

Double network-based method for identifying behavioral patterns in clickstream data

Jesper Bruun^{1*}, Pia Ray Jensen², Linda Udbyt²

¹⁾ University of Copenhagen, Department of Science Education, ²⁾ University of Copenhagen, Niels Bohr Institute
Corresponding Author: jbruun@ind.ku.dk

Keywords: Wiki textbook, Principal component analysis, Clustering, Community detection, Action network, Similarity network

ABSTRACT

We describe a method for finding structural patterns in online user behavior that uses network analyses in two ways. The method relies only on structural network measures for characterization and clustering user sessions. We illustrate the method through by analyzing a collection of about 50.000 interactions drawn over a period of 3 years. This illustrative example pertains to physics students' interactions with an online wiki-formatted textbook on neutron scattering.

- The method is different from e.g. hierarchical and k-means clustering in that it highlights relational structures between user sessions. It also naturally considers overlapping clusters.
- The method produces visualizations – maps – that highlights connections between clusters can be used to guide further analyses.
- The method is likely usable in general for analysis of any time record of connected discrete events.

Method details

Overview

This paper uses an illustrative example to describe a method we have developed to extract structural behaviors of students who interact with a wiki formatted online textbook on Neutron Scattering (Authors & others, 2013). By structural, we mean that it is based on the order in which users perform actions, not on the meaning of those actions. The method is based on clickstream data obtained as a database dump via web-analytics software (Adams, 2012). For the particular dataset we use for illustrative purposes, we do not have user details. However, these would be straightforward to integrate in the method. The method operates with two levels of networks (1) the level of the individual session, called *session networks* or action networks (Author, 2016) and (2) the level of similarity between individual sessions. The structure of the paper is as follows. First, we provide the context from which our example data was produced. Then we provide definitions, which are used throughout the paper. This is followed by a brief overview of the method. The remainder of the paper is an illustrative step-by-step example of how to use the method to extract clusters of similar online behavior. Unless otherwise noted, we have used the R language and environment for statistical computing (R Core Team, 2017) with packages psych (Revelle, 2017) and igraph (Csardi & Nepusz, 2006). Throughout the paper, we provide references to supplementary R-scripts as well as illustrative figures.

Context

We argue that the presented method is usable for clickstream data in general – and indeed any time record of discrete events. For ease of reading, we here provide a brief outline of the wiki-style online textbook that users interacted with to produce the data.

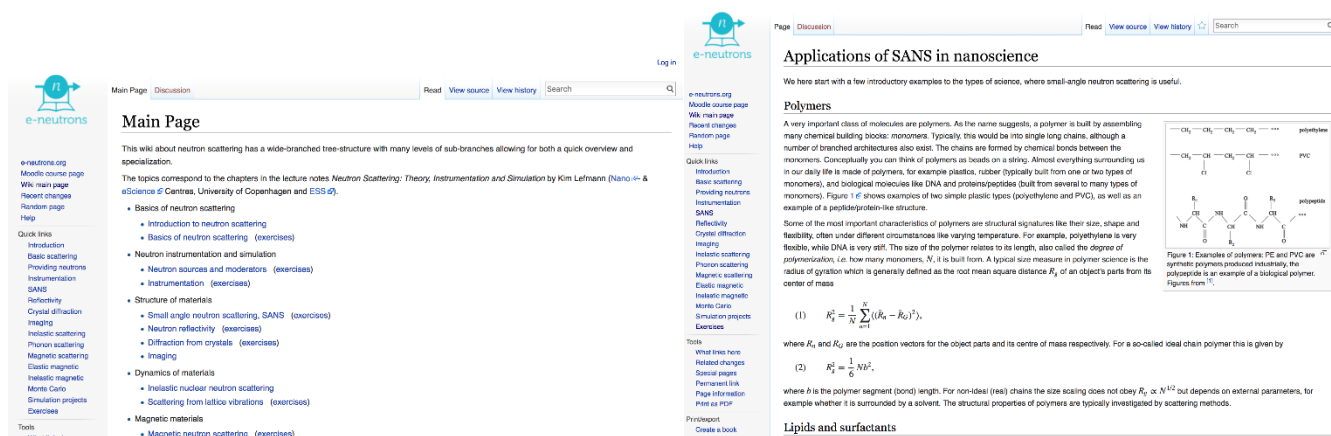


Figure 1 Screen shots showing the main page of the wiki-textbook (left) and an example of text in the wiki text book (right).

The wiki-textbook follows a tree-structure organization that mirrors a textbook. A menu on the main page of the wiki-textbook lists all text chapters as well as a separate chapter for problems (called exercises). See Figure 1.

Problem:Neutron velocity selector

A neutron velocity selector is a drum that spins around an axis parallel to the beam. This axis lies below the guide. From the drum, a series of absorbing neutron blades sticks out radially. The ends of the drum are twisted with respect to each other. A principal sketch is shown in [velocity selector figure](#) on the [Instrumentation](#) page.

This effect of the selector is that only neutrons around a certain velocity (or wavelength) can pass through.

Assume a selector length of $L = 0.25$ m, $n = 68$ blades, and a twisting angle, $\phi = 48.3^\circ$, as for the selector at the SANS-2 instrument at PSI. Calculate the rotation speed you should use to select 10 Å neutrons.

Hint	[show]
Hint	[show]
Solution	[show]

Problem:Neutron velocity selector

A neutron velocity selector is a drum that spins around an axis parallel to the beam. This axis lies below the guide. From the drum, a series of absorbing neutron blades sticks out radially. The ends of the drum are twisted with respect to each other. A principal sketch is shown in [velocity selector figure](#) on the [Instrumentation](#) page.

This effect of the selector is that only neutrons around a certain velocity (or wavelength) can pass through it.

Assume a selector length of $L = 0.25$ m, $n = 68$ blades, and a twisting angle, $\phi = 48.3^\circ$, as for the selector at the SANS-2 instrument at PSI. Calculate the rotation speed you should use to select 10 Å neutrons.

Hint	[hide]
Consider how long time it will take the neutron to travel through the velocity selector.	
Hint	[hide]
What should the angular velocity ω be, when twisting 48.3° during the same period of time it takes the neutron to travel through the velocity selector?	
Solution	[hide]
The velocity, v , of neutron with $\lambda = 10$ Å is	
$v = \frac{h}{m\lambda}$	
and the time it takes the neutron to travel through the drum of the velocity selector, is	
$\Delta t = \frac{L}{v} = 6.34 \cdot 10^{-8}$	
Thereby the angular velocity, ω , is	
$\omega = \frac{\phi}{\Delta t} = 1333.9 \frac{\text{rad}}{\text{s}} = 12738 \text{ rpm}$	

Figure 2. An example problem with hints and solutions hidden (left) and the same problem with hints and solutions shown (right).

The wiki-textbook also contains a number of problems (called exercises in the wiki-textbook) with associated hints and solutions. In this example, users' interaction by clicking show/hid hint/solution plays an important part. See Figure 2 for an example.

Technically, hints and solutions are implemented with several extensions to the MediaWiki engine (MediaWiki, 2019), specifically ShowHide, MathJax and CrossReference (see MediaWiki documentation), whereof the last two are related to the contents rather than the functionality. See Figure 2. We have used the MediaWiki engine to create the wiki-textbook, and make use of multiple extensions and functionalities. Specifically, ShowHide, MathJax, and CrossReference. The wiki-textbook is part of a larger on-line package of teaching materials for neutron scattering, which may be accessed at e-neutrons.org. The software as well as access to the web site is free.

Definitions

An action. We define an action as an event that is recorded by the server and placed in the clickstream server. An action can be identified in our dataset file as a row in the database dump.

A session. A collection of events, which are tied together by a common identification code, called the Session ID. For each user action, a number of identifiers are recorded (see *Creating sessions* below). One of those identifiers is a time stamp. This is the number of seconds from some starting time, t_0 , 1 January 1970. In identifying the duration of sessions, the first time stamp, t_{start} , was subtracted from the last time stamp, t_{end} , to calculate the duration of the session in seconds, $t_{dur} = t_{end} - t_{start}$. We infer that all unique sessions involve the use of one computer. However, they can extend through a long period and can involve more than one user.

Session network. We define a node as an action, A_i , that occurs on a webpage, for example, when a user clicks a particular link on a particular webpage. The server log records each such action with a timestamp, t_i . A link, L_{ij}^k , is created between two actions, A_i and A_j , if $t_j > t_i$ in a session. The time between actions is then, $\Delta t_{ij}^k = t_j^k - t_i^k$. A link between two actions can occur multiple times, which is represented by the index k .

Similarity network. At this level, each node represents a session network, and links represent similarity. As will be shown later, the methodology uses Euclidean distances in a space spanned by rotated principal components of structural network measures as a basis for similarity.

Outline of method

Starting from a table (e.g. a comma separated file) with each action recorded with a timestamp and a Session ID, the method has the following steps.

