Development of an Advanced Method for the Analysis of Topics and Events on Twitter and their Evolution

BY
GIORGIO CAVAGGION
Laurea, Politecnico di Milano, Milan and Politecnico di Torino, Turin, Italy, 2011

THESIS
Submitted as partial fulfillment of the requirements for the degree of Master of Science in Computer Science in the Graduate College of the University of Illinois at Chicago, 2013

Chicago, Illinois

Defense Committee:
Andrew Johnson, Chair and Advisor
Jason Leigh
Pier Luca Lanzi, Politecnico di Milano
This thesis is dedicated to my parents, for their love and constant support.
ACKNOWLEDGMENTS

I wish to thank, first and foremost, Professor Andrew Johnson and Professor Pier Luca Lanzi for their invaluable support that guided me to the completion of this thesis.

I am also indebted to my friends and colleagues who shared the fantastic experience at UIC with me, contributing to make it stimulating and enjoyable. A special thanks goes to Michele Spagnuolo with whom I shared projects and adventures.

Moreover I would like to thank Giovanni Cavaggion for his support and valuable contribution to the analysis of the 2013 Italian elections.

My sweetest thanks go to Alice Merli, for her love and patience and to my family, whose constant support made this experience possible.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>INTRODUCTION</th>
<th>METHOD FOR THE ANALYSIS AND VISUALIZATION OF THE CONTENT OF TWEETS AND ITS EVOLUTION</th>
<th>METHOD FOR THE ANALYSIS AND VISUALIZATION OF PATTERNS IN MENTIONS AND RETWEETS AND THEIR EVOLUTION</th>
<th>METHOD FOR THE IDENTIFICATION OF OPINION MAKERS AND INFLUENTIAL TWEETS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.1 A new method</td>
<td>2.1 State of the art methods for the content analysis of tweets</td>
<td>3.1 Mentions and retweets</td>
<td>4.1 Influence on Twitter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.2 Tag clouds as a mean to visualize changes in text content</td>
<td>3.2 State of the art researches</td>
<td>4.2 Method for the identification of authoritative users in a graph of mentions</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.3 Algorithm for the creation of a tag cloud displaying the content of tweets</td>
<td>3.3 Force-directed graph drawing algorithms</td>
<td></td>
</tr>
</tbody>
</table>
### TABLE OF CONTENTS (continued)

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.3</td>
<td>Method for the identification of opinion makers and influential tweets in a graph of retweets</td>
</tr>
<tr>
<td>5</td>
<td>METHOD FOR THE ANALYSIS OF THE SENTIMENT OF TWEETS AND ITS EVOLUTION</td>
</tr>
<tr>
<td>5.1</td>
<td>State of the art methods for the sentiment analysis of tweets</td>
</tr>
<tr>
<td>5.1.1</td>
<td>Emoticon analysis</td>
</tr>
<tr>
<td>5.1.2</td>
<td>Single words methods</td>
</tr>
<tr>
<td>5.1.3</td>
<td>Text classification</td>
</tr>
<tr>
<td>5.2</td>
<td>My method</td>
</tr>
<tr>
<td>5.2.1</td>
<td>Emoticon classification</td>
</tr>
<tr>
<td>5.2.2</td>
<td>Logistic regression classifier</td>
</tr>
<tr>
<td>5.3</td>
<td>Dataset</td>
</tr>
<tr>
<td>5.4</td>
<td>Performances</td>
</tr>
<tr>
<td>5.4.1</td>
<td>Emoticon classifier</td>
</tr>
<tr>
<td>5.4.2</td>
<td>Logistic regression classifier</td>
</tr>
<tr>
<td>5.4.3</td>
<td>N-Gram number</td>
</tr>
<tr>
<td>5.4.4</td>
<td>Threshold</td>
</tr>
<tr>
<td>5.5</td>
<td>Overall Sentiment Index</td>
</tr>
<tr>
<td>6</td>
<td>STATE OF THE ART APPLICATIONS AND POSSIBLE IMPROVEMENTS</td>
</tr>
<tr>
<td>6.1</td>
<td>Characteristics of an application to analyze an event or topic and its evolution over time</td>
</tr>
<tr>
<td>6.2</td>
<td>Twitter analysis and visualization applications</td>
</tr>
<tr>
<td>6.3</td>
<td>Characteristics of the state of the art applications</td>
</tr>
<tr>
<td>6.4</td>
<td>How to improve</td>
</tr>
<tr>
<td>6.4.1</td>
<td>What is missing</td>
</tr>
<tr>
<td>6.4.2</td>
<td>Possible improvements</td>
</tr>
<tr>
<td>7</td>
<td>APPLICATION DESIGN</td>
</tr>
<tr>
<td>7.1</td>
<td>Tools used</td>
</tr>
<tr>
<td>7.2</td>
<td>Connecting to the Twitter API</td>
</tr>
<tr>
<td>7.3</td>
<td>Application layout</td>
</tr>
<tr>
<td>7.3.1</td>
<td>Tab Organization</td>
</tr>
<tr>
<td>7.3.2</td>
<td>Screen layout</td>
</tr>
<tr>
<td>7.4</td>
<td>Inter-tabs functionality</td>
</tr>
<tr>
<td>7.4.1</td>
<td>The search bar - Search, Import and Export tweets</td>
</tr>
<tr>
<td>7.4.2</td>
<td>The Menu</td>
</tr>
<tr>
<td>7.5</td>
<td>Tab 1 - When it’s happening</td>
</tr>
<tr>
<td>7.5.1</td>
<td>The flux graph</td>
</tr>
<tr>
<td>7.5.2</td>
<td>The languages pie chart</td>
</tr>
<tr>
<td>7.5.3</td>
<td>The sources pie chart</td>
</tr>
</tbody>
</table>
# TABLE OF CONTENTS (continued)

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.6 Tab 2 - What is happening</td>
<td>100</td>
</tr>
<tr>
<td>7.6.1 The controller</td>
<td>101</td>
</tr>
<tr>
<td>7.6.2 The tag cloud</td>
<td>101</td>
</tr>
<tr>
<td>7.6.3 The auto-play functionality</td>
<td>102</td>
</tr>
<tr>
<td>7.7 Tab 3 - Who are the opinion makers</td>
<td>104</td>
</tr>
<tr>
<td>7.7.1 Interactivity</td>
<td>105</td>
</tr>
<tr>
<td>7.7.2 Nodes</td>
<td>105</td>
</tr>
<tr>
<td>7.7.3 Edges</td>
<td>106</td>
</tr>
<tr>
<td>7.7.4 Labels</td>
<td>106</td>
</tr>
<tr>
<td>7.7.5 Time range controller</td>
<td>108</td>
</tr>
<tr>
<td>7.7.6 Auto-play functionality</td>
<td>109</td>
</tr>
<tr>
<td>7.8 Tab 4 - Where it’s happening</td>
<td>110</td>
</tr>
<tr>
<td>7.8.1 The map</td>
<td>111</td>
</tr>
<tr>
<td>7.8.2 Geocoding</td>
<td>111</td>
</tr>
<tr>
<td>7.8.3 Size by followers</td>
<td>112</td>
</tr>
<tr>
<td>7.8.4 Tooltip</td>
<td>113</td>
</tr>
<tr>
<td>7.8.5 Time range controller</td>
<td>114</td>
</tr>
<tr>
<td>7.8.6 Auto play functionality</td>
<td>114</td>
</tr>
<tr>
<td>7.9 Tab 5 - How people feel</td>
<td>116</td>
</tr>
<tr>
<td>7.9.1 The sentiment pie chart</td>
<td>117</td>
</tr>
<tr>
<td>7.9.2 The sentiment graph</td>
<td>117</td>
</tr>
<tr>
<td>7.9.3 Options</td>
<td>118</td>
</tr>
<tr>
<td>7.10 Tab 6 - Credits</td>
<td>119</td>
</tr>
<tr>
<td>8 TEST CASES</td>
<td>121</td>
</tr>
<tr>
<td>8.1 Visualization of the graph of retweets after Pope Benedict XVI's resignation</td>
<td>121</td>
</tr>
<tr>
<td>8.2 Visualization of the pattern of retweets during the 2013 tour of Justin Bieber</td>
<td>125</td>
</tr>
<tr>
<td>8.2.1 March, 04 2013 - London</td>
<td>126</td>
</tr>
<tr>
<td>8.2.2 March, 16 2013 - Barcelona</td>
<td>127</td>
</tr>
<tr>
<td>8.2.3 March, 19 2013 - Paris</td>
<td>128</td>
</tr>
<tr>
<td>8.2.4 March, 23 2013 - Bologna</td>
<td>129</td>
</tr>
<tr>
<td>8.3 Analysis and visualization of tweets in the 2013 Italian elections</td>
<td>130</td>
</tr>
<tr>
<td>8.3.1 The candidates</td>
<td>131</td>
</tr>
<tr>
<td>8.3.2 Flux analysis - identifying the main events</td>
<td>133</td>
</tr>
<tr>
<td>8.3.3 Event (a) - 02/12/2013</td>
<td>134</td>
</tr>
<tr>
<td>8.3.4 Event (b) - 02/15/2013</td>
<td>136</td>
</tr>
<tr>
<td>8.3.5 Event (c) - 02/17/2013</td>
<td>140</td>
</tr>
<tr>
<td>8.3.6 Event (d) - 02/18/2013</td>
<td>142</td>
</tr>
<tr>
<td>8.3.7 Event (e) - 02/20/2013</td>
<td>144</td>
</tr>
<tr>
<td>8.3.8 Event (f) - 02/22/2013</td>
<td>146</td>
</tr>
</tbody>
</table>
# TABLE OF CONTENTS (continued)

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.3.9 Event (g) - 02/25/2013</td>
<td>148</td>
</tr>
<tr>
<td>8.3.10 Event (h) - 02/26/2013</td>
<td>152</td>
</tr>
<tr>
<td>8.3.11 Prediction</td>
<td>157</td>
</tr>
<tr>
<td>9 IMPROVEMENTS TO THE STATE OF THE ART</td>
<td>161</td>
</tr>
<tr>
<td>9.1 Flux analysis and general information</td>
<td>161</td>
</tr>
<tr>
<td>9.2 Content analysis</td>
<td>166</td>
</tr>
<tr>
<td>9.3 Patterns of mentions and retweets</td>
<td>168</td>
</tr>
<tr>
<td>9.4 Geolocation</td>
<td>171</td>
</tr>
<tr>
<td>9.5 Sentiment analysis</td>
<td>174</td>
</tr>
<tr>
<td>9.6 Retrieved tweets and import/export functionality</td>
<td>177</td>
</tr>
<tr>
<td>10 CONCLUSION</td>
<td>179</td>
</tr>
<tr>
<td>CITED LITERATURE</td>
<td>181</td>
</tr>
<tr>
<td>VITA</td>
<td>184</td>
</tr>
</tbody>
</table>

# LIST OF TABLES

<table>
<thead>
<tr>
<th>TABLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>46</td>
</tr>
<tr>
<td>II</td>
<td>46</td>
</tr>
<tr>
<td>III</td>
<td>49</td>
</tr>
<tr>
<td>IV</td>
<td>50</td>
</tr>
<tr>
<td>V</td>
<td>51</td>
</tr>
<tr>
<td>VI</td>
<td>79</td>
</tr>
<tr>
<td>VII</td>
<td>92</td>
</tr>
<tr>
<td>VIII</td>
<td>152</td>
</tr>
<tr>
<td>IX</td>
<td>153</td>
</tr>
</tbody>
</table>

LIST OF POSITIVE EMOTICONS

LIST OF NEGATIVE EMOTICONS

FREQUENCY OF POSITIVE EMOTICONS

FREQUENCY OF NEGATIVE EMOTIONS

EMOTICON CLASSIFIER’S CONFUSION MATRIX

CHARACTERISTICS OF THE STATE OF THE ART APPLICATIONS

TWITTER API OPERATORS

ELECTION RESULTS - SENATE

ELECTION RESULTS - CHAMBER OF DEPUTIES
### LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Flow graph of the algorithm to create a tag cloud from tweets</td>
</tr>
<tr>
<td>2</td>
<td>Tag cloud of tweets without filtering</td>
</tr>
<tr>
<td>3</td>
<td>Tag cloud of tweets with filtering</td>
</tr>
<tr>
<td>4</td>
<td>Tag cloud at 11 am</td>
</tr>
<tr>
<td>5</td>
<td>Tag cloud at 1 pm</td>
</tr>
<tr>
<td>6</td>
<td>Tag cloud at 3 pm</td>
</tr>
<tr>
<td>7</td>
<td>Layout of a social network, Fruchterman-Reingold vs ForceAtlas2</td>
</tr>
<tr>
<td>8</td>
<td>Layout of a graph of mentions - FA2, FA2 with no-overlap and Frucht.-Reing.</td>
</tr>
<tr>
<td>9</td>
<td>Layout of a graph of retweets - FA2, FA2 with no-overlap and Frucht.-Reing.</td>
</tr>
<tr>
<td>10</td>
<td>Flow graph of the algorithm to create a graph of mentions</td>
</tr>
<tr>
<td>11</td>
<td>Example of the creation of a graph of mentions</td>
</tr>
<tr>
<td>12</td>
<td>Flow graph of the algorithm to create a graph of retweets</td>
</tr>
<tr>
<td>13</td>
<td>Example of the creation of a graph of retweets</td>
</tr>
<tr>
<td>14</td>
<td>Layout of a graph of mentions using area=10000, area=6N and area=N</td>
</tr>
<tr>
<td>15</td>
<td>Layout of a graph of mentions using a gravity of 1, 3.5 and 10</td>
</tr>
<tr>
<td>16</td>
<td>Layout of a graph of mentions using a speed of 1, 10 and 200</td>
</tr>
<tr>
<td>17</td>
<td>Effects of the ranking algorithm on a network of mentions</td>
</tr>
<tr>
<td>18</td>
<td>Effects of the modularity algorithm on a network of mentions</td>
</tr>
<tr>
<td>19</td>
<td>Evolution of a network of mentions over time</td>
</tr>
<tr>
<td>20</td>
<td>Evolution of a network of retweets over time</td>
</tr>
<tr>
<td>21</td>
<td>Influence index in a network of mentions, nodes = users, edges = mentions</td>
</tr>
<tr>
<td>22</td>
<td>Influence index in a network of RTs, green nodes = tweets, red nodes = users</td>
</tr>
<tr>
<td>23</td>
<td>Influence index update by back propagation</td>
</tr>
<tr>
<td>24</td>
<td>Accuracy of the classifier given n</td>
</tr>
<tr>
<td>25</td>
<td>Accuracy of the classifier given n and threshold</td>
</tr>
<tr>
<td>26</td>
<td>Twitter Browser</td>
</tr>
<tr>
<td>27</td>
<td>Visible Tweets</td>
</tr>
<tr>
<td>28</td>
<td>Tweet Beam</td>
</tr>
<tr>
<td>29</td>
<td>Twitter Map</td>
</tr>
<tr>
<td>30</td>
<td>Foller.me</td>
</tr>
<tr>
<td>31</td>
<td>Tweet Archivist</td>
</tr>
<tr>
<td>32</td>
<td>Mentionmapp</td>
</tr>
<tr>
<td>33</td>
<td>Twitter Counter</td>
</tr>
<tr>
<td>34</td>
<td>Twitonomy</td>
</tr>
<tr>
<td>35</td>
<td>A World Of Tweets</td>
</tr>
<tr>
<td>36</td>
<td>Cloud.li</td>
</tr>
<tr>
<td>37</td>
<td>Monitter</td>
</tr>
<tr>
<td>38</td>
<td>Twitter Character</td>
</tr>
<tr>
<td>FIGURE</td>
<td>PAGE</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>39</td>
<td>Trendsmap</td>
</tr>
<tr>
<td>40</td>
<td>Revisit</td>
</tr>
<tr>
<td>41</td>
<td>Hashtagify</td>
</tr>
<tr>
<td>42</td>
<td>Spot</td>
</tr>
<tr>
<td>43</td>
<td>Year in Hashtag</td>
</tr>
<tr>
<td>44</td>
<td>Sentiment140</td>
</tr>
<tr>
<td>45</td>
<td>Tweereal</td>
</tr>
<tr>
<td>46</td>
<td>Layout of the application</td>
</tr>
<tr>
<td>47</td>
<td>Example of layout</td>
</tr>
<tr>
<td>48</td>
<td>The initial screen of the application</td>
</tr>
<tr>
<td>49</td>
<td>The search bar</td>
</tr>
<tr>
<td>50</td>
<td>The menu</td>
</tr>
<tr>
<td>51</td>
<td>First tab</td>
</tr>
<tr>
<td>52</td>
<td>The flux graph</td>
</tr>
<tr>
<td>53</td>
<td>The languages pie chart</td>
</tr>
<tr>
<td>54</td>
<td>The sources pie chart</td>
</tr>
<tr>
<td>55</td>
<td>Second tab</td>
</tr>
<tr>
<td>56</td>
<td>The tag cloud controller</td>
</tr>
<tr>
<td>57</td>
<td>The tag cloud</td>
</tr>
<tr>
<td>58</td>
<td>Self evolution of the tag cloud</td>
</tr>
<tr>
<td>59</td>
<td>Third tab</td>
</tr>
<tr>
<td>60</td>
<td>Interaction with the graph (default view on the left)</td>
</tr>
<tr>
<td>61</td>
<td>Particular of a graph of mentions</td>
</tr>
<tr>
<td>62</td>
<td>Labels of the graph of mentions - hide mode</td>
</tr>
<tr>
<td>63</td>
<td>Labels of the graph of mentions - show mode</td>
</tr>
<tr>
<td>64</td>
<td>Labels of the graph of mentions - auto mode</td>
</tr>
<tr>
<td>65</td>
<td>Time range controller of the mention and retweet graphs</td>
</tr>
<tr>
<td>66</td>
<td>Time range controller of the mention and retweet graphs</td>
</tr>
<tr>
<td>67</td>
<td>Auto-play functionality of a graph of mentions</td>
</tr>
<tr>
<td>68</td>
<td>Fourth tab</td>
</tr>
<tr>
<td>69</td>
<td>The geocoding functionality</td>
</tr>
<tr>
<td>70</td>
<td>Size of the markers on the map</td>
</tr>
<tr>
<td>71</td>
<td>Map tooltip</td>
</tr>
<tr>
<td>72</td>
<td>Time range controller of the interactive map</td>
</tr>
<tr>
<td>73</td>
<td>Tweets on the interactive map using two different time ranges</td>
</tr>
<tr>
<td>74</td>
<td>Auto-play functionality of the interactive map</td>
</tr>
<tr>
<td>75</td>
<td>Fifth tab</td>
</tr>
<tr>
<td>76</td>
<td>Sentiment pie chart</td>
</tr>
<tr>
<td>77</td>
<td>Sentiment graph</td>
</tr>
<tr>
<td>78</td>
<td>Options of the sentiment classifier</td>
</tr>
<tr>
<td>79</td>
<td>Visualization of retweets after Pope Benedict XVI’s resignation</td>
</tr>
<tr>
<td>80</td>
<td>Clusters of satirical tweets on the resignation of the Pope</td>
</tr>
</tbody>
</table>
### LIST OF FIGURES (continued)

