constant intermediate results.

is the only class interfacing with the client code. It is initialized with input objects (as defined in Section 2.2.1), the parameter k, and the kernel function. When `cluster_objects()` is called, the clusterer object performs the steps (heuristic) initialization, assignment, and update until total convergence. Internally, it uses `_Cluster` objects as containers for the `_InputObject`s. The cluster objects maintain two lists: those input objects that are assigned to the cluster in the current step and those that will be assigned to it in the next iteration. The latter list is populated during the assignment step and the cluster objects are then advanced to the next iteration step in the update stage of the clusterer. Note that two mathematical terms used in the distance calculation between a single input object and a cluster are (partially) constant. As described in Section 2.2.1, the term A denotes the self-similarity of an input object ($k(x,x)$) and is always constant. The value is thus stored in an `_InputObject` and only calculated once. The term B is constant for a cluster within one *iteration*. It is thus stored in a `_Cluster` object and updates when the cluster is

advanced to the next iteration. The method advancing a cluster returns whether the memberships have changed and the clusterer can thus stop when all clusters stay the same from the previous iteration to the current. The clusterer also provides a method generating a `multidimensional_scaling.MDS` object with input objects in the order that resulted from the clustering. It can be used to generate a plot showing the clusters in a hyperspace, but is not used so far.

The clustering of users by their reference use is started by a call to the function `cluster_users_according_to_domain_classes(..)` of the module `reference_analysis`. The function generates a map as shown in Section 5.2.4 by using the kernel described in that section and the clustering implementation described in this section.

5.3.4 Fetch URLs and Parse Content

The class `ScientificPublicationsProvider` loads the CCSVI-related scientific publications from the CSV file `in/ccsvi_publications.csv` on object instantiation and provides methods to retrieve the information as `Publication` named tuples.

Objects of class `ReferencedPublicationsProvider` access these tuples and provide the core functionality of the approach. The method `fetch_from_web(references)` fetches every URL from the web that was passed in as argument by using class `HyperlinkAnalyzer`. The latter retrieves the content from the web using `urllib2`. Depending on the Content-Type field in the HTTP header, it either parses the text out of an (X)HTML document using `BeautifulSoup`[5] or parses the text out of a PDF document using `pyPdf`[6]. The parsed text is then searched for any contained publication titles or IDs by the `ReferencedPublicationsProvider` object. An interactive console interface then asks the user for manual selection and verification of every URL, that produced a match. The resulting mapping from references (URLs) to publication IDs is then stored in the CSV file `referenced_publications.csv`. The mapping generated by the method `get_publication_timestamps_tuples(posts)` from publications to when they were referenced, is then used as the input for the plot. The plot is generated by the method `timeline_of_referenced_publications()` of class `plots.ReferenceUsePlotCollection` using `pylab / matplotlib`.

5.4 Results: Patterns and Trends in Reference Use

The following four subsections discuss results in the form of plots, tables, and rankings of the aforementioned implementations. The results are interpreted regarding the research questions derived from the problem statement in the beginning of this chapter.

[5] See Section 3.3.

[6] See `https://pypi.python.org/pypi/pyPdf`.

5.4.1 Most Popular Domains

Table 5.1 shows the resulting ranking of the most popular domains. On both sides, the video streaming platform YouTube and the DMSG website itself are linked to the most often. However, the social encyclopedia website Wikipedia, which is regarded a rather factual source[7], ranks 3rd in the full corpus, but is much less important in the CCSVI-related discussion. Interestingly, the social networking site Facebook ranks 5th in the CCSVI-related discussion, but does not show up at all in the 15 most popular domains of the full corpus. Also, a lot of blogs and fora based around CCSVI are popular in the relevant part (`csvi-ms.net, thisisms.com, ccsvi-ms.pl, das-ccsvi.net`), while associations and MS portals comprise the popular references in the full corpus. A possible interpretation of these two findings is, that the CCSVI-related discussion relies more heavily on personally exchanged opinions or stories than a typical discussion on the DMSG layperson forum. Factual sources or widely agreed upon viewpoints seem to be less important than in a usual discussion.

Full Corpus			Relevant Posts		
Opp. rank	Occ.	Domain	Opp. rank	Occ.	Domain
1	3490	youtube.com	1	289	youtube.com
2	1982	dmsg.de	2	166	dmsg.de
6	1100	de.wikipedia.org	11	131	csvi-ms.net
-	414	bilderload.com	-	67	thisisms.com
13	407	ms-forum-weihe.de	-	67	facebook.com
-	359	amsel.de	3	61	de.wikipedia.org
-	349	ms-netz-hamburg.de	-	59	ccsvi-ms.pl
-	316	ms-life.de	-	54	das-ccsvi.net
-	311	msweb.lu	-	35	gastgitarre.de
10	290	aerztezeitung.de	10	34	aerztezeitung.de
3	258	csvi-ms.net	14	33	ncbi.nlm.nih.gov
-	221	de.youtube.com	-	30	ccsvi-tracking.com
-	207	spiegel.de	5	28	ms-forum-weihe.de
11	195	ncbi.nlm.nih.gov	-	27	ctv.ca
-	189	google.de	-	24	ms-info.net

Table 5.1: The 15 most cited domains in the full corpus (left side) and the relevant parts (right side). 'Opp. rank' indicates the rank of the domain on the opposite side. Example: `ms-forum-weihe.de` *is on rank 5 in the full corpus.*

[7] Fallis [16] gives a good summary of literature discussing the reliability of Wikipedia. The outcome is positive.

5.4.2 Visual Representation of Reference Use Over Time

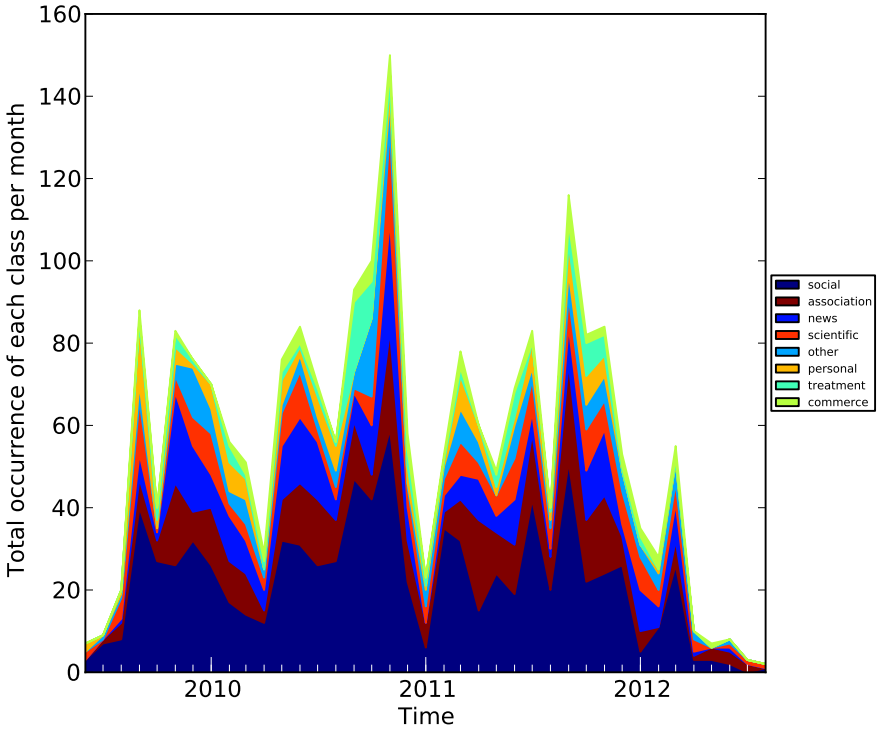


Figure 5.2: Primary domain class occurrence of the relevant parts. The occurrences are aggregated for each month.

Reducing the hyperlinks found in the relevant part of the corpus to their domains resulted in 545 different domains. These were classified manually by taking into account the overall appearance of the website and especially the "about" section. Figure 5.2 shows a plot of the used primary domain classes in the relevant corpus parts as constructed by the algorithm discussed in Section 5.2.3. The horizontal axis of the plot ranges from June 2009 (one month after the original publication by Zamboni) to August 2012. Several interesting findings are evident from the plot.

First of all, social media references are the most popular ones at any given point in time. Association related websites and news sites come second and third. These domain classes also account for the highest variance in total reference use. Secondly, there are large differences in the total amount of posted references per month. The peak was reached

in November 2010 with about 150 different references posted in that month. April 2010 and January 2011 show sharp declines in the number of posted references with 30 and 25, respectively. This correlates rougly with the total number of relevant posts over time, but the external events causing these declines in activity are not yet known. The plot also shows, that the topic of CCSVI caught on quickly in the layperson forum. The original publication of Zamboni was published in May 2009 and it was already in August 2009 that a considerable amount of references discussing the topic were posted. The forum users seemed to have lost the interest in the debate lately, as suggested by the few CCSVI-related references posted in 2012. Thirdly, when the total number of posted references rises from a given point in time to another one, the change is typically reflected in all of the domain classes. This means that certain trends of interest, independent of what they are caused by, typically echo in all kinds of references.

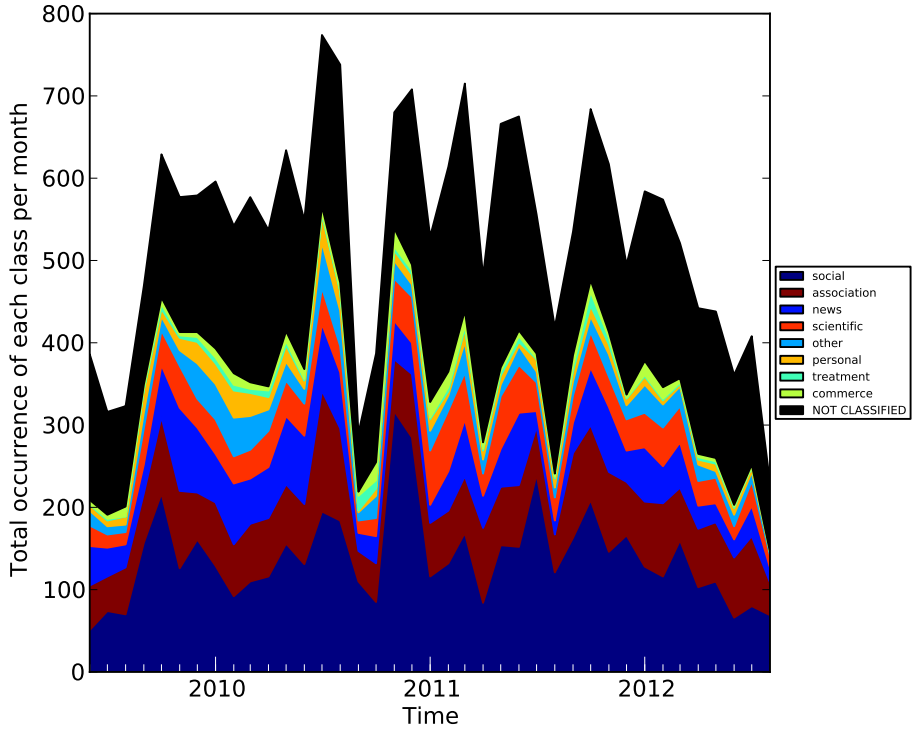


Figure 5.3: Primary domain class occurrence of the full corpus. The black area indicates not classified domains. These domains may belong to any of the mentioned classes.