- Sessions
 - Create sessions; a set of events are grouped together by a joint Session ID and ordered by timestamp.
 - Time class classification; session durations are classified on the basis of kernel density estimate quartiles.
 - Relative difference for show-hide clicks; a measure of a specific type of interaction is calculated for all sessions.
- Session networks
 - Create session networks; each on-line action is represented as a node – links between nodes based on timing.
 - Calculate structural network measures; 23 different structural measures were selected and calculated.
 - Conducting principal component analysis (PCA) over all sessions and all measures resulting in five components
- Similarity network
 - Creating similarity network of sessions in which students visit a page with a problem in neutron scattering. Similarity based an exponential transformation of the Euclidean distance between sessions in PCA space.
 - Applying LANS algorithm (Foti, Hughes, & Rockmore, 2011); a method for removing insignificant links in a dense network

- c. Applying Fuzzy Infomap algorithm (Esquivel & Rosvall, 2011); a method for finding overlapping clusters in a network
- 4. Various analyses
 - a. Examining similarity network; visualizing similarity network and clusters provides insights into relationships between clusters.
 - b. Scrutinizing session networks; a pdf with all session networks is included as a supplementary file.
 - c. Cluster component scores; mean and standard deviations on each score for each cluster.
 - d. Segregation analysis; investigation of over-representation of particular session attributes.
 - e. Groups of clusters; using cluster component scores to find larger groups.
 - f. Analyses of relative difference in show-hide clicks

The sample data file used in this method is available online (Authors, 2019).

Sessions

Loading and preparing data

The accompanying R-scripts contain the scripts used in this illustrative example to load and prepare data. We make a number of preliminary modifications to the data. Some of the modifications are cosmetic, for example, replacing full URLs, which are very common with shorthand versions. Others are needed in further analysis. For example, the *DOM element text column* contains information about whether a page with the name problem, question, exercise, or simulation project was targeted by the action. The same column contained information about whether a show or a hide button was clicked. The code used for loading and preparing data can be found in the R-script, *01clickstreamClusterMethodSessions.r*, lines 1-219.

Creating sessions

In order to create sessions, we first identified all unique Session ID in the dataset. We find 4175 unique sessions. We calculated the duration of each session by identifying the maximum and minimum timestamp values and subtracting them. For this illustrative example, we filtered out sessions shorter than 5 minutes, as we take this period as a threshold for meaningful interaction with the web-page content. We found 2184 sessions with a duration of 5 minutes or more. We then created 2184 tables with all collected parameters for all actions relating to a Session ID.

The code used for creating sessions and selecting sessions in which a web-page with a problem was accessed can be found in the accompanying R-script, *01clickstreamClusterMethodSessions.r*, lines 221-271.

Time class classification

Session durations in the sample vary from 5 minutes to 20 days and follow no clear distribution. For later analyses, we found it useful to divide session durations into discrete classes. In order to classify the duration of each problem session, we produced a kernel density estimate (R Core Team, 2017; Sheather & Jones, 1991). In this example, we saw that 229 out of 231 were in the first quartile ($t_{dur} < 41.3$ hours). Since this classification would hide much of the variation, we chose 3 hours as a threshold and produced another kernel density estimate. The choice of 3 hours is pragmatic; we assume that most students in the course would not spend more than 3 hours at a time on a single problem. We then use the quartiles to divide sessions into a discrete set of time classes, short, middle, long, and extensive. The maximum density estimate was at 3.38 hours, and we included the sessions with durations above 3 hours in the Extensive category. Table 1 summarizes the Time Class categorization.

Table 1: Summary of time classes

Time Class	Duration limits	$N_{sessions}$
Short	5-25 minutes	59
Middle	25 minutes – 1.4 hours	93
Long	1.4-2.4 hours	38
Extensive	> 2.4 hours	41 (30 above the 3 hour threshold)

Since the choice of 3 hours is to some extent arbitrary, we investigate the impact of setting this threshold to 1, 1.5, 2, 2.5, and 3 hours in later analyses. The code used to create the time class classification can be found in the accompanying R-script, *01clickstreamClusterMethodSessions.r*, lines 272-318.

Relative difference for show-hide clicks

We construct a measure for tendency to show hints/solutions in a session, s :

$$\mu_s = \frac{N_{show} - N_{hide}}{N_{show} + N_{hide}} \quad (1)$$

$\mu = 0$, when there is a balance between showing and hiding hints/solutions, while $\mu = 1$, if $N_{hide} = 0$; when a session only displays clicks on show-buttons. If show is never clicked, the measure is undefined. We assume that in order to click a [hide]-button, a show-button must be clicked. This is not guaranteed, since a user could potentially hide a piece of text that was set to be shown on entering a page. Table 2 summarizes our calculations for μ_s for both problem session and other sessions. Clearly problem-solving sessions stand out from non-problem-solving session by having a much smaller fraction of μ_s undefined. A Wilcoxon Rank test

using only on the 582 non-problem-solving and 188 problem-solving sessions, which have well defined μ s, shows that these distributions are different $p < 0.01$.

Table 2. Summary of measure for showing/hiding hints/solutions.

μ values	$\mu < 0$	$\mu = 0$	$0 \leq \mu \leq 1$	$\mu = 1$	μ undefined	Total
Non-problem-solving sessions	2 (0.1%)	24 (1.2%)	216 (11.1%)	340 (17.4%)	1371 (70.2%)	1953
Problem-solving sessions	2 (0.8%)	18 (7.8%)	75 (32.5%)	93 (40.3%)	43 (18.6%)	231
Total	2 (0.1%)	53 (2.4%)	558 (25.5%)	853 (39%)	717 (32.8%)	2184

The code used for calculating μ s can be found in the accompanying R-script, *01clickstreamClusterMethodSessions.r*, lines 319-365.

Session networks

Creating session networks

In order to create sessions, we extracted information about sessions and created a table for each as shown in Table 3, in which Type, Document ID, and Target ID uniquely determines the action. Type represents the type of action, for example, whether a Hide or Show button was pressed, whereas Document ID and Target ID refer to individual pages and specific buttons/links on a particular page.

We now formed networks where nodes represented unique combinations of type, Document ID, and Target ID, and links represented the order. The value of a link was set to the time between actions, Δt . The resulting session network can be seen in Figure 3. In this way, we created session networks from server logs from three years (2012-2014) involving three iterations of a blended graduate level course on Neutron Scattering.

We used Document ID to uniquely identify a viewable document, for example, a page containing a problem and Target ID to identify the target of the action. Targets can be hyper-links to other pages, Show/Hide buttons, or other actions (such as showing an image, scrolling, or dragging). Tags are HTML-tags. To distinguish between different types of actions, we created the Type identifier. Using URL names, the Document Object Model (Wikipedia, 2018) of the wiki-software, and HTML-tags, the Type-identifier labels whether an action was navigation to a wiki-page with a problem or not, if it was a click on Show or on Hide, or other. Pages that involve wiki-textbook problems all have the word problem in the URL. We used this to create a list 231 sessions, which included events on pages with one or several of 25 wiki-textbook problems. We subsequently used this list to keep track of these problem-solving sessions.

Table 3. Information in table for session 1826. This information is used to in the creation of session networks. However, the table contains more information. See the accompanying R-script for details.

Action No.	Tag	Type	Document ID	Target ID	Timestamp	Δt (s)
1	A	To problem	2411356568	2039119516	1411028922	...
2	A	Show	2039119516	4124365635	1411029291	369
3	A	Hide	2039119516	4124365635	1411029298	7
4	A	Show	2039119516	4153568538	1411029299	1
5	A	Show	2039119516	4124365635	1411029313	14
6	DD	Other	3326751606	0	1411029326	13
7	A	Show	3326751606	4124365635	1411030487	1161
8	DIV	Other	3326751606	0	1411030597	110
9	IMG	Other	3326751606	3392475497	1411031858	1261

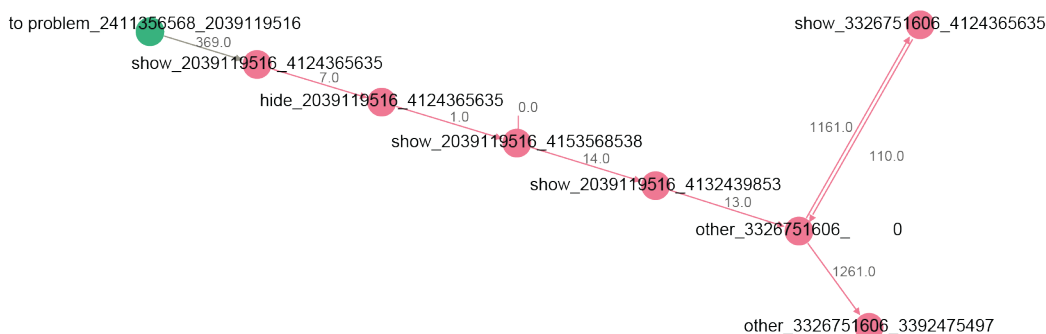


Figure 3. The network of session 1826 made from the data in Table 3.

