Adaptive Social Sensor Event Detection

Abhijit Suprem and Calton Pu

School of Computer Science, Georgia Institute of Technology
Physical Event Detection

• Traditionally performed with physical sensors
• Some domains require global tracking, and some can be performed locally
  • **Global** – Weather/climate tracking
    • Dense physical multi-sensor coverage (barometric pressure, cloud coverage, humidity)
  • **Global** – Earthquakes
    • Semi-dense sensor coverage (near fault-lines especially)
  • **Global/Local** – Rainfall
    • Dense global sensor coverage
  • **Local** – Flooding
    • Local coverage near flood-prone regions
  • **Local** – Yield monitoring
    • Local coverage on corresponding farm
  • **Local** – Subsurface soil/groundwater monitoring
    • Local coverage on corresponding farm’s water source
Global physical event detection

• Goals of physical event detection
  • Near real-time detection
  • Global detection

• Almost-global detection possible, but slow
• Dense global sensor coverage is difficult or expensive
Dense Global Event Detection

• Waste-water disposal earthquakes
  • require continuous deployment of seismometers near fracking wells
  • As wells move, seismometers also move
  • As wells expand, new seismometers deployed

• Landslides occur under a variety of conditions and sensor coverage is expensive
  • Uneven terrain with loose soil post-rain
  • Earthquakes with loose soil or rain
  • Heavy rain and flooding near mountainous or hilly regions

• Traffic jams
  • Dense camera cover with anomaly and video event recognition
  • Current approach (Google, Bing): aggregate phone data of drivers

• Other city events: protests, marches, accidents, fires
• Other disaster type events: hail, forest fire, disease, infection
Social Sensor

• Limiting factor is dense, global sensors
• Social sensors: social media + web data + blogs
• Advantages
  • Dense, global coverage (4B Internet users, 3B social media users)
  • Near real-time (events reported within 1m – 2hr usually)
  • Increasing ubiquity + rich historical & behavioral data
  • Multi-modal data (text, image, video)
  • Multi-perspective data (multiple users and sources)
Event Detection from Social Streams

• Social streams can be leveraged for various real-world events beyond disasters
  • Earthquake detection
  • Landslide/Flooding detection
  • Traffic jams, riots, social events

• Near real-time coverage

• Variety of physical events can be detected with the same framework

¹Earthquake Shakes Twitter Users: Real-time Event Detection by Social Sensors, Sakaki et al
²Social Sensors and Pervasive Services: Approaches and Perspectives, Rosi et al
Challenges in Social Sensor Event Detection

• NLP on Social Data
  • Social data is noisy + low context
  • NLP is more challenging due to lack of context + noise + short text nature

• Difficult to filter irrelevant topics
  • Text/Image/Video data on large variety of topics (not dedicated sensor)
  • No heuristic or simple filtering rules

• Weak-signal events
  • Millions of events represented in data, with a fraction being relevant
  • Relevant class is the minority class (few training data)

• Concept Drift
  • Changes in underlying data distribution exacerbates above problems
Concept Drift in Social Sensors

- A datapoint $P_i$ is a distribution over events $P(E_a | P_i)$
  - $E_a \in E$ (universe of events)
  - $E_{\text{landslide}} \in E_a$
- Independently, each point is a generative model over signals $S$
  - $P(P_i | S)$
- $E_a = \sum_i^k a_i S_i$
Concept Drift in Social Streams

\[ E_a = \sum_{i}^{k} a_i S_i \]

• Concept drift occurs when distribution of \( a_i \) changes (usually over time)
  
  • *Real* concept drift
    • Changes in \( f(a_i) \) cause changes in true decision boundary
  
  • *Virtual* concept drift
    • Changes in \( f(a_i) \) do not cause changes in true decision boundary

• True decision boundary
  • The actual hyperplanes separating classes
  • ML approximates the true hyperplanes
Types of Concept Drift

• **Real concept drift**
  • Several approaches to detecting and adapting to real drift
  • get oracle labels, and compare error rate over time of classifier
  • If error rate increases, drift has occurred
  • Use oracle labels to retrain model

• **Virtual concept drift**
  • Virtual drift – new regions of data space discovered over time
  • New data is dissimilar from training data
  • Sometimes difficult to generalize existing machine learning event detection rules
Our Dataset

• Physical event detection
• Collected from social sources over several years
• Drift
  • Data ingest techniques change over time
  • Data content changes
  • Increasing noise over time
• Events
  • Landslides
  • Flooding
  • Earthquake
Evidence of Real Drift