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>81</td>
<td>Clusters of informative tweets on the resignation of the Pope</td>
</tr>
<tr>
<td>82</td>
<td>Retweets during 2013 Justin Bieber’s Europe Tour - London</td>
</tr>
<tr>
<td>83</td>
<td>Retweets during 2013 Justin Bieber’s Europe Tour - Barcelona</td>
</tr>
<tr>
<td>84</td>
<td>Retweets during 2013 Justin Bieber’s Europe Tour - Paris</td>
</tr>
<tr>
<td>85</td>
<td>Retweets during 2013 Justin Bieber’s Europe Tour - Bologna</td>
</tr>
<tr>
<td>86</td>
<td>Events of the 2013 Italian elections</td>
</tr>
<tr>
<td>87</td>
<td>Flux of tweets to Monti - 02/12/2013</td>
</tr>
<tr>
<td>88</td>
<td>Content of tweets from Monti - 02/12/2013</td>
</tr>
<tr>
<td>89</td>
<td>Content of tweets to Monti - 02/12/2013</td>
</tr>
<tr>
<td>90</td>
<td>Sentiment of tweets to Monti - 02/12/2013</td>
</tr>
<tr>
<td>91</td>
<td>Flux of tweets to Monti - 02/15/2013</td>
</tr>
<tr>
<td>92</td>
<td>Flux of tweets to Bersani - 02/15/2013</td>
</tr>
<tr>
<td>93</td>
<td>Content of tweets from Monti - 02/15/2013</td>
</tr>
<tr>
<td>94</td>
<td>Content of tweets to Monti - 02/15/2013</td>
</tr>
<tr>
<td>95</td>
<td>Sentiment of tweets to Monti - 02/15/2013</td>
</tr>
<tr>
<td>96</td>
<td>Content of tweets from Bersani - 02/15/2013</td>
</tr>
<tr>
<td>97</td>
<td>Content of tweets to Bersani - 02/15/2013</td>
</tr>
<tr>
<td>98</td>
<td>Sentiment of tweets to Bersani - 02/15/2013</td>
</tr>
<tr>
<td>99</td>
<td>Flux of tweets to Grillo - 02/17/2013</td>
</tr>
<tr>
<td>100</td>
<td>Content of tweets from Grillo - 02/17/2013</td>
</tr>
<tr>
<td>101</td>
<td>Content of tweets to Grillo - 02/17/2013</td>
</tr>
<tr>
<td>102</td>
<td>Sentiment of tweets to Grillo - 02/17/2013</td>
</tr>
<tr>
<td>103</td>
<td>Flux of tweets to Berlusconi - 02/18/2013</td>
</tr>
<tr>
<td>104</td>
<td>Content of tweets from Berlusconi - 02/18/2013</td>
</tr>
<tr>
<td>105</td>
<td>Content of tweets to Berlusconi - 02/18/2013</td>
</tr>
<tr>
<td>106</td>
<td>Sentiment of tweets to Berlusconi - 02/18/2013</td>
</tr>
<tr>
<td>107</td>
<td>Flux of tweets to Giannino - 02/20/2013</td>
</tr>
<tr>
<td>108</td>
<td>Content of tweets from Giannino - 02/20/2013</td>
</tr>
<tr>
<td>109</td>
<td>Content of tweets to Giannino - 02/20/2013</td>
</tr>
<tr>
<td>110</td>
<td>Sentiment of tweets to Giannino - 02/20/2013</td>
</tr>
<tr>
<td>111</td>
<td>Flux of tweets to Grillo - 02/22/2013</td>
</tr>
<tr>
<td>112</td>
<td>Content of tweets from Grillo - 02/22/2013</td>
</tr>
<tr>
<td>113</td>
<td>Content of tweets to Grillo - 02/22/2013</td>
</tr>
<tr>
<td>114</td>
<td>Sentiment of tweets to Grillo - 02/22/2013</td>
</tr>
<tr>
<td>115</td>
<td>Flux of tweets - 02/25/2013</td>
</tr>
<tr>
<td>116</td>
<td>Content of tweets to Bersani - 02/25/2013</td>
</tr>
<tr>
<td>117</td>
<td>Content of tweets to Grillo - 02/25/2013</td>
</tr>
<tr>
<td>118</td>
<td>Content of tweets to Monti - 02/25/2013</td>
</tr>
<tr>
<td>119</td>
<td>Content of tweets to Ingrò - 02/25/2013</td>
</tr>
<tr>
<td>120</td>
<td>Flux of tweets - 02/26/2013</td>
</tr>
<tr>
<td>121</td>
<td>Content of tweets to Grillo - 02/26/2013</td>
</tr>
<tr>
<td>122</td>
<td>Content of tweets to Ingrò - 02/26/2013</td>
</tr>
<tr>
<td>FIGURE</td>
<td>PAGE</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>123</td>
<td>156</td>
</tr>
<tr>
<td>124</td>
<td>157</td>
</tr>
<tr>
<td>125</td>
<td>159</td>
</tr>
<tr>
<td>126</td>
<td>162</td>
</tr>
<tr>
<td>127</td>
<td>162</td>
</tr>
<tr>
<td>128</td>
<td>163</td>
</tr>
<tr>
<td>129</td>
<td>163</td>
</tr>
<tr>
<td>130</td>
<td>164</td>
</tr>
<tr>
<td>131</td>
<td>164</td>
</tr>
<tr>
<td>132</td>
<td>165</td>
</tr>
<tr>
<td>133</td>
<td>166</td>
</tr>
<tr>
<td>134</td>
<td>167</td>
</tr>
<tr>
<td>135</td>
<td>168</td>
</tr>
<tr>
<td>136</td>
<td>169</td>
</tr>
<tr>
<td>137</td>
<td>170</td>
</tr>
<tr>
<td>138</td>
<td>170</td>
</tr>
<tr>
<td>139</td>
<td>171</td>
</tr>
<tr>
<td>140</td>
<td>172</td>
</tr>
<tr>
<td>141</td>
<td>173</td>
</tr>
<tr>
<td>142</td>
<td>174</td>
</tr>
<tr>
<td>143</td>
<td>175</td>
</tr>
<tr>
<td>144</td>
<td>176</td>
</tr>
<tr>
<td>145</td>
<td>176</td>
</tr>
</tbody>
</table>
The aim of this thesis is to develop an advanced method for the analysis and visualization of topics and events on Twitter and their evolution over time.

In the first chapters I will illustrate the four components of the proposed method, which are content analysis, analysis of the patterns of mentions and retweets, identification of opinion makers and influential tweets, and sentiment analysis.

After that, I will describe how my method has been implemented in an application for the analysis and visualization of tweets.

Finally, some test cases and comparisons with other applications will be used to show which improvements to the state of the art have been achieved.
CHAPTER 1

INTRODUCTION

Twitter is a huge source of knowledge for identifying and analyzing events and topics, mainly because through its API it provides a large amount of public tweets which can be analyzed to obtain a real time feedback of an event or topic, including its magnitude, its evolution over time and space, what popular sentiment is with respect to it and how this sentiment changes over time, how the information spreads, who are the main vectors of the spreading etc.

Twitter presents information to its users in form of a stream of tweets determined by who a user is following. It has evolved in recent years from a platform to exchange opinions between friends to an extremely powerful tool for sharing information about events and hot topics (1). For this reason, researchers have become increasingly interested in Twitter in recent years.

As observed by Teevan et al. (2), people use Twitter mainly as a source to retrieve two kinds of information:

- Social information: finding other twitter users, to discover their thoughts or to have an overview of people’s opinion on a particular topic.
- Trending information: finding news, trends, hot topics, famous events or events attended by colleagues.

Another interesting aspect which emerged from Teevan’s research is that people tend to repeat the same query on Twitter more than once, to monitor changes and evolutions of the results. This operation
is made complicated by the manner in which Twitter allows to explore its contents; in fact it’s very difficult to identify patterns in topics and events by just looking at the raw tweets presented in chronological order by Twitter, especially when the stream is overcrowded or full of noisy self-informing tweets, since most Twitter users still use this platform to tweet about their thoughts or personal events, while just about the 20% of users use it to share information of public interest (3).

To overcome the limited expressive power of a raw stream, many researches have focused on the analysis and visualization of tweets in a great variety of fields, from politics (4,5) to natural disasters (6). Unfortunately most of these works are either focused on very specific topics and/or based on an offline analysis of previously collected tweets.

As highlighted by Marcus et al. (7), there is the lack of a sound method which can be used by an application to perform an analysis on an unconstrained dataset of real-time collected tweets and visualize the results.

1.1 A new method

The aim of this thesis is to develop a method for the analysis and visualization of events and topics on Twitter and their evolution over time. This method has four main components:

- Content analysis of tweets (Chapter 2)
- Patterns analysis of mentions and retweets (Chapter 3)
- Identification of opinion makers and influential tweets (Chapter 4)
- Sentiment analysis (Chapter 5)
The method will be implemented in an application which fills a gap in the present state of the art, allowing a user to deeply understand the dynamics of an event or topic by the analysis and visualization of real-time collected tweets.

After describing the components of my method, I will analyze the state of the art applications (Chapter 6) and show which improvements can be generated by the implementation of my approach.

I will then describe the design of my application (Chapter 7), illustrate some test cases (Chapter 8) and point out which improvements to the state of the art have been achieved (Chapter 9).
CHAPTER 2

METHOD FOR THE ANALYSIS AND VISUALIZATION OF THE CONTENT OF TWEETS AND ITS EVOLUTION

The aim of the analysis and visualization method described in this chapter is to obtain a dynamic tag cloud from an unconstrained dataset of tweets retrieved from the Twitter API. Through the implementation of this method into my application, the user will be able to visualize a tag cloud, which changes, showing the evolution of the content of tweets over time.

In Section 2.1 the state of the art methods for the content analysis of tweets are presented; Section 2.2 describes the advantages of using a tag cloud for representing the content of tweets; in Section 2.3 my method for the creation of a tag cloud of tweets is presented; Section 2.4 illustrates the filtering process of my algorithm; while in Section 2.5 is described how to give to a tag cloud the ability to evolve over time.

2.1 State of the art methods for the content analysis of tweets

Since a chronologically ordered list of tweets is an heterogeneous and ineffective way to represent their content, in recent years some researches and applications have been developed in order to find a way to represent the content of tweets in a more intuitive and meaningful way.

Many researchers focus on studying the presence of hashtags in tweets, since hashtags are an immediate way for having an idea of the content of a tweet and compare it with others (8). In addition to frequency lists, some other techniques have been adopted to represent the use of hashtags in tweets;
for example a Hashtag Graph Model (9), representing a graph where two hashtags (nodes) are linked if they have appeared in at least one tweet together, or a dynamic graph of hashtags (10), representing the diffusion of hashtags over time.

Another classical way of representing the content of tweets is by means of tag clouds. Tag clouds are visually appreciable groups of words, usually alphabetically ordered, whose size is proportional to their frequency in a text or in a group of texts. They have been often used in the analysis of tweets because of their representative power and intuitiveness. Their most typical use is to represent the content of a set of tweets to identify the most frequent words by their size, but they have been also used for some other purposes, for example identifying the most tweeted terms by a user and which of them are used also by others (11), or to measure the trust and importance of users in relation to a given topic (12).

2.2 Tag clouds as a mean to visualize changes in text content

Tag clouds offer the possibility to visualize the key terms of a text at once. The differences in text size let the user immediately focus on the most frequent terms. The effectiveness of tag clouds with respect to a normal list of words has often been discussed. For example research by Gruen et al. (13) showed how people perform better in a categorization task (looking quickly at some words and then naming the 4 categories they belonged to) using a vertical list instead of a tag cloud. Another experiment by Halvey et al. (14) demonstrated that people perform better also in a selection task (finding the name of a country between 10 countries) using lists with respect to tag clouds. But what has not been considered in these studies, as suggested by Hearst et al. (15), is the subjective reaction of people to the two different layouts. In their research they interviewed 20 people working in web design or information visualization; some comments of the consulted people were: "tag clouds are useful for
I think that these answers suggest that lists may be better for performing tasks, but they don’t have the same expressive power of tag clouds in displaying dynamic social data, like tweets. Searching Twitter for a given query returns huge volumes of unstructured text and a visual instrument like a tag cloud can give a snapshot of the content of tweets, allowing the viewer to immediately focus on the key terms. Moreover, a dynamic tag cloud is better at showing changes in the content of tweets over time with respect to a vertical list, since it allows to spot trends and patterns more intuitively.

In the following sections I will describe a method for the construction of a dynamic tag cloud starting from an unconstrained dataset of tweets. With the implementation of this method in my application the user will be able to input a query and visualize a tag cloud which evolves when she selects a different time range. The application will also feature an "auto-play" functionality to let the tag cloud "evolve" by itself, effectively showing changes in the content of tweets over time. The description of the implementation of the method in the application can be found in Section 7.6.

2.3 Algorithm for the creation of a tag cloud displaying the content of tweets

The aim of this algorithm is to create a meaningful tag cloud from an unconstrained dataset of tweets collected from the Twitter API. Figure 1 presents the flow graph\(^1\) of the algorithm.

\(^1\)Dashed lines represent the else paths
Figure 1: Flow graph of the algorithm to create a tag cloud from tweets

1. TagCloud cloud = new TagCloud()
2. FOR every tweet
3. Split tweet in its words
4. FOR every word in tweet
5. IF word starts with " " || " " || ( || [ ...
6. Remove first character of word
7. IF word ends with " " || ; || ! || ? || ] || ]
8. IF word.length() > 2 && word is not an URL or user
9. IF word is not query-related
10. IF word is not a common word || twitter-related
11. cloud.addWord(word)

end FOR
end FOR
As can be seen in Figure 1, the algorithm iterates over each word of each retrieved tweet and adds it to the tag cloud only if it’s not stopped by a series of filtering steps.

2.4 Filtering

Figure 2 shows the result of creating a tag cloud of tweets matching a given query (in this case "Italy") using my algorithm but without performing any filtering step. The size and opacity of each word are proportional to its frequency and the words are displayed in alphabetical order (a more detailed description of the visual aspects of the method will be given in Section 7.6).

![Figure 2: Tag cloud of tweets without filtering](image)

The most frequent terms are: the query term itself (Italy), some common words and prepositions (in, to, the, from) and the nickname of a user, therefore the result is still not very satisfactory.

To make tag clouds much more meaningful, my method includes a filtering process performed on many levels:

- Query related words: Words related to the current query are removed from the cloud since they are obviously appearing in every tweet. For instance, if a user searches "University of Milano" the tag cloud excludes words as University, #university, Milano, #milano etc.

\(^1\)performed on 01/27/2012
• Common words: Since common words, like for instance prepositions, are preponderant in tweets but not meaningful, common words in English, Spanish and Italian are excluded.

• Final and initial characters: parentheses, quotes, and many other punctuation symbols are removed from the tag cloud.

• Words shorter than 3 characters: words with less then 3 characters are removed from the tag cloud, since they are often not meaningful. An exception has been made for smileys.

• URLs: tweets often contain media URLs; they are removed from the tag cloud.

• Users: nicknames of users are removed to avoid popular users to monopolize the tag cloud. The most mentioned users will be visualized in a dedicate tab of the application.

• Twitter-related terms: terms like “RT”, which stands for retweet, are not considered when building the cloud.

As can be seen in Figure 3, performing the same query of Figure 2, but applying the previously described filters, produces a much more meaningful tag cloud; the most frequent terms are now: *love*, *hope*, *flashmob*, *Liga* (the nickname of a famous Italian singer), *Naples*, *Milan*, *Berlusconi* etc.
2.5 Evolution over time

The addition of the time dimension to tag clouds makes them more powerful, allowing them to show changes in the content of tweets over time. Figure 4, Figure 5 and Figure 6 show the evolution of the most frequent terms in tweets related to the same query ("Italy") performed on January, 27 2013, focusing on three different time steps (11 am, 1 pm and 3 pm).

![Figure 4: Tag cloud at 11 am](image)

At 11 am some people are asking Cody Simpons, an Australian singer, to please come to Italy, while others express their love for this country.

![Figure 5: Tag cloud at 1 pm](image)
At 1 pm some users are talking about the American Singer Austin Mahone and his tour in Europe (#mahonebacktoeurope), some others are listening to music (#soundtracking) or talking about the beautiful places of Italy (#amalficoast, #capri).

![Figure 6: Tag cloud at 3 pm](image)

At 3 pm many people are commenting about Berlusconi saying that Mussolini was influenced by Hitler when performing his bad actions.

From this example it’s clear how giving a tag cloud the ability of evolving over time to reflect changes in the content of tweets allows to obtain a powerful instrument for tracking people’s opinion and its evolution on a given topic or event.
CHAPTER 3

METHOD FOR THE ANALYSIS AND VISUALIZATION OF PATTERNS IN MENTIONS AND RETWEETS AND THEIR EVOLUTION

The aim of the analysis and visualization method described in this chapter is the construction and layout of timed directed graphs which show the evolution of the patterns of mentions and retweets over time. The layout of the graphs is performed by tuning the Fruchterman-Reingold algorithm to obtain an attractive visualization of networks of mentions and retweets. The implementation of the method in my application (described in Section 7.7) will allow a user to obtain in a few seconds a graph related to any query on Twitter, explore it and watch it evolving over time, by means of an "auto-play" functionality.

Section 3.1 introduces the concepts of mention and retweet; in Section 3.2 the state of the art searches in this field are presented; Section 3.3 describes two of the most popular force-directed graph drawing algorithms; Section 3.4 presents my method for the creation of a graph of mentions and a graph of retweets; in Section 3.5 my method for the visualization of patterns of mentions and retweets is described; while Section 3.6 shows how to add the temporal dimension to the graphs created with my method.

3.1 Mentions and retweets

A mention is any reference to a username contained in a tweet. Replies are a particular kind of mention which are posted by clicking the Reply button, they begin with the name of the creator of the
original tweet. Replies have "in reply to username" written at the bottom of the tweet. Both mentions and replies of a user can be found in the Mentions tab on the Connect page of any user profile.

A retweet is the sharing of someone else’s tweet. A retweet is distinguished by the retweet icon and the name of the user who retweeted it. Sometimes people write "RT" at the beginning of a retweet to signal that they are quoting the content of someone else.

3.2 State of the art researches

In recent years, some researches have started focusing on retweets and, more marginally, on mentions.

For what concerns mentions, they have been used as a parameter to compute the influence of a user, for example by Morris et al. (16), or to understand how the communication between people has evolved after the advent of this new microblogging platform.

The research on retweets is instead focusing on two main points: the conversational aspects of retweeting (see for example Boyd et al. (17)) and their role and impact in the diffusion of information (see for example (18)). This recent research by Kwak et al. proposes an interesting model for visualizing the retweets contained in a dataset: a "tree" of retweets, that is a sort of graph in which two nodes (tweets) are connected if one is the retweet of the other. Unfortunately, the disconnected components of the graph are considered separately, no ranking algorithm is applied and no time information is associated to the nodes, therefore it’s difficult to have a clear overview of the retweeting pattern.

3.3 Force-directed graph drawing algorithms

Force-directed (or force-based) algorithms are a class of graph drawing algorithms whose aim is to display graphs in a clear and visually appreciable way. The idea behind these algorithms is to simulate
the graph as if it was a physical system. Eades (19) used the metaphor of substituting vertices by steel rings and edges by springs to form a mechanical system, then put the vertices in an initial layout and let the spring forces do the rest.

In the following subsections I will describe two of the most popular force-directed algorithms: Fruchterman-Reingold and ForceAtlas2, comparing their capabilities in displaying networks of mentions and retweets.

3.3.1 The Fruchterman-Reingold algorithm

The Fruchterman-Reingold algorithm (20) is based on two main principles:

1. Vertices connected by an edge should be drawn near each other.
2. Vertices should not be drawn too close to each other.

Vertices are seen as atomic particles or celestial bodies, characterized by repulsive and attractive forces.

In the initial configuration, the vertices are positioned randomly in the space, then the algorithm performs a series of iterations. Each iteration is characterized by three steps:

1. Calculate the effects of attractive forces on each vertex,
2. Calculate the effects of repulsive forces on each vertex,
3. Cooling phase.

The cooling phase consists in decreasing at each iteration the maximum displacement, called temperature, of each vertex; in this way, as the layout becomes better, the final iterations will perform finer adjustments.
In order to reduce the time complexity of the algorithm some simplifications with respect to the physical model have been performed; in fact vertices are considered to be attracted only to their neighbors; this has $O(|E|)$ complexity (where E is the set of edges). Unfortunately computing the repulsion of each vertex has $O(|V|^2)$ complexity (where V is the set of vertices). To reduce also this component of the complexity, a grid-variant of the algorithm has been developed. It divides the screen into a grid of squares and repulsive forces are computed only between vertices in nearby squares.

The total complexity of the algorithm is $O(|V| + |E|)$ if the grid variant is used, $O(|V|^2 + |E|)$ otherwise.

### 3.3.2 The ForceAtlas2 algorithm

The ForceAtlas2 algorithm (21), implemented in the Gephi software\(^1\), combines different techniques to obtain a network which is clearly readable. The main characteristics of the algorithm are:

- **Clarity**: to produce a spatialization that is easily understandable by the users.
- **Efficiency**: reaching the convergence quickly, maintaining a good precision in nodes layout.
- **Versatility**: allow the user to tune the shape of the network.