Let us compare the plot reflecting the relevant parts of the corpus to Figure 5.3, which shows a plot of the whole (unfiltered) corpus. The magnitude of the vertical axis indicates that the users generally post a vast amount of hyperlinks discussing topics other than CCSVI. The large black area in the plot indicates references whose domains have not been classified. These show up, because the manual classification was only done for the domains of the relevant corpus parts. The relative importance of the domain classes looks similar to the one in Figure 5.2. However, because a large part of the domains are not classified, a bias is possible. What can be said for sure is that the whole corpus follows own trends, that do not necessarily match those of the relevant part. For example, Figure 5.3 shows a sharp decline in posted references between September and October 2010, that does not show up in the relevant part. On the other hand, declines in the relevant part plot do not have an impact on the whole corpus plot. This means that the forum is used continuously for all kinds of discussions. If the users do not discuss CCSVI so much in a given month, they tend to discuss something else.

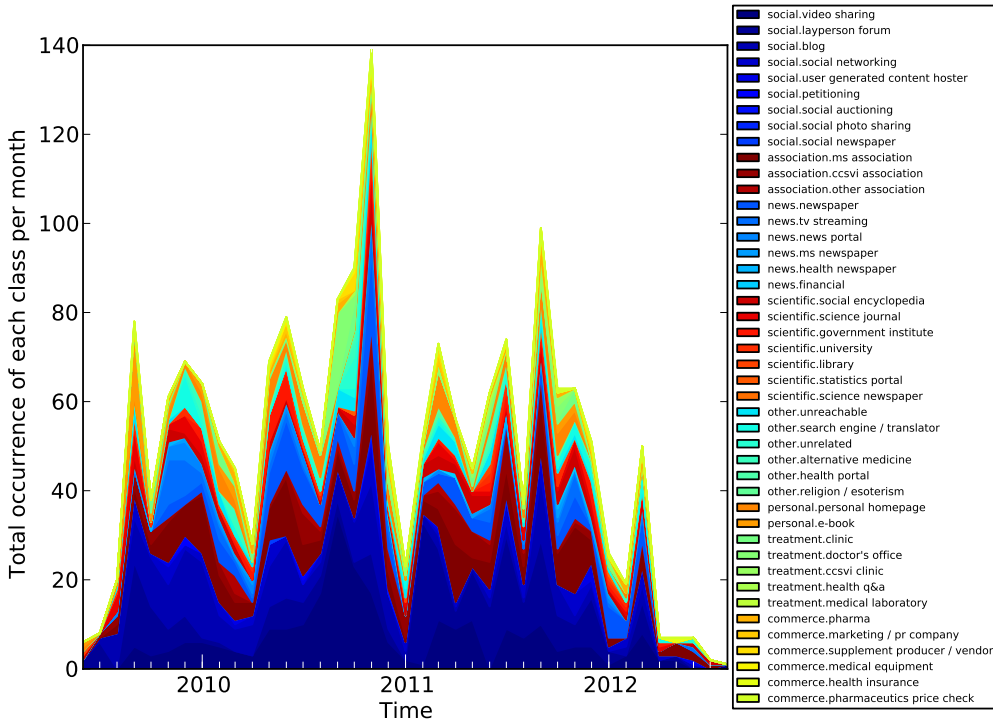


Figure 5.4: Secondary domain class occurrence of the relevant parts. Note that all secondary classes of a single primary class share the same shade of color. Example: All social classes are depicted in a shade of blue.

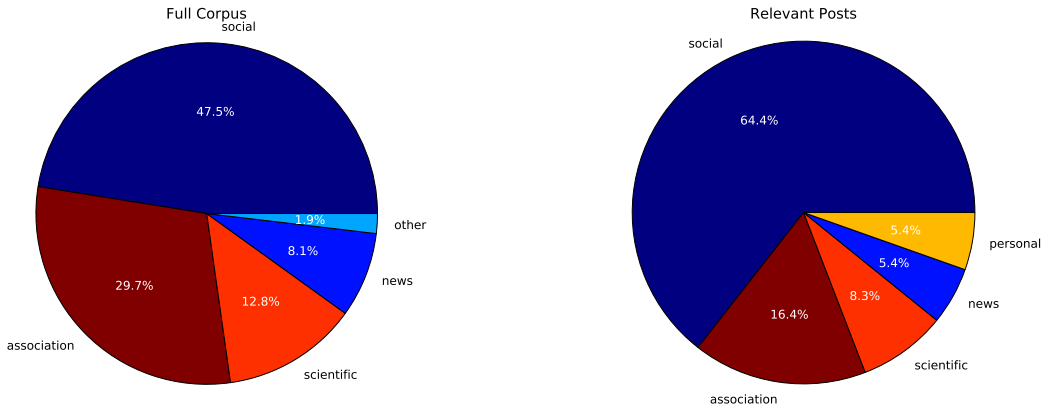


Figure 5.5: Pie charts showing the relative amount of occurrences of the primary domain classes of the top 15 domains. The whole (unfiltered) corpus is on the left hand side while the relevant parts are on the right hand side.

The manual classification used a two-level taxonomy of primary and secondary domain classes. The aim was to retain detail while also having enough abstraction for meaningful interpretation. Figure 5.4 shows a plot that includes these secondary classes. Because of the specific choice of colors, the plot looks very similar to the one in Figure 5.2. It is difficult to derive additional meaning from this more fine grained representation, but it can be used as a source of information for future questions that require more detail to answer them.

Because a classification of domains is available, the top 15 domains discussed in the previous section can be analyzed further. Figure 5.5 shows two pie charts, one of them based on the whole corpus and the other one on the relevant parts. The chart on the right hand side is based directly on Table 5.1 and the other chart on an equivalent. The comparison of the two charts supports the finding of the previous section, which stated that social media seems to be more important in the CCSVI-related discussion than on average. This dominance of social media comes at the cost of association-provided information and scientific sources.

5.4.3 User Patterns in Reference Use

Two different user clusterings were performed. The first one used usernames that were not cleaned for anomalies. By "anomalies" we mean the fact that the forum software puts brackets "`()`" around a user name in old posts (posted before September 2010), but does not do so for posts after that date. As this is not touched in the first clustering, `username` and `(username)` are regarded different entities. When the clustering is performed with k=5, the

#	Members				
0	(u_02771)	**(Elizabeth)**	**Elizabeth**	**Owen**	(u_07545)
	u_07416	u_06522	(u_07177)	u_12555	(u_12835)
	u_00724	u_05725	u_06222	(u_10012)	(u_03525)
	u_05872	u_00224	(u_01251)	**(James_1)**	
1	(u_01752)	(u_06282)	u_08936	u_11474	u_09623
	u_08635	u_12897	u_09468	(u_12104)	u_02548
	(Owen)	**Margaret_1**	**(Margaret_2)**	u_03525	u_05013
	(u_07827)	u_04138	(u_06775)		
2	**(Richard)**	(u_00748)	u_00999	u_04307	**Richard**
	(u_05323)	**(James_2)**	u_01167	(u_02465)	u_00831
	u_07545	(u_10778)	u_06747	(u_08979)	u_09774
	u_07736.	**(James_3)**			
3	(u_12444)	(u_00917)	(u_05406)	(u_03459)	(u_10052)
	(u_05493)	**(William)**	(u_08075)	(u_06222)	u_08078
	u_10668	**William**	(u_00243)	(u_00637)	
4	u_07878	(u_11880)	**(James_4)**	(u_03435)	u_06796

Table 5.2: User clusters calculated from reference use with non-clean users. Note that `(user)` *and* `user` *tend to appear in the same cluster.*

five clusters seen in Table 5.2 are obtained. Note that we anonymized the user names for privacy. We gave actual English first names to users who are discussed in the text.

Interestingly, a username and the counterpart with brackets around it tend to appear in the same cluster. This holds for users `Elizabeth, William, Richard`, but not for `Owen`. It is also likely that `(Margaret_2)` and `Margaret_1` are accounts of the same person, because the original user names are different spellings of a quite unique name. Both are located in the same cluster. There is a set of 4 user names (`James_1, James_2, James_3, James_4`) that appear to be variants of the same name and they are located in 3 different clusters. If we assume that the same person is behind these accounts, the person either changes their taste in references over time, behaves differently when using different accounts, or simply has no distinctive taste in references. Besides that, we have already identified four users whose different accounts show up in the same cluster. Thus, the clustering results can be regarded plausible, because the preference of a person towards certain domain classes seems to be consistent across accounts.

After the test for plausible results, the clustering was performed with cleaned user names (the anomalies were removed). Again, k=5 was used and the results are shown in Table 5.3. This time, the clusters were less balanced in size as cluster 2 has by far the most members. The names for the clusters were derived manually from the information shown in Table 5.4. This means that a human interpretation of the aggregated reference use behaviour was the basis of the cluster names, which is common in exploratory data analysis. The clustering of users according to their reference use results in several interesting findings.

Most of the users are regarded Video Sharers. They prefer videos from websites such as `youtube.com` over written sources. This, however, does not say anything about the type of information these users propagate, because video sharing sites contain all kinds of different material. Video Sharers also tend to post references to MS associations, newspapers, blogs

#	Description	Members				
0	Undefined	u_03459	u_00243	u_08936		
1	Video Sharers	Owen	u_00999	u_07878	u_06522	u_12555
		u_12897	u_11474	u_09623	u_00724	u_06025
		u_12835	u_12104	u_05725	u_06282	u_02771
		u_09468	u_01251	u_06796	u_07177	u_02548
		u_11880	u_03435	u_05872	u_07827	u_01752
		u_06222	u_10012	James_4	u_07754	u_00224
		u_03525	Margaret_1	u_05013	Elizabeth	u_07736
		Margaret_2	u_04138			
2	Balanced Communicators	u_10778	u_07416	James_2	u_01167	u_08979
		James_3	u_05323	u_09774		
3	Homepage Promoters	u_06775	u_00748	Richard	u_02465	u_00831
		u_07545	u_06747			
4	Bloggers	u_04307	u_00917	u_08078	u_12444	u_10668
		William	u_10052	u_00637	u_08075	u_05493
		u_04058	u_05406	James_1		

Table 5.3: User clusters calculated from reference use with cleaned users.

