Tactical Analytics

Ed Morris
Dr. Jaime Carbonell
Ben Bradshaw
Kevin Pitstick
Petar Stojonov
Daegun Won
Distribution Statements

Copyright 2016 Carnegie Mellon University

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Department of Defense.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN “AS-IS” BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[Distribution Statement A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

DM-0004127
Edge Analytics Pipeline for Streaming Situational Awareness

Previous Work:
- Developed a platform for building and testing data analytics for streaming textual data
  - Tested multiple analytics in public safety settings; Multi-day music festivals, Little League World Series, Visit of Pope Francis
  - Retrospective analysis: Cairo and Benghazi

Current Focus:
- Recognize Events
  - Focus for FY17; add video & improve text
- Determine Credibility
  - Related Work*
- Extract “Patterns of Life”
  - Primary Focus, Ongoing
- Interpolation and Extrapolation
  - Future Work

*Information about this work is available during poster session
Patterns of Life and Scripts

• A “Pattern of Life” represents stereotypical sequence of events and interactions in a particular context

• The research community calls these “scripts” (Schank & Abelson, 1977)

• Scripts help analysts relate emerging situations to what is already known
  - “I have seen that before” (recognition of instance of a script)
  - “He probably entered through Turkey” (what happened previously – interpolation)
  - “Since he is an American with good language skills, they will probably use him as a spokesperson” (what may happen next -- extrapolation)

• Scripts are a good way to introduce data to a new analyst or an analyst in a new job
Pattern of Life examples

ISIL script for takeover of a village¹:
- List the powerful families
- Name the powerful individuals
- Find out income sources
- Identify names, sizes, and control of rebel brigades
- Identify illegal activities (Sharia Law) that could be used for blackmail

Other Examples²:
• Russian aggression in Crimea²:
  ▪ Transporting or erecting missile launcher: Apr 6 2012: “[Russian] military has begun deploying S-400 mobile surface-to-air missiles in Kaliningrad”
  ▪ Mobilization of forces; Russian military activity increased in the 12 months leading up to the Crimean invasion. Information was available on the web indicating this activity
• North Korean nuclear test preparation e.g. vehicle activities, site activities, etc.

¹From notes of Samir Abd Muhammad al-Khilifawi (Haji Balr), considered the architect of ISIS, killed in a firefight with Syrian rebels. Found with org charts, lists, & schedules describing ISIS strategy <http://www.spiegel.de/international/world/islamic-state-files-show-structure-of-islamist-terror-group-a-1029274.html>
²Recorded Future Blog <https://www.recordedfuture.com/russian-military-activity/>
Goals and Challenges

Long term goal: build the pipeline to recognize then validates events, recognizes patterns (scripts) and allows for interpolation (previous events) and extrapolation (prediction of future events)

Objective for FY16: Understand the challenges of script learning

• Event recognition and ordering
  - Recognizing events in free-form text
    - DARPA DEFT is improving single and multiple sentence event recognition
    - No viable solution for multiple document event recognition
  - Establishing relationships among events (e.g., order, causality)

• Credibility analysis of events (more information available)

• Script creation and modification
  - Determining if event sequences represent a new script or an instance of an existing script
  - Preventing invalid pathways from being incorporated into scripts
Our Approach

Events = \{\text{Actor, Activity, Time, Location} \ldots \}

Problems with event recognition forced us to use a proxy dataset with easily recognizable events

- Box scores for major league baseball are readily available and events are very easy to recognize (runs, outs, etc.)

Deal with script extraction and matching; not on performance and volume issues (simple scripts aren’t small)

Learn constraints based on initial portion of the dataset in real-time

Generate the script that represents the data

- Structure of the DAG (directed acyclic graph) representing \( \frac{1}{2} \) inning
- Frequency for branches in the DAG (based on box score data)
Algorithm 1 Script Building Algorithm
Input: a list \( S \) of \( N \) sequences of events, distance metric threshold \( R \), timer threshold \( T \)
Output: a resulting list of scripts, \( L \)

1. \( S = \emptyset \)
2. for \( s \in S[1 - 500] \) do
3. & Generate uniqueness constraints \( U \) and pairwise order constraints \( P \).
4. end for
5. for \( s \in S[501 - \text{end}] \) do
6. & if \( L == \emptyset \) then
7. & add \( s \) as a script to \( L \)
8. else
9. & for \( l \in L \) do
10. & for traversal \( t \) in \( l \) do
11. & if \( \text{dist}(t, s) \leq R \) then
12. & mark \( l \) for adding
13. end if
14. end for
15. end for
16. if an \( l \in L \) is marked for adding then
17. for traversal \( t \) in \( l \) do
18. & if \( \text{dist}(t, s) \) is smallest and no constraints in \( U \) and \( P \) are violated then
19. & add \( s \) to \( l \)
20. end if
21. end for
22. else
23. & add \( s \) as a script to \( L \)
24. end if
25. end if
26. end if
27. for \( l \in L \) do
28. & \( \text{timer} += 1 \)
29. & if \( \text{timer} > T \) then
30. & remove \( l \) from \( L \)
31. end if
32. end for
33. end for
34. return \( L \)

