

Visual Analysis of Wikipedia Edits

Vast 2008 Mini-Challenge 1

Devdhar Patel | Compsci 690v | 12/18/2017

# Introduction

The VAST 2008 mini-challenge 1 focuses on the analysis of edits on the Wikipedia page of the Parisio movement. The Paraiso movement is controversial and is having considerable social impact in a specific area of the world. This fictional movement is similar to many real-world movements and revolutions and just like in any real-world situation, there are different factions who support and oppose the movement. The task of the mini-challenge is to identify these factions and the people in it by analyzing the Wikipedia edits and the terse comments of these edits. The dataset does not give us the actual changes in the edits which adds to the challenge of the task.

# Dataset

The data is in the form of a text file. Each line represents an edit. An example of one edit:

*# (cur) (last) 23:44, 15 January 2007 Alonzo (Talk | contribs) m (100,571 bytes) (?Scientific criticism of Paraiso beliefs - link)*

We can get the following information from the sample:

1. Timestamp
2. Username
3. Minor edit (represented by the highlighted m)
4. Page size after the edit (in Bytes)
5. Comment/Description of the edit

The list became the dimensions/columns of my data. From the above data, there were a few quick ways to analyse the data and categorise the users. Therefore, I tried those methods to see if they would give an acceptable result so that I could build my solution on top of it.

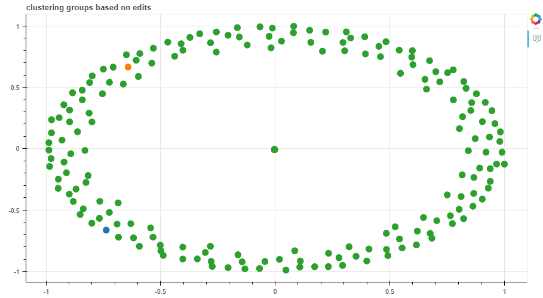
# Solutions Exploration

The efforts described in this section can be viewed in the jupyter notebook “Initial Solutions Exploration.ipynb”.

## 1. Term Frequency-Inverse document Frequency feature extraction and clustering

The first step was to use the Tf-idf feature extraction and cluster the users according to it. The Tf-idf feature extraction method was the first since it is most commonly used to group documents of text together. One document was the text of all the edits of one user appended together. Therefore, the number of documents is equal to the number of users. The idea was that users more inclined towards a particular subject might use a particular set of words more often than less inclined users. These words would be identified as features, and upon clustering the documents/users, we would be able to identify the different factions.

### Result

The clustering did not give the desired results. Almost all the users were grouped into one cluster. The reason for this is because the size of the comments is very small and a lot of comments contain similar language generally used with edits. An example of the top features selected for each cluster when we extract features with the n-gram range of 1 to 2 are:

**Cluster 1:** 'template links', 'xxxxxxxxx gregoria', 'unsourced opinion'

**Cluster 2:** 'extraneous information', 'auditing extraneous', 'edits 75'

**Cluster 3:** 'assertion lead', 'lead citation', 'mighty'

As we can see above, usernames are sometimes picked as features. This is problematic since people from both factions could use the username to attack, defend or support the username. Some of the features picked such as 'edits 75' do not have any meaning. Almost all of the reverts are grouped into one cluster which is not what we want since reverts are the means of edit warring between the different factions. Therefore, this method was not suitable for classifying users into factions.

## 2. Creating groups via reverts and mentions

The second method is the most obvious one. It is logical that people from one faction are more likely to revert the edits from the opposing factions and not the edits from their own faction. It would also be the case that users from the same faction would revert the edits back to the edits of users from their own faction. Based on this, all the comments were a username was mentioned were analyzed and a relationship matrix of each user with another user was created. The rules for the relationship matrix were:

1. if the mention is used in a positive way, we will add 1 to the relationship
2. if the mention is used in a negative way, we will subtract 1 to the relationship
3. All relationships start at 0

The positive and negative mentions were determined by whether the word ‘to’ appeared before the username mention. This is because most of the usernames are mentioned in a negative context. When an edit is reverted ‘to’ the edit of a user, the username is mentioned in a positive context.

### Result

Most of the relationships identified were negative. This is unsurprising as most of the usernames are only mentioned in a negative way and there are very few positive comments with username mentions.