#	Cluster Name	Domain Class	Total Occurrences
0	Undefined	other.search engine / translator	8
		news.tv streaming	4
		social.blog	4
		association.ms association	3
		other.alternative medicine	1
1	Video Sharers	social.video sharing	202
		association.ms association	113
		news.newspaper	74
		social.blog	56
		scientific.social encyclopedia	41
2	Balanced Communicators	social.layperson forum	140
		association.ms association	39
		association.ccsvi association	31
		social.video sharing	21
		social.social networking	15
3	Homepage Promoters	personal.personal homepage	58
		association.ms association	12
		news.newspaper	12
		social.layperson forum	10
		association.ccsvi association	5
4	Bloggers	social.blog	84
		social.layperson forum	41
		social.video sharing	37
		social.social networking	31
		association.ccsvi association	21

Table 5.4: Total domain class occurrences of the top five domains of each clusters. The numbers stem from an aggregation over all the users from within a cluster.

and even social encyclopedias, which emphasizes the diversity of interest within the cluster. A small but important group are the Balanced Communicators. These users bring together different communication channels by posting references to other layperson fora, as well as social networking and video sharing sites. They promote the formation of opinions by presenting both, generic MS associations and dedicated CCSVI associations. The second largest group are the Bloggers. The blog is their medium of choice and other rather opinion-based sources are also important. Bloggers use CCSVI associations, which might explain

how the topic could catch on. Interestingly, there is a group, that can best be described as Homepage Promoters. What distinguishes them primarily is their preference of personal homepages. These websites feature static content authored by a single person and already existed in the early era of the Internet. Cluster 0, though, has just 3 members and can be regarded an artifact of the clustering.

5.4.4 Delay in Use of Scientific Publications

The human-assisted search for referenced scientific publications revealed that out of the 62 publications dealing with CCSVI, 7 were referenced by the forum users. Figure 5.6 shows each of these publications in a separate area. Interestingly, the original CCSVI publication by Zamboni (represented at the very top) was introduced to the forum already two months after publication. It was then referenced again multiple times throughout the following years until October 2011. Other (mostly critical) publications were referenced as well, with typical delays of 1 or 2 months. However, a total of 19 references to scientific publications over the course of four years is much less then expected. It remains unclear whether some references could not be identified properly or the forum users do indeed disregard scientific publications to a large extent.

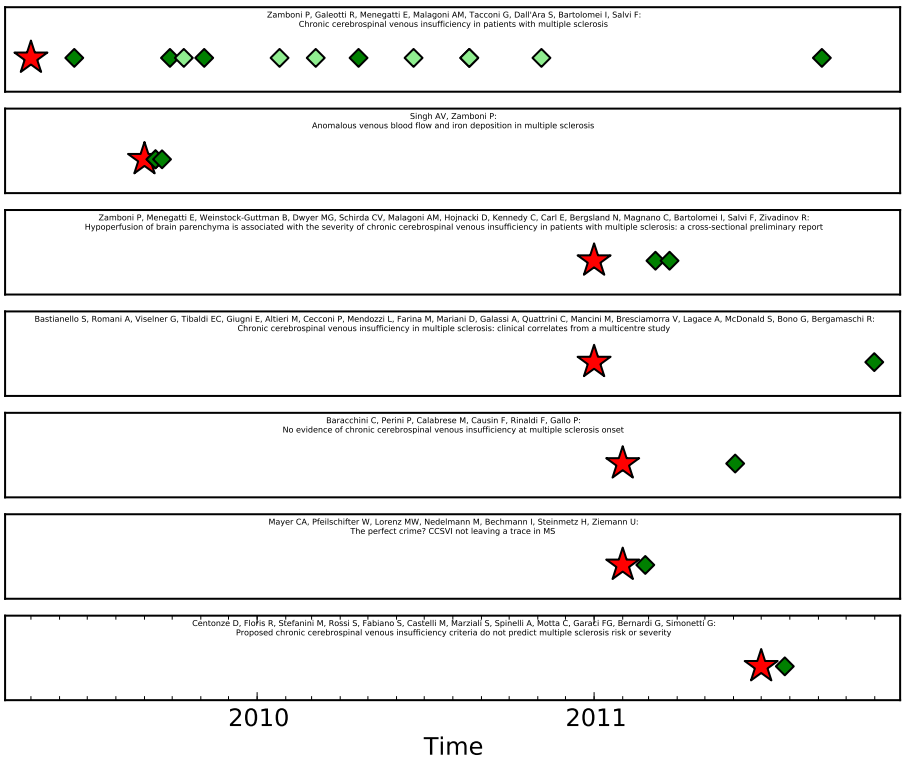


Figure 5.6: Scientific Publications referenced by forum users. Each publication has a separate area. The star indicates the date of publication, whereas the diamonds indicate points in time, when the publication was referenced in a forum post. A dark green diamond indicates a direct reference to the publication, whereas a light green diamond indicates an indirect reference. All of these areas share the same x-axis.

6 Describing User Behavior and Influence

The second entity type of interest are the forum users. We want to find ways to describe them based on the data we have extracted. As a first step, we formulate specific problems to address.

6.1 Problem: Describe User Behavior and Influence Based on Limited Information

As we want to find out more about user behavior, relationships, and influence, the following questions are of interest:

1. In the study of social networks, an often observed attribute of individuals is homophily. The term was originally introduced in [25] and describes the tendency of people to associate with people similar to themselves. Although the concept itself was already described in the 1950s, it has drawn interest again in the automated social network analysis (see for example [35]) and, more importantly, it is also used in the analysis of medical communities [8][10]. As we have already clustered users according to their reference use, it is of interest to analyze, whether there is some sort of homophily regarding reference use. The question is, whether users tend to associate more often with people of the same reference use cluster. To do so, we need to find a way to describe the behavior of the users in an aggregated way and compare the results to the case of the hypothetical irrelevance of reference cluster membership.

2. How can we identify patterns in user behavior? How can we assign roles to users based on their general behavior? To do so, we need to find descriptions of what, how, and when users post. From these descriptions, we aim to find distinct roles by grouping similar users together.

3. How can we assess the influence of users on the discussion? In doing so, can we identify the 50 most influential users and make statements about reference use and general behavior of these users?

The following section discusses approaches to answering these questions.

6.2 Approach: Create Graphs and Define User Features

This section contains an important preface that discusses two opposing approaches to the creation of graphs from forum data. The shown concepts are used again in the subsequent three subsections that discuss approaches to the research questions formulated above.

6.2.1 Preface: Two Approaches to Graph Creation

Several methods intended to describe user behavior or user influence in online communities stem from the field of Social Network Analysis (SNA). These methods require a social network, which is a graph consisting of nodes representing users and edges representing communication relationships[1]. In an online forum though, explicit communication relationships do not exist. Instead, users post to threads, one post after another and visible to everyone. However, researchers often assume that these communication relationships exist implicitly. A mapping is thus required from the sequential posts-in-a-thread data structure to a graph connecting users.

The common approach is the so-called reply-graph. Here, if user B makes a post directly after a post of user A, the created graph will contain a directed edge from B to A. In cases when an undirected graph is required, an undirected edge can be used. The approach is quite popular. For example, [31] calls the reply-relationship the "most obvious relationship among users" while other sources using this approach include [9][17][42][45]. If we want to apply this graph creation method to only reflect communication of relevant content, we end up with regarding a series of relevant posts as a separate thread. The reply-graph is based on the assumption that a user posts in order to answer the previous post and thus assumes information exchange only happened between these two people. This assumption can be disputed. If, for example, a user comes home from work, visits the forum and notices a new thread was started, they might read all the previous posts and then decide to comment. The comment is thus not necessarily a response to the last post alone.

We propose another approach that is especially suited for cases when relevant parts are scattered across threads. We define a continuous discussion as the longest uninterrupted sequence of relevant posts. Then, we create a bipartite graph connecting user nodes with continuous discussion nodes if they posted in such a series of relevant posts. We then project the bipartite graph onto the users. The result is a graph where users are connected if they were connected to the same continuous discussion nodes. The two approaches are illustrated in Figure 6.1 in an example case of three relevant posts.

We want to get an overview on how these different approaches vary in the number of graph neighbors that a typical user has. We thus compute the degrees of each node without using weights, effectively counting the number of neighbors. Comparing the two histograms in Figure 6.2, we notice that both distributions look similar. Also, both follow the power

[1] To be precise, SNA is not restricted to modeling communication.

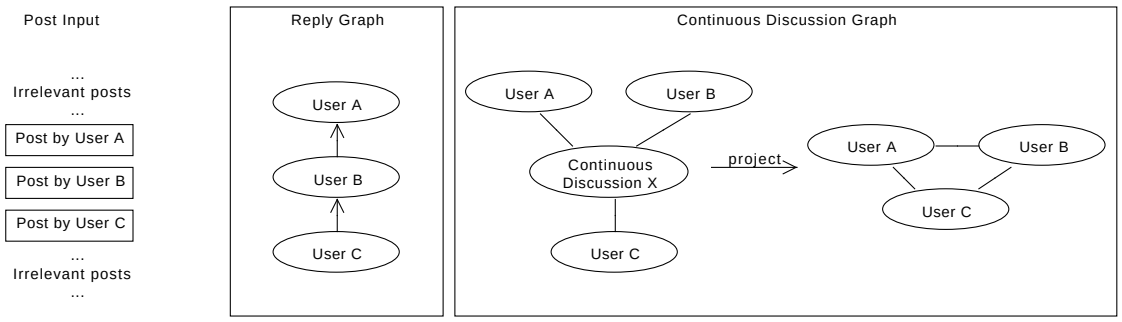


Figure 6.1: Illustration of the two graph creation approaches. The exemplary input is shown on the left. The center shows the classical reply graph resulting from the input. On the right, the user-user-graph, obtained from projecting a user-discussion-graph onto the users, is shown.

law, which is expected in a situation of online communication. The continuous discussion approach though results in increased node degrees. We prefer to use this approach when we need to represent user communication about CCSVI, because the assumptions seem more plausible.