The energy model is based on the Fruchterman-Reingold and LingLog (22) algorithms, with an addition: the repulsion between the nodes is made dependent on their degree. The idea behind this change is to obtain an algorithm which is optimized to layout web graphs and social networks, which are characterized by few highly connected nodes and by the presence of different "communities", by bringing poorly connected nodes closer to nodes with a higher degree.

---

\(^1\) An interactive visualization and exploration platform for networks (https://gephi.org)
This algorithm has also many options that can be activated in its Gephi implementation by the user. One of the most useful is the "Prevent Overlap" mode; it considers the size of two nodes when computing the distance between them, determined by the attraction and repulsion forces. This option slows down the performances of the algorithm, so it’s better to use it only during the final iterations of the algorithm to refine the layout of the graph.

3.3.3 Comparison of the two algorithms in displaying mentions and retweets

As stated by its creators, the ForceAtlas2 algorithm has been designed to visualize social networks. I compared the two algorithms, or more specifically their implementation in the Gephi software, in displaying a social network.

The dataset I used for this first comparison is the network of my mutual friendship relationships on Facebook. It is an undirected graph composed by 340 nodes and 3374 edges, the average degree is 19.8, while the graph density is 0.059.

![Figure 7: Layout of a social network, Fruchterman-Reingold vs ForceAtlas2](image)
As can be seen in Figure 7, the ForceAtlas2 algorithm (on the right) does a better job clustering the different communities that are present in the network, while using Fruchterman-Reingold (on the left) the communities are still visible because of the high density of the edges but they are not clearly clustered.

Now I will apply both algorithms to a dataset of mentions and one of retweets to compare their layout performances. The datasets have been collected by me with the method explained in the next section and are related to a random query: "Pulcinella" (a classical character of the Italian theater).

The dataset of mentions is a directed graph composed of 269 nodes (users) and 185 edges (an edge from node A to node B means "user A mentioned user B in a tweet"). The average degree is 0.7, while the graph density is 0.003.

The dataset of retweets is a directed graph composed of 576 nodes (tweets) and 401 edges (an edge from node A to node B means "B is a retweet of A"). The average degree is 0.7 and the graph density is 0.001.
As can be seen in Figure 8 and Figure 9, the two graphs are disconnected and characterized by a low density. As pointed out by Fruchterman and Reingold in their work, the problem of disconnected graphs is that there are no edges that hold together the different parts of the graph; because of this, the disconnected parts tend to move away from each other.

This problem is clearly evident in the layouts performed by ForceAtlas2, in which the repulsion forces are dependent on the degree of the nodes, therefore the nodes with a high degree (most mentioned users for mentions and most retweeted tweets for retweets) tend to attract the connected nodes on themselves and to push away the other nodes. ForceAtlas2 without the no-overlap option (on the left of the figures) produces a low quality layout in which edges are not visible at all. This problem is partially solved by activating the no-overlap option (in the center of the figures), which slows down the execution of the algorithm but produces a better result; the different clusters are still too dense internally and too far away from each other.

The Fruchterman-Reingold algorithm (on the right of the figures) avoids the problem of the disconnected parts of the graph flying apart by partitioning the graph into its connected components and
reserving a part of the frame for each component. Another advantage of this algorithm is that the graph
is contained in a circle of a size which can be determined by the user, therefore it is easier to display
it in an application and it’s easier for the user to explore it. This shape is also better for displaying the
labels of the nodes with less overlapping.

For these reasons I chose to base my method on the Fruchterman-Reingold algorithm. As can be
seen from the figures, the algorithm by itself produces a layout which has some advantages but it’s still
not fully satisfactory. A good tuning of the algorithm, described in Section 3.5, with the addition of
ranking and modularity algorithms will produce an optimal result for the visualization of networks of
mentions and retweets.

3.4 Algorithms for the creation of a graph of mentions and one of retweets

In this section I will describe the algorithms I developed to create a graph representing mentions in
tweets and a graph representing retweeting patterns starting from any dataset of tweets.

3.4.1 Algorithm for the creation of a graph of mentions contained in Tweets

The aim of this algorithm is to obtain a directed graph representing the pattern of mentions in an
unconstrained dataset of tweets. Each node of the graph is a user, while an edge from node $A$ to node $B$
means "user $A$ mentioned user $B$ in a tweet". Figure 10 presents the flow graph\(^1\) of the algorithm.

\(^1\)Dashed lines represent the else paths
Figure 10: Flow graph of the algorithm to create a graph of mentions
As can be seen in the flow chart, the algorithm iterates on the tweets and their mentions to add the necessary nodes and edges to the graph, avoiding duplicates.

Since the algorithm iterates only on the tweets which contain at least one mention and then on the mentions, considering that the percentage of tweets containing at least one mention is approximately 53% and the average number of mentions per tweet is approximately 1.2 (23), the time complexity of the algorithm is $O(N)$.

To better show how the algorithm works, let’s suppose to have 3 incoming tweets:

1. **Tweet1**: user *George* mentioning *Paul* and *Alice*.
2. **Tweet2**: user *George* mentioning *Alice*, *Kim* and *Frank*.
3. **Tweet3**: user *Alice* mentioning *Kim*.

![Diagram of graph creation]

Figure 11: Example of the creation of a graph of mentions
Figure 11 shows the steps of the creation of the network and the numbers representing the flow followed in the flow chart of the algorithm.

3.4.2 Algorithm for the creation of a graph representing the pattern of retweets

The aim of this algorithm is to obtain a directed graph representing the pattern of retweets in a dataset of tweets. The tweets are represented as nodes, while an edge from node $A$ to node $B$ means "$B$ is a retweet of $A$", in this way, following the edges allows to follow the flow of information. If a tweet has been retweeted more than $R$ times (this information is given by the Twitter API), then also its creator is added to the graph. In this way, it will be possible to see not only how tweets get retweeted but also who are the opinion makers. Figure 12 presents the flow graph\(^1\) of the algorithm.

\(^1\)the dashed lines represent the else paths
Figure 12: Flow graph of the algorithm to create a graph of retweets

1. FOR every tweet
2. IF tweet is a retweet
3. Node target = new Node(tweet)
4. graph.addNode(target)
5. IF graph contains the original tweet
6. Node source = graph.getNode(original)
7. Node source = new Node(original)
8. graph.addNode(source)
9. graph.addEdge(new Edge(source, target))
10. IF original tweet has more than R RTs
11. IF graph contains the user of original tweet
12. Node user = new Node(user)
13. graph.addNode(user)
14. graph.addEdge(new Edge(user, source))
End For
As can be seen in the flow chart, the algorithm iterates on the incoming tweets which are retweets, adding the necessary nodes and edges to the graph, avoiding duplicates.

The time complexity of the algorithm is clearly $O(N)$.

To better show how the algorithm works, let’s suppose to have these incoming Tweets, with $R = 1$:

1. Bob: tweet1
2. Alice: RT tweet1
3. Bob: tweet2
4. John: RT tweet2
5. Mark: RT tweet1
6. Alice: RT tweet2

Figure 13: Example of the creation of a graph of retweets
Figure 13 shows the steps of the creation of the network and the numbers representing the flow followed in the flow chart of the algorithm.

3.5 Method for the visualization of patterns of mentions and retweets

As previously said, this visualization method is based on the tuning of the Fruchterman-Reingold algorithm to attractively visualize patterns of mentions and retweets, and on the addition of a ranking algorithm to highlight the hubs of the network and of a modularity algorithm to highlight the different "communities" which are present in the graph.

3.5.1 Tuning of the Fruchterman-Reingold algorithm

The Gephi implementation of the Fruchterman-Reingold algorithm, which will be used in my application, allows the user to set 3 parameters:

- Area,
- Gravity,
- Speed.

The Area parameter is the area of the graph, that means the area of the circle in which the graph should be contained. Obviously, for a consistent layout, the area should be proportional to the number of nodes. After many tries on many networks of mentions and retweets, the experimental result is that a value of 6 times the number of nodes $N$ gives the best visual results.

As can be seen in Figure 14, a too high area value (on the left) makes the nodes too small when the user zooms out to have an overview of the graph; this will create problems in distinguishing the colors of the nodes and in attaching a label to them in the application. On the other hand, if the area value is
too small (on the right), the nodes are too close; this will create problems in recognizing the edges and their directions and in avoiding overlapping labels.

Figure 14: Layout of a graph of mentions using area=10000, area=6N and area=N

The Gravity parameter is used to attract the nodes to the center, avoiding the previously described problem of the dispersion of the disconnected components. The experimental result, reached after applying different gravity values on many networks of mentions and retweets is that the gravity should not be dependent on the single graph and its value should be around 3.5.

As can be seen in Figure 15, a low gravity value (on the left), creates the problem that was present in the ForceAtlas2 algorithm: the disconnected components are too far one from the other, while a high gravity value (on the right) makes impossible to distinguish the different components of the network.
Figure 15: Layout of a graph of mentions using a gravity of 1, 3.5 and 10

The *Speed* parameter is the speed at which the algorithm reaches the convergence; increasing the speed creates a loss in precision. I found out that the optimal value of the speed, to obtain a good visual result in a short time, is 10.

Figure 16: Layout of a graph of mentions using a speed of 1, 10 and 200

As can be seen in *Figure 16*, using a speed value of 1 (on the left) produces a high quality result; unfortunately this is achieved after 2715 iterations. A speed value of 10 (in the center) produces a
layout which is perfectly comparable in terms of quality with the previous one; this result is obtained after 376 iterations. A speed value of 200 (on the right) converges after 10 iterations but the result is not appreciable.

Summing up, these are the parameters that are used in my method:

- Area = 6N (where N is the number of nodes)
- Gravity = 3.5
- Speed = 10

3.5.2 Nodes Ranking algorithm

As previously said, my method includes the use of two ranking algorithms to highlight the most authoritative users (in the network of mentions) and the most influential users and tweets (in the network of retweets).

Figure 17: Effects of the ranking algorithm on a network of mentions
The visual benefits of the use of the ranking algorithms can be seen in Figure 17, while a complete
description of them can be found in Chapter 4.

3.5.3 Modularity algorithm

A modularity algorithm is used to automatically find communities inside a network. My method
includes the use of the algorithm developed by Blondel et al. (24) and my application takes advantage
of its implementation in the Gephi software to allow a user to better visually spot the different groups
of mentions and retweets in the previously built network.

Modularity is an indicator of the quality of a partition performed on a network; it compares the
internal links of each community with the external links with other communities; it ranges between -1
(worst quality) and 1 (best quality). The formula for computing modularity is: (25)

$$Q = \frac{1}{2m} \sum_{i,j} [A_{ij} - \frac{k_i k_j}{2m}] \delta(c_i, c_j)$$

where $A_{ij}$ is the weight of the edge between nodes $i$ and $j$, $k_i = \sum_j A_{ij}$ is the sum of the weights of
each edge connected to node $i$, $c_i$ is the community of node $i$, the function $\delta(u, v)$ is 1 if $u = v$ and 0
otherwise and $m = \frac{1}{2} \sum_{i,j} A_{ij}$.

The method described in the previously cited paper and adopted in my method is able to obtain a
complete hierarchical partition in communities of a network in a very short time\(^1\) and its based on the
iterative repetition of two phases:

\(^1\)The network of the users of a big phone company, composed by 2.04M nodes, was partitioned in 0.76 seconds.
At the beginning each node is assigned to a different community, therefore the number of communities is equal to the number of nodes.

In the first phase, for each node $i$, the algorithm computes if there is a gain in modularity by removing the node from its community and putting it in the community of any of its neighbors $j$. After considering all the neighboring nodes, the node $i$ is put in the community with the maximum gain (obviously only positive gains are considered). This phase stops when no more improvements are possible.

In the second phase a new network is built; in this new network each community identified in the previous phase is a new node; a node $A$ is connected to a node $B$ if in the previous network there was at least one link between an element of community $A$ and an element of community $B$.

The two phases are performed one after the other until a maximum of modularity is reached. Since at each pass of the algorithm the number of nodes greatly decreases, the most of the computation time is spent in the first pass.

One of the advantages of using this algorithm is that it has linear complexity and its execution time is negligible on a few hundred nodes; moreover there is already an implementation of it in the Gephi software, so it will be easier to use it in my application.

*Figure 18* shows the effects of partitioning the nodes of a network of mentions using the modularity algorithm previously described. The different communities of users mentioning each other can now be easily distinguished.
3.6 Evolution over time

My method, as described so far, allows to obtain a static graph of mentions and retweets. In this section I will describe how to add time to them in order to obtain graphs which can change, representing the evolution of the patterns of mentions and retweets in a dataset of tweets.

3.6.1 Adding time to a network of mentions

Let’s consider a previously constructed network of mentions. Let’s call $t_i$ the initial time of a node or an edge in the network and $t_f$ the final time of a node or an edge in the network. Let’s then call $M$ a mention (edge), $S(M)$ its source and $T(M)$ its target.

Using this notation, the initial and final times of the source of a mention, are given by:

$$t_i(S(M)) = t_f(S(M)) = \text{date(tweet}(M))$$

where $\text{date(tweet}(M))$ is the creation date of the tweet containing mention $M$. 

Figure 18: Effects of the modularity algorithm on a network of mentions
The same relation holds for the initial and final times of the target of a mention:

\[ t_i(T(M)) = t_f(T(M)) = date(tweet(M)) \]

If a node is the target of many mentions, its initial time is set to:

\[ t_i(T(M_1, M_2, \ldots, M_n)) = \min(date(tweet(M_1)), date(tweet(M_2)), \ldots, date(tweet(M_n))) \]

and its final time is set to:

\[ t_f(T(M_1, M_2, \ldots, M_n)) = \max(date(tweet(M_1)), date(tweet(M_2)), \ldots, date(tweet(M_n))) \]

The initial and final times of an edge representing a mention are instead:

\[ t_i(M) = t_f(M) = t_i(S(M)) = t_f(S(M)) \]

Using the previous relations, an initial and a final time are assigned to each node and edge of the network. Selecting any time range will visualize all the components of the graph whose initial time is greater or equal the start of the selected range or whose final time is less than the end of the selected range.
Figure 19: Evolution of a network of mentions over time

*Figure 19* shows the evolution of a network of mentions obtained by the implementation of the method in my application (described in Section 7.7).

### 3.6.2 Adding time to a network of retweets

Let’s consider a previously constructed network of retweets. Let’s call $O$ an edge representing the creation of the original tweet, $S(O)$ the source of the original tweet (the user), $T(O)$ the node representing the original tweet, $R$ an edge representing the creation of a retweet, $S(R)$ the source of the retweet (the original tweet) and $T(R)$ the node representing the retweet.

Using this notation, the initial and final times of a node representing a retweet are given by:

$$t_i(T(R)) = t_f(T(R)) = date(tweet(R))$$

where $date(tweet(R))$ is the creation time of the retweet.

The initial and final times of an edge representing the creation of a retweet are given by:
\[ t_i(R) = t_f(R) = date(tweet(R)) = t_i(S(R)) = t_f(T(R)) \]

The initial and final times of a node representing the original tweet are given by:

\[ t_i(S(R_1, R_2, ..., R_n)) = min(t_i(S(R_1), t_i(S(R_2), ..., t_i(S(R_n))) \]

and

\[ t_f(S(R_1, R_2, ..., R_n)) = max(t_i(S(R_1), t_i(S(R_2), ..., t_i(S(R_n))) \]

The initial and final times of a node representing the creator of the original tweet and the edge representing the creation are given by:

\[ t_i(O) = t_i(S(O)) = t_i(O) \]

and

\[ t_f(O) = t_f(S(O)) = t_f(O) \]

Using the previous relations, an initial and a final time are assigned to each node and edge of the network. Selecting any time range will visualize all the components of the graph whose initial time is
greater or equal the start of the selected range or whose final time is less than the end of the selected range.

Figure 20: Evolution of a network of retweets over time

*Figure 20* shows the evolution of a network of retweets obtained by the implementation of the method in my application (described in *Section 7.7*).
CHAPTER 4

METHOD FOR THE IDENTIFICATION OF OPINION MAKERS AND INFLUENTIAL TWEETS

In this chapter I will describe two methods: one for identifying the authoritative users in a network of mentions and one for identifying opinion makers and influential tweets in a network of retweets. Both methods are based on a numerical index; these indexes will be used to rank the nodes of the respective graphs and to determine their size in the visualization of the networks performed by my application (See Section 7.7).

Section 4.1 introduces the concept of influence on Twitter; Section 4.2 explains how to use my method to identify authoritative users in a graph of mentions; while Section 4.3 shows how to identify the opinion makers and influential tweets in a graph of retweets.

4.1 Influence on Twitter

Finding the influential users in a social network has been a very important issue in the last decades, also because of its commercial value, in fact the identification of the opinion makers could allow to spread some ideas more quickly, to make a more effective advertising campaign or to analyze the potential diffusion of a new product.

One of the problems in this kind of task, as correctly pointed out by Bakshy et al. (26), is that the totality of the network is often unobservable and is therefore necessary to perform some estimates.
Twitter, because of its publish-subscribe structure, is a very powerful means of information diffusion and it’s quite simple to crawl it to construct "follower graphs".

Two naive approaches which can be used to compute the influence of a user or a tweet on Twitter are to use the number of followers for a user and the number of retweets for a tweet. This information is easily retrievable from the Twitter API, but it cannot be used a a precise index because of these reasons: first of all the number of followers of a user is not an indicator of how much influence a person has with respect to a given topic, for example if I am retrieving from the Twitter API tweets related to fishing and, for instance, Barack Obama is mentioned just once and he never tweeted about this topic, it would not be correct to assign him an influence value which is proportional to his number of followers. The number of retweets, instead, takes into account just the first level of the "tree" of retweets, without considering the potential exponential spread due to retweets of retweets.

The methods proposed in the two following sections propose the use of two indexes which overcome these problems.

### 4.2 Method for the identification of authoritative users in a graph of mentions

A directed graph of mentions is composed by nodes representing users, while an edge from node $A$ to node $B$ means "$A$ has mentioned $B$ in a tweet".

My idea to rank the nodes of a network of mentions was to use an index which is inspired to PageRank but it doesn't take into account the number of links of the "linking" node, since, if a user mentions one or more users, this does not cause the authoritativeness of the mentioned users to decrease. The index, instead, takes into account the number of followers of the mentioning user, since being mentioned by a user with many followers gives more visibility to the mention.
The authoritativeness $A$ of a node $n$ is computed as follows:

$$A_n = f + \sum_{i=1}^{k} (F_i + d(A_i))$$

where $f$ is the average number of followers of the users composing the network, $F_1, k$ are the number of followers of the $k$ nodes mentioning $n$, $d$ is a damping factor and $A_i$ is the authoritativeness of node $i$, mentioning node $n$.

As can be seen from the formula, the authoritativeness of a user is dependent on the number of followers of the users mentioning him and, by a damping factor, on the authoritativeness of the mentioning users. In my application I used a damping factor of 0.1 to give more importance to the number of followers which plays an essential role in giving visibility to a mention on Twitter.

Figure 21: Influence index in a network of mentions, nodes = users, edges = mentions
Figure 21 shows a sample network of mentions and the authoritativeness values of each node. The size of each node (as it will be done in the application) has been made proportional to its authoritativeness value.

Combining my method for the construction of a graph of mentions with my method for computing the authoritativeness of a user allows to easily identify the most authoritative users with respect to a given topic (the largest nodes in the figure).

4.3 Method for the identification of opinion makers and influential tweets in a graph of retweets

A directed graph of retweets is composed by nodes representing tweets, while an edge from a node $A$ to a node $B$ means "$B$ is a retweet of $A"$, moreover, if a tweet has more than a given threshold of retweets $R$, also its creator is added as a node to the graph.

What we would like to have is an index which allows us to rank both the influence of tweets and users in information diffusion. Fortunately we have used an algorithm for the construction of the graph which allows us to track the information diffusion by following the direction of the edges. The idea behind my algorithm is to start from the "leaves" of the graph (nodes without children) and to follow the flow of information upstream, incrementing the influence values of the nodes.

The influence $I$ of a node $n$ is considered equal to the number of retweets of which $n$ is the root; it can be recursively computed by using the following formula:

$$I_n = \sum_{i=1}^{k} (I_i + 1)$$
where $I_{1,k}$ are the influence values of the $k$ children nodes of $n$. The recursion stops when a node has no children, in that case the influence index can be simply computed as:

$$I_n = 0$$

Note that if a tweet has never been retweeted its influence value will be 0, since no user has decided to share its information.