The code used for creating session networks can be found in the accompanying R-script, *02clickstreamClusterMethodSessionNetworks.r*, lines 1-72

Calculating structural network measures

Inspired by Faust (2006), this method aims at comparing session networks by various network measures. A multitude of network measures exist, and we selected 23 measures of global network characteristics. Some are basic network measures and derivatives, such as the number of nodes, N , the number of links, L , and the density, ρ . Others, such as diameter, d , and Target Entropy, TE , provide information about the overall structure of the network. Finally, we included connected triads –*motifs*– in the analysis. Connected triads have been described as the building blocks of networks (Milo et al., 2004, 2002), and may provide more detailed understanding of the structure of session networks.

The measures are described in technical detail by Costa, Rodrigues, Travieso, & Villas Boas (2007) and Wasserman & Faust (1994). Here, we give a brief description of them. We have already described nodes, N and links, L . The density, ρ , can be seen as a measure of how many unique events are connected; in a high-density session network a user has navigated between a large fraction of the unique events. Mutual links, N_{\leftrightarrow} , indicate that a user has navigated back and forth between two unique actions.

The notion of *shortest paths* from unique actions to other actions are central to the next four measures. A shortest path between two nodes, A_i and A_j is the least number of possible other nodes one has to visit in order to get from A_i to A_j . Shortest paths are often used in navigational models, for example, the average (shortest) path length, \bar{l} . The diameter, d , is the longest shortest path in the session network and is often taken as a linear measure of the size of the network. (Costa et al., 2007; Wasserman & Faust, 1994)

Target entropy, TE and search information SI (Author & other, 2013; Rosvall, Trusina, Minnhagen, & Sneppen, 2005) use shortest paths to gauge the activity around an action and how difficult it is to find A_j from A_i , respectively. In this study, we calculate TE and SI for the whole network. Doing this for TE will yield information about the predictability of a session network. For example, if a session network consists of a linear string, $TE = 0$, while it will be higher for more complex networks. SI will yield information about the navigability of the network; a low SI will signify easy navigation from action to action on average.














Below, in Table 4 and Table 5, we list the measures, short description of them, and descriptive statistics for the 2184 sessions.

Table 4. Ten of the network measures, which we have calculated in this study.

Symbol	Description	Descriptive statistics			
		Mean	Median	Min	Max
N	The number of nodes in a network. Represents unique actions.	9.5	7	1	194
L	The number of (directed) links in the session network. Represents the ordering and timing of actions.	10.9	8	0	243
ρ	The fraction of number of links out of the number of possible links, for directed networks: $\frac{L}{n(n-1)}$ (Wasserman & Faust, 1994)	0.22	0.16	0.005	1
N_{\leftrightarrow}	Number of mutual links.	2.15	0	0	62
d	Diameter of the network. The longest shortest path in a network.	7.13	6	0	166
\bar{l}	Average length of a shortest path from one node to another.	3.03	2.59	1	58.3
TE	Target Entropy. Gauges average predictability of traffic around nodes in a network. (Rosvall et al., 2005)	0.11	0.06	0	1.02
SI	Search Information. Gauges the average number of questions needed to navigate between two nodes in a network. (Rosvall et al., 2005)	0.65	2.59	1	8.59
C	Transitivity, the number of closed triangles relative to the number of connected triplets. (Wasserman & Faust, 1994)	0.08	0	0	1
S	The entropy of the distribution of the number of edges (degree distribution). From Costa et al. 2007	0.69	0.68	0	2.08

Table 5. Network motifs (Milo et al 2002).

Symbol	Description	Descriptive statistics			
		Mean	Median	Min	Max

	V-in. Two unique actions, which both precede one unique action at some point during the session.	1.19	0	0	54
	Chain. Unique actions following each other at some point during the session	8.6	5	0	292
	Mutual-in. Two unique actions which followed each other at some point in the session, and one action that preceded one of the unique actions at some point.	1.8	0	0	85
	V-out. One unique action preceding two unique actions at some point during the session.	0.94	0	0	43
	Feed-forward-loop. Like a chain, but additionally one action preceding the other at some point.	0.11	0	0	11
	Regulated-mutual. Like V-out but with a mutual connection between the two actions.	0.02	0	0	3
	Mutual-out. Like mutual-in but with the single link reversed.	1.47	0	0	85
	Mutual-V. A combination of mutual-in and mutual-out.	0.77	0	0	76
	Three-loop. A closed loop of unique actions that followed each other at some point during the session.	0.35	0	0	7
	Regulated-3-loop. Like three-loop, but with an extra link between two unique actions.	0.14	0	0	11
	Regulating mutual. Like regulated-mutual but with non-mutual links reversed.	0.02	0	0	3
	Semi-clique. Like V-in with an extra link between two unique events.	0.05	0	0	8
	Clique. Three actions that all followed each other at some point during the session.	0.02	0	0	3




The code used for calculating these measures can be found in the accompanying R-script, *02clickstreamClusterMethodSessionNetworks.r*, lines 73-114.

Principal Component Analysis

We used the Psych package (Revelle, 2017) to perform principal component analysis on the 23 network measures over the 2184 session networks. We opted to use rotated components since this will "drive loadings towards zero or towards their maximum possible absolute value" Jolliffe (2002, p. 271), thus maximizing differences between components. Thus, we expected each rotated component RC to highlight different structural aspects of session networks.

We performed the rotated principal component analysis on the 2184 session networks with $t_{dur} > 300s$. The parallel analysis suggested that five principal components would be sufficient for our purposes. Running the PCA with five components and varimax rotation we found that each component accounted for 8% or more of the variance (adding up to a total of 77%). Adding an extra rotated component accounted for an additional 5%, and we chose to keep five components. Table 6 below lists the loadings for each rotated component above an absolute threshold value of 0.4. Each component captures some dimension of structure, and session networks with high scores on a component should exhibit network measures in accordance with the loadings on that component.

Table 6. Rotated component loadings from the PCA.

	Component 1: Linear length	Component 2: Mutuality	Component 3: Navigation	Component 4: Complexity	Component 5: Trial & Error
N	0.91				
L	0.80		0.41		
ρ	0.61				
d	0.96				
\bar{l}	0.98				
N_{\leftrightarrow}		0.84			
			0.81		
	0.79		0.53		
		0.86			

			0.80		
			0.70		
					0.68
		0.85			
		0.88			
			0.71		
		0.48			0.48
					0.72
		0.66			0.43
		0.52			
TE				0.83	
SI		0.54	0.49	0.44	
C				0.60	
S				0.82	

Authors (2019a) analyze, interpret, and name components as shown in Table 6. As could be expected, we found no correlations between component scores ($p > 0.9$ for all correlations). The code used for the PCA accompanying R-script, *clickstreamClusterMethodSessionNetworks.r*, lines 115-118.

Similarity network

Creating similarity network

Following Valtonen et al. (2009), we used the rotated component scores for each network as a basis for similarity. Treating the rotated components found above as spanning a vector space, we calculated the Euclidean distance between each pair of session networks. This procedure produced a symmetric distance matrix. We then converted the distance matrix, D_{ij} , to a similarity matrix, W_{ij} , using the transformation $W_{ij} = e^{-D_{ij}}$. Thus, the similarity is a number between 0 and 1, with 1 representing perfect similarity. The matrix W_{ij} describes a similarity network.

The code used for generating the similarity network can be found in *03clickstreamClusterMethodSimilarityNetwork.r*, lines 1-13.

Applying LANS algorithm

Since all session networks will be at a finite distance from each other, the similarity network will be fully connected. Also, a session will be similar to itself. These two conditions make it hard for community detection algorithms to find cluster structure. The standard way to overcome self-similarity is to remove the diagonal. To overcome the finite-distance problem, we follow (Author & others, 2016) and use local adaptive networks sparsification (LANS) (Foti et al., 2011) to remove insignificant connections. The principle behind LANS is to find out which connections are important for each node. Technically, for a link, L_{ij} , the LANS algorithm compares the link weight, W_{ij} , with all other weights of links attached to the node. If W_{ij} is greater than or equal to a predefined fraction of other links, $f = 1 - \alpha$, where α can be interpreted as a p-value, the link is kept. Otherwise, LANS will mark the link for deletion. However, the same link is attached to two nodes. In our implementation of LANS, a link can survive if it is significant to just one of the two nodes it connects. For the current illustration, we chose $\alpha = 0.05$. Table 7 below is a summary of the resulting similarity network.

The code used for generating the sparsified – or backbone – network can be found in the accompanying R- script, *03clickstreamClusterMethodSimilarityNetwork.r*, lines 14-20.

Table 7. Summary description of the similarity network.