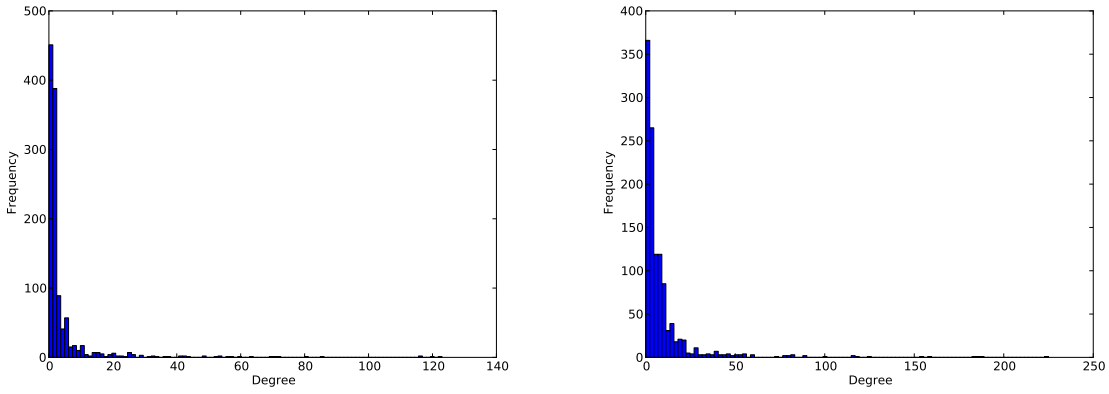
6.2.2 Assess User Communication

The user-user-graph is first reduced in size by removing those users from it, that do not show up in the reference use clustering (because they posted too few references). The graph can then be described as $G = (U, E)$, where $U = \{u_i\}$ denotes the users. The edges can be defined as a set of two-element subsets of the nodes. For example, $E = \{\{u_1, u_2\}, \{u_2, u_3\}\}$ means the nodes (users) u_1 and u_2 are connected with an edge as well as u_2 and u_3. We denote the function mapping users to their reference use cluster by $m : U \to C$. In order to examine whether there is some sort of homophily regarding the reference cluster membership of the users, we project the graph onto the clusters. This means, we construct a graph with (in this case five) nodes where each node represents a cluster. Each cluster i is assigned the size attribute equal to the number of cluster members:

$$size(i) = |\{u | u \in U, m(u) = i\}|.$$

Each node is connected to every other node and will also have a self loop associated. The edge between node i and j will have a weight corresponding to the number of users of the two clusters that had a connection in the original user-user-graph. The weight between cluster nodes i and j is thus defined as

$$weight(i, j) = |\{\{u_a, u_b\} : \{u_a, u_b\} \in E, m(u_a) = i \wedge m(u_b) = j \vee m(u_a) = j \wedge m(u_b) = i\}|.$$



(a) The classical reply graph.

(b) The continuous discussion approach.

Figure 6.2: Histograms of unweighted user degrees. These figures are computed over relevant corpus parts only.

The resulting graph can be drawn with node diameters proportional to the size attribute and edge thicknesses proportional to the edge weights.

However, from the reference cluster connection graph alone, it is not evident, whether users tend to talk more often to users from the same cluster. This is the case, because the clusters have very different sizes (number of members) in the first place. If a cluster is very large, a thick self loop does not imply that the cluster members prefer to communicate with their own kind rather than with users from other clusters. If users chose their communication partners randomly, a thick self loop would also be expected for a large cluster, because a randomly chosen partner is more likely to stem from it. To overcome this problem and provide a meaningful interpretation of the graph, we need to compare it to a graph reflecting the situation, where reference cluster membership does *not* influence the communication behavior of the users. In other words, the baseline situation must be one, where every communication relationship is random.

A simple stochastic model is thus derived, that generates the data used for comparison. The purpose is to have a graph with the same cluster sizes and number of communication relationships as in the observed case, but with edges stemming from a random choice of communication partners. The model takes as input the observed cluster sizes $\{c_0, c_1, \ldots c_k\}$ of the clusters $\{0, 1, \ldots, k\}$ and the number of observed communication pairs denoted by $n = |E|$. The model then describes the edge weights of the complete reference cluster graph. Such an edge weight connecting clusters i and j is denoted by e_{ij}. The model assumes that the fixed n pairwise communication relationships are distributed randomly. This means, a communication relationship (u_1, u_2) is modeled by two random variables: the first user and

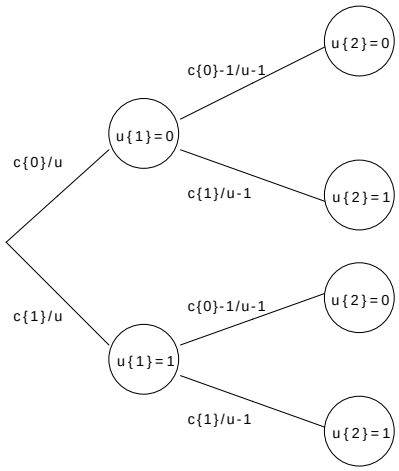


Figure 6.3: Probability tree for a model with two reference use clusters.

the second one. Both are drawn from the hypothetical pool of u users ($u = \sum_{i=0}^{k} c_i$). The probability of the first user being from cluster i is thus $P(u_1 = i) = \frac{c_i}{u}$. Because a user cannot have a communication relationship with themselves, the users are said to be drawn from the user pool without replacement. The probability distribution for the second user thus depends on the first user. The probability of the second user being from the same cluster is $P(u_2 = i|u_1 = i) = \frac{c_i-1}{u-1}$. The probability of the second user being from another cluster $i \neq j$ is $P(u_2 = j|u_1 = i) = \frac{c_j}{u-1}$. Figure 6.3 shows an illustration for a model with two clusters.

The joint probabilities are then used to compute the edge weights e_{ij}. Because the edges are undirected, $P(u_1 = i, u_2 = j)$ and $P(u_1 = j, u_2 = i)$ both account for the same edge. Because the joint probabilities are intended to distribute the n communication pairs over the edges, we need to multiply them with n. Every edge weight is thus defined by the stochastic model as:

$$e_{ij} = \begin{cases} \frac{c_i}{u} \cdot \frac{c_i-1}{u-1} \cdot n & \text{if } i = j \\ (\frac{c_i}{u} \cdot \frac{c_j}{u-1} + \frac{c_j}{u} \cdot \frac{c_i}{u-1}) \cdot n & \text{if } i \neq j, \end{cases}$$

which can be shortened to:

$$e_{ij} = \begin{cases} \frac{c_i(c_i-1)n}{u(u-1)} & \text{if } i = j \\ \frac{2c_ic_jn}{u(u-1)} & \text{if } i \neq j. \end{cases}$$

6.2.3 Cluster Users by Behavior Features

Assigning roles to users of social media is not a new approach. More specifically, the task has also been done for online fora before, in various ways. The approaches mainly vary in three different aspects: Which and how many roles are defined, how these roles are assigned, and what assumptions these assignment methods are based on. Several approaches define just two roles up front and thus reduce the role finding task to a task of binary classification. For example, [22] distinguishes between information seekers and providers, while [3] defines answer and discussion persons. Sometimes, the approach is to separate the answer person from other user types like in [42]. Other work attempts to describe the behavior of the users in more detail and thus defines more than two roles. For example, [5] defines 5 roles and [9] defines 8. Welser et al. [41] define 4 roles, but their work is based on the social encyclopedia website Wikipedia and thus the used data source is different in structure and semantics from an online forum. The approaches also vary in the way they assign the roles. Welser et al. [41] use a very simple approach and define fixed decision rules based on the intuition of the authors. They define, for example, that "if a user has more than 60% content edits and <*some other conditions hold*> the user is assigned the role of the Vandal Fighter". Such an approach is highly subjective and relies on the researches ability to have an intuition on what user roles might be prevalent in a given community. Other approaches include a survey of users (in [5]) or the informal definition of user roles derived from looking at several feature visualizations [36][39]. The most sophisticated methods use mathematical models or machine learning techniques in order to assign roles automatically [3][9][22][42].

Out of these solutions, the work presented in [9] is the most important one and comes very close to what we want to achieve. They assign a high number of roles, which corresponds to a detailed description of user behavior. Their role assignment is based on a clustering algorithm and thus automated and less subjective. Finally, there are no assumptions about possible user roles made, as clustering is an exploratory data analysis. We thus use the same overall approach:

1. We define a set of features based on meta data of the extracted corpus. These features are discussed below.
2. We perform clusterings with different numbers of clusters and use different internal evaluation metrics to find the optimal number of clusters. Here, we use the internal evaluation metrics discussed in Section 2.2.2.
3. We then explore the resulting clusters and name them according to what behavior the users expose.
4. We create a plot showing the cluster sizes.

We chose 9 different features, that aim to describe what and how a user posts. Some of them are taken from [9] or other sources and are described below. If not noted otherwise,

the features are defined over the whole, unfiltered corpus, because user behavior is assumed to be independent of the topic at hand.

avg_msg_length $\in [0; \inf)$ (from [5]): Average post content length in characters without counting references. The message length is an indicator of the amount of effort, that is put into a post by a user, and it also tells us something about the discussion style of a user. Some users prefer elaborated, essay-like contributions while others use the forum in a more conversational way.

avg_posts_per_day $\in (0; \inf)$: Average number of posts per day, that the user made. This is the most important activity feature of a user and it also provides an insight into the selectiveness of the user. A user with a high number of posts per day over a long time period can be expected to be a frequent visitor, who makes post regardless of outside events.

avg_refs_per_post $\in (0; \inf)$: Average number of distinct references that are included in a post. The feature describes the tendency of a user to bring new sources of information to the forum and may also describe the ability to support the stance of the user with evidence.

avg_threads_per_day $\in (0; \inf)$: Average number of different threads a user posts to per day. While this is also an activity feature, it provides an insight into the focus of interest a user has. A low value may indicate a preference to discuss only specific topics while a high value may indicate a preference to join any sort of discussion.

days_active $\in [1; \inf)$ (from [5]): Number of days between the first post and the last one. The feature indicates the consistency of the contribution behavior of a user and is an important piece of context information when interpreting the other features.

fraction_of_posts_cited $\in [0; 1]$: Fraction of the posts that have been cited at least once. While it can only be assumed what users try to express when they use the citation function, the feature is expected to show the tendency to provoke direct responses from other forum participants.

fraction_relevant_posts $\in [0; 1]$: Fraction of the posts that were classified as relevant by the Information Retrieval algorithm. The feature is a solid indicator of the interest in CCSVI, that the user has. While it can not be inferred from this feature alone, whether the user has a pro-CCSVI or contra-CCSVI stance, it seems plausible that users with a high interest in CCSVI believe in the hypothesis.

fraction_threads_initiated $\in [0; 1]$ (from [9]): Fraction of the threads the user has initiated, based on the total number of threads the user contributed to. This feature

measures the tendency of a user to start discussions, which is often related to the introduction of new information to the forum.

relevant_user_coverage $\in [0; \text{inf})$: Number of users the user discussed CCSVI with divided by the total number of posts the user made. This means we use the user-discussion-graph discussed earlier and project it onto the users. In the resulting graph, a user is connected with every user that took part in at least one of the same continuous discussions. Thus, the feature can be described as the efficiency in opinion exchange about CCSVI.

Note that the selected features differ from existing work in several ways.

- We make use of corpus meta data that has not been used in the work we know of, namely citations and references.
- We use the tendency of a user to discuss the topic of CCSVI in features, which can be attributed to the specific context of this work.
- We count threads and posts per day, because they provide an important activity measure.
- We decide not to use several reply graph based features used in [9] like in- and out-degree exponents, % of bidirectional neighbors and % of bidirectional threads. Preliminary experiments have shown that these features do no discriminate the users well. We assume that there are two reasons. Firstly, [9] base their metrics on Q&A related boards, but the DMSG forum is to a much lesser extent a Q&A forum. This is illustrated by the fact that typically 90% of the users of a Q&A forum only post once (because they have received a response to their question), but in the DMSG forum, this percentage of "lurkers" is only 62.9%. Secondly, the reply-graph itself is based on assumptions that we must criticize (see Section 6.2.1).