False negatives in 2018

False positives in 2018
Real drift – False negatives

- Each data point from 2014-2018 encoded with w2v
- tSNE used for dimensionality reduction on entire dataset (positive + negative)
- For classifier trained on 2014 data only (orange)
- Positive instances of 2018 data indistinguishable from negative samples in 2014
- False negative errors
Real drift – False positives

- For classifier trained on 2014 data only (orange)
- Negative instances (2018) indistinguishable from positive samples in 2014
- False positive errors
Evidence of Virtual Drift

• Shift in positive samples
• Positive samples in 2018 lie in different region than positive samples in 2014
• Virtual drift can lead to real drift
• ML approximates true decision boundary
• So virtual drift can overstep an incorrectly generalized boundary
Putting it together

• Our approach addresses two broad challenges
  • ML-based event detection on social streams
  • Drift detection and adaptation for continuous learning

• ML-based Event Detection framework
  • Our framework is designed to be deployable for various event types
  • Real-time streaming from social sources,
  • Continuous data collection from reputable sources
  • Data processing using pub/sub
  • Event detection with ML classifiers

• Drift detection and adaptation
  • Automated drift detection without oracle labels
  • Drift adaptation without human/oracle labels
Social Stream Event Detection

• Traditional event detection assumptions do not hold

• Event characteristics do not exhibit changes
  • Concept drift phenomenon causes changes in underlying data distribution

• Event detection rules do not fluctuate continuously
  • Concept drift phenomenon causes changes in decision boundaries

• Raw sensor data are not easily calibrated and do not have noise
  • Social sensor data is highly noisy
  • Relevant class is minority class/weak-signal
  • Trend-based methods not feasible for weak-signal events
  • Statistical and deep ML methods useful for social sensor data
Event Detection Framework

**High Confidence Dataflow**
- High latency
- Streamer downloads news articles, government reports
- Event identification to perform event detection
- High confidence sources are stable, with little to no drift

**Social Source Dataflow**
- Low latency, abundant, noisy, global coverage
- Process datapoint
- Heterogeneous Data Integration for labeling (5%)
- ML-Based Event Detection on the rest (95%)
ASSED Environment Setup

• ASSED framework
• Streamers (**High-confidence** and **Social source**)
  • ASSED supports Twitter API, Google Search API, NewsAPI
• ASSED process
  • Primitives for framework process
  • ASSED processes communicate with each other with Apache Kafka

1. Process $M$ exports output as $<$topic-data$>$ pair into Kafka with registered export-key
2. Kafka keeps output until it is requested or 3 days have passed
3. Process $N$ continuously reads data from its import-key topic
4. Process $N$ records key offset for recovery
Streamers

- Each data point is saved on disk and sent to Kafka pub/sub
- Each ASSED process is assigned an \textit{import}- and \textit{export}- key
- Buffers between multiple-input processes
  - Kafka does not deal with multiple ingests
  - A topic item can be processed exactly once or continuously until expire
  - With ASSED, we create a buffer process that manages MI dataflow
  - Buffer ingests single-input and pushes copies for each input in MI flow

<table>
<thead>
<tr>
<th>export-key attributes</th>
<th>Social-source</th>
<th>Reputable-source</th>
</tr>
</thead>
<tbody>
<tr>
<td>streamer</td>
<td>‘ss’</td>
<td>‘rs’</td>
</tr>
<tr>
<td>lang</td>
<td>Any language supported by ASSED application (‘en’, ‘fr’, etc)</td>
<td>‘en’, ‘fr’, etc for reputable text sensors (e.g. news articles), or ‘num’ for numeric data</td>
</tr>
<tr>
<td>keyword</td>
<td>Physical event designation of application (‘landslides’)</td>
<td>Physical event designation of application (‘landslides’)</td>
</tr>
<tr>
<td>source</td>
<td>Name of social network (‘Twitter’)</td>
<td>Name of reputable source (‘NOAA’)</td>
</tr>
<tr>
<td>url</td>
<td>URL of post “twitter.com/.../1072933351441526784”</td>
<td>URL of source; ‘NULL’ if source is a physical sensor endpoint</td>
</tr>
<tr>
<td>post_id</td>
<td>Local auto-incrementing numeric ID</td>
<td>Local auto-incrementing numeric ID</td>
</tr>
<tr>
<td>streamer_timestamp</td>
<td>Local timestamp of commit to \textit{R.Store}</td>
<td>Local timestamp of commit to \textit{R.Store}</td>
</tr>
</tbody>
</table>