	N	L	ρ	d	\bar{l}	S	TE	SI
Similarity network ($\alpha = 0.05$)	231	2074	0.08	7	3.47	2.43	3.42	5.50

Applying Infomap algorithm

This methodology now employs community detection for finding clusters of similar session networks. This strategy can be seen to have three significant advantages over commonly used clustering techniques, such as k -means and hierarchical clustering (Dutt, Aghabozrgi, Ismail, & Mahroeian, 2015). First, the quality of the clustering can be assessed in terms of a measure called the modularity (Q), which is the fraction of connections within a cluster (as opposed to between clusters) minus what could be randomly expected. For $Q < 0.3$ (Newman, 2004), there would be no significant community structure to detect, and most community detection methods search the solution with the highest Q (Lancichinetti & Fortunato, 2009). Second, more information about the structure of a community is kept; it is not given that the clustering structure is hierarchical or flat. It could be either or in-between. Third, for the particular clustering algorithm we use, Fuzzy Infomap (Esquivel & Rosvall, 2011), clusters can overlap, which can give additional information about inter-cluster structure. Network community detecting thus provides means to investigate such structures in more detail.

Fuzzy Infomap relies on an information theoretical correspondence between compression and regularity detection. The algorithm can be described as a random walker traversing the network via links. In the similarity network, the walker is expected to spend a lot of time walking between similar session networks, because they are tightly linked. Fuzzy Infomap will exploit the fact that it will be easier to compress information about the walk if similar session networks are grouped into clusters to partition the network. In some cases, a session network will lie on the border between two clusters and assigning the session network to two or more clusters (with a given percentage belonging to each cluster) will allow for more compression of the information about the walk. The end product of this procedure is an assignment of each session network to one or more clusters.

Having installed Infomap and created a folder *output*, we used the following command in a terminal to invoke Fuzzy Infomap:

```
./Infomap backbone.net --overlapping output
```

This produces a file named *backbone.tree*, which is formatted as shown in Box 1. This file is placed in a folder called *output*. Infomap is continuously developing, and depending on the version, the clustering will be slightly different. However, we find that our results are not changed significantly in the newest current version (version 0.20).

Box 1 Sample .tree file. The first line shows the options and summary of the results. In this iteration the 231 nodes were split into 287 in overlapping communities.

```
# 'backbone.net --overlapping output' -> 287 nodes partitioned in 1s from codelength 11.720174752 in one level to codelength
5.759363881 in 2 levels.
# path flow name node
1:1 0.0103803 "470" 69
1:2 0.0103803 "810" 99
1:3 0.0103803 "839" 102
1:4 0.0103803 "1047" 111
```

We convert the tree file to a .csv-file (1:1 is replaced by e.g. 1[space]1). The result of this conversion can be found in the accompanying file *overlappingGroups.csv*. To load this file use *03clickstreamClusterMethodSimilarityNetwork.r*, lines 29.

Infomap calculates the flow-value (Rosvall, Axelsson, & Bergstrom, 2009) of each node. The flow-value quantifies how many times a node is visited relative to the rest of the network. Flow-values sum up to 1. With overlapping clusters, the flow-value is distributed between the clusters. Thus, a node i , which is part of both Cluster A and Cluster B, has a flow-value associated with each of these clusters, α_i^A and α_i^B . Following Esquivel & Rosvall (2011), the fraction of membership of Cluster A is

$$f_i^A = \frac{\alpha_i^A}{\alpha_i^A + \alpha_i^B} \quad (2)$$

For nodes, which are part of more than two clusters, the sum in the denominator is over all clusters.

Method validation: Suggestions for analysis

This section illustrates the kind of analysis this method makes possible. For an interpretation of the data in light of educational research, we refer to Authors (2019).

Examining similarity network

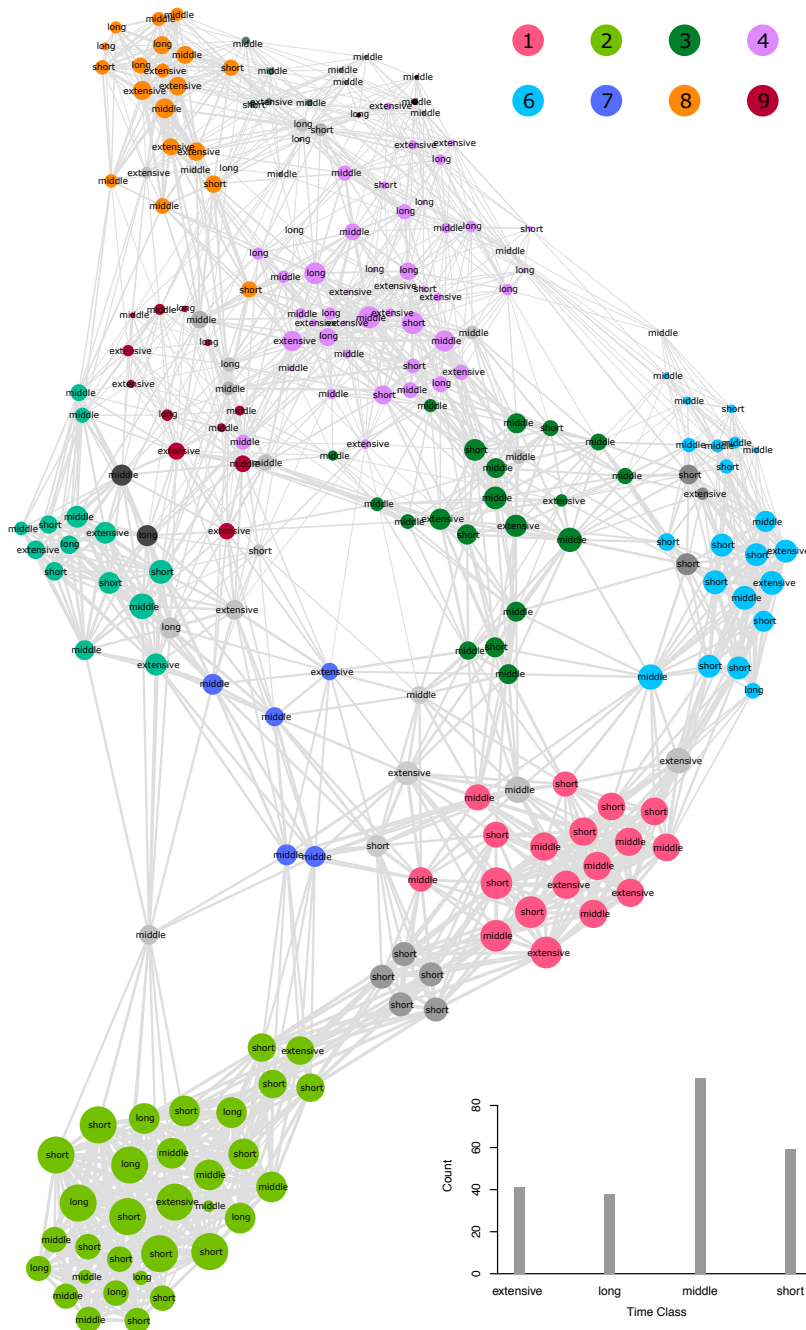


Figure 4. Similarity network with $\alpha=0.05$ and groups as found by Fuzzy Infomap. Size of nodes represent the sum of links; their accumulated similarity.

Figure 4 shows a plot of the similarity network. Colors represent the 10 clusters. The thickness of each link indicate similarity; thicker links represent more similar sessions. Nodes sizes represent the sum of links for each node; the weighted degree. In this network it can be seen as the nodes accumulated similarity. Finally, the label on each node illustrate the time class (3h) variable. In Figure 4, we have used the Yifan-Hu algorithm (Hu, 2005), which is one of the many force-based layout algorithms available. In this algorithm, each node repels the other with a force inversely proportional to the distance between them. A link between two nodes with result in an attractive force proportional to the square of the distance. For a given placement of all nodes, the total energy of the system can be calculated, and Yifan-Hu uses adaptive cooling to find the placement of nodes that minimizes the energy. This ensures minimum overlap of links and often that sessions that are clustered together also end up close to each other visually.

To arrive at a plot similar to the one in Figure 4, use software Gephi version 0.92 (Bastian, Heymann, & Jacomy, 2009) and a cluster file (.csv-version) as shown in Box 1. Select the built-in version of Yifan-Hu's algorithm to produce a plot that is close to isomorphic to Figure 4. Readers can use the supplemental gephi file, or experiment with the instructions given in Box 2 below.

Box 2. Instructions for making a graph in Gephi 0.92.

To visualize a graph in gephi, start in R:

1. `> write.graph(backbonenetwork,"backbone.net", "pajek")` # requires you running appropriate supplemental scripts beforehand

2. Open Gephi and load the network as an undirected network.
3. Go to Gephi's "Data Laboratory" tab and export node table. Open the table in a spreadsheet editor and provide a new column called "cluster".
4. Copy Infomap cluster number to corresponding nodes in node table. That is, match nodes in overlappingGroups.csv file to nodes in node table from Gephi. Note that some nodes are part of multiple clusters, so assign a unique value to these (i.e. all nodes that are part of both cluster 1 and cluster 2 are assigned the value "12" or similar). This can be done manually.
5. It is important to use the node table as the ID column in that table is used by Gephi to match labels to correct nodes.
6. Import node table to Gephi, select "append to existing workspace".
7. Go to "Overview" and select the color palette in the Appearance tab. Choose "partition" and select the cluster attribute. Press Apply.
8. In the Statistics tab, calculate Avg. Weighted Degree by pressing run. Close the report.
9. In the Appearance tab, choose the size (concentric circles) icon, "ranking" and choose Weighted Degree as attribute. Set Min. Size and Max. Size to 10 and 40 respectively. Press Apply.
10. In the Layout tab, choose Yifan-Hu and Run.