Because these features have very different scales, we perform a z-score normalization before the clustering. That means, we adjust each feature value by subtracting the feature value mean from it and dividing it by the standard deviation. If $x[d]$ is the value of feature d of object x we set it to $x[d] := \frac{x[d]-\mu_d}{\sigma_d}$. This ensures that features with large absolute values (like days active) do not dominate the distance calculations. This normalization technique is rather robust to outliers [37]. For the clustering we use the kernelized K-Means algorithm, that has already been used in the clustering of users according to their reference use. As a kernel function, we use the dimension-value-map kernel, which simply maps the objects to a linear feature space where every of the 9 features is represented by one dimension each:

$$k_{dvm}(x, y) = \sum_{d \in Features} x[d] \cdot y[d].$$

The use of a kernel is thus more an implementational detail.

6.2.4 Compare Several User Influence Measures

Several approaches to assessing the influence of users regarding the discussion about CCSVI exist. All of them have in common, that they assign scores to users. From these scores, a ranking of users can be generated with the most influential user having rank 1. We want to compare the different approaches according to how similar their results are in this practical case. Note that we are not interested in the difference in score values, because they can be regarded an implementational detail of a ranking approach. Instead we are interested in comparing the resulting rankings themselves.

Spearman's ranking correlation coefficient is a measure that can describe the dependence of the two rankings and is thus a measure of similarity. It was originally proposed in [32]. For two rankings, a and b, it is defined as

$$r_s = \frac{\sigma_{a,b}}{\sigma_a \cdot \sigma_b},$$

where σ_a describes the standard deviation of the ranks of a and $\sigma_{a,b}$ describes the covariance of the ranks of a and b. Note that this definition is the same as the one of the popular Pearson product-moment correlation coefficient ρ, except for the fact that r_s is based on ranks. A value of $r_s = 1$ means that the two rankings are identical, $rs = -1$ means one ranking is the reverse of the other one, and $r_s = 0$ means the rankings have nothing in common. If we define the set of users to rank as $U = \{u_i\}$ and a ranking function $rg_a : U \to \mathbb{R}$, we state the formula more explicitly and adapted to our needs as

$$r_s = \frac{\frac{1}{|U|} \cdot \sum_{u \in U} (rg_a(u) - \bar{r})(rg_b(u) - \bar{r})}{\sqrt{\frac{1}{|U|} \cdot \sum_{u \in U} (rg_a(u) - \bar{r})^2} \cdot \sqrt{\frac{1}{|U|} \cdot \sum_{u \in U} (rg_b(u) - \bar{r})^2}} \quad \text{with} \quad \bar{r} = \frac{|U| + 1}{2}.$$

The ranking function $rg : U \to \mathbb{R}$ has to assign averaged ranks in the case of ties. For example, if two users have the same score value and are ranked second and third, both get the rank $\frac{3+2}{2} = 2.5$ assigned. We want to compare 7 different measures, that are mostly based on the graph discussed in Section 6.2.1:

post_count Naive approach counting the number of relevant posts per user.

continuous_discussion A user's degree in a user-user-graph projected from a user-discussion-graph.

continuous_discussion_weighted Takes edge weights into account. This means, if a user posted twice in a continuous discussion, the degree is increased.

reply_graph A user's degree in an unweighted, undirected reply graph over the relevant parts.

reply_graph_weighted A weighted, but undirected variant of the reply graph.

reply_graph_directed An unweighted, but directed variant of the reply graph.

reply_graph_directed_weighted A weighted and directed variant of the reply graph.

We want to compare these measures by calculating pairwise r_s values and placing them in a correlation matrix. We want to visualize the correlation matrix by plotting squares of a color spectrum. Then, we want to find the 50 most influential users according to a selected measure and generate pie charts showing their membership in (1) reference use clusters and (2) user behavior clusters. Because the charts do not show the overlap of the two role types, we want to show a contingency table in addition.

6.3 Implementation: Reuse Previous Implementation and Use Network Libraries

The following three subsections discuss implementations of the three aforementioned approaches. The implementations are discussed rather briefly, because they are often straightforward and/or reuse existing functionality. Unless denoted otherwise, all mentioned functions and classes reside in the module `user_analysis`.

6.3.1 Create and Draw Graphs

The construction of the two graphs is carried out by the class `ReferenceClusterAnalyzer`. On instantiation, the class requires an undirected `networkx` user-user-graph and users assigned to clusters. The class creates the two graphs, but the drawing is carried out by the function `plot_graph_with_selfloops(graph,file_name)` in the `plots` module. Because the drawing capabilities of `networkx` are very limited (for example no self loops), the traditional `graphviz` library[2] has to be used. Unfortunately, some layout adjustments are necessary, as well as manual node and edge labeling with image processing software.

6.3.2 Calculate Features and Reuse K-Means

Fortunately, the generic kernelized K-Means implementation found in the module `kernelized_k_means` can be reused. We extend it to support the calculation of the Dunn, modified Dunn, and Davies Bouldin indices. The module `user_analysis` contains the part specific to the clustering of users according to their behavior. Most importantly, the class `UserFeatureCalculator` implements the calculation of the features. It requires the data sources it uses in the constructor (including the full corpus, reduced corpus variants, and a facade to

[2] Available at `http://www.graphviz.org/`.

the relevance algorithm) and `calculate_features(user)` generates a dictionary containing a feature name to value map suitable for use with the dimension-value-map kernel. The class `UserBehaviorClusterer` makes use of the feature calculating class. It has the responsibility to select users, get the features, normalize them, and perform different clusterings. This includes repeated clusterings with $k = 1,\ldots,14$ and the definitive clustering with a chosen k. Plots showing the different indices over various values of k and the distribution of final cluster sizes are plotted by the class `UserBehaviorPlotCollection` from the module `plots`.

6.3.3 Compare Measures and Rank Users

The class `UserInfluenceAnalyzer` requires the corpus reduced to relevant parts, a corpus containing full partly relevant threads, and a relevance facade. From these, the class uses the class `GraphCreator` from module `graph_projections` to generate all required graphs. The method `spearmans_correlation_matrix` calculates and returns a correlation matrix. It does so by performing the following first for every measure:

1. Calculate the score for every user.
2. Sort the users descendingly by score value.
3. Map every user to a (possibly real-valued) rank while averaging ranks of ties.

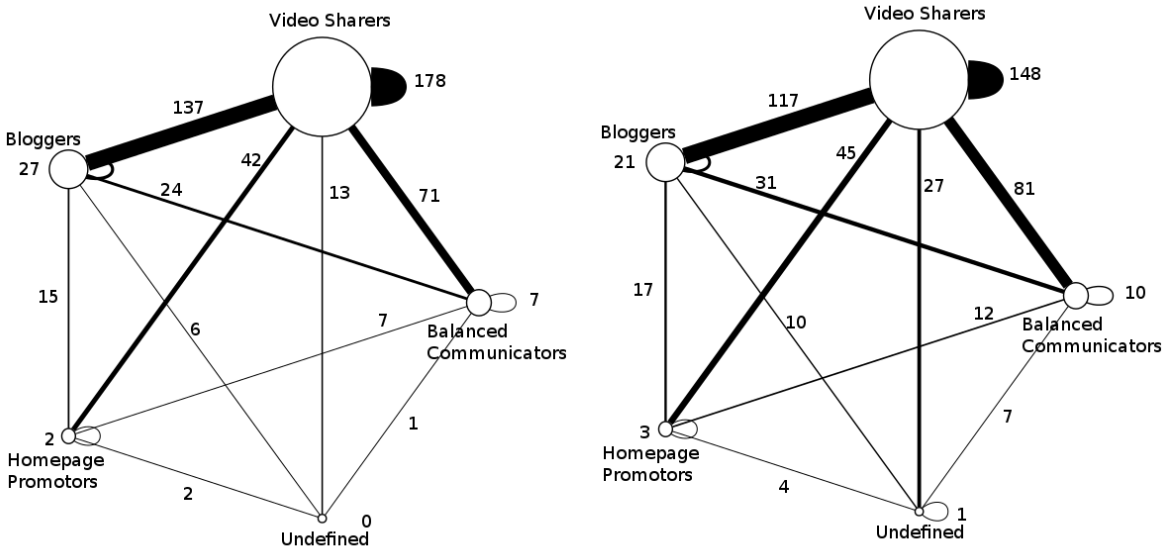
The r_s value between two rankings of two measures is then carried out by a direct implementation of the explicit formula shown in Section 6.2.4. The class `UserInfluencePlotCollection` from `plots` uses the matrix to plot the visualization. Fortunately, `matplotlib` already provides the required functionality under the name "pseudocolor plot". The plotting of the distribution of reference use and user behavior cluster membership among the 50 most influential users is performed by the same class. The calculation of role distribution and overlap is performed by functions from the `user_analysis` module that make use of Python's built-in set operations.

6.4 Results: User Behavior and Influence

The following three subsections discuss results stemming from the use of the aforementioned implementations. The results are interpreted regarding the research questions derived from the problem statement at the beginning of this chapter.

6.4.1 Weak Homophily in Reference Use

If we compare the observed situation shown on the left hand side of Figure 6.4 to the random scenario shown on the right hand side, we notice that the two graphs are similar. However,



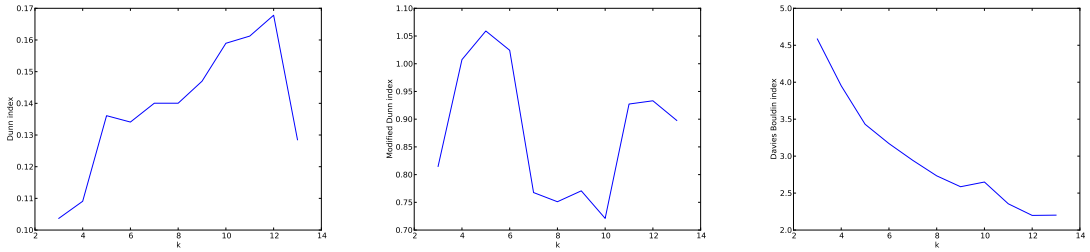
(a) The actual observed situation. *(b) A scenario with random communication.*

Figure 6.4: Reference use clusters and their relations.

the two largest reference use clusters do indeed show increased self loop weights compared to the random scenario. Video Sharers have their self loop weight increased by about 20% and Bloggers by about 29%. This observation supports the hypothesis of homophily, although only weakly. Note that the other three clusters show contradicting evidence. Here, the users seem to prefer talking to users of clusters other than their own. Because these clusters are very small, the significance of the observation is questionable. In summary, it is not entirely clear whether homophily in reference use exists among the users, although we have identified weak signs. Further work is encouraged to deduct a formal hypothesis test based on the data presented here.

