Visualizing Social Media Content with SentenTree


Presented by: David Johnson
Unstructured Text Documents

Twitter/Social Media collections are many unstructured text documents

Unstructured text documents are hard to analyze!

Many authors, redundant information

Can accumulate many of these documents in short time
Summarizing Unstructured Documents

Could extract common information & present a world cloud

Word clouds good at a glance to gain overarching theme

World clouds lose concepts and structure

How do we maintain semantic representation?
SentenTree
SentenTree

Node-link visualization with force-directed placement

Edge between words indicates occurrence in same tweet

Spatial arrangement is syntactic ordering

Large font indicates high frequency of occurrence
Frequent Sequential Patterns

Initialization steps:

- Normalize tweets
- Perform tokenization
- Root node of tree of sequential patterns is initial pattern
- Initial pattern contains no words
- Grow new sequential patterns from the root
Frequent Sequential Patterns

all tweets (189450)

1

goal (74554) (114896)
Frequent Sequential Patterns
Frequent Sequential Patterns
Frequent Sequential Patterns

![Diagram showing frequent sequential patterns with nodes labeled goal, first goal, and watching, and associated counts such as 74554, 189450, etc.](image)
Frequent Sequential Patterns
Frequent Sequential Patterns
Frequent Sequential Patterns

```
all tweets (189450)

goal (74554)

first goal (41344)

first goal world (36136)

first goal world cup (36081)

first goal world cup own (14861)

watching (11248) (103648)

own goal (13233) (19977)

(33210)

(5208)

(55)

(114896)

...```

13
Frequent Sequential Patterns

![Graph showing frequent sequential patterns](image-url)
Frequent Sequential Patterns
Interaction Demo

https://twitter.github.io/SentenTree/
Visual Encoding

SentenTree uses a \textit{constrained} force-directed placement algorithm.

Placement constraints: \textit{word order}, \textit{vertical}, \textit{horizontal}
Visual Encoding

Only **word order** constraint applied
Visual Encoding

Only *word order* constraint applied

*Horizontal* and *vertical* constraints added
Considerations: Tokenization

Stop words and punctuation removed

Numbers, hashtags, urls, @ handles are matched

No stemming performed
Critique

The **Bad:**

No stemmer

Final visualizations are still sometimes ambiguous
Critique

The **Good:**

System accomplishes design goals

Well written paper, easy to understand examples

Scalable
Thanks!

Questions?