\textbf{export-key template}

```
*streamer : lang : key : src : url : id : timestamp*
```

\textbf{value format}

\[ P_i = \{ p_i, l_i, t_i, h_i, u_i \} \]
Metadata Extraction

• Event detection requires location
• NER fails on short-text streams (low context)
• We integrate high-confidence dataflow
• High-confidence events’ locations stored in Metadata cache (Redis)
• Locations used as substring match for Social Source data

• Additional metadata
  • User information
Heterogeneous Data Integration

• Traditional event detection approach
  • Generate model on training data
  • Use initial model for all events
• This fails in drifting environments
  • Virtual drift – generalization failure
  • Real drift – model must be updated
• High-confidence sources are ground-truth data
• Social posts in same spatio-temporal region are labeled as relevant events
• Remaining posts are passed through ML-based Event Detection
• On average, 5% of social posts can be so labeled
Heterogeneous Data Integration

HDI-Labeled Social-source Data

# Social-Source Posts

Jul | Aug | Sept | Oct | Nov | Dec

Unlabeled Data  | 189 | 106 | 193 | 249 | 885 | 223
HDI-Labeled Data | 7205 | 14245 | 4867 | 15847 | 7084 | 4873

Distributed and Event Based Systems, 2019
ML-Based Event Detection

**Filter Generation/Updates**
- a. ASSED generates new filters + updates existing filters
- b. Filters are sent to $F_{\text{Store}}$ with training data

**Event Detection**
- 1. Data cleaning and encoding
- 2. Processed data sent to ASSED
- 3. ASSED matches data to $F_{\text{Store}}$ filters (k-NN)
- 4. Selected filters create an ensemble
- 5. Detected events are sent to $PE_{\text{RDB}}$

**ML-based Event Detection**

- Mapped to Physical Events in **Heterogeneous Data Integration**
- Training Data:
  - Filter Selection Method: b.
- Prediction Data:
  - Prediction: 4.
  - Not Mapped to Physical Events: 2.

**Heterogeneous Data Integration**

- $F_{\text{Store}}$
- Real-time dataflow (Social-source)
- Scheduled dataflow (Reputable-source)

Distributed and Event Based Systems, 2019
Event Detection - Learning

• HDI-Labeled data, where available, is used to generate new classifiers/filters
• Each filter is stored in a Filter database (*F_Store*)
• A filter is referred to using its compressed training data
  • Centroid of training data

• **Concept drift adaptivity**
  • Filters continuously and automatically updated using HDI labels
  • HDI labels do not require human intervention, so no latency in labeling
  • No human cost in labeling/updates either
Event Detection – Classifier filtering

• ASSED allows several modes to filter classifiers for ensemble selection
  • Recent-New
    • Only most recent (prior update/generate window) newly created classifiers
  • Recent-Updates
    • Only most recent updated classifiers
  • Recent
    • All recent classifiers, either new or updated
  • Historical-New
    • All classifiers newly created
  • Historical-Updates
    • All updated classifiers
  • Historical
    • All classifiers created in operational history
Event Detection – Classifier selection

- After classifier filtering, ASSED allows the following selection methods
  - No-further-filtering
    - All filtered classifiers are used to create an ensemble.
    - Ensemble can be unweighted, or weighted on classifier performance
    - Ensemble can also be weighted on distance of classifier centroid to data point
    - Classifiers performance on most recent HDI test-set
  - Top-k Performance
    - Classifiers tested on HDI test-set (stored in F_Store)
    - Top-k performant classifiers used in ensemble
    - Weights: unweighted, performance, or distance
  - Top-k Nearest
    - Top-k nearest classifiers to data point
    - Distance measured using training centroid
    - Weights: unweighted, performance, or distance
Event Detection – Prediction

• Generated ensemble used for prediction

• Evaluation
  • Tested static and adaptive approaches
  • Static – learner trained in 2014 and never updated
  • Adaptive – use ASSED framework
  • LITMUS – Landslide Detection System
  • Built with ASSED Framework

• [https://grait-dm.gatech.edu/demo-multi-source-integration/](https://grait-dm.gatech.edu/demo-multi-source-integration/)
  • Only ASSED version (does not include static version)
Results Preview