One way of conceptualizing the plot in Figure 4 is as a map. Notice on Figure 4 that clusters 1, 2, and 6 all lie on the "East coast" of the map. Internally, each cluster is very similar; thick links connect them and on average sessions in clusters 1, 2, and 6 are larger than sessions in other clusters. The reason for them being large is that each of clusters 1, 2, and 6 are very dense both in terms of number of links and in terms of the thickness of each link. Thus, we can expect them to be structurally very homogeneous. In contrast, cluster 10 consists of sessions which are small and have very thin links connecting them. We can expect the sessions, which these nodes represent to be more diverse than sessions in clusters 1, 2, and 6 respectively.

Session networks

A file with plots of all session networks can be generated. These are used to get a sense of the variation of networks belonging to a cluster. The plots also show the fractional membership of the cluster. The code for generating this file with R is given in *03clickstreamClusterMethodSessionSimilarityNetwork.r* lines 31-349.

From these plots it is apparent, for example, that clusters 1, 2, and 6 can structurally be described as beads on a string – the difference primarily being the length of the string. Another possibility is to locate particular sessions in the similarity network, which are shared between clusters. Then to compare these sessions visually with typical clusters from each session. We have yet to explore this possibility.

Cluster network measures

We calculate mean cluster values in two ways. The first considers fractional membership. Thus, for a given measure, X , the cluster mean is:

$$X_c = \frac{\sum_{i \in c} f_i^c X_i}{\sum_{i \in c} f_i^c} \quad (3)$$

The standard deviation is

$$\sigma_c = \frac{\sum_{i \in c} f_i^c (X_i - X_c)^2}{\sum_{i \in c} f_i^c} \quad (4)$$

The second way of calculating mean cluster values relies only on nodes with full membership in a group and is calculated in the standard way for mean and standard deviation. Table 8 and Table 9 lists the results.

Table 8. Mean values per cluster for each network measure considering fractional memberships. N_{ses} shows the number of members [fractional number of members] in each cluster.

Name	Clus 1	Clus 2	Clus 3	Clus 4	Clus 5	Clus 6	Clus 7	Clus 8	Clus 9	Clus 10
N_{ses}	28 [25.7]	36 [31.4]	30 [22.7]	61 [54.0]	19 [16.5]	28 [23.5]	13 [7.1]	25 [19.9]	23 [16.2]	20 [12.0]
N	9.8(5)	5.1(5)	17(1)	21(3)	5.4(7)	22(4)	9(1)	8(1)	12(3)	20(7)
L	9.1(6)	4.1(5)	18(2)	29(6)	5.8(9)	22(5)	9(2)	9(2)	17(6)	33(13)
ρ	0.107(5)	0.22(2)	0.071(5)	0.08(6)	0.26(4)	0.053(7)	0.144(3)	0.20(4)	0.14(3)	0.10(2)
d	8.8 (4)	4.0(5)	14(1)	13(2)	4.1(6)	21(4)	6.7(9)	6(1)	9(2)	14(5)



























\bar{l}	3.6(2)	2.0(2)	5.3(3)	4.8(6)	1.9(2)	8(1)	2.9(3)	2.5(2)	3.2(4)	4(1)
N_{\leftrightarrow}	0.5(4)	0.0(1)	2.4(8)	6(2)	2.6(8)	0.5(5)	2.6(8)	1.3(8)	8(3)	10(6)
	0.04(8)	0(0)	1.1(4)	6(2)	0.1(1)	0.6(5)	0.0(1)	1.1(4)	1(2)	7(4)
	7.71	3.13	16.26	26.40	2.21	21.20	5.28	4.84	7.76	24.88
	0.2(2)	0(0)	1.4(5)	8(4)	1.3(2)	0.3(3)	1.3(8)	0.9(7)	6(4)	9(7)
	0.0(1)	0(0)	1.0(4)	5(1)	0(0)	0.4(5)	0(0)	0.6(3)	1(1)	5(3)
	0(0)	0(0)	0.03(7)	0.6(3)	0(0)	0.0(1)	0(0)	0.2(2)	0.0(1)	0.5(6)
	0(0)	0(0)	0(0)	0.09(9)	0(0)	0(0)	0(0)	0(0)	0.04(9)	1.0(4)
	0.1(1)	0.0(0)	1.0(5)	7(3)	0.7(3)	0.2(3)	0.8(7)	0.8(6)	5(4)	8(6)
	0.02(5)	0(0)	0.1(1)	3(3)	0.2(2)	0(0)	0.3(5)	0.1(2)	3(4)	3(5)
	0(0)	0(0)	0.1(1)	1.4(4)	0(0)	0.1(1)	0(0)	1.0(1)	0.2(3)	1.4(7)
	0(0)	0(0)	0(0)	0.6(4)	0.01(5)	0(0)	0(0)	0.3(3)	0.3(6)	1(1)
	0(0)	0(0)	0(0)	0(0)	0(0)	0.06(2)	0(0)	0(0)	0(0)	0.8(4)
	0(0)	0(0)	0(0)	0.2(2)	0(0)	0(0)	0(0)	0(0)	0.4(5)	1(1)
	0(0)	0(0)	0(0)	0.04(7)	0(0)	0(0)	0(0)	0(0)	0.3(3)	0.3(3)
TE	0.01(1)	0(0)	0.09(1)	0.19(2)	0.21(3)	0.02(1)	0.08(3)	0.20(5)	0.28(6)	0.33(6)
SI	0.05(5)	0.01(2)	0.9(2)	2.2(3)	0.4(1)	0.3(2)	0.3(3)	0.7(2)	1.7(4)	2.8(9)
C	0(0)	0(0)	0.01(2)	0.15(2)	0.00(2)	0.01(1)	0(0)	0.37(5)	0.1(6)	0.25(4)
S	0.53(2)	0.63(4)	0.75(6)	0.95(7)	1.02(7)	0.39(6)	0.80(7)	1.0(1)	1.2(1)	1.4(2)

Table 9. Mean values per cluster for each network measure considering only full membership.

Name	Clus 1	Clus 2	Clus 3	Clus 4	Clus 5	Clus 6	Clus 7	Clus 8	Clus 9	Clus 10
N_{ses}	18	30	18	42	13	21	5	18	12	10
N	10.0(5)	5.0(5)	17(1)	20(3)	5.0(7)	21(4)	9(1)	7.2(9)	11(1)	19(7)
L	9.3(5)	4.0(5)	18(2)	27(4)	5.2(8)	21(4)	9(2)	8(1)	16(2)	31(13)
ρ	0.104(5)	0.22(2)	0.071(5)	0.075(7)	0.28(5)	0.055(7)	0.13(1)	0.21(4)	0.15(3)	0.11(2)
d	9.1(4)	4.0(5)	14(1)	12(2)	3.8(6)	20(4)	7.0(6)	5.6(8)	9(2)	13(5)
\bar{l}	3.7	2.0	5.2	4.6	1.8	7.7	3.0	2.4	3.1	4.0
N_{\leftrightarrow}	0.3(4)	0(0)	2.3(9)	5(2)	2.3(4)	0.4(4)	2.4(8)	1.1(6)	8(1)	9(6)
	0.1(1)	0(0)	0.9(4)	6(2)	0.1(2)	0.4(4)	0(0)	1(4)	0.8(7)	6(3)
	8.06	3.07	15.83	25.10	1.92	20.05	5.60	4.33	6.42	23.00
	0.1(1)	0(0)	1.3(5)	7(4)	1.2(2)	0.2(2)	1.2(4)	0.8(4)	5(2)	9(6)

	0.1(1)	0(0)	1.0(4)	4(1)	0(0)	0.2(3)	0(0)	0.6(3)	0.6(7)	5(2)
	0(0)	0(0)	0(0)	0.5	0(0)	0(0)	0(0)	0.1	0(0)	0.4
	0(0)	0(0)	0(0)	0(0)	0(0)	0(0)	0(0)	0(0)	0(0)	1(5)
	0.1(1)	0(0)	0.9(4)	6(3)	0.6(3)	0.1(2)	0.8(7)	0.6(4)	4(1)	7(4)
	0(0)	0(0)	0.1(1)	2(1)	0.1(2)	0(0)	0.2(4)	0.1(1)	2(1)	2(3)
	0(0)	0(0)	0.1(1)	1.2(3)	0(0)	0.05(9)	0(0)	1.1(1)	0.2(2)	1.5(7)
	0(0)	0(0)	0(0)	0.5(3)	0(0)	0(0)	0(0)	0.2(2)	0.1(2)	1.1(8)
	0(0)	0(0)	0(0)	0(0)	0(0)	0(0)	0(0)	0(0)	0(0)	0.9(5)
	0(0)	0(0)	0(0)	0.1(2)	0(0)	0(0)	0(0)	0(0)	0.3(3)	1(1)
	0(0)	0(0)	0(0)	0(0)	0(0)	0(0)	0(0)	0(0)	0.3(3)	1.0(3)
TE	0.01(1)	0(0)	0.09(1)	0.18(2)	0.22(3)	0.01(1)	0.08(1)	0.20(5)	0.30(7)0	0.34(6)
SI	0.06(6)	0(0)	0.9(2)	2.1(3)	0.3(2)	0.2(2)	0.3(3)	0.6(2)	1.7(4)	3(1)
C	0(0)	0(0)	0.01(2)	0.13(2)	0(0)	0.003(5)	0(0)	0.38(5)	0.1(6)	0.25(3)
S	0.52(2)	0.62(4)	0.76(5)	0.92(7)	1.05(7)	0.36(5)	0.80(7)	0.94(9)	1.19(9)	1.5(2)

Code that can be used to calculate these measures can be found in *04aclickstreamClusterMethodSuggestionsAnalysis.r*, lines 38-450.