Note also that a node representing a user will have a minimum influence of $R + 1$ since only users which created at least one tweet with at least $R$ retweets are added to the graph. Therefore, using the proposed method, the influence of a user will be determined by the sum of the influence of her tweets plus the number of her tweets (obviously we are considering only the tweets which are part of the graph).

Figure 22: Influence index in a network of RTs, green nodes = tweets, red nodes = users
Figure 22 shows a sample network of retweets and the influence values of each node. The size of each node (as it will be done in the application) has been made proportional to its influence value.

Combining my method for the creation of a graph of retweets with my method for computing the influence of the nodes of the graph allows to easily spot the most influential tweets (the largest nodes in green) and the opinion makers (the largest nodes in red).

Figure 23: Influence index update by back propagation

Figure 23 shows how using this method it’s easy to update the influence values by back propagation when a new retweet is added to the graph.
CHAPTER 5

METHOD FOR THE ANALYSIS OF THE SENTIMENT OF TWEETS AND ITS EVOLUTION

In this chapter I will present my method, based on the use of an Emoticon Classifier and a Logistic Regression Classifier, to assign a sentiment to any tweet and to track the evolution of an index representing the overall sentiment.

Section 5.1 presents the state of the art researches on the sentiment analysis of tweets; in Section 5.2 my method is described; Section 5.3 presents the dataset used; in Section 5.4 the performances of my method are reported; while Section 5.5 describes the index used to quantify the sentiment contained in tweets.

5.1 State of the art methods for the sentiment analysis of tweets

In this section I will describe the state of the art researches in the sentiment analysis of tweets, identifying three main currents: emoticon analysis, methods based on the analysis of single words or short sentences, and methods based on the use of text classifiers.

5.1.1 Emoticon analysis

One approach for the sentiment analysis of tweets is using the emoticons contained in them. This technique, used for example by Logunov et al. (27), consists in categorizing tweets based on the positivity or negativity of their emoticons.
This method is often used as a naive sentiment classifier of tweets, due to his numerous advantages. First of all this technique is very easy to implement, it just needs to check if a tweet contains an emoticon, if it positive or negative, and decide what to do in case of conflicting emoticons (in its simplest implementation these cases are not taken into account). Another advantage is that emoticons are used in almost any language, therefore they can be used to classify the sentiment of multi-language datasets. Moreover this method has a high accuracy (see Section 5.4).

A sentiment analysis based on emoticons has also its disadvantages. One of the greatest drawbacks is that it is obviously applicable only to tweets containing emoticons (approximately the 5%). Another disadvantage is that people tend to use more emoticons when they are happy than when they are sad, therefore the "ordinary" ratio between tweets containing positive and negative emoticons has to be taken into account to avoid a bias towards positiveness. Finally, in some countries, especially in Asia, the emoticon system is very different from the Western one.

This technique is used also by Twitter, in fact searching "school :)" will retrieve all the tweets containing the word school and a positive emoticon ( :) , :D , :p etc.), while searching "school :(" will retrieve all the tweets containing the word school and a negative emoticon ( :(, :'( etc.).

5.1.2 Single words methods

These methods consist in classifying the sentiment of tweets by looking at some particular words or short expressions which are indicative of a given mood (e.g hate for Anger and smile for Happiness).

One of the simplest ways to obtain a sentiment index for a period $T$ using this method is to take each tweet, count the words which are symptomatic of each possible mood and assign the mood with the most words to the tweet, then assign the the mood with the most tweets to the period $T$. 
A more significative approach, proposed by Akcora et al. (28), is to choose $n$ possible moods and to assign a list of symptomatic words or expressions to each of them. A vector $v = (s_1, s_2, ..., s_n)$, where $s_i$ is a boolean equal to 1 if sentiment $i$ is present, 0 otherwise. For each time interval containing $N$ tweets the centroid of the set of vectors $V = (v_1, v_2, ..., v_N)$ is computed. The similarity between two time intervals is computed using the cosine similarity between their centroid vectors.

A weakness of a method based on the analysis of single words or short expressions is that it fails when the mood of a tweet can be guessed only by contextualizing the symptomatic terms, for example "I used to love this movie, but now I don't". Another problem emerges during holidays when expressions like "happy new year" increase the positiveness of the overall sentiment even if they are not indicative of the real mood of the user.

5.1.3 Text classification

Since most of the researches in the sentiment analysis of tweets are based on a pre-collected set of tweets, they are often manually classified or, when the dataset is larger, they are "socially classified", for example by using Amazon Mechanical Turk (4). More rarely, when the tweets are too many to be manually classified by a human (or humans), some state of the art tools for sentiment analysis are used (for example by Bollen et al. (29)).

My method will use a state of the art tool but, since it has to be applied to any unconstrained dataset of tweets, the tool will be used to train a classifier on a dataset of random tweets.

The state of the art research, more than on the way the tweets are classified, differs in how they compute an index to track the changes of sentiment over time.
5.2 My method

The method I will be using for analyzing the evolution of sentiment in tweets is based on a sentiment index, obtained by the use of an emoticon classifier, when possible, and a three way (positive, negative, neutral) logistic regression classifier. Moreover a threshold on the conditional probability of the classification will be used: tweets classified with a confidence below the threshold will be classified as neutral to avoid excessive speculations. As shown in Section 5.4, choosing the right value as threshold allows to increase the performances of the classifier.

5.2.1 Emoticon classification

I’ve decided to include a classification based on emoticons for many reasons: first of all this kind of classification has a high accuracy (see Section 5.4); moreover it is not influenced by a problem of traditional machine learning techniques, which is to be less accurate when the testing set is different from the training one for domain, topic or time; emoticons instead are independent of domain, topic and time (as highlighted by Read (30)); another very important reason is that emoticons are language independent, therefore the implementation of the model in my application will allow a user searching for a non-English topic to turn off the logistic regression classifier (which has been trained only on tweets written in English) and to base the sentiment analysis only on emoticons. Another reason is that emoticons allow us to disambiguate tweets that could be very difficult to classify without considering the emoticon, for example the sentence "I really love Monday mornings :S" could be considered positive without taking into account the emoticon which reveals the real sentiment behind the sentence.

In my method I will consider the following emoticons as positive:
TABLE I: LIST OF POSITIVE EMOTICONS

<table>
<thead>
<tr>
<th>Glyph</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>:-) or :) or =) or (:</td>
<td>smile</td>
</tr>
<tr>
<td>:-) or :)</td>
<td>wink</td>
</tr>
<tr>
<td>:-P or :P</td>
<td>sticking tongue</td>
</tr>
<tr>
<td>:-D or :D or =D</td>
<td>wide smile</td>
</tr>
<tr>
<td>^^</td>
<td>joy</td>
</tr>
<tr>
<td>&lt;3</td>
<td>love</td>
</tr>
<tr>
<td>XD</td>
<td>happy</td>
</tr>
</tbody>
</table>

and the following ones as negative:

TABLE II: LIST OF NEGATIVE EMOTICONS

<table>
<thead>
<tr>
<th>Glyph</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>:-) or :( or :)</td>
<td>sad</td>
</tr>
<tr>
<td>:'( or :'(</td>
<td>crying</td>
</tr>
<tr>
<td>-. or _- or _-- or __.</td>
<td>disappointed</td>
</tr>
<tr>
<td>:-/ or :/ or :l or :\</td>
<td>disappointed</td>
</tr>
<tr>
<td>:-S or :S</td>
<td>confused</td>
</tr>
</tbody>
</table>
A tweet containing negative emoticons will be considered negative, while one containing positive emoticons will be considered positive; tweets containing contrasting emoticons will not be considered and will be classified by the logistic regression classifier.

### 5.2.2 Logistic regression classifier

For the tweets which do not contain emoticons a 3 way (positive, negative and neutral) classifier has been trained. It is based on a n-gram ("a contiguous sequence of n items from a given sequence of text or speech"\(^1\)) level logistic regression classifier.

The \(n\) level of the n-gram analysis will be 12 and the threshold of confidence in the classification will be 0.6; tweets classified with a confidence which is less than the threshold will be considered neutral.

Logistic regression is a discriminative model which can be used to estimate \(p(y|x)\) directly. This is done by representing each example by a feature vector \(x\). Let’s consider a positive/negative classification. \(x\) is first mapped to a real number, where a very positive number is more likely to make the tweet associated to \(x\) be classified as positive and a very negative number is more likely to make the tweet be classified as negative. After this linear mapping, the range of values is mapped to \([0, 1]\), in order to obtain a percentage (31).

The classifier has been trained and tested using the Java library LingPipe\(^2\), a powerful toolkit which uses computational linguistics.

---


\(^2\)http://alias-i.com/lingpipe/index.html
5.3 Dataset

There are some set of labeled tweets on the Internet which can be used to train and test sentiment classifiers, even if the Twitter API policy right now forbids publishing sets of collected tweets. One of these is the Twitter Sentiment Corpus\(^1\) which consists of 5513 hand-classified tweets related to 4 famous technology brands. Another publicly available dataset is the Twitter Sentiment Dataset 2008 Debates\(^2\) which contains 3238 labeled tweets from the 2008 Presidential debate.

I decided not to use these datasets but to create another one by myself for some reasons: one of them is that both datasets are specific of a given topic (politics and technology), therefore training a sentiment classifier on these sets could produce a biased model. Moreover the 2008 Debates dataset is labeled in an unconventional way, that is with some sentiment scores (the number of scores can change) from 1 to 4 given by different people.

The dataset I collected is composed by 2548 random tweets taken from the Twitter Stream API. The tweets have been collected during different days at different time of the day to avoid biases in sentiment due for example to the weekend or some particular events. The tweets have been labeled manually by me using 3 classes: positive, negative and neutral.

5.4 Performances

In this section I will describe the testing performed on the two classifiers and the resulting performances: 90.4% for the one based on emoticons and 75.1% for the one based on logistic regression.

\(^1\)available at: http://www.sananalytics.com/lab/twitter-sentiment

\(^2\)http://www.infochimps.com/datasets/twitter-sentiment-dataset-2008-debates
5.4.1 Emoticon classifier

To measure the performances of my classifier based on the use of emoticons I collected a testing dataset (the model obviously doesn’t need any training, since it just associates a “positive” or “negative” class to a tweet depending on the emoticons which are present in it) of 2181 random tweets from which I extracted only the tweets containing emoticons, which were 104, the 4.8%.

Table III shows the number of occurrences of each positive emoticon in the testing dataset and their percentage in relation to the number of tweets containing positive emoticons and to the number of tweets containing any emoticon.

<table>
<thead>
<tr>
<th>emoticon</th>
<th>occurrences</th>
<th>% positive</th>
<th>% emoticons</th>
</tr>
</thead>
<tbody>
<tr>
<td>:-) or :) or =) or (:</td>
<td>44</td>
<td>53.01%</td>
<td>42.31%</td>
</tr>
<tr>
<td>:-) or ;)</td>
<td>14</td>
<td>16.87%</td>
<td>13.46%</td>
</tr>
<tr>
<td>&lt;-3</td>
<td>8</td>
<td>9.64%</td>
<td>7.69%</td>
</tr>
<tr>
<td>:-D or :D or =D</td>
<td>7</td>
<td>8.43%</td>
<td>6.73%</td>
</tr>
<tr>
<td>:-P or :P</td>
<td>5</td>
<td>6.02%</td>
<td>4.81%</td>
</tr>
<tr>
<td>^</td>
<td>3</td>
<td>3.61%</td>
<td>2.88%</td>
</tr>
<tr>
<td>XD</td>
<td>2</td>
<td>2.41%</td>
<td>1.92%</td>
</tr>
<tr>
<td>sum</td>
<td>83</td>
<td>100%</td>
<td>79.81%</td>
</tr>
</tbody>
</table>
Table IV, instead, shows the number of occurrences of each negative emoticon in the testing dataset and their percentage in relation to the number of tweets containing negative emoticons and to the number of tweets containing any emoticon.

**TABLE IV: FREQUENCY OF NEGATIVE EMOTIONS**

<table>
<thead>
<tr>
<th>emoticon</th>
<th>occurrences</th>
<th>% negative</th>
<th>% emoticons</th>
</tr>
</thead>
<tbody>
<tr>
<td>:-(' or :( or :)</td>
<td>9</td>
<td>42.86%</td>
<td>8.65%</td>
</tr>
<tr>
<td>:-/ or :/ or :l or :\</td>
<td>6</td>
<td>28.57%</td>
<td>5.77%</td>
</tr>
<tr>
<td>-.- or -_ or -____ or -_</td>
<td>4</td>
<td>19.05%</td>
<td>3.85%</td>
</tr>
<tr>
<td>:'( or :'(</td>
<td>1</td>
<td>4.76%</td>
<td>0.96%</td>
</tr>
<tr>
<td>:-S or :S</td>
<td>1</td>
<td>4.76%</td>
<td>0.96%</td>
</tr>
<tr>
<td><strong>sum</strong></td>
<td><strong>21</strong></td>
<td><strong>100%</strong></td>
<td><strong>20.19%</strong></td>
</tr>
</tbody>
</table>

To test the performances of the classifier I manually labeled the tweets of the testing dataset using 3 classes: positive, negative and neutral. Then the performances of the classifier, based on the presence of the emoticons (note that it only distinguishes between positive and negative tweets) have been tested.

The Emoticon Classifier correctly classified the 90.4% of the tweets. Here below the confusion matrix:
5.4.2 Logistic regression classifier

The sentiment analysis classifier has been trained on a set of 2283 tweets taken from the dataset described in Section 5.3 and tested on the remaining 265 tweets, therefore there are no overlaps between the training and testing dataset. The classifier reaches its maximum accuracy, 75.1%, using a n-gram level of 12 and a threshold of 60%.
5.4.3 N-Gram number

As can be seen in Figure 24, the classifier reaches an accuracy over 74% between 10 and 13 n-gram level, in particular the maximum value, 75.1% is obtained with $n = 12$.

5.4.4 Threshold

As can be seen in Figure 25¹, the threshold level has been chosen equal to 60% since it is the one which allows to reach the best performances, In particular, as shown before, it reaches the maximum for a n-gram level of 12.

¹the y axis starts from 0.3
5.5 Overall Sentiment Index

Since neutral tweets don’t contribute in determining the overall sentiment (and many of them are non-English tweets or contain just a link), they will not be considered when computing the Overall Sentiment Index. The Overall Sentiment Index of a time interval from \( t_1 \) to \( t_2 \) is given by:

\[
S(t_1, t_2) = \frac{P(t_1, t_2) - N(t_1, t_2)}{P(t_1, t_2) + N(t_1, t_2)}
\]
where $P(t_1, t_2)$ is the number of positive tweets in the interval, and $N(t_1, t_2)$ is the number of negative tweets in the interval. The index ranges from $-1$, when only negative (and neutral) tweets are present, to $+1$, when only positive (and neutral) tweets are present.

Even if it does not appear in the formula, the number of neutral tweets should not be neglected, since comparing it to the number of positive and negative tweets allows us to obtain information on the quantity of sentiment, in addition to the quality of sentiment represented by the index. For this reason the percentage of positive, negative and neutral tweets will always be present along with the Overall Sentiment Index in the application.
In this chapter I will present the state of the art applications for the analysis and visualization of tweets and I will suggest some possible improvements which will be adopted by my applications.

In Section 6.1 describes which characteristics should be present in an application in order to analyze an event or topic on Twitter and its evolution over time. In Section 6.2 20 of the most famous Twitter analysis and visualization applications are presented, in Section 6.3 a table summarizes the characteristics possessed by them, while in Section 6.4 the improvements of my applications with respect to the state of the art are illustrated.

6.1 Characteristics of an application to analyze an event or topic and its evolution over time

Based on my analysis of the state of the art applications (presented in Section 6.2) and my experience in building data analysis and visualization tools, I was able to identify the characteristics that an application should have in order to allow a user to spot trends and evolution in time of topics and events on Twitter:

- **Online collection analysis**: Most of the research related to the analysis of tweets is based on pre-collected datasets. I think it would be essential for an analysis application to allow its users to collect the most recent tweets related to a given query in just one click; otherwise it would be more an application which presents a pre-computed result rather than an application which gives the user the possibility to analyze an arbitrary event or topic at any time.
• **Multi topic:** Much of the research has focused on and analyzed a particular event or topic, while a very useful feature for users would be searching at any time for any topic, having the possibility of taking advantage of the query functionalities of the Twitter API (excluding terms, searching for an hashtag, filtering messages from and to a given user, filter by tweets’ location, date or sentiment).

• **Free and opensource:** I think that one of the most important features of an application built for research and analysis purposes is to make it free and opensource, allowing people from all over the world to use it in their analyses.

• **Raw data:** An analysis application should also allow a user or a researcher to download the collected raw data to archive it and possibly perform further analyses.

• **Flux analysis:** The analysis of the flux of tweets over time allows a user to understand the entity of an event or topic and to identify peeks in tweeting activity to focus on.

• **Content analysis:** Obviously the evolution in number of tweets is not enough if not paired with a valid analysis of their content. My idea is to allow a user to visualize not only the raw content of tweets but to instantaneously have an overview of tweets content by means of tag clouds. Adding also a temporal aspect to this, will allow a user to visualize dynamic tag clouds which change showing the evolution of tweets’ content over time.

• **Sentiment analysis:** Sentiment analysis of tweets is a challenging task some researches have focused on, but, also in this case, the majority of them are focused on the sentiment analyses of a pre-collected datasets (4, 32, 33). I think that it would be interesting for a user to obtain a sentiment analysis of recent tweets on a topic with just one click.
• **Geolocation**: Geolocation is a very useful information in order to analyze trends in events and topics on Twitter. Unfortunately just <1% of the tweets have a precise geo-tag.

• **Time aspect**: Time representation can be exploited to depict changes in content, position and sentiment of tweets.

• **Tracking Retweets and Mentions**: Tracking the patterns of retweets and mentions in a dataset of tweets allows to obtain a clear visualization of the information flow and to identify the opinion makers.

• **Retrieved tweets limit**: In the majority of the applications that perform online analysis of tweets, the number of retrieved tweets is fixed to a few hundred, therefore the maximum number of tweets retrievable by the Twitter Search API is not fully exploited.

• **GUI**: The user interface should be intuitive also for non-expert users, As shown later, I will take advantage of many intuitive components as sliders, search boxes, radio buttons etc.

### 6.2 Twitter analysis and visualization applications

In this section I will present 20 of the most popular Twitter analysis and visualization applications; I decided to include in this selection only free and ready-to-use applications, excluding commercial products or applications which are still in a development phase. I also excluded the applications which allow a user to collect tweets without any kind of additional analysis or visualization.
Twitter Browser

Twitter Browser\textsuperscript{1} is a web visualization which represents tweets as replicating neurons. It is more an artistic experiment rather than a data analysis visualization, since the user has no idea of which tweets he is visualizing. A message on the webpage of the project suggests that the tool used to have more functionalities but, when the Twitter APIs were modified, it has not been updated.

\textsuperscript{1}http://www.neuroproductions.be/twitter_friends_network_browser
Visible Tweets

Visible tweets\(^1\) is a web application which allows to input a query and visualize the related tweets. The application takes advantage of fancy transitions between the various tweets but a real analysis is not performed. For each tweet, its content, its approximate creation time, the nickname of the user and her avatar are displayed. The application can be a good tool to visualize tweets in background at a conference but it cannot be considered as an actual analysis application. The transition effect between the tweets can be customized.

\(^1\)http://visibletweets.com
Tweet Beam

Tweet Beam is a web application which visualizes tweets related to any query using a mosaic composed by the avatars of the users. The application does not provide much information about the tweets (e.g. date) and it’s not clear how the size of the avatars is determined. An automatic transition highlights one tweet at a time, while mouse hovering a particular avatar highlights the related tweet for some seconds. The tweets are replaced after some seconds, probably some sampling is performed to avoid tweets to keep changing when choosing a very popular hashtag. The full-screen mode can be used as a visual support for conferences or public events. Since no additional information on the tweets is displayed, this application cannot be considered a real analysis tool.

1http://www.tweetbeam.com
Twitter Map

Figure 29: Twitter Map

Twitter Map\(^1\) is a simple tool which displays tweets related to any query on Google Maps. The search is limited to 500 tweets and, since only a small percentage of tweets are geo-located, just a few markers appear on the map (for example searching “obama” produced only 17 results, while less popular queries produced just one or two markers). When hovering the mouse on a marker, the user, the content and the location of the tweet are displayed, no creation date is shown. The markers appear on the map one after the other, the rate seems regular, so it’s probably not proportional to creation times; it’s not possible to verify if the order of appearance is chronological, since, as previously said, the user cannot see time related data.