6.4.2 Six User Roles

Figure 6.5 shows three internal clustering evaluation metrics for $k = 3, \ldots, 13$. The different indices vary to a great extent regarding their suggestion of an optimal k. The original Dunn index shown on the left hand side strongly suggests the use of a $k \geq 5$ and describes $k = 12$ as the optimal value for the clustering. The modified Dunn index shown in the center suggests the use of $k = 5$, but makes $k = 4$ and $k = 6$ also look acceptable. The Davies Bouldin index shown on the right hand side, which says that smaller index values are better, decreases exponentially with an increase of k. The index does not help in finding



(a) Dunn index. *(b) Modified Dunn index.* *(c) Davies-Bouldin index.*

Figure 6.5: Internal evaluation metrics for different values of k. The original Dunn index shown in (a) uses complete linkage for intra-cluster similarity and single linkage for inter-cluster similarity. The modified version shown in (b) uses average linkage in both cases.

the right k in this case. We also look at the cluster sizes resulting from different values of k. We observe that with $k > 6$ clusters with size 1 are generated. Given this observation and the supporting evidence of the Dunn indices, we decide to use $k = 6$ for the following user clustering.

Using $k = 6$, we obtain 6 clusters with various sizes. Each of these clusters ideally describes a pattern of user behavior, which is a role for short. In order to describe these patterns qualitatively and give names to the roles, we look at the individuals of each cluster and, more importantly, at the summary in Table 6.1. Doing so, we obtain the following role descriptions:

Sophisticated Contributors ($|c_0| = 4$): These users put a lot of effort into their contributions. Their posts are typically 3 times as long as the average post and contain up to 5 times more references. The number of contributions per day varies greatly within the small cluster. One of the members is dedicated to the topic of CCSVI, while the others are not. Surprisingly, despite the effort they put into their contributions, they are not cited more often than other users[3]. From the perspective of information exchange, the users play key roles in the forum.

Short-Lived CCSVI Spammers ($|c_1| = 4$): These users were active for very few days but posted an average of 5.7 messages per day. During their short but intense contribution period, their messages were much shorter than average and contained nearly no references. The vast majority of the posts were about CCSVI, but the users did not initiate any threads. With these few posts, their efficiency in covering other users in CCSVI-related discussions is extremely high.

[3]Future work is thus encouraged to find out, under which circumstances users use the citation function and what the intended meaning is.

Cluster #		0	1	2	3	4	5
avg_msg_length	μ	1293.9	271.0	448.0	496.0	510.8	380.5
	σ	387.7	097.7	226.6	269.0	338.4	168.6
avg_posts_per_day	μ	0.301	5.667	0.593	0.246	3.971	0.560
	σ	0.272	1.958	0.580	0.222	1.907	0.537
avg_refs_per_post	μ	0.917	0.050	0.185	0.150	0.202	0.533
	σ	0.280	0.087	0.178	0.141	0.159	0.254
avg_threads_per_day	μ	0.061	0.958	0.239	0.147	1.339	0.255
	σ	0.038	0.298	0.201	0.122	0.418	0.217
days_active	μ	467.0	001.8	793.1	292.0	554.1	515.2
	σ	325.1	000.8	427.5	293.0	441.2	447.1
fraction_of_posts_cited	μ	0.131	0.330	0.149	0.210	0.202	0.164
	σ	0.111	0.196	0.103	0.146	0.059	0.150
fraction_relevant_posts	μ	0.264	0.839	0.104	0.534	0.099	0.379
	σ	0.213	0.278	0.092	0.156	0.069	0.159
fraction_threads_initiated	μ	0.309	0.000	0.117	0.089	0.064	0.458
	σ	0.257	0.000	0.089	0.097	0.108	0.130
relevant_user_coverage	μ	0.351	1.078	0.178	1.143	0.061	0.500
	σ	0.236	0.517	0.168	0.765	0.074	0.377

Table 6.1: Mean and standard deviation of the features shown for all of the six clusters.

Average Users ($|c_2| = 108$): These users are "average" in a way, there is nothing specifically outstanding about them. This, however, does not mean that the users are very similar to each other. The average user makes a post every other day and includes a reference in 18.5% of the posts. Only about 10% of the posts are dedicated to the topic of CCSVI.

CCSVI Focused Responders ($|c_3| = 28$): These users were active for less than a year and during their rather short contribution period, they did not engage in many threads nor did they make many posts per day. Interestingly, most of their posts are about CCSVI, but the users initiated threads very rarely. They appear to wait for CCSVI discussions to come up and then contribute their opinions. This strategy is very efficient regarding their CCSVI-related opinion spread.

Highly Active Relational Posters ($|c_4| = 10$): These users are highly active in the sense that they contribute a lot of posts (4 per day!) to a lot of different threads. However, they very rarely initiate a thread. They do not show a focus on CCSVI, but take part in all kinds of discussions. This group of users is very important from a community building point of view, as these people are expected to have personal

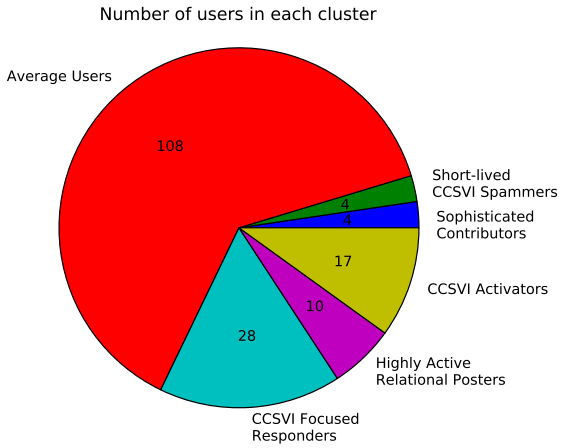


Figure 6.6: Pie chart showing the number of users assigned to each role.

relationships with other users. A lot of personal small talk is prevalent in the forum (identified manually) and it is assumed that a substantial amount can be attributed to these users.

CCSVI Activators ($|c_5| = 17$): These users play a crucial role in fueling the discussion about CCSVI. They initiate a lot of threads, mostly about CCSVI, and provide about 3 times as many references as average users. Their level of general activity is average and they reach a decent amount of users with the discussions they fuel, although not as many as the CCSVI Focused Responders. The messages they post, are shorter than on average. Manual analysis of the cluster members showed that 2 out of the 17 members have been banned, which happens rarely in the forum at hand.

The obtained clusters have very different sizes as illustrated by the pie chart shown in Figure 6.6. Note that most of the users can only be described as "average", which is similar to the situation observed in [9]. The characteristics of these users provide a baseline for comparison with the other user roles. Yet, we do not know whether the proposed solution is not specific enough to differentiate the average users even further, or if most users do indeed not stand out. The reasons for this conformity may be of sociological nature and may not be directly observable in the data. However, nearly half of the users were assigned more detailed roles making the results valuable.

6.4.3 Measure Correlation and Roles of Most Influential Users

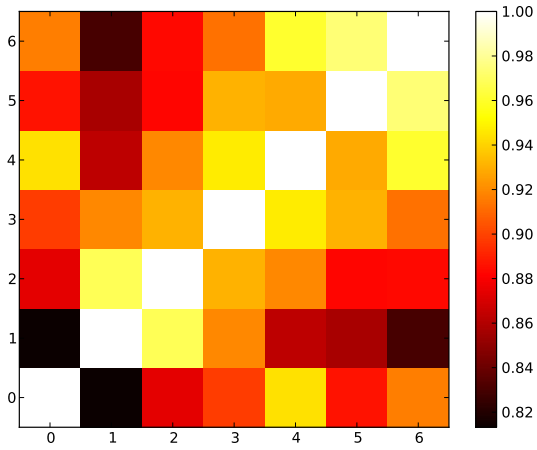
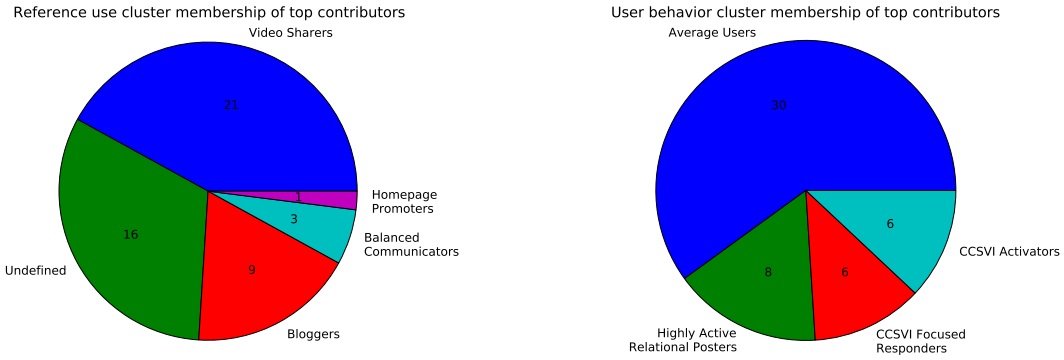


Figure 6.7: Spearman's rank correlation coefficient of the user influence measures.
0: post_count
1: continuous_discussion
2: continuous_discussion_weighted
3: reply_graph
4: reply_graph_weighted
5: reply_graph_directed
6: reply_graph_directed_weighted

Examining the correlation matrix shown in Figure 6.7 yields several insights. Firstly, all influence measures are correlated with at least $r_s = 0.82$. This quite high correlation suggests that the question of what measure is the most appropriate one, has little impact on the resulting ranking. Secondly, the measures highly correlate with the simple post counting approach. Especially weighted variants of the reply graph are very similar to the results produced by post counting. The continuous discussion based approaches have the lowest correlation with all other measures, which can be attributed to the very different assumptions about user behavior. Finally, the weighted, but undirected, reply graph shows the highest correlation with all other approaches. Thus, the approach seems to be a good compromise with respect to the different assumptions. We decide to use this measure for the identification of the 50 most influential users.

Figure 6.8 shows what reference use patterns the 50 most influential users expose and what user roles they play. We notice that, unfortunately, 16 users exhibit undefined reference use patterns. The reason is, that they did not post enough references and thus could not be included in the clustering by reference use. It is also evident, that Bloggers are



(a) Reference use. *(b) User behavior.*

Figure 6.8: Cluster membership among the 50 most influential users according to the weighted undirected reply graph ranking.

overrepresented among the 50 most influential users while Homepage Promoters are underrepresented. The user role "Sophisticated Contributor" is not present among the users while the Highly Active Relational Posters are overrepresented. Table 6.2 is a contingency table of the two role types and thus provides are more detailed overview on how the roles overlap. Interestingly, the two behavior roles associated with the intense discussion of CCSVI show a large overlap with Bloggers. Because blogs are often regarded an opinion based medium, this might be an indicator of an opinion based core influence in the forum.