Cluster component scores

We calculate cluster scores in the same two way as we did the network measures. Figure 5 shows the differences between the fuzzy membership and a hard partitioning. Relevant code can be found in *04aclickstreamClusterMethodSuggestionsAnalysis.r*, lines 451-550.

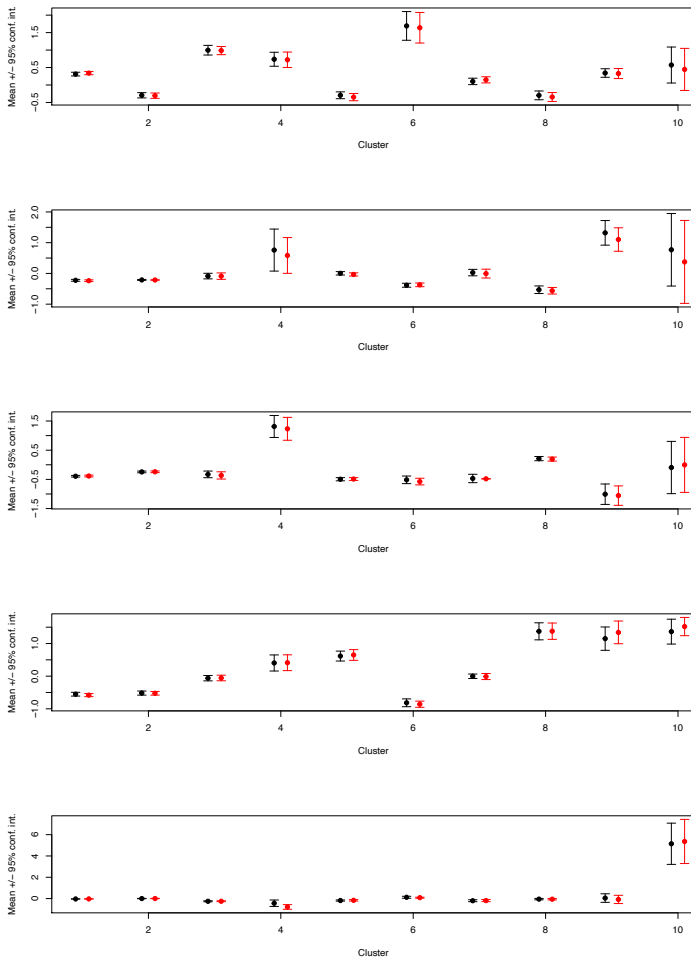


Figure 5: Cluster scores on each component. Left: Cluster scores over fractional members. Right: Cluster scores over full membership nodes.

Analysis of time and date variables

Each network was given attributes for year, week, starting day, and starting hour. Figure 6 shows the distribution of sessions on each of these attributes.

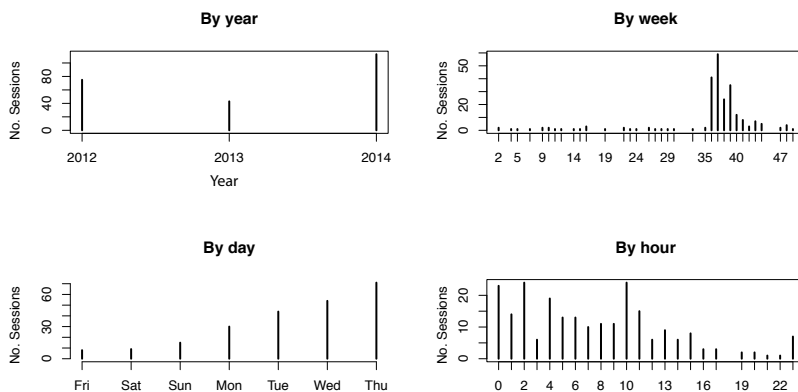


Figure 6. Descriptive statistics for the 231 problem solving sessions.

During the years 2012-2014 the wiki textbook was used in a university course in Weeks 36-45. The course days were Tuesday (half a day) and Thursday (full day). Relevant code can be found in `04aclickstreamClusterMethodSuggestionsAnalysis.r`, lines 578-616.

Cluster duration distributions

One way of comparing clusters would be to compare the average time spent in sessions belonging to each cluster. Since the clustering relies only on structural network measures, we cannot expect each cluster to have a characteristic duration. This can be verified by plotting kernel density plots of cluster durations. See Figure 7. Instead, we use the time class measure to characterize clusters. Relevant code can be found in `04aclickstreamClusterMethodSuggestionsAnalysis.r`, lines 617-649.

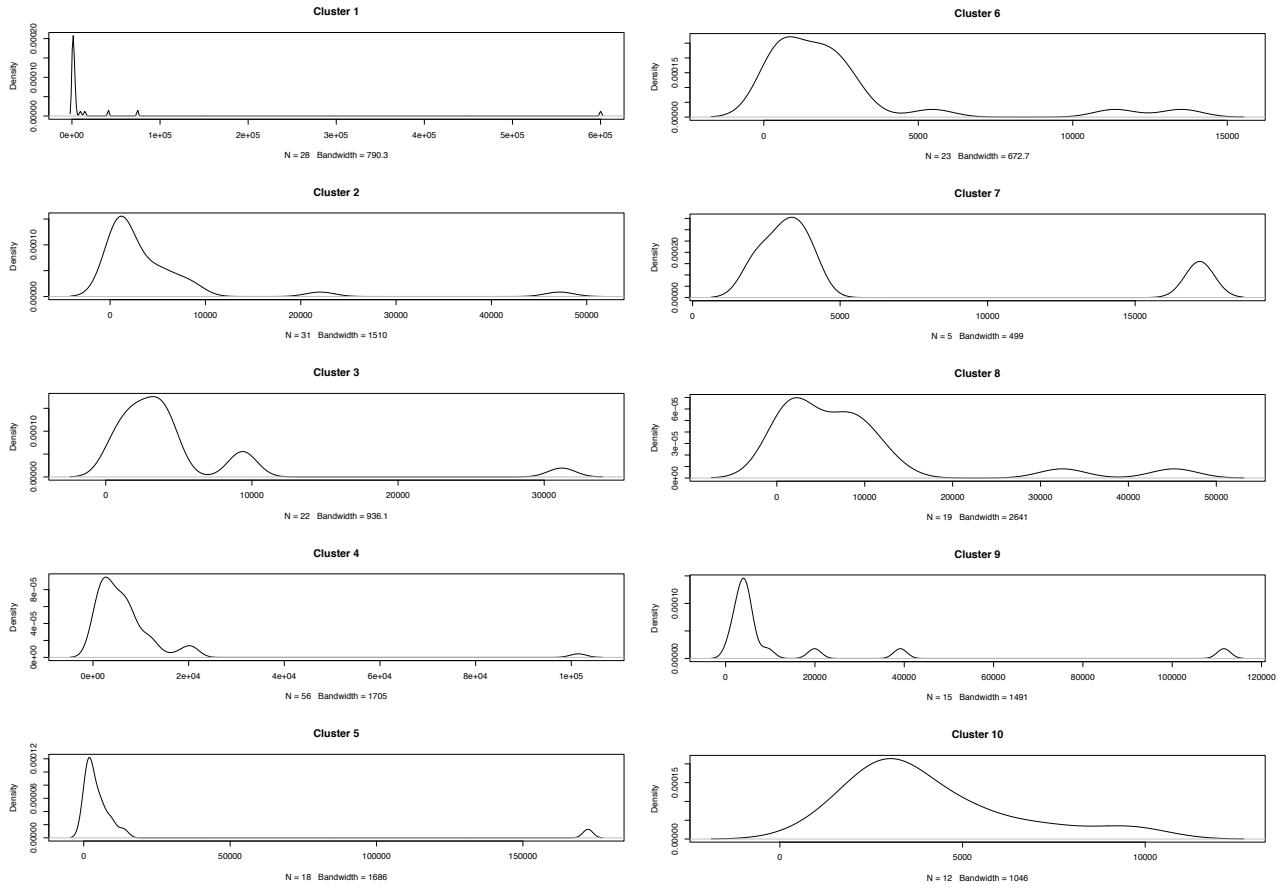


Figure 7. Cluster duration distributions for the 10 clusters of problem solving sessions.