\(^1\)http://twittermap.appspot.com
Foller.me

Foller.me\(^1\) is a web application which analyzes the tweeting history of a user; this includes general information about the user (retrieved from her profile), tweeting activity (number of following, followers, followers ratio etc.), topics (in form of a tag cloud), most used hashtags, mentions and replies, a quantitative analysis of the last 100 tweets (percentage of retweets, tweets with mentions, medias or links etc.), a very simple sentiment analysis, considering if the tweet contains happy or sad smileys, and number of tweets for each time of the day. A section showing a map of where do followers come from is still under development.

\(^1\)http://foller.me
Tweet Archivist

Used to be an offline application which has been later converted to a web application which analyzes tweets related to a query. The analysis includes: tweet volume over time, top users, language, top words, top URLs, sources, user mentions, hashtags and images. The pie charts sometimes are really difficult to understand since they have more than 15 slices and no filtering is applied, moreover, in my opinion, the information related to the Top Users shouldn’t be displayed in form of a pie chart since the total doesn’t sum up to 100%. The application allows to download the retrieved tweets as a .xls or .zip file. Signing in allows a user to subscribe to a given query and to receive hourly updates.

1http://http://www.tweetarchivist.com (The advanced options of the application have become paid from March 2013)
Mentionmapp

Figure 32: Mentionmapp

Mentionmapp\(^1\) is a tool which creates an interactive map of mentions where each node, i.e. a user, is connected to the people and hashtags he/she mentioned the most in recent tweets. The central node is the one chosen by the user. It is possible to click on a neighboring node to focus on it and proceed exploring. The application asks for Twitter authentication and for the permission to "read Tweets from your timeline, see who you follow and follow new people, update your profile, post Tweets for you" without any reason.

\(^1\)http://http://mentionmapp.com (requires a Twitter account)
Twitter Counter

Figure 33: Twitter Counter

Twitter Counter\(^1\) is a very simple tool which plots the number of followers and following of any Twitter user. The application allows the user to compare up to 3 users and to predict how their number of followers will grow in the future.

\(^1\)http://twittercounter.com
Twitonomy is a web application which allows the user to have a good analysis of its own account or the one of any other user. Given a user, the tool analyzes the number of tweets per day, most retweeted/mentioned users, most used hashtags, most retweeted tweets, most active days of the week and hours of the day etc. The application requires Twitter permission to "read Tweets from your timeline, see who you follow and follow new people, update your profile, post Tweets for you" without any reason, since the user is able to search for any twitter account, she could also search for her own without logging in.

---

1http://www.twitonomy.com (requires a Twitter account)
A World of Tweets

A World of Tweets\(^1\) is a web tool which shows a heat map of tweets being tweeted right now, showing how many of them, in percentage, come from each Country. No analysis of the content or date of the tweets is performed. It is possible to change some aspects of the visualization.

\(^{1}\text{http://aworldoftweets.frogdesign.com/}\)
Cloud.li

Image: Figure 36: Cloud.li

Cloud.li\(^1\) is a web application which allows the user to input some terms to perform a query and possibly some terms to be excluded. The result is a dynamic tag cloud showing which words are present the most in tweets related to the inputted query. How the dynamic tag cloud updates and how many tweets are considered in forming it is not declared.

\(^1\)http://cloud.li
Monitter

Monitter\(^1\) is an application showing streams of tweets related to user queries. It is possible to filter tweets by their geographical location but the functionality doesn’t seem to work properly, since many of the retrieved tweets say "location: Planet Earth" or other places which are in another continent with respect to the location chosen by the user.

\(^1\)http://www.monitter.com
Twitter Character

Twitter Character\(^1\) is a twitter visualization tool by Visual.ly which allows to analyze a twitter account or to compare two, after choosing an avatar to represent it/them. The avatar is loaded automatically if a popular person is chosen. The result is an infographic presenting and comparing the main features of the user(s). The analysis includes tweets per day, followers/following ratio, number of followers per day, topics, chattiness, enthusiasm and interestingness (how these last three parameters are calculated is not declared). A good point of this tool is the representation of topics as words in a Venn Diagram; this allows to see which are the shared topics between two users.

\(^1\)http://visual.ly/twitter
Trendsmap

Figure 39: Trendsmap

Trendsmap\(^1\) is a web tool which overlays the most common words contained in tweets on a world map; in this way it is possible to discover trends in topics depending on geographical areas. Clicking on a word makes appear a tooltip showing the last tweets containing it and two flux graphs that are too small to be understandable. Often the map becomes too overcrowded.

\(^1\)http://www.trendsmap.com
Revisit

Revisit\(^1\) is an application which allows the user to input some search terms and to obtain an interactive visualization of conversational threads established by retweets and @replies. When new tweets are available, they are added to the visualization. Unfortunately the application becomes quite laggy when the limit of tweets is set to more than 500.

---

\(^1\)http://http://moritz.stefaner.eu/projects/revisit/
Hashtagify

With Hashtagify it’s possible to select an hashtag and visualize a graph of the hashtags that are the most related with it.

---

1http://hashtagify.me
Spot

![Spot Visualization](image)

Figure 42: Spot

Spot\(^1\) is a Twitter visualization in which tweets are represented as particles. Depending on the chosen view, the tweets/particles rearrange themselves dynamically forming clusters. Often, many tweets are left uncategorized out of the clusters. It’s possible to group tweets by similarity, time when they were sent, user who wrote them, or words contained. This visualization has a limit of 200 tweets. The application is a bit laggy.

\(^1\)http://www.neoformix.com/spot
Year in Hashtag

Year in Hashtag\(^1\) is an application which shows the most important events of 2011 by means of tweets, Youtube videos and photos. It is possible to filter events by month or place. No analysis on the tweets is performed.

\(^1\)http://yearinhashtag.com
Sentiment140

Figure 44: Sentiment140

Sentiment140\(^1\) is a Twitter sentiment analysis tool which classifies tweets depending on their content; the possible outcomes are: Positive, Negative or Unclassified. It is possible to search for tweets in English or Spanish, both gives good accuracy even if the best results are obtained with the English search. The number of retrieved tweets is not declared.

\(^1\)http://www.sentiment140.com
Tweereal

Tweereal\textsuperscript{1} is a visualization showing real-time geo-located tweets on an interactive map. The website doesn’t say how the color and the dimension of the circles representing the tweets are determined. It is possible to customize some parameters of the visualization but no analysis on the tweets is performed, and it’s not possible to visualize the content of the retrieved tweets.

6.3 Characteristics of the state of the art applications

In this section I will present a table showing which of the characteristics described in Section 6.1 are present in the state of the art applications previously presented. For reasons of space, the names of the applications in the table have been substituted by abbreviations. The pairing is here reported:

\textsuperscript{1}\url{http://tweereal.com}
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Twitter Browser</td>
</tr>
<tr>
<td>2.</td>
<td>Visible Tweets</td>
</tr>
<tr>
<td>3.</td>
<td>Tweet Beam</td>
</tr>
<tr>
<td>4.</td>
<td>Twitter Map</td>
</tr>
<tr>
<td>5.</td>
<td>Foller.me</td>
</tr>
<tr>
<td>6.</td>
<td>Tweet Archivist</td>
</tr>
<tr>
<td>7.</td>
<td>Mentionmapp</td>
</tr>
<tr>
<td>8.</td>
<td>Twitter Counter</td>
</tr>
<tr>
<td>9.</td>
<td>Twitonomy</td>
</tr>
<tr>
<td>10.</td>
<td>A world of tweets</td>
</tr>
<tr>
<td>11.</td>
<td>Cloud.li</td>
</tr>
<tr>
<td>12.</td>
<td>Monitter</td>
</tr>
<tr>
<td>13.</td>
<td>Twitter Character</td>
</tr>
<tr>
<td>14.</td>
<td>Trendsmap</td>
</tr>
<tr>
<td>15.</td>
<td>Revisit</td>
</tr>
<tr>
<td>16.</td>
<td>Hashtagify</td>
</tr>
<tr>
<td>17.</td>
<td>Spot</td>
</tr>
<tr>
<td>18.</td>
<td>Year in Hashtag</td>
</tr>
<tr>
<td>19.</td>
<td>Sentiment140</td>
</tr>
<tr>
<td>20.</td>
<td>Tweereal</td>
</tr>
</tbody>
</table>

**table legend:**

✓ = the characteristic is present

✗ = the characteristic is not present

~ = some aspects of the characteristic are present, but it’s not fully implemented

NA = it’s not possible to determine if the characteristic is present
### TABLE VI: CHARACTERISTICS OF THE STATE OF THE ART APPLICATIONS

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
<th>sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi topic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Opensource</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Raw data</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flux analysis</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Content an.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✔</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sentiment an.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Geolocation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time evolution</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RTs/@</td>
<td>✔</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High # tweets</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td></td>
<td></td>
<td></td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>✔</td>
<td>NA</td>
<td>✔</td>
<td>NA</td>
<td></td>
<td></td>
<td>NA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nice GUI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sum</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3.5</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>2.5</td>
<td>3.5</td>
<td>4</td>
<td>3</td>
<td>3.5</td>
<td>5</td>
<td>4.5</td>
<td>3.5</td>
<td>4.5</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>
6.4 How to improve

6.4.1 What is missing

As can be seen from the table, the state of the art applications have on average 3 of the 11 previously defined characteristics, in particular they are often focused on a specific task (e.g. sentiment analysis, map visualization), but it wasn’t possible to find a tool which allows a user to perform a wide range of analyses on a given topic to easily spot trends related to the content and location of tweets, retweeting patterns etc. Having a tool with all the previously described characteristics, would allow to obtain a 360 degrees view of an event or a topic and to understand its evolution over time.

Let’s for example think of a concert: a standard tool would allow the user to see on a map that most of the tweets come from the concert area, that the majority of them have a positive content and that the most used words are concert and music. A complete application, as the one I am going to develop, would allow the user to see how the position and number of tweets change over time, watching them getting nearer to the concert location, noticing, by means of a dynamic tag cloud, how the most common words change, for instance, from begin to awesome to best and ever seen. A dynamic analysis of sentiment would also allow the user to see how the number of positive tweets increases as the start of the concert is getting closer, while the tweets becomes more negative when the concert ends and people are sad.

6.4.2 Possible improvements

After analyzing how the characteristics described in Section 6.1 are implemented in the state of the art applications, I will suggest some improvements that my application will implement. Most of these improvements comes from the implementation of the methods developed in my thesis and described in Sections from 2 to 5.
• **Multi topic:** Approximately half of the applications gives the user the possibility to input any query, while others restrict the query to tweets from a specific user or talking about a specific event. In my application I will give the user the possibility to input any query that is acceptable by the Twitter API; this allows to exclude terms, filter by date, geographic position, user, hashtag and sentiment.

• **Open source:** None of the considered applications is open source.

• **Raw data:** Raw data is downloadable only with one of the 20 applications I analyzed. I think that allowing a user to archive the raw tweets he retrieved, especially for advanced users, is very important in order to have a dataset on which further analyses can be performed. The saved dataset could also be later loaded in the app to replay its analyses. In my application I will allow the user to download the retrieved tweets and to load them again, in this way the application will allow both the storage and analysis of tweets.

• **Flux analysis:** Flux analysis is present in 4 applications. These applications present a clear graph that shows how the number of tweets on a given topic changes over time. In my application I will produce a similar graph that will provide the exact numbers on mouse hover.

• **Content analysis:** Content analysis, which should be one of the most important features to analyze an event or topic on twitter, is present in only 5 applications out of 20 and, when present, is often performed just by enumerating the most used hashtags or, more rarely, by means of a tag cloud. My idea is to add dynamicity to tag clouds, letting them change as the content of tweets changes. In this way a user will be able not only to have an overview of the content of tweets
talking about a given topic, she will also be able to understand how this content changes over time.

- **Sentiment analysis**: Sentiment analysis is present in only one application and it’s used just to show the number of positive vs. the number of negative tweets. I think that adding temporal information to this analysis would result in an interesting visualization of sentiment evolution in time.

- **Geolocation**: Geolocation information is present in only 4 applications and the location of tweets is based only on the information provided by Twitter. Since the percentage of geolocated tweets is <1%, a lot of information is not being exploited. My idea is to also take advantage of the position determined by the user profile of the creator of the tweet.

- **Time evolution**: I think that time evolution, which is really present in only 4 applications, is an essential aspect to analyze trends in events and topics. In my application I will consider this aspect which allows to depict changes in flux, content, patterns, position and sentiment of tweets.

- **Tracking Retweets and Mentions**: Retweets and mentions are considered in just 3 applications. In only one of them the user has a clear overview of the retweeting pattern. Since tracking the patterns of retweets and mentions allows to have a clear visualization of the evolution of a conversation, I will allow the user to clearly understand and visualize this patterns.

- **Limit in retrieved tweets**: Many of the analyzed applications don’t say how many tweets are being analyzed, while some others limit the search to a few hundreds. In my application I will allow the user to fully exploit the Twitter API limits by a dynamic retrieving rate which slows down when the rate limit of the API is being exceeded.
• **GUI:** Almost all of the 20 applications have a clear and user-friendly interface. A clear and friendly GUI is for sure a good way to let a user interact with the application more naturally, exploiting all of its features. As I will describe in the next section, I will use visual elements which will show useful information and analyses in a simple and intuitive way.
CHAPTER 7

APPLICATION DESIGN

In this chapter I will describe how the methods developed in my thesis and described in Chapters from 2 to 5 have been implemented in my application.

In Section 7.1 the tools used to develop the application are presented; in Section 7.2 it’s explained how the application interacts with the Twitter API; in Section 7.3 the layout of the application is described; in Section 7.4 the search bar, the menu and the import/export functionality is presented; in Sections from 7.5 to 7.10 the six tabs of the application are described.

7.1 Tools used

The main tool used to build the application is Processing. Processing\(^1\) is an open source environment to design and build visualization and data analysis applications.

The decision of using Processing came from its many advantages:

- It’s open source

- It’s inter-platform and it allows to create double-clickable inter-platform applications

- The documentation is updated and very well written, many learning examples are available

- The Forum\(^2\) is very useful since it’s frequented by very helpful people

---

\(^1\)http://processing.org

\(^2\)https://forum.processing.org
One of the few cons of Processing is its IDE, since it’s designed for simple projects rather than complex Object Oriented ones. Fortunately it is possible to import Processing as a Java library in Eclipse\(^1\). In this way both the previously described advantages of Processing and the advantages coming from using Eclipse (easy packages and classes management, autocomplete functionality etc.) can be fully exploited.

### 7.2 Connecting to the Twitter API

In order to allow my application to retrieve tweets from Twitter, I used Twitter4J\(^2\), a Java library which allows to interact with the Twitter API.