	CCSVI Focused Responders	Average Users	CCSVI Activators	Highly Active Relational Posters	Sum
Balanced Communicators	0	2	1	0	3
Bloggers	3	2	3	1	9
Homepage Promoters	0	1	0	0	1
Undefined	2	10	1	3	16
Video Sharers	1	15	1	4	21
Sum	6	30	6	8	50

Table 6.2: Contingency table of the cluster memberships of the 50 most influential users.

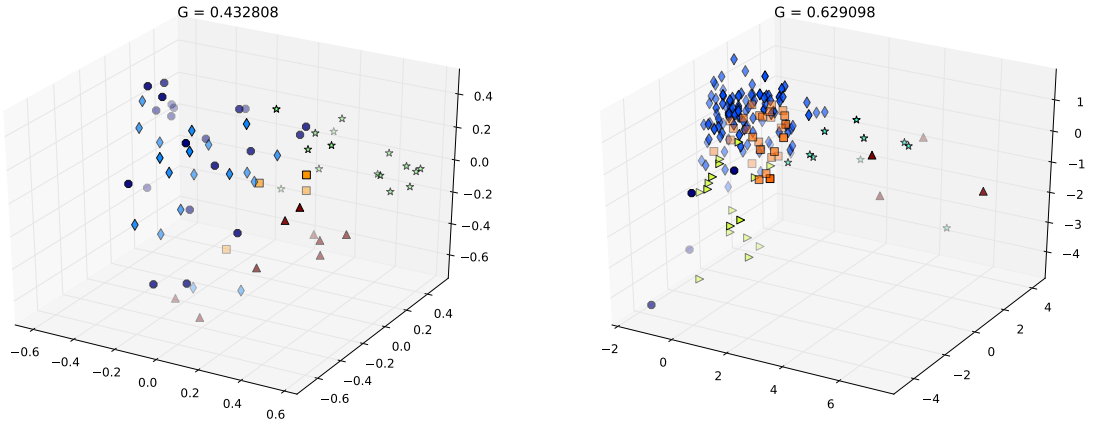
7 Threats to Validity

It is important to note that several aspects of the presented work threaten the validity of the discussed results. The most serious threat is inaccuracy in information retrieval. The algorithm we developed clearly outperforms the naive keywords based approach, but it only achieves an average MCC of 0.699 in the 10-Fold Cross Validation on annotated posts. This implies that there is still a number of posts in the corpus that are relevant, but do not get recognized as relevant by the algorithm (False Negatives). Furthermore, we can expect several irrelevant posts to be classified as relevant by the algorithm (False Positives). This inaccuracy has a great impact on the work, because the presented analyses are either based on the posts classified as relevant alone or compare the relevant part with the whole corpus. The results may not only be incomplete, but they may also be biased, because the information retrieval algorithm likely prefers more explicit posts over implicit ones.

The analysis about reference use over time is threatened by the reduction of URLs to their domain part. We made the simplifying assumption, that both `www.example.com/a` and `www.example.com/b` point to similar content. However (and especially in the case of social media), the content of these websites is user-provided and thus very different in nature. One YouTube video may be a scientific debate while another one may be a patient experience report. Hence, this simplification may falsify parts of the results.

The analysis about the delay in scientific publications also faces threats. We assumed that the list of CCSVI publications provided in [29] is complete, which is not necessarily the case. While fetching the URLs, we also noted that many of them were not reachable anymore. Some of them were no permalinks and thus pointed to content that changes over time. Therefore, we cannot know what kind of content was originally referenced.

When we assigned roles to users based on reference use and general behavior, we faced the problem of very small population sizes (68 and 171, respectively). The clustering approaches can be criticized as they rely on several assumptions. The most fundamental assumption is, that distinct patterns in reference use and user behavior exist among the users. We further assumed that the features we defined are capable of describing these patterns. The features we used are either calculated by averaging over the whole corpus or over relevant parts. We do not account for the fact, that users may expose different behavior in different situations. For example, messages in joke telling threads are expected to be shorter than messages in more serious threads. Finally, we assumed that these patterns can be represented by spherical forms in the constructed hyperspace. Because clustering is an exploratory data analysis by definition, there is no objective way of saying how successful the clustering



(a) Users by reference use. *(b) Users by general behavior.*

Figure 7.1: 3-dimensional MDS visualizations of the individuals in the hyperspace. Each point represents a user. The corresponding symbol and color represent cluster membership.

was. We merely present one possible interpretation of the data. However, it is evident that the data points in the hyperspace are not easily separable. This is supported by the contradicting evaluation metrics shown in Section 6.4.2 and is further illustrated by the Metric Multidimensional Scaling (see Section 2.2.3) representations shown in Figure 7.1. It is evident, that users assigned to different clusters are often very close to each other in the constructed low dimensional space. However, we have to keep in mind that the shown 3-dimensional space only captures 43.3% and 62.9% of the variance of the two discussed feature spaces. In any case, people's behavior is expected to be multifaceted and we did our best to provide a meaningful interpretation.

8 Conclusions and Outlook

In a concluding interpretation of the entire presented work, we have to keep in mind two important aspects. The first aspect is the existence of threats to validity, that have been discussed in the previous chapter. The second aspect is, that the analyzed forum discussion is just one possible example of a CCSVI-related debate among laypersons. Although the forum is open to everyone, it is expected to attract a certain target audience. Thus, the obtained results cannot be generalized to the nature of debate among laypersons.

Keeping these limitations in mind, we can contribute a deeper understanding of online fora and especially controversial debates therein. We showed that structural features of a forum can be utilized to increase the success of information retrieval. Regarding the discussion about CCSVI, we made several observations. We observed, that the topic caught on quickly after the initial publication, but interest declined since the beginning of 2012. In the debate, social media was referenced more often than any other type of resource. The dominance of social media is a general trend in the observed forum. However, in the CCSVI-related debate, social media (especially video sharing websites, but also other fora and blogs), as well as personal homepages, are much more important than in a typical discussion in such a forum. Scientific publications, on the other hand, were referenced only very rarely. We found only seven different CCSVI-related publications that were discussed by the forum users. They were brought quickly to the forum, but they were referenced only one or two times. In conclusion, it is evident that strictly factual sources were not used a lot. Instead, we found sources that we assume to be mostly opinionated.

We were able to describe user behavior in detail. More specifically, we identified four distinct patterns of reference use. The results seemed plausible, because different identities assumed to belong to the same person often showed up in the same group. We observed a weak tendency of users to discuss with people who use similar kinds of references. We also identified six behavior induced user roles. About half of the analyzed users did not stand out and could only be described as average. However, we also identified two sets of users that were especially valuable to the community (the Sophisticated Contributors and the Highly Active Relational Posters). Two other user roles (the CCSVI Focused Responders and the CCSVI Activators) were responsible for fueling the debate on CCSVI. We noticed that several approaches to the assessment of user influence were highly correlated in practice. The users associated with the CCSVI-fueling roles all showed up among the fifty most influential users and showed a strong overlap with the reference use group Bloggers. We can thus expect a certain core influence on the CCSVI debate to stem from these people.

Future work is encouraged to address several unsolved problems. The usefulness of structural forum data in Information Retrieval is evident from this work, but our algorithm is open to improvement. In this context, it is still unknown why citations seem to play such a little role and under which circumstances users cite other posts. More sophisticated algorithms could be developed that take into account dynamic user behavior. Dynamic change of user behavior patterns is also of interest in the context of the user roles we assigned. It is still unknown whether these user roles are stable or change under certain circumstances. Besides that, many findings presented in this work lack a sociological explanation. We do not know, for example, why social media is the dominant type of resource. We also do not know which external events caused the fluctuations in discussion volume. It also remains unanswered why scientific publications were referenced rarely and why users show the behavioral patterns we identified.

The used methods and implementation can be applied to a different extracted forum and a different, possibly unrelated, topic. The implementation is generic and thus only a few manual definitions and labels are required in order to analyze different material. The same analysis steps carried out on a different corpus can be used to verify the results obtained in this work and provide further insight into the questions whether the results generalize to a certain extent.

Abbreviations and Acronyms

XML Extensible Markup Language

DOM Document Object Model

HTML Hypertext Markup Language

DMSG Deutsche Multiple Sklerose Gesellschaft (Engl.: German MS Society)

MCC Matthews Correlation Coefficient

MDS Multidimensional Scaling

SVD Singular Value Decomposition

EA Evolutionary Algorithm

PDF Portable Document Format

SNA Social Network Analysis

CCSVI Chronic Cerebrospinal Venous Insufficiency

MS Multiple Sclerosis

CSV Character Separated Values

List of Figures

3.1 Screenshot of the top of a DMSG forum thread. The first post shows a reference to another website, the second post cites the first one. 18
3.2 UML class diagram of the data access layer providing an abstraction from the underlying XML. Note that several methods are left out, because they are not required to understand the design principle. 21

4.1 Thread size (measured in # of posts) distribution of threads containing at least one keyword. 25
4.2 Fraction of keyword-containing posts per thread. If, for example, a thread has 4 posts and 1 of them contains a keyword, the fraction of keyword-containing posts for this thread is 0.25. This histogram shows the mentioned fraction for every thread in the corpus. 26
4.3 Relative position of keyword-containing posts in their containing threads. Relative position is defined by post_index/thread_length. A value of 0 means the post is the first one in the thread, a value of 1 means it is the last one. . . 27
4.4 Intuitive graph-based visualization of an example thread. The circular nodes denote posts, the squared nodes denote features contributing relevance to the posts. Posts also inherit fractions of relevance scores from other posts by following them or citing them. 29
4.5 UML class diagram showing the transformation of threads into a more efficient representation with respect to the Evolutionary Algorithm. 31

5.1 Class diagram of the kernelized K-Means implementation. The plus (+) symbol indicates public attributes and the minus (-) sign private ones. The structure is a simple three level aggregation that is intended to cache constant intermediate results. 44
5.2 Primary domain class occurrence of the relevant parts. The occurrences are aggregated for each month. 47
5.3 Primary domain class occurrence of the full corpus. The black area indicates not classified domains. These domains may belong to any of the mentioned classes. 48

5.4 Secondary domain class occurrence of the relevant parts. Note that all secondary classes of a single primary class share the same shade of color. Example: All social classes are depicted in a shade of blue. 49
5.5 Pie charts showing the relative amount of occurrences of the primary domain classes of the top 15 domains. The whole (unfiltered) corpus is on the left hand side while the relevant parts are on the right hand side. 50
5.6 Scientific Publications referenced by forum users. Each publication has a separate area. The star indicates the date of publication, whereas the diamonds indicate points in time, when the publication was referenced in a forum post. A dark green diamond indicates a direct reference to the publication, whereas a light green diamond indicates an indirect reference. All of these areas share the same x-axis. 54