Based on the relationship matrix, all the users were grouped into four groups:

1. Bots
2. Neutral users
3. Pro Parisio
4. Against Parisio

We started grouping from the most active user, i.e. the user with the most non-zero values in their row in the relationship matrix. We grouped users for and against each other starting from the most active users to the least active users. The remaining non-bot users were put into the neutral group. This approach gave a result that was very similar to the final answer of mini-challenge 1. However, there were a few problems in this:

1. The relationship was categorised as positive only if the word ‘to’ appeared before the username. This was based on the data of Wikipedia edits of another page: “2016-17 Kashmir unrest”. For the current database, there were very few positive relationships appearing. Therefore, we needed more ways to extract the positive relationships.
2. For many edits, the username is not mentioned during revert and this method does not take that into account.
3. For many edits, the username for the edit to which the revert is performed is not mentioned and this method does not take that into account.
4. The relationship does not depend on the other words. For example, a revert that insults a user in the comment and a normal revert are given the same weight.

## 3. sentiment analysis

The last method to categorise the users was according to the sentiment of their comments. This method would categorise the users into positive, negative and neutral groups. Each user would be categorised by the sentiment of all their comments combined.

### Result

Sentiment analysis on the comments of edits did not give the desired result. I believe it is because the sentiment analyser that was used was designed for larger documents. For example:

?Membership - cleaning up my own reference mess

The above sentence is categorised as negative with a compound of -0.36. (Compound values higher than 0.2 are categorised as positive and lesser than -0.2 are categorised as negative). However, it refers to the users own commit and therefore the sentiment here is not negative.

However, when the users are categorised, the positive group contains many of the most active users who are anti-Parisio and the negative group contains many of the most active users who are pro-Parisio. This suggests that most of the edits of the anti-Parisio factions are positive and most of the edits of the pro-Parisio faction are negative. Therefore, this method could be used to show the general sentiments of each faction and group the users into those factions. However, this result was discovered after I solved the challenge and could compare the final answer with the result of this method. Therefore, I will not be using this method in my final solution.

# Final Approach

After trying out the several different methods to solve the problem as described above, I decided to build upon the second method since it gave the most promising results. For the visualisation, instead of a relationship matrix heat map, I chose a conflict network since it better represents the relationship between entities and is easier to decipher.

## Initial data visualization

Before starting any data analysis, it is important to get a feel of the data that you will be working on. The best way to do this is by a visualisation. The initial data visualisation serves this purpose.

This visualization can be viewed in the file: “Initial Data Visualization.ipynb”.

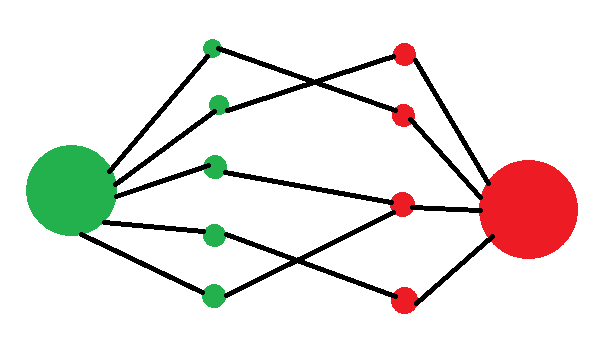
Each of the dot represents and edit. The size of the dot represents the size of the edit. The x-axis represents time and the y-axis represents the users. The colour blue represents bytes added and the colour red represents bytes removed.

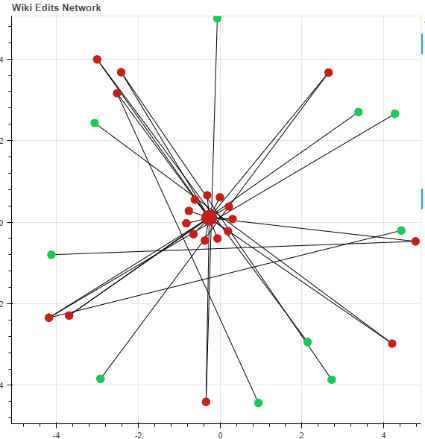
Upon inspecting the edits and the visualisation, we can make the following observations:

1. There is a huge difference in between the big edits and the small edits.
2. The big red edits are due to cases of extreme vandalism where the entire Wikipedia page has been removed or replace with a picture or a sentence, etc.
3. The big blue edits are due to the page being restored back after it has been deleted or replaced. Therefore, we can observe that every big red dot is followed by a big blue dot on the timeline.
4. There are periods of activity and inactivity in the edits.