Detected by both LITMUS-ASSED and LITMUS-static

Detected only by LITMUS-ASSED

Extraneous information
Extraneous information, Missing context
Extraneous information
Low context, multiple events
Experimental setup

• We tested four broad approaches (including variations)
• We cover overall results here

<table>
<thead>
<tr>
<th>Approach</th>
<th>Description</th>
<th>Available Training Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>N_RES</td>
<td>Non-resilient encoding/classifier without HDI</td>
<td>2014 Data</td>
</tr>
<tr>
<td>RES</td>
<td>Resilient encoding/classifier without HDI</td>
<td>2014 Data</td>
</tr>
<tr>
<td>N_RES-HDI</td>
<td>Non-resilient encoding/classifier with HDI</td>
<td>HDI-Labeled Social data</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(07/18 - 12/18)</td>
</tr>
<tr>
<td>RES-HDI</td>
<td>Resilient encoding/classifier with HDI. (Uses kNN scheme)</td>
<td>HDI-Labeled Social data</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(07/18 - 12/18)</td>
</tr>
</tbody>
</table>
Precision

- **Statistical vs Deep**
  - No significant difference between either in precision
  - N_RES (deep) has slightly more variability in late 2018

- **HDI vs Non-HDI**
  - HDI confers adaptivity from beginning
  - HDI-based updates allow RES-HDI and to outperform N_RES
  - RES-HDI performance begins increasing in late 2018
Recall

- **Statistical vs Deep**
  - Significant variability in recall
  - Recall: higher false negatives

- **HDI vs Non-HDI**
  - HDI confers adaptivity from beginning
  - HDI-based updates allow RES-HDI and to outperform N_RES
Throwback: Drift

False negatives in 2018

False positives in 2018

Negative samples (2014) and positive samples (2018)

Positive samples (2014) and negative samples (2018)
F-Score

- F-score: harmonic combination of precision and recall

- **Statistical vs Deep**
  - Deep learners have variance in performance in drifting conditions without adaptivity
  - Statistical learners deteriorate as well due to low recall

- **HDI vs Non-HDI**
  - HDI confers clear adaptivity
  - HDI-based ensemble (under kNN selection and weighting, with historical filter)
  - F-score: 0.988 for RES-HDI (deep)
Event detection improvement

LITMUS-original vs. LITMUS-ASSED

Normalized Additional Events
Event detection improvement

• We compare LITMUS-ASSED to LIMUS-static

• Events detected in LITMUS-static were also detected in LITMUS-ASSED

• Both Events
  • Events detected in both LITMUS-static and LITMU-adaptive

• LITMUS-adaptive only
  • Events in 2018 detected only with ASSED
  • Concept drift adaptivity improves event detection
  • In each case, LITMUS-ASSED detects additional events not detected by LITMUS-static
Event detection improvement

- Comparing additional events detections by LITMUS-ASSED only
- Over time, increasing numbers (and fraction) of events are detected by LITMUS-ASSED
- LITMUS-static fails to recognize increasing numbers of true events
- LITMUS-static is more susceptible to the noise
Results – Global LITMUS Coverage

Detected by both LITMUS-ASSED and LITMUS-static

Detected only by LITMUS-ASSED
## HDI-Based Improvement

<table>
<thead>
<tr>
<th>Data Window</th>
<th>Pct of Labeled Data</th>
<th>Improvement</th>
<th>Additional Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jul-2018</td>
<td>2.62%</td>
<td>125.5%</td>
<td>183%</td>
</tr>
<tr>
<td>Aug-2018</td>
<td>0.74%</td>
<td>159.2%</td>
<td>206%</td>
</tr>
<tr>
<td>Sept-2018</td>
<td>3.97%</td>
<td>156.7%</td>
<td>241%</td>
</tr>
<tr>
<td>Oct-2018</td>
<td>1.57%</td>
<td>126.1%</td>
<td>229%</td>
</tr>
<tr>
<td>Nov-2018</td>
<td>12.49%</td>
<td>225.7%</td>
<td>252%</td>
</tr>
<tr>
<td>Dec-2018</td>
<td>4.58%</td>
<td>132.0%</td>
<td>348%</td>
</tr>
</tbody>
</table>