I isolated all the code interacting with the Twitter API in a class called `TwitterManager`. As can be seen from the pseudo-code below, this class basically connects to the Twitter API and retrieves the tweets that match a query defined by the user, stopping when a limit defined by the user is reached and sleeping for some seconds if the rate limit is being exceeded.

```java
//imports

public class TwitterManager {

    ...

    ArrayList<Tweet> tweets;

    //In the constructor the authentication is performed
    public TwitterManager(){
        -perform authentication

1http://processing.org/learning/eclipse

2http://twitter4j.org/en/index.html
public void performQuery(String inputQuery) {
    //create a query using the input String
    -try to convert tweets
    -catch exception
}

public void convertTweets(Query query) {
    do {
        -retrieve tweets of the current page of the query
        -fill the ArrayList of Tweets.
        -sleep if the Twitter API rate limit is being exceeded
    } while (query has pages && a limit set by the user is not exceeded);
}

7.3 Application layout

7.3.1 Tab Organization

A drawback of an analysis and visualization tool which allows the user to have a complete view and understanding of the data is the risk to overwhelm her. For this reason, I decided to keep my application
organized in tabs, showing general data at the beginning and allowing the user to visualize more specific analyses later. The application has been divided in 6 tabs:

- **Tab 1 - When it’s happening:** The first tab is automatically displayed when the tweets are retrieved. By means of a flow graph it allows the user to spot peeks in tweeting activity, while some pie charts give some other general information about the retrieved tweets.

- **Tab 2 - What is happening:** The second tab shows a visualization of the content of tweets by means of a dynamic tag cloud, created using the method described in Section 2. A combination of the tag cloud and a flow graph allows the user to visualize how the content evolves over time.

- **Tab 3 - Who are the opinion makers:** The third tab contains an analysis of the most influential users and tweets for the current query, representing the patterns of mentions and retweets. These patterns are represented by means of directed graphs, realized using the methods described in Section 3 and 4.

- **Tab 4 - Where it’s happening:** The fourth tab contains an interactive map showing the location of the retrieved tweets, it’s also possible to observe how the position of tweets changes over time.

- **Tab 5 - How people feel:** The fifth tab contains a sentiment analysis of the retrieved tweets. Changes of sentiment over time are represented. The analysis is performed using the method described in Section 5.

- **Tab 6 - Credits:** The sixth tab contains the credits of the application.
7.3.2 Screen layout

The application has a width/height ratio of 16:9. To simplify elements positioning in the screen, I created some "virtual" points which help me in referring to a given position of the screen.

As can be seen in Figure 46, I divided the page in 4 sectors, each of which is divided in 4 other sectors. Points labeled with an A are the centers of the 4 main sectors of the page; points labeled with a B are positioned in the center of the 4 arms of the cross dividing the page in 4; C points are the corners of the page and the centers of the 4 sides of the page; D points are positioned between C points.
This kind of layout greatly simplifies the positioning of graphical elements in the screen; for example, if we wanted to obtain the layout of Figure 47, if we suppose that each element has a constructor of the kind `public Element(float posX, float posY, float width, float height)`, without these points we should write:

\[
\begin{align*}
\text{Element element1} &= \text{new Element(pageW/2, pageH/4, pageW, pageH/2);} \\
\text{Element element2} &= \text{new Element(pageW/4, pageH*3/4, pageW/2, pageH/2);} \\
\text{Element element3} &= \text{new Element(pageW*3/4, pageH*3/4, pageW/2, pageH/2);} \\
\end{align*}
\]

Using the reference points, defined in Processing as `PVector(float posX, float posY)`, the code becomes:

\[
\begin{align*}
\text{Element element1} &= \text{new Element(B2.x, B2.y, pageW, pageH/2);} \\
\text{Element element2} &= \text{new Element(A4.x, A4.y, pageW/2, pageH/2);} \\
\text{Element element3} &= \text{new Element(A3.x, A3.y, pageW/2, pageH/2);} \\
\end{align*}
\]
7.4 Inter-tabs functionality

As can be seen in Figure 48, when the user opens the application, an initial screen is displayed. It contains a brief guide on how to use the application. Two items are always present during the use of the application, they are:

- A search bar;
- a navigation menu.
7.4.1 The search bar - Search, Import and Export tweets

The search input field is implemented using the Processing library controlP5\(^1\). It allows the user to input a term or a complex query using Twitter API’s operators. The search bar is always present in foreground, allowing the user to perform a new query at any time.

A summary of all the operators which are allowed by the Twitter API is reported in Table VII:\(^2\)

\(^1\)http://www.sojamo.de/libraries/controlP5

\(^2\)Adapted from: http://www.tweetarchivist.com/about/operators
### TABLE VII: TWITTER API OPERATORS

<table>
<thead>
<tr>
<th>Example</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>twitter search</td>
<td>containing both &quot;twitter&quot; and &quot;search&quot;. This is the default operator.</td>
</tr>
<tr>
<td>&quot;happy hour&quot;</td>
<td>containing the exact phrase &quot;happy hour&quot;.</td>
</tr>
<tr>
<td>love OR hate</td>
<td>containing either &quot;love&quot; or &quot;hate&quot; (or both).</td>
</tr>
<tr>
<td>beer -root</td>
<td>containing &quot;beer&quot; but not &quot;root&quot;.</td>
</tr>
<tr>
<td>#haiku</td>
<td>containing the hashtag &quot;haiku&quot;.</td>
</tr>
<tr>
<td>from:alexiskold</td>
<td>sent from person &quot;alexiskold&quot;.</td>
</tr>
<tr>
<td>to:techcrunch</td>
<td>sent to person &quot;techcrunch&quot;.</td>
</tr>
<tr>
<td>@mashable</td>
<td>referencing person &quot;mashable&quot;.</td>
</tr>
<tr>
<td>&quot;happy hour&quot; near:&quot;miami&quot;</td>
<td>containing the phrase &quot;happy hour&quot; and sent near &quot;miami&quot;.</td>
</tr>
<tr>
<td>cat since:2010-12-27</td>
<td>containing &quot;cat&quot; and sent since &quot;2010-12-27&quot; (yyyy-mm-dd).</td>
</tr>
<tr>
<td>ftw until:2010-12-27</td>
<td>containing &quot;ftw&quot; and sent up to date &quot;2010-12-27&quot;.</td>
</tr>
<tr>
<td>movie -scary :)</td>
<td>containing &quot;movie&quot;, but not &quot;scary&quot;, and with a positive attitude.</td>
</tr>
<tr>
<td>flight :(</td>
<td>containing &quot;flight&quot; and with a negative attitude.</td>
</tr>
<tr>
<td>traffic ?</td>
<td>containing &quot;traffic&quot; and asking a question.</td>
</tr>
<tr>
<td>hilarious filter:links</td>
<td>containing &quot;hilarious&quot; and linking to URLs.</td>
</tr>
<tr>
<td>news source:twitterfeed</td>
<td>containing &quot;news&quot; and entered via TwitterFeed</td>
</tr>
</tbody>
</table>
A radio button allows the user to set the limit of retrieved tweets. The possible values are: 100, 250, 1000, 2500, 5000. The maximum value has been fixed to 5000 since retrieving more than this amount of tweets without exceeding the Twitter API rate limit can take up to some hours. In any case the user is able to perform the same query more than once and then merge the retrieved tweets in a single dataset, this operation can be performed by means of the import/export buttons.

The export button creates an "output.csv" file containing all the retrieved tweets and, for each of them, the information needed to load and display the tabs, that is:

- ID (the ID of the tweet)
- Date of creation (date of creation of the tweet)
- Text (the text of the tweet)
- User (the screen name of the creator)
- Source (the source of the tweet, e.g. Twitter for iPhone)
- Language of the user (taken from his/her profile)
- Latitude
- Longitude
- Real location data (indicates if the tweets contains real location data)
- Geocoding active (indicates if the location data has been generated using geocoding)
- Sentiment (the sentiment of the tweet)
- Number of followers of creator (taken from the profile of the user)
• Avatar of the creator (taken from the profile of the user)

• A list of mentions (all users mentioned in the tweet)

• ID of the original tweet (if it is a retweet)

The application also generates two network files (mentions.gexf and retweets.gexf) which represent the network of mentions and the network of retweets created by the application and can be archived and further analyzed with network analysis tools such as Gephi.

The import button imports a "input.csv" file contained in the same folder of the application, without header and matching the following format:

• ID: 18 digits

• Date of creation: with the following format: Tue Feb 19 00:00:04 CET 2013

• Text: any string

• User: any string, the @ is not needed

• Source: any string

• Language of the user: any string, it’s preferable to use the ISO codes

• Latitude: any decimal number

• Longitude: any decimal number

• Real location data: "true" or "false"

• Geocoding active: "true" or "false"

• Sentiment: "neg", "pos" or "neu"
• Number of followers of creator: any positive number

• Avatar of the creator: a link to the avatar of the user

• A list of mentions: a list of mentioned users, separated by commas

• ID of the original tweet (if it is a retweet): any integer number

If some attributes are missing, the related tab will be empty, for example if the sentiment attribute is not present for all the tweets, then the sentiment analysis tab will not visualize any result.

The geocoding radio button can be used to switch on/off the geocoding functionality which allows to impute the location of a user from the information given in her profile. When geocoding is turned on, the collecting process can take up to 10 times more.

7.4.2 The Menu

![The menu](image)

Figure 50: The menu

The menu is implemented using controlP5 and is always present in foreground, allowing the user to switch between the various data analysis and visualization tabs of the application. As can be seen in Figure 50, the currently selected tab remains highlighted.
7.5 Tab 1 - When it’s happening

After the user inputs a query, matching tweets are retrieved and he/she is redirected to the first tab which displays general information about flux, languages and sources of tweets. It contains three main elements:

- A flux graph;
- A languages pie chart;
- A sources pie chart.
7.5.1 The flux graph

![The flux graph](image)

Figure 52: The flux graph

The flux graph displays the number of tweets matching the query over time. The x axis usually represents days; the last day is the current one (or the most recent one in case no tweets were created in the current day), while the first one depends on the limits imposed by the Twitter API and on the popularity of the topic, for example retrieving the most recent tweets talking about "pizza" will return tweets created in the last minutes, while choosing a more specific query will return tweets up to a week ago. The y axis represents the number of tweets created on each day.

The graph adapts itself to fit the current curve. The x axis, as said before, usually represents days but it automatically switches to hour-level resolution if the query returns tweets which don’t cover more than two days. The y axis automatically uses a step which is a multiple of 5 and proportional to the highest peek.

In order to allow the user to visualize the exact number of tweets created on a given day, the graph has been made interactive: when the position of the mouse approaches a day (or hour) the exact number of tweets appears (see Figure 52).
7.5.2 The languages pie chart

![Languages of tweets](image)

Figure 53: The languages pie chart

This pie chart represents the percentages of tweets being tweeted by a user speaking a given language. This information is obtained from the profiles of the creators of the collected tweets.

To avoid an incomprehensible result, due to the excessive number of languages, the pie chart is designed to group all the elements whose percentage is less than a given threshold (5%) under a label called "OTHERS".

7.5.3 The sources pie chart

![Sources of tweets](image)

Figure 54: The sources pie chart
This pie chart represents the percentages of tweets being tweeted from a given source (e.g. Web, Twitter for iPhone, Facebook etc.).

Since the possible sources of a tweet are countless (for instance any blog or website with a Twitter plugin), also in this case the sources with a percentage less than 5% have been grouped under the label "OTHERS".

Since the Twitter API, when asked for the source of a tweet, returns an HTML tag of the kind: 