Segregation analysis

We used the Segregation measure as employed by Author & other (2014). The Segregation measures any over representation of a particular node attribute in a group. For example, a cluster might consist of sessions that happened more at a particular weekday than could be randomly expected. This would result in a high Segregation when compared to random variation. We assign attributes randomly and calculate the Segregation for the random assignment. We do this $N = 10^4$ times and calculate the Z-score, $Z = \frac{X - \bar{X}}{\sigma_X}$. For $Z > 1.96$, Segregation is significantly different from random (Author & other, 2014). Segregation has been developed for hard partitioning only, so we use a hard partitioning where a node is placed in the cluster within which it has the highest fractional membership.

In this example, we calculate Z-scores for hour of day, day of week, week of year and year over the 10 clusters and find no evidence of significant Segregation. However, for all time class variables, we find significant Segregation. See Figure 8

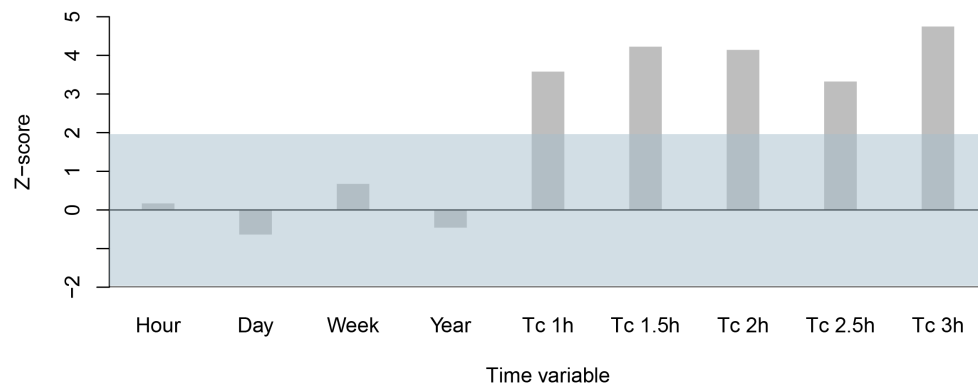


Figure 8. Segregation for various time related variables. The blue rectangle represents zone of insignificance. Tc represent various versions of the time class variable.

We calculated the Segregation per clusters finding that clusters 1, 2, and 6 all show an over-representation of short sessions. Cluster 3 shows an over-representation of middle duration sessions. Cluster 4 shows an over-representation of long duration clusters. See Table 10.

Table 10. Clusters that contribute to Segregation.

Cluster	Short 5-25 min	Middle 25min-1.4h	Long 1.4-2.4h	Extensive 2.4h+	Sum
1	0.46	0.36	0	0.18	1
2	0.45	0.26	0.23	0.06	1
3	0.18	0.64	0	0.18	1
4	0.14	0.34	0.30	0.21	1
6	0.48	0.39	0.04	0.09	1
All	0.26	0.40	0.16	0.18	1

Relevant code can be found in `04aclickstreamClusterMethodSuggestionsAnalysis.r`, lines 650-699.

Groups of clusters

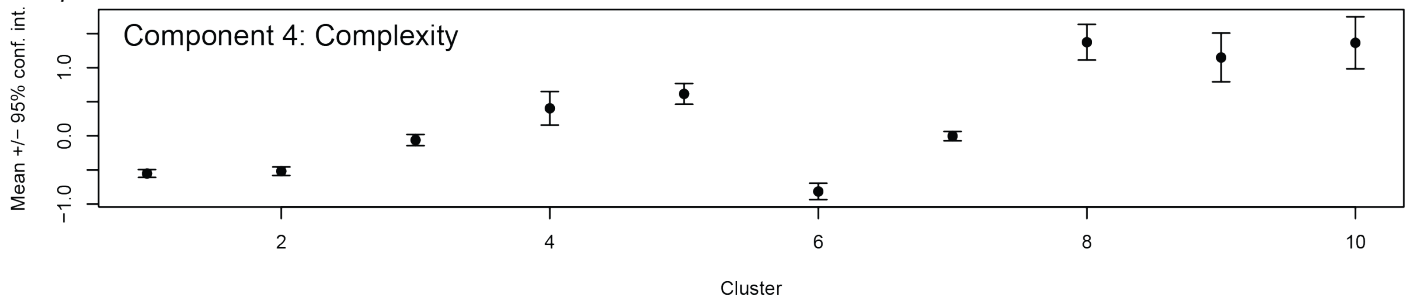


Figure 9. Cluster scores on Component 4: Complexity.

While Infomap finds distinct clusters, we find that careful scrutiny of cluster scores on each component reveals interesting insights. Here, we highlight this part of the method by focusing on Component 4 (Figure 9, see also Authors (2019)). Clusters 1, 2, and 6 scores negatively on Component 4. Clusters 3 and 7 score around zero. Clusters 4 and 5 score above 0 but below 1, while clusters 8, 9, and 10 score more than 1. Different arguments could be made for different groupings (see Authors, 2019).

Show-hide analysis

Of the 231 problem-solving sessions, the relative show-hide difference, μ , can be defined for 186. Figure 10 shows the mean value and 95% confidence intervals per cluster for fractional membership.

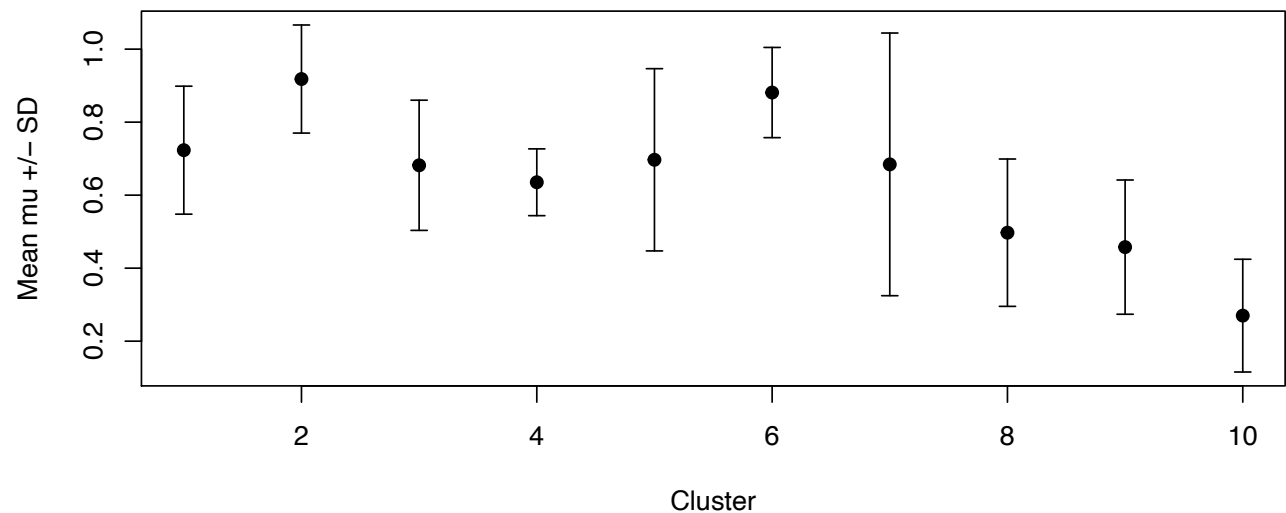


Figure 10. Relative difference of show-hide clicks. Mean and standard deviations for each cluster.

The uncertainties seem too large to say anything meaningful, we use groupings found by scrutinizing Component 4 cluster scores (See Figure 9). Figure 11 shows results for different groupings of clusters.