## Initial Network

Based on the above observation, we can very easily categorise the big edits into pro-Parisio and anti-Parisio depending on whether bytes were added or removed in the edit. The vandals are anti-Parisio and the users who restore the page are pro-Parisio. Based on this assumption, the users responsible for the big edits are categorised. To create a conflict network, where it is easy to decipher the users from each faction, I decided to create a conflict network based on the following idea:



The idea is to create two big nodes, that represent the anti-Parisio and pro-Parisio sentiment and these nodes will have positive weights with the anti-Parisio and pro-Parisio users. The users from each faction will have negative edges with users from the opposing factions. This would create a visual partition on the conflict network between the two networks. However, I soon realized that the users who revert the vandals cannot be called pro-Parisio as they could be neutral too. Therefore, we can only comment on the faction of the vandals with some confidence. Therefore, the initial network turned out to be a lot different from the idea:

This can be viewed in “Final Solution.ipynb”.

## Getting conflict edits

Conflict edits are all the edits that mention the username of a person or have words related the rv., undo, undid, etc. in their comment. The next step would be to create relationships from these edits by extracting the sentiment and strength of the sentiment toward the user. I am focusing on particular actions like reverts and mentions. To get an idea of the context of each comment, I built a visualisation that shows all the selected edits. Upon clicking each edit, we can view the user, comment, bytes changed, previous edit, etc. The usernames are highlighted which makes it easier to determine the context in which the username is used. From this visualisation, it is easy to determine that most of the reverts where the username is not mentioned refer to the previous edit. This is useful to extract the username when no username is given. From this visualisation, we also extract a few edits that are not conflicts since the user is reverting his/her own edit. We remove these edits from the conflict edits. This visualization can be viewed in “Final Solution.ipynb”.

## Tf-idf feature extraction

After finalising the conflict edits, features are extracted from the conflict edits by the Tf-Idf method. This is done so that we can get the most relevant and the most commonly used words in the edits and attach weights to them. To help in selecting the features, I created another visualization similar to the previous one where we can select a feature and view all the edits that contain that feature. We also highlight the username and features to help identify the context and the structure related to each feature.

We select the following features:

1. Undo/undid
2. Revert/reverted/rv/rvt/reverting
3. Vandalism
4. Pov
5. Good faith

These features are selected as they are the most meaningful and appear in most of the conflict edits. We can see from the visualisation that most of the edits with the features undo/undid and Revert/reverted/rv/rvt/reverting have a similar structure. We use this structure to extract the for-user and against-user for each edit.

This visualization can be viewed in “Final Solution.ipynb”.

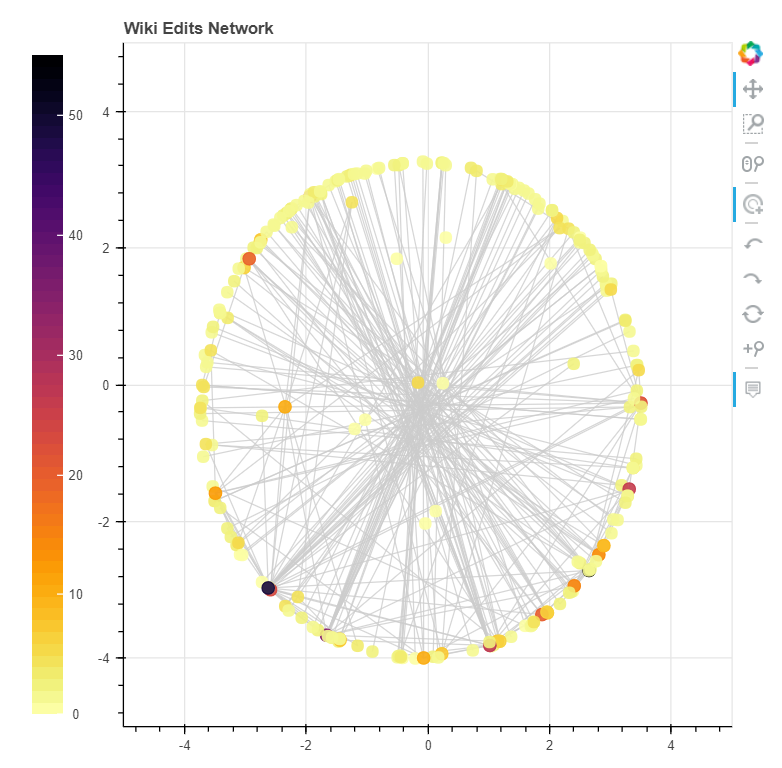
## Attaching weights to features

We attach the following weights to features and actions:

1. Revert:-3
2. Accusation:-1 (keyword:pov; this means that the user is accused of adding their point of view into the page without any reference.)
3. Vandalism:-2
4. For: 3
5. Good faith:3

The revert represents when a user reverts another users edit. The for represents when a user reverts to another users edit. These actions are opposite to each other and therefore are given opposite weights. Accusation is a mildly negative activity and therefore is given the weight -1. Vandalism is a very negative word and therefore is given the weight -2. When ‘good faith’ is used in a revert, the user does not harbour any negative sentiment and therefore to neutralize the revert, it is given the weight of 3. These weights have been set after a lot of testing and are finetuned for this particular dataset.