```html
<a href="URL of the source">Name of the source</a>
```

containing the name of the source and its URL, a regular expression has been used to parse the names of the sources.
7.6 Tab 2 - What is happening

The second tab contains a visualization of the content of tweets by means of a dynamic tag cloud, implemented using the method described in Chapter 2, and a controller. The user is able to choose a time span and visualize the related tag cloud. This tab also implements an "auto-play" functionality, which makes the tag cloud "evolve" by itself, showing to the user how the content of tweets varies over time.

Figure 55: Second tab
7.6.1 The controller

![Image of the tag cloud controller](image)

Figure 56: The tag cloud controller

The tag cloud controller is implemented by means of a flux graph and a Range controlP5 element. The combination of these elements allows the user to easily select a time span, possibly focusing on the peaks shown in the flow graph. The selection of the range is fluid and has a precision to the level of minutes. As soon as the user releases the control, the tag cloud updates. The starting and ending times of the selection are always displayed and updated in real-time to allow the user to properly select the desired time interval.

The controller element also features three buttons which can be used to control the auto-play functionality, which is described in the last subsection of this section.

7.6.2 The tag cloud

![Image of the tag cloud](image)

Figure 57: The tag cloud
The tag cloud is based on all the retrieved tweets included in the interval determined by the controller. The computation of the weight of each word is performed using OpenCloud Java library\(^1\) and it is used when computing the size of the text of each word; the size of each word is related to its weight by the following formula:

\[ s = S \left( \frac{w}{S} \right)^{0.6} \]

where \( s \) is the font size of the word, \( S \) is the maximum desired font size, and \( w \) is the weight of the word.

The drawing of the words is performed in order to fill the space under the controller without exceeding the sides of the screen and trying to keep the cloud symmetric at the same time.

The words are "colored" with colors ranging from black to white, where white words are the ones with the highest frequency. This choice was made to let the user immediately spot the most important words, not only by their size but also by their brightness; this also helps to keep track of the words when the auto-play functionality is turned on.

In order to make the tag cloud more meaningful, the filtering process described in Section 2.4 is performed.

### 7.6.3 The auto-play functionality

This functionality gives more expressive power to the tag cloud, making it "evolve" by itself; in this way the user can sit back and better spot the changes in the content of the retrieved tweets over time. The user can control this functionality by means of three buttons.

\(^1\)http://opencloud.mcavallo.org
The **Play** button makes the animation start from the beginning using the time span selected by the user (if no time interval is selected, the application automatically chooses the best one) and a step which is proportional to the time interval covered by the entire dataset.

The **Pause** button makes the animation pause. It can be restarted from where it was by pressing the **Play** button.

The **Stop** button stops the animation. If the Play button is pressed afterwards the animation starts from the beginning.

To prevent the auto-play from interrupting the draw cycle of Processing, it has been implemented as a Timer object, which executes at each second, moving the time interval of the selection forward and updating the tag cloud. When the selection reaches the end the Timer stops automatically.

![Figure 58: Self evolution of the tag cloud](image)

*Figure 58* shows the evolution of the tag cloud by means of the auto-play functionality at three different time steps.
7.7 Tab 3 - Who are the opinion makers

This tab contains a visualization of an interactive dynamic graph representing the pattern of mentions in the retrieved tweets, and another one representing the pattern of retweets. Both graphs are constructed using the method described in Chapter 3. It’s possible to switch between the two visualizations by means of a radio button. The user is also able to interact with the graph and analyze the evolution of the patterns over time.

Figure 59: Third tab
7.7.1 Interactivity

As can be seen in Figure 60, the default view of the application makes the graph automatically fit in the screen. The user is then able to interact with the graph by dragging and zooming in and out using the mouse wheel, this allows to better explore the graph, even when the auto-play functionality is active.

7.7.2 Nodes

Figure 61: Particular of a graph of mentions
The nodes are colored based on the modularity algorithm described in Section 3.5, which allows to highlight the different "communities" of users and tweets.

The size of the nodes is computed using the methods described in Section 4 which allow to immediately spot the most authoritative users (in the graph of mentions) and the most influential users and tweets (in the graph of retweets).

The result can be seen in Figure 61, where the various "communities" are easily distinguishable and the "authorities" are easily detected (in this case @Youtube)

7.7.3 Edges

The edges are depicted as arrows representing, as described in Section 3.4, mentions (in the graph of mentions) or retweets (in the graph of retweets). The user is able to set the color of the edges to be white, the same color as the source node, or of the same color as the target node. It is also possible to hide/show the edges or to let the application decide; in this last case the edges will be visible only after a given zoom level has been reached.

7.7.4 Labels

The labels of the graph, representing the screen names of the users, have been made proportional in size to the size of the nodes, in this way they help better identifying the opinion makers. There are three options to choose from for the display of labels.

The first option is to always hide them (see Figure 62).
Another option is to always show them (see Figure 63).

In this case, even if my layout method greatly facilitates the positioning of labels, some overlaps are possible. To cope with this problem I’ve made the size of the labels also dependent on the zoom level, in this way, in case of overlaps, zooming in will eliminate the problem.

A third option is the "auto" one (see Figure 64).
In this case the application automatically decides which labels to show; in particular all the labels are hidden when the zoom level is minimum, then, zooming in will make the names of the most important users appear depending on an order based on the size of the nodes, until all the labels are shown. This option allows a user to immediately spot the opinion makers and prevents her from being overwhelmed by too many names.

### 7.7.5 Time range controller

The controller of this tab (Figure 65) allows the user to select any time range from the retrieved dataset; only the nodes and edges which belong to the selected interval, depending on the method described in Section 3.6, are displayed.
Figure 66: Time range controller of the mention and retweet graphs

Figure 66 shows the same pattern of mentions when two different time ranges are selected.

7.7.6 Auto-play functionality

The auto-play functionality is very useful to understand the evolution of patterns over time, in fact it is possible to watch users appearing and disappearing (in the network of mentions) and tweets "propagating" when being retweeted (in the network of retweets). The user can play, pause or stop the animation and select any time range, while the step between each update of the graph is set to 1 second.

Figure 67: Auto-play functionality of a graph of mentions
7.8  Tab 4 - Where it’s happening

This tab displays the retrieved tweets on an interactive map. The user is able to zoom, pan, change some visualization aspects, choose any time range and activate the "auto-play" functionality. In this way it is possible to obtain a full overview of the dynamics related to the location of tweets and their possible evolution over time.

Figure 68: Fourth tab
7.8.1 The map

The map has been implemented starting from the Modest Maps library\(^1\). The user is able to pan an zoom using the mousewheel. I’ve also added a functionality which allows to pan and zoom on a given point by double clicking it. The user is also able to switch between three different views: Road, Terrain and Mixed.

7.8.2 Geocoding

Since the percentage of geolocated tweets is <1\%, using just the real coordinates of the tweets would allow to represent on the map just a small percentage of them. To cope with this problem I’ve implemented a geocoding module which tries to convert the information related to the location of the user, contained in her profile, into coordinates. With this technique it is possible to represent approximately the 60\% of tweets. My application is using the MapQuest API\(^2\), which doesn’t have a rate limit, to perform geocoding, therefore it is possible to geocode up to thousands of tweets. Since a request to the API has to be made for each tweet and since the answer can take up to 1 second, this functionality slows down the loading of the tabs. It can be enabled or disabled by the user.

\(^1\)http://modestmaps.com

\(^2\)http://developer.mapquest.com
As can be seen in Figure 69, activating the geocoding functionality (geocoded tweets are colored in red, while the green ones are those containing true geo-location information) allows to represent a much greater number of tweets.

7.8.3 Size by followers
As can be seen in Figure 70, the user can choose to set the size of the markers (tweets) on the map to be always the same (on the left), or to set it to be proportional to the number of followers of the user (on the right). In this way it’s possible to have an idea of how many people a tweet will reach. The figure highlights also a problem caused by geocoding: many users in their profile don’t write their exact location but the largest city nearby (for example Rome); this causes some overlaps on the map. Fortunately the user can avoid this problem by reducing the time range or by activating the auto-play functionality, since it is unlikely that two tweets overlap both in space and time.

### 7.8.4 Tooltip

![Figure 71: Map tooltip](image)

When clicking on a tweet on the map, a tooltip automatically appears, showing the content of the tweet, the screen name of its author and also her profile picture.
7.8.5 Time range controller

The time range controller allows a user to select any time interval from the retrieved dataset, filtering only the tweets which belong to the interval.

![Figure 72: Time range controller of the interactive map](image)

The time dimension is very useful for identifying (if present) patterns in the evolution of the location of tweets over time.

7.8.6 Auto play functionality

![Figure 74: Auto-play functionality of the interactive map](image)
The auto-play functionality can be accessed by means of three buttons: Play, Pause and Stop. It is implemented in the same way as in Tab 2 and 3.
7.9 Tab 5 - How people feel

The fifth tab of the application contains the visualization of the results of the sentiment analysis method described in Section 5. The user is able to visualize and understand the evolution of sentiment over time by means of some options.
7.9.1 The sentiment pie chart

This pie chart represents the percentages of positive, negative and neutral tweets, based on the options chosen by the user. When the tab is open for the first time, the pie chart is related to the whole dataset, the user is then able to select any time range using the controller of the graph, this will update the pie chart.

7.9.2 The sentiment graph

Figure 76: Sentiment pie chart

Figure 77: Sentiment graph
The graph shows the value of the Overall Sentiment Index, as defined in Section 5.5, for each day (it automatically switches to hour-level resolution if needed) of the retrieved dataset. In this way it’s possible to observe the evolution of the index over time.

The y axis doesn’t display any legend because it’s not important to have the exact number, since the scale it’s not important. The top of the graph represents the maximum positive index, while the bottom represents the maximum negative index. A line in the middle of the graph and a light green/red background helps the user in better understanding the positiveness/negativeness of the index.

The graph also features a time range controller which can be used to "feed" the pie chart with the tweets belonging to the selected time interval. The minimum and maximum dates of the interval are displayed respectively in the top-left and top-right corners.

7.9.3 Options

![Classifier Options](image)

Figure 78: Options of the sentiment classifier

The user is given the possibility to choose which classifier to use between the emoticon classifier and the logistic regression classifier (she can also use both combined). A message suggests the user to turn on only the emoticon classifier if she is analyzing tweets which are not in English, since this is the language of the tweets the logistic regression classifier has been trained on.
The sixth tab of the application contains the credits of the application, that is all the external libraries and services which have been used in my application. The list is here reported:

- *controlP5*\(^1\) is a GUI and controller library for Processing, written by Andreas Schlegel. It has been used to implement some GUI elements of the application.

- *Gephi Toolkit*\(^2\) packages the essential Gephi modules (Graph, Layout, Filters, IO etc.) in a standard Java library. It has been used to implement the graph of mentions and the graph of retweets.

- *LingPipe 4.1.0*\(^3\) is a tool kit for processing text using computational linguistics. It has been used to train the sentiment classifier based on logistic regression.

- *MapQuest API*\(^4\) is a free, unlimited geocoding Web Service. It has been used to implement the geocoding module of my application.

- *Modest Maps*\(^5\) is a small, extensible, and free library for designers and developers who want to use interactive maps in their own projects. It has been used to implement the interactive map on which the geo-located tweets are displayed.

\(^1\)http://www.sojamo.de/libraries/controlP5

\(^2\)https://gephi.org/toolkit

\(^3\)http://alias-i.com/lingpipe

\(^4\)http://developer.mapquest.com

\(^5\)http://modestmaps.com
- **OpenCloud**\(^1\) is a Java library for generating and managing tag clouds. It has been used to compute the frequency of the terms in the retrieved tweets to build the dynamic tag cloud of the second tab.

- **opencsv 2.3**\(^2\) is a very simple csv (comma-separated values) parser library for Java. It has been used to implement the import/export functionalities of my application.

- **Processing 2.0b7**\(^3\) is an open source programming language and environment for people who want to create images, animations, and interactions. It is the core of the application, since it has been used to implement many of its visualizations.

- **Twitter4J**\(^4\) is an unofficial Java library for the Twitter API. It has been used to make my application retrieve the most recent tweets matching a given query.

\(^1\)http://opencloud.mcavallo.org

\(^2\)http://opencsv.sourceforge.net

\(^3\)http://processing.org

\(^4\)http://twitter4j.org/en/index.html
CHAPTER 8

TEST CASES

In this chapter I will present some test cases which allow to better understand the analysis and visualization capabilities of the methods developed in this thesis and implemented in my application.

Section 8.1 presents a graph which visualizes the pattern of retweets created after the resignation of Pope Benedict XVI; Section 8.2 shows some visualizations of tweets and retweets during the 2013 Europe Concert of Justin Bieber; while Section 8.3 contains an analysis of the 2013 Italian elections based on the flux, content and sentiment analysis methods described in the previous chapters.

8.1 Visualization of the graph of retweets after Pope Benedict XVI's resignation

On the 11th of February 2013, Pope Benedict announced his pending resignation, effective 28 February 2013, because of "lack of strength of mind and body"\(^1\).

I used my application to build and visualize a network of retweets related to this event. The network (the final result can be seen in Figure 79) is a directed graph composed by 11239 nodes and 9178 edges. The tweets were collected from 11:45 AM of the 11th of February 2013 to 10:20 PM of the same day.

To obtain the result in the figure I modified some parameters of the application as the maximum number of retrieved tweets and the color of nodes and edges. In particular I used a darker color to highlight users, while tweets are represented in light gray. The layout of the graph took approximately 2 minutes to obtain the final result.

\(^1\)http://en.wikipedia.org/wiki/Pope_Benedict_XVI

121
Figure 79: Visualization of retweets after Pope Benedict XVI’s resignation
In the middle of the figure it is possible to distinguish some big clusters composed by an influential tweet, in the center, and its retweets, all around; some other clusters are far from the center, while many tweets with just a few retweets fill the circular shape.

Surprisingly, the most influential tweets are not the ones created by news authorities but the ones belonging to satirical/funny profiles. I isolated these clusters in Figure 80.

Figure 80: Clusters of satirical tweets on the resignation of the Pope

The @Queen_UK profile is a satirical profile pretending to be Elizabeth II of England; as can be seen in the figure, it generated 5 tweets related to the resignation of the Pope. The text of the largest one is "Text from Pope Benedict XVI: "Smell you later b**es! I'm outta here!" " and, with its 1347 retweets, it’s the most influential tweet collected, while its creator is the most influential user of the network.

Another cluster which can be seen in the figure is composed by two tweets generated by @Charles_HRH, a fictional parodic profile pretending to be Charles, Prince of Wales. The content of the first tweet is "The economy must be bad. Even God’s making people redundant. #Pope", while the second one says
"Queen Beatrix of the Netherlands and Pope Benedict XVI have both stepped down to give someone else a chance. How very thoughtful."

The third cluster is composed by a single tweet and its retweets. It was created by @OMGFacts, a profile tweeting about surprising facts. The content of the tweet is "Pope Benedict XVI announced that he’s resigning from the papacy later this month. He’s the first pope to resign since 1415".

![Figure 81: Clusters of informative tweets on the resignation of the Pope](image)

Other clusters are composed by the tweets created by news media. The most influential users, shown in Figure 81, were @nytimes (The New York Times), @BreakingNews (Breaking News), @BBCWorld (BBC News) and @BloombergNews (Bloomberg News).

This test case demonstrates that the method implemented in my application allows to build and visualize a complete graph of retweets related to an event, which can be used to analyze the retweeting pattern, the diffusion of information and the main vectors of the diffusion.
8.2 Visualization of the pattern of retweets during the 2013 tour of Justin Bieber

In this section I will show the results of an analysis and visualization of the pattern of retweets on a map during four stages of the 2013 Europe tour of Justin Bieber. The analysis is based on the use of three features of my application: geocoding, the possibility of exporting the retrieved tweets and the capability of exporting the graph of retweets. For each stage I performed various steps in order to obtain the final visualizations. First of all I activated the geocoding functionality of my application and then exported the .csv file containing the most recent 10000 tweets. I filtered the file, keeping only those tweets containing true coordinates and the ones from which it was possible to obtain the coordinates through geocoding. After that I opened the .gexf file exported by my application, containing the graph of retweets, and added to it the tweets contained in the filtered .csv file. In this way I obtained a graph where the nodes are all the retrieved tweets with their true or geocoded location and an edge between two nodes indicates that a tweet is a retweet of the other one. Then I opened the resulting .gexf file using Gephi and used the GeoLayout plugin\(^1\) to layout the tweets based on their coordinates, then I made the edges curved and tuned the color and opacity of the nodes to let them appear as lights. Finally I exported the resulting graph as a .svg file and overlaid it on a night world map.

\(^1\)https://marketplace.gephi.org/plugin/geolayout/
8.2.1 March, 04 2013 - London

Figure 82: Retweets during 2013 Justin Bieber’s Europe Tour - London

Figure 82 shows the most recent 10000 tweets and retweets matching the query "Justin Bieber" at the starting time of the concert in London. It is possible to see how many tweets are concentrated in the Southern area of the United Kingdom, near the concert location. It's also interesting to notice how the information "flows" from the area of the concert to North and South America and Asia. These pattern of retweets is mainly caused by a tweet by Justin Bieber, some hours before the concert, saying "In a good mood. Got my friends and team with me here and ready for Day 1 at the O2. London get ready! #BELIEVEtour".
Figure 83 represents the tweets and the pattern of retweets related to Justin Bieber obtained analyzing the most recent 10000 tweets at the starting time of his concert in Barcelona. As can be seen from the figure, there is a high tweeting activity in Spain and in particular in the area of the concert. It is also possible to notice a wide flow of information from and to Spain, in particular, a tweet from Los Angeles saying "If Justin Bieber and One Direction collaborated together they would beat Gangnam Styles record." has been largely retweeted in the concert area.
8.2.3 March 19 2013 - Paris

As can be seen in Figure 84, many tweets, among the most recent 10000 collected at the starting time of the event in Paris, are located in the concert area. There is also a lot of retweeting flow directed from and to Paris, most of which is caused by a tweet by Bieber, saying "blast it at midnight. me and iamwill —– #THATPOWER", referring to the new single from will.i.am which features Justin Bieber. Another consistent flux of retweets is created by a tweet from Mexico in which the author says that he doesn’t like Justin Bieber but he loves the passion of his fans; since this tweet is written in Spanish, it is retweeted almost exclusively in Spain and South America.
Figure 85: Retweets during 2013 Justin Bieber’s Europe Tour - Bologna

Figure 85 shows the most recent 10000 tweets collected at the starting time of the concert in Bologna. As can be seen in the figure, a lot of tweets have been generated in Italy, in particular near the concert area. Moreover, a tweet from Justin Bieber saying "Italy I hear you!! #BELIEVEtour" is retweeted all over the world, especially in the United States and in South America.

This test case demonstrates that my method for the construction of a graph of retweets, if combined with the geocoding capabilities of my application, allows to obtain an interesting visualization of the geographical spread of information during an event.
8.3 Analysis and visualization of tweets in the 2013 Italian elections

In this section I will show the result of an analysis of tweets related to the 2013 Italian elections performed using the methods developed in this thesis and implemented in my application. The analysis includes four main aspects:

- Collection: All the Tweets created between 02/11/2013 and 02/27/2013 from politicians and from people mentioning politicians (this was possible thanks to the from: and to: operators of the Twitter API) have been collected using the Export functionality of my application.

- Flux analysis: The flux analysis tab of my application has been used to visualize the number of tweets per day from and to politicians in order to identify peaks which correspond to events.

- Content analysis: The content analysis tab of the application has been used to generate a tag cloud of the tweets from and to each politician for each day in order to visualize and compare the thoughts of politicians with the thoughts of the people.

- Sentiment analysis: The sentiment analysis tab of the application has been used to obtain a graph of sentiment over time which allows to visualize the positive/negative reactions of people to political events and statements. To obtain a more significative result I trained a sentiment classifier on a dataset of 2459 manually labeled tweets from and to the six candidates, created in the two weeks before the start of the analysis.

- Outcome prediction: In the last paragraph I will compare some indexes which can be useful to predict the outcome of the elections.
The combination of these aspects allowed me to obtain a clear and complete analysis of the elections, identifying the events which caused an increment in number of tweets, visualizing what politicians are saying and how people react in terms of content and sentiment.

### 8.3.1 The candidates

I focused my analysis on all the tweets from and mentioning the six candidates for the elections (Silvio Berlusconi, Pier Luigi Bersani, Mario Monti, Beppe Grillo, Antonio Ingroia and Oscar Giannino). Here below a brief overview of each of them:

*Silvio Berlusconi* is an Italian politician, entrepreneur and owner of the A.C. Milan football club. He served three times as Prime Minister of Italy from 1994 to 1995, 2001 to 2006 and 2008 to 2011. He is often referred to as "Il Cavaliere" (The Knight) for his Order of Merit for Labour. He is the leader of the Pdl party ("Il Popolo della Libertà", The People of Freedom).

*Pier Luigi Bersani* is an Italian politician and Secretary of the PD ("Partito Democratico", Democratic Party), Italy’s leading center-left party. Bersani was Minister of Industry, Commerce and Craftsmanship from 1996 to 1999, Minister of Transport from 1999 to 2001, and Minister of Economic Development from 2006 to 2008.

*Mario Monti* is an Italian economist who has been the Prime Minister of Italy since 2011, leading a government of technocrats in the wake of the Italian debt crisis. Monti resigned as Prime Minister on 21 December 2012 after the passing of his second Budget. Monti served as a European Commissioner

1 adapted from Wikipedia
from 1995 to 2004. Monti has also been Rector and President of Bocconi University in Milan for many years.

_Beppe Grillo_ is an Italian comedian, actor, blogger and politician. He has been involved in politics since 2009 as founder of the M5S ("MoVimento 5 Stelle", Five Star Movement). He used to perform in theatres talking about themes which include energy usage, corruption, finance, freedom of speech, globalization, and technology. He is a strong proponent of Internet freedom. His movement is often defined as "Anti-Politics" because of his idea that politicians are only subordinates of the people and that they should work for the country only for a short time, they should not have criminal records, and should focus on thinking about the problems of the country without any other conflicts of interest.

_Antonio Ingroia_ is an Italian magistrate and politician and leader of "Rivoluzione Civile" (Civil Revolution), a left-wing coalition of political parties. Ingroia is also the director of a UN investigation against narcotraffic in Guatemala. He is also a writer and contributes regular columns to the daily newspaper Il Fatto Quotidiano.

_Oscar Giannino_ is an Italian journalist and politician, he is the former president of the free-market oriented party "Fare per Fermare il Declino" (Act to Stop the Decline).
8.3.2 Flux analysis - identifying the main events

Figure 86: Events of the 2013 Italian elections

*Figure 86* represents the number of tweets per day mentioning each of the six candidates. From the trend of the graph it’s possible to identify seven main events which caused an increment of flux for one or more candidates:

(a) 02/12/2013: Monti is interviewed at Tg La7, a newscast, Uno Mattina, a talk show, and Radio Capital, a radio network.
(b) 02/15/2013: Monti accuses Berlusconi from the talk show Agorà, Bersani attends a meeting in Bologna.

(c) 02/17/2013: Grillo holds a meeting in Genova and refuses the only television interview that he had ever granted.

(d) 02/18/2013: Berlusconi attends a meeting in Milan with Roberto Maroni, center-right candidate President of Lombardy.

(e) 02/20/2013: Oscar Giannino resigns from the presidency of his movement after it was discovered that he had fabricated his resume by adding false academic claims.

(f) 02/22/2013: Grillo holds the final meeting of his campaign in Rome.

(g) 02/25/2013: Results of the elections.

(h) 02/26/2013: Reactions after the elections.

### 8.3.3 Event (a) - 02/12/2013

![Figure 87: Flux of tweets to Monti - 02/12/2013](image)

As can be seen in Figure 87, there is a clear peak in the number of tweets per day (2123) to Mario Monti caused by the three public appearances of the outgoing Prime Minister.
Figure 88: Content of tweets from Monti - 02/12/2013

*Figure 88* represents a tag cloud of the tweets created by Mario Monti during this day. They are focused on his appearances (#unomattina, #radiocapital and #tglas7) and on his program which aims to increase the number of women workers (#agendadonne and donne).

Figure 89: Content of tweets to Monti - 02/12/2013

As can be seen in *Figure 89*, the tweets directed to Mario Monti are not focused on the public appearances themselves but on their content (#donne, women) and on more general topics like what Monti has to do (fare), what he did (fatto), the tax burden he imposed (tasse). Some people are also commenting on Monti saying that the populism of Grillo is devastating (devastante) for Italy (l’Italia).
Figure 90: Sentiment of tweets to Monti - 02/12/2013

In *Figure 90* is possible to notice that the general sentiment towards Monti’s statements is negative, but with a slight increment with respect to the previous day.

8.3.4 **Event (b) - 02/15/2013**

Figure 91: Flux of tweets to Monti - 02/15/2013

Figure 92: Flux of tweets to Bersani - 02/15/2013
Figure 91 and Figure 92 show respectively a peek in tweets (1768) to Monti caused by his participation to the TV show Agorà and a peek in tweets (1547) to Bersani due to his meeting in Bologna, the capital of his home Region.

Figure 93: Content of tweets from Monti - 02/15/2013

The content of Monti’s tweets (Figure 93) highlights his appearance at the talk show (#agorà) but no specific topics emerge, in fact the most frequent terms are #sceltacivica (the name of Monti’s “Party”), persone (people) and politica (politics).

Figure 94: Content of tweets to Monti - 02/15/2013
Also the content of the tweets directed to Monti contain quite general topics like *politica* (politics), *governo* (government) and *paese* (Country). Some people comment on *Buffon*, an Italian goalkeeper, captain of the Italian National Team, declaring his support to Monti. Some tweets make fun of this fact associating the name of the goalkeeper "Buffon" to an insult to Monti, "buffone" (buffoon).

![Figure 95: Sentiment of tweets to Monti - 02/15/2013](image)

The overall sentiment of people talking about the outgoing Prime Minister is more positive than negative and it’s stable with respect to the two previous days.

![Figure 96: Content of tweets from Bersani - 02/15/2013](image)

From the content of Bersani’s tweets (*Figure 96*) it’s possible to deduce his participation to a meeting in Bologna (#bologna) and the content of his speech: it’s necessary to lower the fidelity tax (*fedeltà*...
fiscale) on the working class (#lavoro and lavoro) in order to lower (abbassare) the tax burden. Bersani also congratulates the administrators (amministratori) of the Region Emilia-Romagna for their diligence in repairing the damage caused by the earthquake of 2012. The dimension of the term "noi" (we) underlines the personal commitment to deliver the promises of the election program.

Figure 97: Content of tweets to Bersani - 02/15/2013

No specific topic emerges from the content of the tweets directed to Bersani but it’s interesting to notice the contraposition between the terms "noi" and "voi" ("we" and "you"), and "siamo" and "siete" ("we are" and "you are").

Figure 98: Sentiment of tweets to Bersani - 02/15/2013
The overall sentiment increased from the previous day but it’s still negative, mainly because many supporters of PD (Democratic Party) accuse Bersani of not being very incisive in his election campaign.

8.3.5 Event (c) - 02/17/2013

On the 17th of February a clear peek (3403) in the number of tweets is caused by Grillo’s meeting in Genova, his home town, and his decision, communicated by means of a tweet, to cancel the interview which was supposed to be televised by Ski.