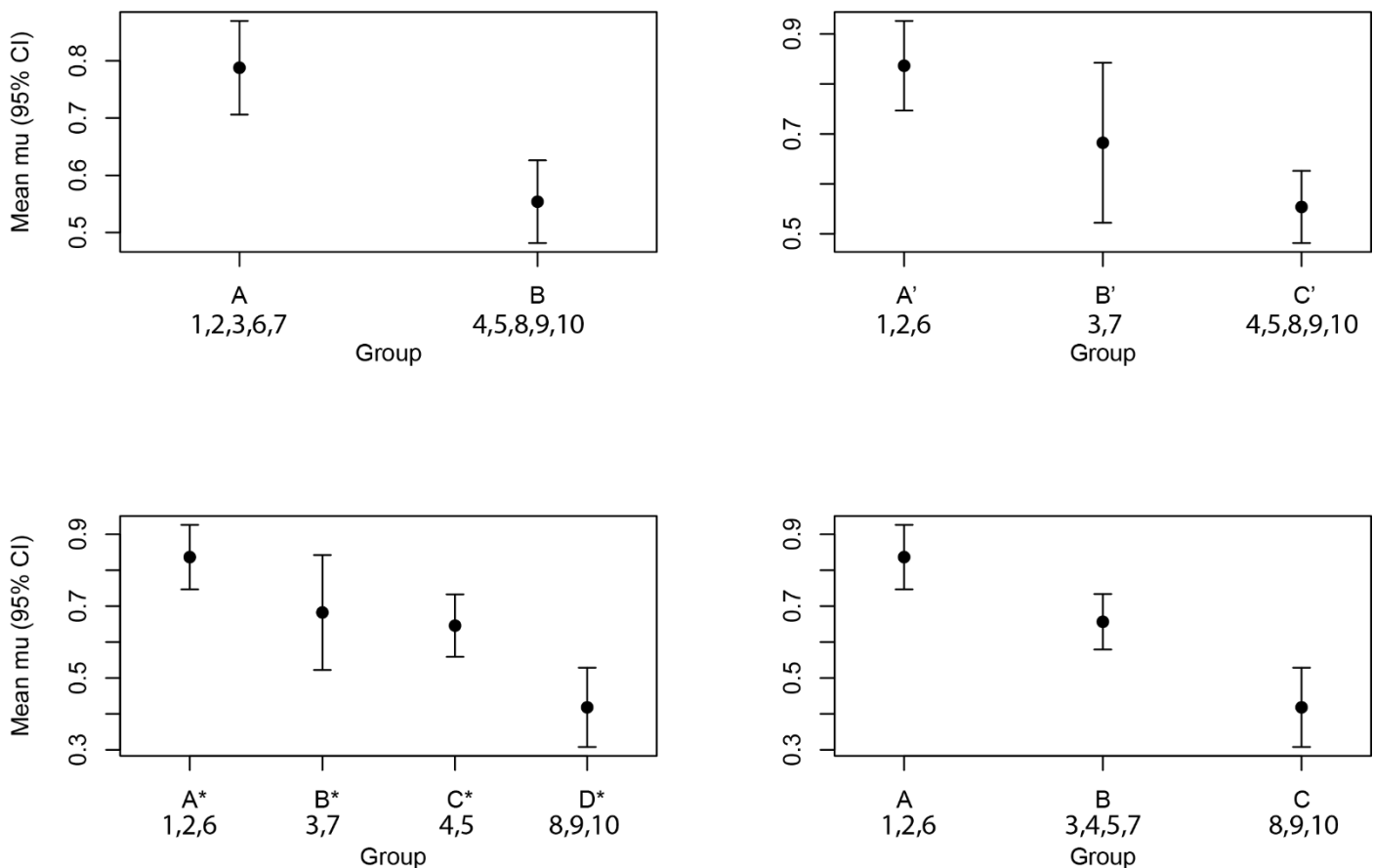


Figure 11. Mean relative show-hide differences for different groupings of clusters.

Relevant code for this analysis can be found in *04aclickstreamClusterMethodSuggestionsAnalysis.r*, lines 700-992.

Table 11. Distribution of fractional memberships on each grouping of clusters shown in Figure 11.

Top left		Top right		Bottom left		Bottom right	
N_A	81.8	$N_{A'}$	56.03	N_A^*	56.03	N_A	56.02
N_B	104.2	$N_{B'}$	25.77	N_B^*	25.77	N_B	87.88
		$N_{C'}$	104.2	N_C^*	62.10	N_C	42.10
				N_D^*	42.10		
Total	186	Total	186	Total	186	Total	186

This concludes our suggestions for analyses and our illustration of the method. As mentioned in the introduction, we believe the method to be applicable to analyses of click-stream data in general. Some measures will likely need modification for other contexts. These include the relative difference in show-hide clicks, and the time class variable. Parts of the method may also be used while others left out or modified. For example, nodes in a similarity network need not be networks, but could be other entities. The similarity measures need not be exponentials of distances. On the other hand, session networks need not be the only type of networks applicable for this method. A similar method has been developed for networks depicting dialogue (Author & others, 2018). Finally, this particular illustration has focused on structural network measures – individual meanings of nodes have not been considered.

- Supplementary material and/or Additional information:** OWA_clickstream20nov2014.csv
- 01clickstreamClusterMethodSessions.r
- 02clickstreamClusterMethodSessionNetworks.r
- 03clickstreamClusterMethodSessionSimilarityNetwork.r
- 04aclickstreamClusterMethodSuggestionsAnalysis.r

- f. tarEnt.r
- g. searchInf.r
- h. backboneExtraction.r
- i. characteristicsAll.csv
- j. overlappingGroups.csv
- k. similarityNetwork.gephi

References:

- Adams, P. (2012). *Open Web Analytics (version 1.5)*. Retrieved from <http://www.openwebanalytics.com/>
- Author. (2016). Article title. *Proceedings*, pages.
- Author, & other. (2013). Article title. *Journal*, X(Y), Title.
- Author, & other. (2014). Article title. *Journal*, X(Y), pages.
- Author, & others. (2016). Article title. *Journal*, X(Y), pages.
- Author, & others. (2018). Chapter. In *Book title* (p. Pages). Publisher.
- Authors. (2019). Title. *Journal*.
- Bastian, M., Heymann, S., & Jacomy, M. (2009). Gephi: an open source software for exploring and manipulating networks. *Third International AAAI Conference on Weblogs and Social Media*.
- Costa, L. da F., Rodrigues, F. A., Travieso, G., & Villas Boas, P. R. (2007). Characterization of complex networks: A survey of measurements. *Advances in Physics*, 56(1), 167–242.
- Csardi, G., & Nepusz, T. (2006). The igraph software package for complex network research. *InterJournal, Complex Sy*, 1695. Retrieved from <http://igraph.org>
- Dutt, A., Aghabozrgi, S., Ismail, M. A. B., & Mahroeian, H. (2015). Clustering algorithms applied in educational data mining. *International Journal of Information and Electronics Engineering*, 5(2), 112.
- Esquivel, A. V., & Rosvall, M. (2011). Compression of flow can reveal overlapping-module organization in networks. *Physical Review X*, 1(2), 21025.
- Faust, K. (2006). Comparing social networks: size, density, and local structure. *Metodoloski Zvezki*, 3(2), 185.
- Foti, N. J., Hughes, J. M., & Rockmore, D. N. (2011). Nonparametric sparsification of complex multiscale networks. *PloS One*, 6(2), e16431.
- Hu, Y. (2005). Efficient, high-quality force-directed graph drawing. *Mathematica Journal*, 10(1), 37–71.
- Jolliffe, I. T. (2002). Principal component analysis and factor analysis. *Principal Component Analysis*, 150–166.
- Lancichinetti, A., & Fortunato, S. (2009). Community detection algorithms: a comparative analysis. *Physical Review E*, 80(5), 56117.
- MediaWiki. (2019). *MediaWiki --- MediaWiki{,} The Free Wiki Engine*. Retrieved from <https://www.mediawiki.org/w/index.php?title=MediaWiki&oldid=3275990>
- Milo, R., Itzkovitz, S., Kashtan, N., Levitt, R., Shen-Orr, S., Ayzenshtat, I., ... Alon, U. (2004). Superfamilies of evolved and designed networks. *Science*, 303(5663), 1538–1542.
- Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., & Alon, U. (2002). Network motifs: simple building blocks of complex networks. *Science*, 298(5594), 824–827.
- Newman, M. E. J. (2004). Fast algorithm for detecting community structure in networks. *Physical Review E*, 69(6), 66133.
- R Core Team. (2017). *R: A Language and Environment for Statistical Computing*. Retrieved from <https://www.r-project.org/>
- Revelle, W. (2017). *psych: Procedures for Psychological, Psychometric, and Personality Research*. Retrieved from <https://cran.r-project.org/package=psych>
- Rosvall, M., Axelsson, D., & Bergstrom, C. T. (2009). The map equation. *The European Physical Journal-Special Topics*, 178(1), 13–23.
- Rosvall, M., Trusina, A., Minnhagen, P., & Sneppen, K. (2005). Networks and cities: An information perspective. *Physical Review Letters*, 94(2), 28701.
- Sheather, S. J., & Jones, M. C. (1991). A reliable data-based bandwidth selection method for kernel density estimation. *Journal of the Royal Statistical Society. Series B (Methodological)*, 683–690.
- Valtonen, T., Kukkonen, J., Dillon, P., & Väisänen, P. (2009). Finnish high school students' readiness to adopt online learning: Questioning the assumptions. *Computers & Education*, 53(3), 742–748.
- Wasserman, S., & Faust, K. (1994). *Social network analysis: Methods and applications* (Vol. 8). Cambridge university press.
- Wikipedia. (2018). *Document Object Model*. Retrieved from https://en.wikipedia.org/wiki/Document_Object_Model