## Final conflict network

After taking into account all the selected features and actions. We create a final conflict network. Since we do not yet know the faction of each user, we shade each node according to their impact. Users with a very high number of conflicts are shaded darker than users with few conflicts.

We can see from the graph that most of the users have less than five conflicts. The visualisation also has a button to hide the low impact users. Uplon clicking any of the nodes, we can view the relationship of that particular user with other users.

This visualization can be viewed by running the python file: conflictNetwork-shaded.py

By running the following command:

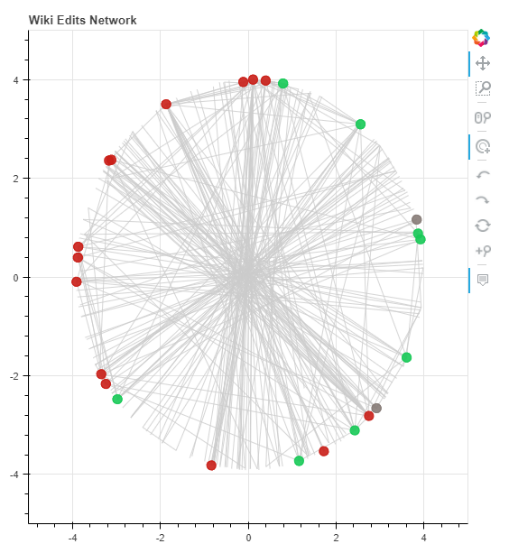
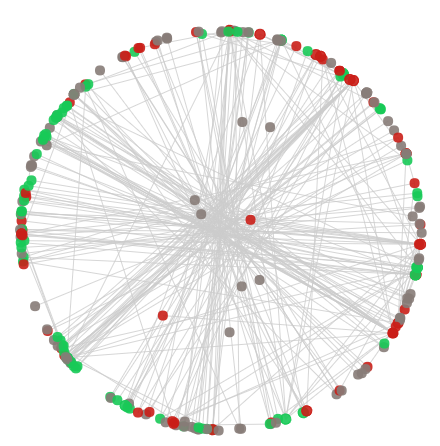
bokeh serve --show conflictNetwork-shaded.py

## categorizing into factions

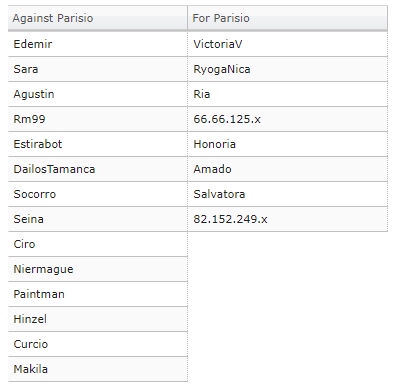
Along with the data given to us, we are also given the wiki discussion page for the Parisio movement. Upon reading through this page, we can see that the user VictoriaV is strongly pro-Parisio. This is also evident in our visualisation as the user is also has the second highest conflicts. On selecting the user in our visualisation, we can confirm the pro-Parisio sentiments of the user. Thus we categorise VicoriaV into the pro-Pariso faction. We then categorise all the users with positive weights with VictoriaV into the pro-Parisio faction and the users with negative weights with VictoriaV into the anti-Parisio faction. We then do this for all users going from the highest impact user to the lowest impact user. Thus we have effectively categorised all the users into factions. The users without factions are neutral. Since we are supposed to identify the major players in each faction, to improve our confidence in the grouping, we remove the low impact users from each faction. The conflict network with each faction coloured can be viewed by running the python file: : conflictNetwork.py

By running the following command:

bokeh serve --show conflictNetwork.py



## factions

****

# Conclusion

The method used to solve this mini-challenge can be used for any Wikipedia page where edit warring occurs. The interactive conflict network is a great visualization for viewing the conflicts and edit warring. This solution can be used for many other types of data like debates, tweets, code commits, etc. with some modification.

## Future work

There are a few questions that can be answered with a little more work:

1. Are users hiding behind IPs?
2. Which events trigger edit activity?

To answer the first question, we would need to analyse the time patterns of edits of each user and the language of comments to find a matching signature between the ip user and actual user. I started work on this but was unable to complete it. I have added the initial visualisation for it in the editFrequency.py. (bokeh serve –show editFrequency.py)