Algorithm 2 Script Building Algorithm
Input: a list \( S \) of \( N \) sequences of events, distance metric threshold \( R \), timer threshold \( T \)
Output: a resulting list of scripts, \( L \)

1. \( S = \emptyset \)
2. for the first 500 sequences \( s \) do
3. & Generate uniqueness constraints \( U \) and pairwise order constraints \( P \).
4. end for
5. for \( s \in \) the remaining sequences in \( S \) do
6. & if Working list of scripts \( L \) is empty then
7. & add \( s \) as a script to \( L \)
8. else
9. & for each script \( l \in L \) do
10. & Check if any of the traversals of \( l \) have a small enough distance metric compared to candidate \( s \), and if so, mark \( l \) for addition.
11. end for
12. & if any \( l \in L \) is marked for adding then
13. & Add \( s \) to the minimum distance traversal of \( l \) if no constraints are violated
14. else
15. & add \( s \) as a script to \( L \)
16. end if
17. end if
18. end for
19. Remove any scripts from \( L \) that have not been incremented in \( T \) iterations.
20. end for
21. return \( L \)
Recognize if a New Sequence Represents a New Branch of an Existing Script

Script X and sequence Z

Similarity

\[ s(X, Z) = 1 - \arg \min_{p \in \text{Path}(X)} \left[ \sum_{x \in p} \beta_x \delta(x, Z) \right] \]

\[ \delta(x, Z) = \begin{cases} 
0 & \text{a match} \\
\alpha_1 & \text{if B.T.} \\
\alpha_2 & \text{insertion} \\
\alpha_3 & \text{deletion} 
\end{cases} \]

\[ \beta_x = 1 \text{ unless specified otherwise by the user} \]
Preventing Invalid Pathways in Scripts

Apply constraints to prevent invalid scripts:

- Input from an expert analyst
  - Direct feedback
  - Learned from curated reports
- Learn from the data (*constraint induction*)
  - Joint learning (learn constraints and scripts while processing operational data) is ideal, but hard
  - Observe initial M sequences to infer constraints before inferring scripts
  - Learn from labeled data

When updating a script, new pathways are checked against *constraints*
Constraints Induction: Order and Uniqueness

Order: Pairwise Constraints

ex) Event X happens before event Y “X >> Y”

• If no order exists we should see each pair ~50/50
• If order does exist we should see much higher frequency for “X >> Y” or “Y >> X”

• Uniqueness: Similar strategy can be applied by looking at the probability of a specific event.
  • When X happens, X happens only once some high percentage of time
  • \( P(X \text{ happened only once in } \geq M \text{ sequences ; N Observations}) > \text{Threshold(} \sim 0.01) \)?
Example Using Box Score Data

- Generated using streaming events extracted from 5000 baseball half-innings
- Constraints identified in first 500 half-innings, rest of data used to generate script
- Some imperfections in the algorithm (e.g., some pathways should be collapsed
Subscript Instances

Most common subscript (3 straight outs)

Multiple runs followed by a triple play

A “rain out” half-inning

Rare half-inning, left out of the generated script (run scores on double play)
Lessons Learned

Scripts can be induced from streaming data
  • Assuming events are correctly identified

Constraints are necessary to avoid obviously invalid pathways
  • Some of them can be learned directly from the data
  • More needed to improve accuracy

Even a simple test case is very complicated
  • We chose a situation with easily identifiable and correctly tagged events
  • We simplified the types of events we considered (e.g. X Runs scored instead of 1 Run, 2 Runs, etc.)
  • We focused on events that were directly related to the length of the script
  • We duplicated sequences of events rather than allowing cycles -- DAGs
FY17 Focus

• Event recognition and extraction
  • Use curated data sets to train classifiers to recognize events in un-curated text
  • Develop approach to recognize events across multiple sources
    • Build on work performed under DARPA contract at CMU and elsewhere

• Extending script learning algorithm to more meaningful data
  • More sophisticated approaches to learning constraints (constraint induction)
  • Direct analyst input for constraints and feedback during script generation

• Recognize that a partial set of events is part of an existing script
  • Necessary for both interpolation and extrapolation