6.1 Illustration of the two graph creation approaches. The exemplary input is shown on the left. The center shows the classical reply graph resulting from the input. On the right, the user-user-graph, obtained from projecting a user-discussion-graph onto the users, is shown. 57
6.2 Histograms of unweighted user degrees. These figures are computed over relevant corpus parts only. 58
6.3 Probability tree for a model with two reference use clusters. 59
6.4 Reference use clusters and their relations. 66
6.5 Internal evaluation metrics for different values of k. The original Dunn index shown in (a) uses complete linkage for intra-cluster similarity and single linkage for inter-cluster similarity. The modified version shown in (b) uses average linkage in both cases. 67
6.6 Pie chart showing the number of users assigned to each role. 69
6.7 Spearman's rank correlation coefficient of the user influence measures. 0: post_count 1: continuous_discussion 2: continuous_discussion_weighted 3: reply_graph 4: reply_graph_weighted 5: reply_graph_directed 6: reply_graph_directed_weighted 70
6.8 Cluster membership among the 50 most influential users according to the weighted undirected reply graph ranking. 71

7.1 3-dimensional MDS visualizations of the individuals in the hyperspace. Each point represents a user. The corresponding symbol and color represent cluster membership. 74

List of Tables

2.1 A confusion matrix showing all possible combinations of prediction and reality in binary classification. 4

3.1 Size of the extracted corpus. 22

4.1 Occurrences of manually defined search terms in the corpus. 24
4.2 Intended content of the four largest threads. 25
4.3 Results of the 10-Fold Cross Validation using the tk-model. 34
4.4 Results of the 10-Fold Cross Validation using the tkcf-model. 34
4.5 Results of the 10-Fold Cross Validation using the tkcfu-model. 34

5.1 The 15 most cited domains in the full corpus (left side) and the relevant parts (right side). 'Opp. rank' indicates the rank of the domain on the opposite side. Example: `ms-forum-weihe.de` is on rank 5 in the full corpus. 46
5.2 User clusters calculated from reference use with non-clean users. Note that `(user)` and `user` tend to appear in the same cluster. 51
5.3 User clusters calculated from reference use with cleaned users. 52
5.4 Total domain class occurrences of the top five domains of each clusters. The numbers stem from an aggregation over all the users from within a cluster. . 52

6.1 Mean and standard deviation of the features shown for all of the six clusters. 68
6.2 Contingency table of the cluster memberships of the 50 most influential users. 71

Bibliography

[1] E. Alba and C. Cotta. Evolutionary algorithms. *Handbook of Bioinspired Algorithms and Applications. Boca Raton: Chapman and Hall/CRC*, 2006.

[2] E. Alpaydin. *Introduction To Machine Learning.* MIT Press, Cambridge, Massachusetts, 2004.

[3] V. Barash, M. Smith, L. Getoor, and H. T. Welser. Distinguishing knowledge vs social capital in social media with roles and context. *Proceedings of the ICWSM*, 9, 2009.

[4] M. J. Bommarito II, D. M. Katz, J. L. Zelner, and J. H. Fowler. Distance measures for dynamic citation networks. *Physica A: Statistical Mechanics and its Applications*, 389(19):4201–4208, 2010.

[5] A. J. Brush, X. Wang, T. C. Turner, and M. A. Smith. Assessing differential usage of usenet social accounting meta-data. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, page 889–898, 2005.

[6] R. Cai, J.-M. Yang, W. Lai, Y. Wang, and L. Zhang. iRobot: an intelligent crawler for web forums. In *Proceeding of the 17th international conference on World Wide Web*, WWW '08, page 447–456, Beijing, China, 2008. ACM. ACM ID: 1367558.

[7] O. Carugo. Detailed estimation of bioinformatics prediction reliability through the fragmented prediction performance plots. *BMC Bioinformatics*, 8:380, Oct. 2007. PMID: 17931407 PMCID: PMC2148069.

[8] D. Centola. An experimental study of homophily in the adoption of health behavior. *Science*, 334(6060):1269–1272, 2011.

[9] J. Chan, C. Hayes, and E. M. Daly. Decomposing discussion forums and boards using user roles. In *AAAI Conference on Weblogs and Social Media*, page 215–218, 2010.

[10] T. Chomutare, E. Arsand, L. Fernandez-Luque, J. Lauritzen, G. Hartvigsen, et al. Inferring community structure in healthcare forums. an empirical study. *Methods of information in medicine*, 52(2), 2013.

[11] J. Couto. Kernel k-means for categorical data. *Advances in Intelligent Data Analysis VI - 6th international Symposium on Intelligent Data Analysis, IDA 2005 Proceedings*, pages 46–56, 2005.

[12] D. L. Davies and D. W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (2):224–227, 1979.

[13] F. Ding, Y. Liu, H. Cheng, F. Xiong, X.-m. Si, and B. Shen. Read and reply behaviors in a BBS social network. In *2010 2nd International Conference on Advanced Computer Control (ICACC)*, volume 4, page 571–576. IEEE, Mar. 2010.

[14] J. C. Dunn. A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, 3(3):32–57, 1973.

[15] L. Fahrmeir, A. Hamerle, and G. Tutz. *Multivariate statistische Verfahren*. Walter de Gruyter, Berlin, 2nd edition, 1996.

[16] D. Fallis. Toward an epistemology of wikipedia. *Journal of the American Society for Information Science and Technology*, 59(10):1662–1674, 2008.

[17] D. Fisher, M. Smith, and H. T. Welser. You are who you talk to: Detecting roles in usenet newsgroups. In *Proceedings of the 39th Annual Hawaii International Conference on System Sciences. HICSS'06.*, volume 3, 2006.

[18] T. Georgiou, M. Karvounis, and Y. Ioannidis. Extracting topics of debate between users on web discussion boards. In *Proceedings of the 1st International Conference for Undergraduate and Postgraduate Students in Computer Engineering, Informatics, related Technologies and Applications (EUREKA 2010)*, Patras, 2010.

[19] J. Handl, J. Knowles, and D. B. Kell. Computational cluster validation in post-genomic data analysis. *Bioinformatics*, 21(15):3201–3212, 2005.

[20] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, Jan. 2009.

[21] K. A. D. Jong. *Evolutionary computation: a unified approach*. MIT Press, Cambridge, Massachusetts, Feb. 2006.

[22] J.-H. Kang and J. Kim. Analyzing answers in threaded discussions using a role-based information network. In *IEEE third international conference on privacy, security, risk and trust (PASSAT) and 2011 IEEE third international conference on social computing (SOCIALCOM)*, page 111–117, 2011.

[23] W. J. Krzanowski. *Principles of multivariate analysis: A user's perspective*. Oxford University Press, New York, 1988.

[24] J. M. Lattin, J. D. Carroll, and P. E. Green. *Analyzing multivariate data*. Thomson Brooks/Cole, Belmont, CA, 2003.

[25] P. F. Lazarsfeld and R. K. Merton. Friendship as a social process: A substantive and methodological analysis. In M. Berger and T. Abel, editors, *Freedom and control in modern society*. Van Nostrand, New York, 1954.

[26] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297, 1967.

[27] B. W. Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et biophysica acta*, 405(2):442, 1975.

[28] V. Nannen, S. Smit, and A. Eiben. Costs and benefits of tuning parameters of evolutionary algorithms. *Parallel Problem Solving from Nature–PPSN X*, page 528–538, 2008.

[29] F. Philipp. Dynamic citation network for CCSVI. Technical report, University of Göttingen, Faculty of Mathematics and Computer Science, Göttingen, 2012.

[30] C. J. V. Rijsbergen. *Information Retrieval*. Butterworth-Heinemann, Newton, MA, USA, 2nd edition, 1979.

[31] X. Shi, J. Zhu, R. Cai, and L. Zhang. User grouping behavior in online forums. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, page 777–786, Paris, France, 2009. ACM. ACM ID: 1557105.

[32] C. Spearman. The proof and measurement of association between two things. *The American journal of psychology*, 15(1):72–101, 1904.

[33] W. S. Torgerson. Multidimensional scaling: I. theory and method. *Psychometrika*, 17(4):401–419, 1952.

[34] W. S. Torgerson. *Theory and methods of scaling*. Wiley, New York, 1958.

[35] M. Tsvetovat and A. Kouznetsov. *Social Network Analysis for Startups: Finding Connections on the Social Web*. Oreilly & Assoc Inc, Oct. 2011.

[36] T. C. Turner, M. A. Smith, D. Fisher, and H. T. Welser. Picturing usenet: Mapping computer-mediated collective action. *Journal of Computer-Mediated Communication*, 10(4), 2005.

[37] N. K. Visalakshi and K. Thangavel. Impact of normalization in distributed k-means clustering. *International Journal of Soft Computing*, 4(4):168–172, 2009.

[38] S. V. N. Vishwanathan and N. M. Murty. Kernel enabled k-means algorithm. `http://eprints.iisc.ernet.in/archive/00000010`, 2002.

[39] F. B. Viégas and M. Smith. Newsgroup crowds and authorlines: Visualizing the activity of individuals in conversational cyberspaces. In *Proceedings of the 37th Annual Hawaii International Conference on System Sciences*, page 10–pp, 2004.

[40] Y. Wang, J.-M. Yang, W. Lai, R. Cai, L. Zhang, and W.-Y. Ma. Exploring traversal strategy for web forum crawling. In *Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval*, SIGIR '08, page 459–466, Singapore, 2008. ACM. ACM ID: 1390413.

[41] H. T. Welser, D. Cosley, G. Kossinets, A. Lin, F. Dokshin, G. Gay, and M. Smith. Finding social roles in wikipedia. In *Proceedings of the 2011 iConference*, page 122–129, 2011.

[42] H. T. Welser, E. Gleave, D. Fisher, and M. Smith. Visualizing the signatures of social roles in online discussion groups. *Journal of Social Structure*, 8(2):564–586, 2007.

[43] J.-M. Yang, R. Cai, Y. Wang, J. Zhu, L. Zhang, and W.-Y. Ma. Incorporating site-level knowledge to extract structured data from web forums. In *Proceedings of the 18th international conference on World wide web*, WWW '09, page 181–190, Madrid, Spain, 2009. ACM. ACM ID: 1526735.

[44] P. Zamboni, R. Galeotti, E. Menegatti, A. M. Malagoni, G. Tacconi, S. Dall'Ara, I. Bartolomei, and F. Salvi. Chronic cerebrospinal venous insufficiency in patients with multiple sclerosis. *Journal of Neurology, Neurosurgery & Psychiatry*, 80(4):392–399, 2009.

[45] K. Zhongbao and Z. Changshui. Reply networks on a bulletin board system. *Physical Review E*, 67(3):036117, 2003.

[46] M. Zhu, W. Hu, and O. Wu. Topic detection and tracking for threaded discussion communities. In *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, WI-IAT'08.*, volume 1, page 77–83, 2008.

All URLs have been verified on April 26, 2013.