Figure 99: Flux of tweets to Grillo - 02/17/2013

Figure 100: Content of tweets from Grillo - 02/17/2013
The tweets from Grillo (*Figure 100*) reveal the content of his speech in Genova (*genova* and *#genova*): he doesn’t like to appear on television because he prefers to be among the people (*gente*) in a public square (*piazza*).

![Content of tweets to Grillo - 02/17/2013](image)

*Figure 101: Content of tweets to Grillo - 02/17/2013*

Most of the tweets directed to Beppe Grillo are focused on the interview (*l’intervista*) for the TV channel *Sky* he cancelled. Many people insinuate that he could have cancelled it because he was afraid (*paura*) to answer challenging questions (*domande*)..

![Sentiment of tweets to Grillo - 02/17/2013](image)

*Figure 102: Sentiment of tweets to Grillo - 02/17/2013*
As could be guessed from the tag cloud, people’s reactions are negative and cause the sentiment index to sharply decrease with respect to the previous day. The sentiment towards Beppe Grillo will reach its minimum on the following day.

8.3.6 Event (d) - 02/18/2013

The maximum number of tweets directed to Berlusconi is reached on the 18th of February, when he attends a meeting in Milan, another one organized by Confindustria, the Italian entrepreneurs’ federation, in Monza, and then is interviewed by the talk show "Quinta Colonna".

Figure 103: Flux of tweets to Berlusconi - 02/18/2013

Figure 104: Content of tweets from Berlusconi - 02/18/2013
The tweets from Berlusconi (Figure 104) are focused on his meeting in Milan (#fieramilano), his meeting in Monza (#confindustria) and his interview on TV (#quintacolonna).

Figure 105: Content of tweets to Berlusconi - 02/18/2013

In his meeting in Milan Berlusconi announces overtaking PD at the polls. The general people’s reaction is clear from the tag cloud (Figure 105), they say "In all (tutti) these years (anni) you didn’t do (avete fatto) anything good".

Figure 106: Sentiment of tweets to Berlusconi - 02/18/2013

The content of the tweets is reflected in a decrement of sentiment with respect to the previous day.
8.3.7 Event (e) - 02/20/2013

A great increment (3240) in the number of tweets directed to Oscar Giannino is reached on the 20th of February. Luigi Zingales, a member of his party and finance professor at The University of Chicago Booth School of Business, left the party on the 18th, accusing Giannino of never obtaining a Master Degree at his University. On the 19th Giannino resigned from the presidency of the party, admitting that he had never obtained the Master Degree. On the 20th Giannino confesses that also his two Italian Degrees in Law and Economics were never obtained.

Figure 108: Content of tweets from Giannino - 02/20/2013
The only term which stands out from the tag cloud of tweets created from Giannino (Figure 108) is *dimissioni* (resignation).

Figure 109: Content of tweets to Giannino - 02/20/2013

The tweets directed to Giannino (Figure 108) are mostly focused on the fake Master Degree discovered by Luigi Zingales.

Figure 110: Sentiment of tweets to Giannino - 02/20/2013

As can be seen in Figure 110 the sentiment towards Giannino had reached its maximum positiveness on the 17th of February, after his speech against University Lobbying, it greatly decreases on the 18th, after Zingale leaves, it slightly increases when Giannino admits his faults and resigns but it decreases again on the 20th when it’s discovered that also his two Italian Degrees have been faked and it reaches its
maximum negativeness on the 21th, when another lie it’s discovered: Giannino never attended Zecchino d’Oro, a famous Italian children’s song festival.

8.3.8 Event (f) - 02/22/2013

A peek (2931) in the flux of tweets directed to Beppe Grillo on the 22th of February is caused by his final meeting at St. John Square in Rome.

The tweets from Grillo (Figure 112) invite his followers to watch (seguite) the stream (diretta) of his meeting from St. John (San Giovanni) Square (piazza).
The tweets mentioning Grillo (Figure 113) thank (*grazie*) him for his effort, saying that he is great (*grande*) and repeat one of Grillo’s most famous mottoes "mandiamoli *tutti a casa*", which means "let’s send them all home", referred to the politicians, accused of being old corrupt traditionalists.

The overall sentiment, as could be guessed from the content of the tweets, is positive and reaches its maximum on the day of the results of the elections.
The 2013 Italian elections ended up with extremely surprising results, that did not match the outcome that was forecast by polls experts. The first instant polls tended to confirm the polls of the months before, showing a strong victory for Bersani’s coalition, the defeat of Berlusconi’s PDL, a good but not exceptional result for Grillo’s Movimento a 5 Stelle, a quiet disappointing result for Monti’s Scelta Civica and the complete defeat of Ingroia’s Rivoluzione Civile and Giannino’s Fare per Fermare il De-
clino. As soon as the first projections on the real votes started to come out, it was clear that the previous polls were off by a long shot: the official Ministry projections showed a huge victory for Berlusconi over Bersani in 5 key regions (Lombardia, Veneto, Sicilia, Campania, Puglia), a surprising result for Grillo, projected over 20% of total votes, in the hunt for becoming the first Italian party, a decent but not exceptional result for Monti, and the defeat of Ingroia and Giannino, projected to be out of the Parliament. Given those projections, the outcome would have been a victory for Berlusconi at the Senate (something that was thought to be out of the realm of possibility just a couple of hours before). The aftermath of those shocking projections is clearly outlined by the number (Figure 115) and content of tweets of the 25th.

![Tweet Content](image)

Figure 116: Content of tweets to Bersani - 02/25/2013

The tweets towards Bersani (Figure 116) outline the feeling of defeat (perdere, perso) and the fact that Bersani is to blame for that, because of his weak campaign and therefore should now (ora) be sacked (dimissioni, dimettiti). They also show that the general consensus is that if Bersani’s opponent in PD’s primary elections, Florence’s major Matteo Renzi (Renzi), was the candidate, he would have easily won. Many people make fun of the fact that Bersani, in his campaign, compared Berlusconi to a jaguar
(giaguaro), and said that he would have erased all his spots, but ironically he seemedly got defeated by that very jaguar.

Figure 117: Content of tweets to Grillo - 02/25/2013

The tweets towards Grillo (Figure 117) reflect the amazing result that was being projected in those hours, and people encourage (forza) and thank (grazie) Grillo, saying that he is great (grande). They also show a strong focus on what is to be done now (ora) and a sentiment of collective victory (siamo). It is also to be noticed that many tweets highlight the fact that Grillo’s party (m5s) was in the run to become the first (primo) Italian party.

Figure 118: Content of tweets to Monti - 02/25/2013
The tweets towards Monti (Figure 118) highlight the sentiment of gratitude (grazie) of his electors, despite the somewhat disappointing result (risultato). The tweets are thankful also for what Monti did (fatto) for the Italians (italiani) with his government of technocrats to bring Italy out of the economical crisis. They also comment on the fact that, given the first projections, Monti’s allies, long time political leaders Gianfranco Fini (Fini) and Pier Ferdinando Casini (Casini) were at risk of not meeting the minimum number of votes required in order to enter the Parliament.

Figure 119: Content of tweets to Ingroia - 02/25/2013

Also the tweets towards Ingroia (Figure 119) show a strong sentiment of gratitude (grazie) despite (comunque) the extremely poor result of his party. They show that his electors believed (creduto) in him. A great amount of tweets mock Ingroia, by saying that now that he has not been elected he should go back (torni) in Guatemala and quit with politics, since his campaign was a failure.
### TABLE VIII: ELECTION RESULTS - SENATE

<table>
<thead>
<tr>
<th>Party/Coalition</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Center-left coalition (PD et al.)</td>
<td>31.63%</td>
</tr>
<tr>
<td>Center-right coalition (PdL et al.)</td>
<td>30.72%</td>
</tr>
<tr>
<td>Movimento 5 Stelle (Grillo)</td>
<td>23.79%</td>
</tr>
<tr>
<td>Scelta Civica (Monti)</td>
<td>9.13%</td>
</tr>
<tr>
<td>Others</td>
<td>4.73%</td>
</tr>
</tbody>
</table>
### TABLE IX: ELECTION RESULTS - CHAMBER OF DEPUTIES

<table>
<thead>
<tr>
<th>Party/Coalition</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Center-left coalition (PD et al.)</td>
<td>29.55%</td>
</tr>
<tr>
<td>Center-right coalition (PdL et al.)</td>
<td>29.18%</td>
</tr>
<tr>
<td>Movimento 5 Stelle (Grillo)</td>
<td>25.55%</td>
</tr>
<tr>
<td>Scelta Civica (Monti)</td>
<td>10.56%</td>
</tr>
<tr>
<td>Others</td>
<td>5.16%</td>
</tr>
</tbody>
</table>
The 26th, a few hours after midnight, the official results of the elections were handed out. Despite a great showing, and the victory in many critical regions, Berlusconi did not manage to obtain a relative majority in the Senate, thanks to Bersani’s victory by a minimum amount of votes in the region of Piemonte. Bersani got 31.63% of votes, equaling 123 senators, while Berlusconi got 30.72%, equaling 117 senators. The situation was somewhat the same for the Chamber of Deputies, where Bersani got 29.55%, while Berlusconi got 29.18%. Thanks to the majority prize that the Italian electoral law applies
for the Chamber, however, Bersani got 297 deputies, while Berlusconi only got 98. Beppe Grillo and
his Movimento 5 Stelle received an astonishing 23.79% of votes for the Senate (which equates to 54
senators) and 25.55% of votes for the Chamber (which equates to 109 deputies), becoming the most
voted Italian party. Monti barely made it above the quorum, with 9.13% of votes for the Senate (19
senators) and 10.54% of votes for the Chamber of Deputies (47 deputies). Ingroia (1.79%, 2.25%) and
Giannino (0.9%, 1.11%) did not reach the quorum, and therefore did not get any representative into the
Parliament. In the light of these results, Bersani won the Chamber of the Deputies by a wide margin
(despite an advantage of just around 100,000 votes over Berlusconi) thanks to the majority prize. The
situation of the Senate was more complex, because despite the relative victory of Bersani, no party had
enough senators to form a govern (158 senators are required). Given the improbability of Berlusconi,
Grillo or Bersani forming an alliance between them, and the low number of senators elected by Monti,
the ultimate outcome was the ungovernability of the Senate. This outcome is reflected in the flux of
tweets of the 26th (Figure 120).

Figure 121: Content of tweets to Grillo - 02/26/2013
The tweets towards Grillo (Figure 121) show a strong sentiment of celebration, thanking (grazie) him and saying once again that he is great (grande). This time they not only thank Grillo, but all of the party (tutti) for this result. They also show the growing understanding among his electors that now (ora) that they are the first Italian party, they should focus on what is to do (fare) in the upcoming future (adesso).

Figure 122: Content of tweets to Ingroia - 02/26/2013

The tweets towards Ingroia (Figure 122) continue to show the gratitude (grazie) of his electors, with a slight decrease in the jokes about him going back to Guatemala and quitting politics.

Figure 123: Content of tweets to Berlusconi - 02/26/2013
The tweets towards Berlusconi (Figure 123) show the sentiment of victory (vinto) that goes through his electors, since such a good result was really unexpected and victory was inches away. They thank (grazie) him (presidente, silvio) and say that he is great (grande), because once again (ancora) did what in the beginning of the campaign seemed almost impossible. They focus also on the possibility of reaching an agreement (accordo) with Bersani to form a government, and on the promise of removing the IMU tax (l’imu).

8.3.11 Prediction

Figure 124: Flux of positive tweets - 02/26/2013
My first idea in order to find an index which could be useful for predicting the surprising results of the elections was to use the number of positive tweets per day towards the candidates, since it represents people’s support to them and it could be directly related to the voting intentions of the electors. Figure 124 shows the number of positive tweets mentioning the six candidates in the weeks leading to the electoral weekend of the 24th and 25th of February. It is easy to notice some main points. Firstly, the ever increasing amount of positive tweets generated by Grillo’s Movimento 5 Stelle. Secondly, the high amount of positive tweets generated by Monti’s Scelta Civica, which apparently did not translate into electoral success. Thirdly, the decreasing at first, and finally stable, but still low if compared to Grillo’s, amount of positive tweets generated by Bersani’s PD. Fourthly, the low amount of positive tweets generated by Ingroia’s Rivoluzione Civile and Giannino’s Fare per Fermare il Declino (taking out of the equation the aftermath of the degrees scandal that involved him). Lastly, the low amount of positive tweets generated by Berlusconi’s Il Popolo della Libertà. In order to properly predict (to a certain degree) the outcome of the elections, it is necessary to take into account the number of tweets posted by the candidates; it is clear that a candidate that tweets a lot tends to excite his electors, that will respond to him, creating a virtuous circle leading to an always increasing amount of positive tweets. Figure 125 shows what happens when dividing, for each day, the number of positive tweets mentioning the candidates by the number of tweets generated by them on that day.
Firstly, it should be noticed how the amount of "spontaneous" positive tweets mentioning Grillo stays very high, and peaks right before the election days: it can be concluded that the amazing electoral result of Movimento 5 Stelle could have been predicted by taking this indicator into account. Secondly, the amount of "spontaneous" positive tweets generated by Bersani increases proportionally, but is still not on Grillo’s level, therefore that could have been an indicator of the fact that the Partito Democratico was strong, but was not the strongest party in the country, that the electors were not that excited about the party, and that the results of Bersani would have been a lot worse than what anticipated in the polls.
Thirdly, Monti’s, Ingroia’s and Giannino’s amounts of positive sentiment, when adjusted to the tweets that they posted, strikes as significantly lower than the two aforementioned candidates. In particular, regarding Monti, it can be concluded that his frequent usage of Twitter and his many appearances on television increased the amount of tweets around his party, and that is why all the polls forecast him around 15-18%. When the number of tweets from the candidate is taken into account, on the other hand, it is easy to see how that forecast was inflated, just as the amount of positive tweets shown in Figure 12.4. Lastly, the amount of spontaneous positive tweets generated by Berlusconi is still surprisingly low. This can be explained by comparing it with the data regarding the profile of Berlusconi’s typical elector: he is very strong, and therefore gets voted, by women and men over 60 years of age. It is easy to see the way this influences this metric, since that is a category of people that seldomly, if at all, use computers or the Internet, let alone subscribing on Twitter.

It can be concluded that the data coming from Twitter could be used in order to predict the outcome of the Italian elections, but only to a certain extent. The prediction is correct if referred to young people, but it fails to reveal the intentions of the elder ones. The validity of the prediction is far more accurate in the 18-50 range, while it clearly becomes less indicative as the age range grows. However, by using my method, a certain amount of predictions that the polls failed to forecast could have been made, and specifically:

- the amazing success of Grillo, who was the real winner of these elections
- the disappointing results of Bersani
- the fact that Monti’s possibilities of success were overrated
- the fact that Ingroia and Giannino had no chances of reaching the required quorum
CHAPTER 9

IMPROVEMENTS TO THE STATE OF THE ART

In this chapter I will compare my application with the state of the art applications, on different aspects, using the same queries, highlighting the improvements which have been achieved. Since there isn’t an application which allows to perform all and the same analyses my application does, I will compare it with different applications for each aspect.

Sections from 9.1 to 9.6 are dedicated, in order, to flux analysis, content analysis, patterns of mentions and retweets, geo-location, sentiment analysis, import/export functionality.

9.1 Flux analysis and general information

Here I will compare the results produced by the first tab of my application with the ones produced by Tweet Archivist\(^1\).

Using a query which is not world-wide popular, in this case "Renato Zero", an Italian singer, the flux graphs produced by the two applications are very similar. The query has been performed on the 21st of February 2013.

\(^1\)described on page 63
Both applications retrieve all the tweets returned from the Twitter API, both graphs (Figure 126 and Figure 127) correctly show dates on the x axis and number of tweets on the y axis, using an adequate resolution. Both application allow to obtain the actual number of retrieved tweets for each day by mouse hovering.

If a more popular query is performed, the results become very different. In this example I chose the query "haiku" (a very short form of Japanese poetry), still performed on the 21st of February 2013.
In this case the flux graph produced by my application (Figure 128) correctly switches to an hour-level resolution, while the graph of TweetArchivist (Figure 129) displays just a singleton representing the total number of tweets created on the day the query was performed.

Therefore my application gives the possibility to correctly visualize the flux of tweets also when the query is so popular that all the retrieved tweets have been created in the last few hours.
Both TweetArchivist and my application display a pie chart related to the languages of tweets and one related to the sources of tweets. To compare the two applications I’ve performed again the query "Renato Zero" on the same day.

The pie chart produced by TweetArchivist (Figure 131) displays many languages ranging from 76.7% to 0.1%. The labels on the pie chart are overlapping and mostly unreadable. The 3D effect can...
distort the actual dimension of the "slices". The greenish colors are very similar and distinguishing between them can result quite difficult for the user.

The pie chart produced by my application (Figure 130) avoids the problem of displaying many irrelevant languages by grouping them under the label "OTHERS". The result is a clear and simple pie chart.

Looking at the percentages of the two pie charts it is possible to spot some differences, in fact both agree that the percentage of tweets in Italian is around the 80% but mine shows English as the second language while the other says it is Portuguese. As described in Section 7.5, my pie chart uses the language information retrieved by the user profile, since it’s quite complex to guess the actual language of the retrieved tweets. Unfortunately Tweet Archivist doesn’t declare how the language information is determined.

![Sources of tweets](image)

Figure 132: Sources pie chart of my application - query "Renato Zero"
Figure 133: Sources pie chart of Tweet Archivist - query "Renato Zero"

The two pie charts of languages (Figure 132 and Figure 133) display the same percentages. My pie chart uses again the "OTHERS" label to group the sources with a percentage <5% while the other displays 25 sources, many of which with a percentage <1%. Again the labels of the Tweet Archivist’s pie chart are difficult to read and half of the sources are associated to a greenish color.

9.2 Content analysis

To compare the content analysis capabilities of my application with the state of the art I chose Cloud.li\(^1\), the only application I found which allows to visualize a dynamic tag cloud of tweets. For this comparison I performed the query "#sochi2014", which is related to the Sochi 2014 Winter Olympics, on the 22nd of February 2013.

As can be seen in Figure 134, Cloud.li produces a tag cloud in which the darkness of the words and their size is related to their frequency. The tag cloud updates at regular intervals but it’s impossible to

\(^1\)described on page 68
understand how many tweets are being considered each time, if their number is constant or is the time range considered to be constant. Moreover the big nickname monopolizes the tag cloud, making many of the other words too light and small to be properly read.

As can be seen in Figure 135, the flux graph and time range selector of my application allow a user to choose an arbitrary time range and to generate the related tag cloud, in this way the user can focus on peeks to see if and how an important event is causing changes in the content of tweets. Moreover the auto-play functionality allows to choose any time range and then watch how the tag cloud evolves over time and, if something interesting is spotted, it’s possible to pause the animation. Moreover a minimum size and brightness of the words is used in my tag cloud to make them readable in any case.
9.3 Patterns of mentions and retweets

To compare the capabilities of my application to visualize the patterns of mentions and retweets with the state of the art I chose Revisit\(^1\). To make this comparison I performed the query "#pope" on March 13, 2013, during the Papal conclave.

Revisit (see Figure 136) performs a very interesting visualization of the mentions and retweets among the retrieved tweets, highlighting in light blue the flow of retweets and in light green the conversational "paths".

\(^1\)described on page 72
The analysis is very detailed, since it allows to visualize the content of the single tweets. It also takes into account the time dimension, representing the most recent tweets on the right, allowing the user to follow the evolution of the flows of retweets and mentions in time. A drawback of this application is that the level of detail is too high, the user is immediately overwhelmed by the connections between the tweets, but is not able to have an overview of the patterns of mentions and retweets.

My application creates a network of mentions (Figure 137) and a network of retweets (Figure 138).
Figure 137: Patterns of mentions of my application - query "#pope"

From the graph of mentions it’s possible to immediately spot, thanks to my ranking algorithm and the gradual appearance of the labels, the most authoritative users with respect to the topic of the query, in this example they are two news media and a journalist.

Figure 138: Patterns of retweets of my application - query "#pope"
In the network of retweets it is possible immediately spot the most influential users, in this case a journalist and a radio, and the retweeting pattern associated to their tweets.

In both the networks the modularity algorithm implemented in my application allows to distinguish, by means of different colors, the various "communities" of users and tweets. Moreover my application allows to focus on any time range to have a snapshot of the networks at a given time or to activate the auto-play functionality, which allows to sit back and watch the self evolution of the networks over time.

9.4 Geolocation

To compare the capability of my application to display the retrieved tweets on an interactive map, I chose Twitter Map\(^1\), an application with similar functions. To test the two applications I used the query "#chicago".

![Interactive map of tweets of Twitter Map - query "#chicago"](image_url)

Figure 139: Interactive map of tweets of Twitter Map - query "#chicago"

\(^1\)described on page 61
Twitter Map retrieves the most recent 500 tweets and displays, as can be seen in Figure 139, the only five of them which contain true geo information on an interactive map powered by Google.

My application instead (Figure 140) retrieves the last 5000 tweets and displays more than 3000 of them using also the geocoding of the information contained in the profile of the users.

The two applications have some similar capabilities: both display the tweets containing true geolocation information, both use an interactive map to display the tweets and both give the possibility to select a specific tweet to visualize its content.

However, my application includes many additional features; first of all it allows to retrieve up to 5000 much more than the 500 retrieved by Twitter Map and it is able to display almost 1000 times more tweets by means of the geocoding functionality. Moreover my application gives the possibility to make
the dimension of the markers on the map proportional to the number of followers of the creators of the tweets, in order to represent the potential influence that a tweet could reach (Figure 141).

Figure 141: Interactive map of my application, dimension by followers - query "#chicago"

Another very important advantage of using my application is the possibility of selecting any time range to display only the tweets which belong to it, moreover, by means of the auto-play functionality it is also possible to let the markers appear and disappear in chronological order, revealing how the location of tweets evolves over time. Let’s for example think of a concert and how the tweets "approach" the location of the event as the start time gets closer.
9.5 Sentiment analysis

I compared the sentiment analysis capabilities of my application with the ones of Sentiment140\(^1\), one of the most famous sentiment analysis applications for tweets. I had to try many different queries before finding one to compare the two applications, since Sentiment140 very often says "Sorry, there were no results. Try broader query, like obama". In the end I decided to test the two applications using two very polarized queries: "I love when" and "I hate when". Both searches have been performed on the 22nd of February 2013.

![Figure 142: Sentiment analysis of Sentiment140 - query "I love when"](image)

When searching "I love when" Sentiment 140 (Figure 142) returns an analysis based on just the 72 most recent tweets (covering approximately a time range of 20 minutes). Some tweets are not classified and excluded, neutral tweets are not considered, therefore the pie chart incorrectly sums up to 100%. The percentages of the pie chart could seem wrong also because the query was very biased towards positiveness, therefore the percentage of positive tweets should be much greater, but it must be taken

\(^1\)described on page 76
into account that many tweets matching the query are sarcastic, for example: "I love when my job doesn’t pay me #sarcasm #gimmemymoney", and express a negative sentiment.

Figure 143: Sentiment analysis of my application - query "I love when"

My application, on the query "I love when" (Figure 143) returns an analysis performed on the 5000 most recent tweets, which cover a period of approximately a week. The percentages of the pie chart are similar to the results of Sentiment140, but this time also the percentage of neutral tweets is shown. Moreover the graph of sentiment allows to track the evolution of sentiment over time. The time range selector on the graph can be used to focus on any time interval and make the pie chart update.
When searching for "I hate when", Sentiment 140 (Figure 144) analyzes just the 65 most recent tweets (covering a time range of approximately 3 minutes). This time the majority of tweets is negative.

My application, performing the query "I hate when" (Figure 144), returns the last 5000 tweets, covering a time range of approximately 4 hours. It is possible to notice from the pie chart that the
The majority of tweets is negative, while there are less positive and neutral tweets. From the graph it can be seen that the sentiment index is always negative but has become more positive in the last two hours.

From the two tests we can conclude that both applications perform an accurate classification but mine is performed on a greater number of tweets, it allows to track the evolution of sentiment over time, to update the pie chart selecting any time range and it shows the percentage of neutral tweets. Sentiment140 let the user choose between two languages (English and Spanish) while mine allows to turn on the emoticons classifier to analyze tweets in any other language. Sentiment140 shows a list of the retrieved tweets and their sentiment, while my application writes these information in the exportable .csv file.

9.6 Retrieved tweets and import/export functionality

The only analysis application, between the ones described in Section 6.2, which allows to export the retrieved tweets is Tweet Archivist\(^\text{1}\). It allows to export all the retrieved tweets (there is a limit for unsubscribed users) in a .xls file. For each tweet all the information which are made disposable by the Twitter API are saved. My application allows to export all the retrieved tweets, without limits, in a .csv file. Most of the information retrieved from the Twitter API and some others coming from the profiles of the users (for example the number of followers, needed to compute the authority of a user) are saved.

I wasn’t able to find any application which allows to perform an analysis or visualization of any pre-collected dataset of tweets. My application instead allows to import any dataset previously exported by the application or any other adequately formatted dataset. All the tabs which have enough data to

\(^{1}\text{described on page 63}\)
show are loaded (for example if the coordinates of the tweets are missing in the .csv file the geo-location tab won’t display any point on the map).

The import/export feature of my application is essential for research purposes, since it allows to archive any interesting dataset, being able to perform all the analysis and visualization features allowed by the application at any time.
CHAPTER 10

CONCLUSION

The aim of this thesis was to develop an advanced method for the analysis of topics and events on Twitter and their evolution over time, this method has been implemented in an application to perform analyzes and visualizations on any unconstrained dataset of tweets; this application was designed to fulfill the lacks present in the state of the art, described by Marcus et al. (7).

My application correctly implements the four components of my method, that is the method for the creation and visualization of a dynamic tag cloud of tweets (Chapter 2), the method for the creation and visualization of patterns in mentions and retweets (Chapter 3), the method for the identification of authoritative users, opinion makers and influent tweets (Chapter 4), and the method for the analysis of sentiment in tweets and its evolution (Chapter 5).

Moreover I have demonstrated, through some test cases (Chapter 8), how my method and my application can successfully be used to analyze many aspects of unexpected events (Section 8.1), events composed by multiple stages (Section 8.2) and events lasting for weeks (Section 8.3).

Finally I have illustrated how the implementation of my method in my application brings a significant improvement to the state of the art (Chapter 9).

From this work I realized that many interesting improvements to my method and my application could be made. Planned future work includes the creation of a logistic regression classifier and an emoticon classifier trained on a larger dataset and tested in a more rigorous way, the creation of a set of logistic regression classifiers trained to classify sentiment in tweets related to a specific topic (e.g.
politics and economics), and the combination of the methods developed in this thesis to obtain a more powerful overview of a topic or event on Twitter, for example a dynamic geo-located graph of retweets would allow to visualize the evolution of the retweeting pattern in time and space, a dynamic geo-located graph of retweets with the use of my influence index would allow to visualize the most influent users and tweets and how their influence behaves and evolves in time and space.


# VITA

**NAME:** Giorgio Cavaggion  

**EDUCATION:**  
- Laurea di I Livello in Information Technology Engineering, Politecnico di Milano and Politecnico di Torino, 2011  
- Master of Science in Computer Science, University of Illinois at Chicago, 2013  

**WORKING EXPERIENCE:**  
- June - August 2013 - Fellow of the Eric & Wendy Schmidt Data Science for Social Good Summer Fellowship 2013  
- February 2013 - Poll clerk at Italian elections  
- June 2007 - Internship at Town Hall Press Office of San Donato Milanese as press officer and designer