### Data Window Pct of Labeled Data Improvement Additional Events

<table>
<thead>
<tr>
<th>July</th>
<th>LITMUS &amp; L-ASSED</th>
<th>Corrected</th>
<th>Additional</th>
<th>Both applications</th>
<th>Corrected Events</th>
<th>L-ASSED Increase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jul-2018</td>
<td>480</td>
<td>44</td>
<td>398</td>
<td>54.7%</td>
<td>5.0%</td>
<td>82.9%</td>
</tr>
<tr>
<td>Aug-2018</td>
<td>644</td>
<td>75</td>
<td>681</td>
<td>48.6%</td>
<td>5.7%</td>
<td>105.7%</td>
</tr>
<tr>
<td>Sept-2018</td>
<td>365</td>
<td>20</td>
<td>513</td>
<td>41.6%</td>
<td>2.3%</td>
<td>140.6%</td>
</tr>
</tbody>
</table>

Distributed and Event-Based Systems, 2019
HDI-Based Improvement

- LITMUS-ASSED leverages HDI to significantly improve event detection
- With a fraction of labeled data, LITMUS-ASSED provides classification improvements of > 150% in drifting conditions
  - Compared to typical, static event detection approaches
- LITMUS-ASSED’s drift adaptivity is also oracle-independent
  - No human labeler expense
  - No human labeling latency
- Classification improvement leads to detection improvements over time

<table>
<thead>
<tr>
<th>Data Window</th>
<th>Pct of Labeled Data</th>
<th>Improvement</th>
<th>Additional Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jul-2018</td>
<td>2.62%</td>
<td>125.5%</td>
<td>183%</td>
</tr>
<tr>
<td>Aug-2018</td>
<td>0.74%</td>
<td>159.2%</td>
<td>206%</td>
</tr>
<tr>
<td>Sept-2018</td>
<td>3.97%</td>
<td>156.7%</td>
<td>241%</td>
</tr>
<tr>
<td>Oct-2018</td>
<td>1.57%</td>
<td>126.1%</td>
<td>229%</td>
</tr>
<tr>
<td>Nov-2018</td>
<td>12.49%</td>
<td>225.7%</td>
<td>252%</td>
</tr>
<tr>
<td>Dec-2018</td>
<td>4.58%</td>
<td>132.0%</td>
<td>348%</td>
</tr>
</tbody>
</table>
Conclusions

• Physical event detection from Social Streams
  • Social Streams are ubiquitous
  • Can operate as a variety of sensors simultaneously
  • Existing dense global coverage and increasing
  • Used for large-scale event detection (earthquakes)

• We develop an approach for general purpose event detection

• Our approach avoids limiting assumptions
  • Handles weak-signals and noisy events
  • Handles changing event characteristics (concept drift)
  • Handles changing decision boundaries and rules (concept drift)
Conclusions

• Our approach does not rely on human labelers
  • Human/oracle labelers are expensive and time consuming
  • We exploit reputable sources to automatically assign labels

• Auto-labeling improves model creation throughput
  • Once auto-label is available, models are immediately tested and updated as and when needed
  • Do not require oracle labelers

• Drift adaptation
  • Deal with real-time, live data
  • Avoid closed data assumptions – not realistic
Acknowledgements: This research has been partially funded by National Science Foundation by CISE’s SAVI/RCN (1402266, 1550379), CNS (1421561), CRISP (1541074), SaTC (1564097) programs, an REU supplement (1545173), and gifts, grants, or contracts from Fujitsu, HP, Intel, and Georgia Tech Foundation through the John P. Imlay, Jr. Chair endowment.
## Raw data - Improvement

<table>
<thead>
<tr>
<th>Window</th>
<th>Performance</th>
<th>Statistics</th>
<th>HDI-Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Static</td>
<td>Augmented</td>
<td>Unlabeled</td>
</tr>
<tr>
<td>2014-Data</td>
<td>0.911</td>
<td>0.9668</td>
<td>NA</td>
</tr>
<tr>
<td>Jul-2018</td>
<td>0.703</td>
<td>0.882</td>
<td>7205</td>
</tr>
<tr>
<td>Aug-2018</td>
<td>0.566</td>
<td>0.901</td>
<td>14245</td>
</tr>
<tr>
<td>Sept-2018</td>
<td>0.5769</td>
<td>0.904</td>
<td>4867</td>
</tr>
<tr>
<td>Oct-2018</td>
<td>0.7</td>
<td>0.8827</td>
<td>15847</td>
</tr>
<tr>
<td>Nov-2018</td>
<td>0.3825</td>
<td>0.8634</td>
<td>7084</td>
</tr>
<tr>
<td>Dec-2018</td>
<td>0.7493</td>
<td>0.9888</td>
<td>4873</td>
</tr>
</tbody>
</table>