Conjoint Modeling of Temporal Dependencies in Event Streams

Ankur Parikh
Asela Gunawardana
Chris Meek
APPLICATION 1:
MAKING ADVERTISING MORE RELEVANT
Display Advertising

Ads are targeted to page content
Display Advertising

Ads are targeted to page content
Behavioral Advertising

User acts, expressing information need...
Behavioral Advertising

User acts, expressing information need...

Depression Symptoms, Warning Signs, Types, and Complications
www.webmd.com/depression/guide/depression-symptoms-and-types
Learn more about the symptoms of depression, depression types, major depression
Behavioral Advertising

...Future ads are targeted to need
Behavioral Advertising

...Future ads are targeted to need
Behavioral Advertising

User actions (e.g. queries issued, pages viewed) and ads are categorized into a hierarchy.

E.g.:

Health & Wellness
   Men’s Health
   Mental Health
   Nutrition

Internet
   Hosting
   Web Design
Behavioral Advertising

User actions (e.g. queries issued, pages viewed) and ads are categorized into a hierarchy.

E.g.:

Health & Wellness
  Men’s Health
  Mental Health
  Nutrition

Internet
  Hosting
  Web Design

Users that engage in actions belonging to a category are later shown ads belonging to that category.
Improving Behavioral Advertising

Goal: Show users information they need, when they need it.
Improving Behavioral Advertising

Goal: Show users information they need, when they need it.

Need better modeling of the dynamics of user’s needs
Improving Behavioral Advertising

Goal: Show users information they need, when they need it.

Need better modeling of the dynamics of user’s needs

  Forecast needs before they are expressed

    Optimal time to provide information may be before expression of need
Improving Behavioral Advertising

Goal: Show users information they need, when they need it.

Need better modeling of the dynamics of user’s needs

Forecast needs before they are expressed
  Optimal time to provide information may be before expression of need

Forecast the end of existing needs
  Don’t advertise after need has passed
Intervene Before Need Is Expressed

Suicide Methods How to Commit Suicide - Alternative Health
www.newhopehealthclinic.com/painless_methods_of_how_to_commit...
Suicide methods how to commit suicide and easy suicide is painless... ways to commit suicide, painless suicide methods painless ways to commit suicide or ...

Suicide methods - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/Suicide_methods
Bleeding · Drowning · Suffocation · Hypothermia · Electrocution
People who commit suicide in this manner usually stay at or around the place for
Stop Ads After No Longer Relevant

car stereos

1,020,000,000 RESULTS

car stereos
crutchfield.com is rated ★★★★★ on PriceGrabber (11329 reviews)

Insane Car Audio Discount
Stop Ads After No Longer Relevant

169,000,000 RESULTS

**UPS: Tracking Information**
www.ups.com/tracking/tracking.html

**UPS Tracking** options for deliveries, shipments, freight, and cargo

- Tracking
- Track Shipments
- Call Customer Service
- Quantum View
Stop Ads After No Longer Relevant

Kenwood Car Stereo Wiring Instructions | eHow.com
www.ehow.com/how_7214289_kenwood-car-stereo-wiring-instruc

Kenwood Car Stereo Wiring Instructions. Although it is true that each model of car stereo that Kenwood has ... How to Install a Kenwood Car Audio. Kenwood ...
Modeling Problem

Model times and categories of user’s future actions given times and categories of user’s past actions

What will the user be interested in when?
APPLICATION 2:
UNDERSTANDING AND FORECASTING FAILURES IN DATA CENTERS
Datacenter

Machine 1

Machine 2

Machine 3
Datacenter

Controller

DiskWrite: Fail

Machine 1
XYZ Sales App

Machine 2
XYZ Email DB
XYZ Sales DB

Machine 3
ABC Email DB
IJK Email DB

...
Datacenter

Controller

Machine 1
XYZ Sales App
XYZ Sales DB

Machine 2

Machine 3
ABC Email DB
IJK Email DB

Insert: Fail
Datacenter

Controller

Machine 1
XYZ Sales App

Machine 2
XYZ Email DB
XYZ Sales DB

Machine 3
ABC Email DB
IJK Email DB

ReadMail: Success
Datacenter

Controller

DiskWrite: Fail

Machine 1
XYZ Sales App

Machine 2
XYZ Email DB
XYZ Sales DB

Machine 3
ABC Email DB
IJK Email DB

...
Datacenter

Controller

Machine 1
- XYZ Sales App

Machine 2
- XYZ Email DB
- XYZ Sales DB

Machine 3
- ABC Email DB
- IJK Email DB

Reboot...
Datacenter

Controller

Machine 1
- XYZ Sales App

Machine 2
- XYZ Email DB
- XYZ Sales DB

Machine 3
- ABC Email DB
- IJK Email DB

ReadMail: Success
DiskWrite: Fail

Machine 1
XYZ Sales App

Machine 2
XYZ Email DB
XYZ Sales DB

Machine 3
ABC Email DB
IJK Email DB
Datacenter Management System

Controller chooses actions in response to events according to a policy

Policy hand-authored by datacenter managers

Difficult: Complex interactions between hardware, OS, services, and policies.
Datacenter Management System

Controller chooses actions in response to events according to a *policy*

Policy hand-authored by datacenter managers

Difficult: Complex interactions between hardware, OS, services, and policies.

Would like to:

Diagnose failures: what past events predict failures?
Understanding policies: where can the policy be improved?
Eventually automate policy selection
Modeling Problem

Model dependencies of events on types and times of past events:

Why did failures happen?

In what situations are particular actions taken?
MODELING EVENT STREAMS AS MARKED POINT PROCESSES
Events Streams as Marked Point Processes

Wish to model dynamics of random processes rather than dependencies between random variables.
Events Streams as Marked Point Processes

Wish to model dynamics of random processes rather than dependencies between random variables.

Model stream of actions from a user or stream of events from a machine as a marked point process.
Events Streams as Marked Point Processes

Wish to model dynamics of random processes rather than dependencies between random variables.

Model stream of actions from a user or stream of events from a machine as a marked point process.
Events Streams as Marked Point Processes

Wish to model dynamics of random processes rather than dependencies between random variables.

Model stream of actions from a user or stream of events from a machine as a marked point process.
Modeling Marked Point Processes

Forward in time likelihood:

\[ p(t_1, l_1, ..., t_n, l_n) = \prod_{i=1}^{n} p(t_i, l_i | h_i) \]

History \( h_i = (t_1, l_1), ..., (t_{i-1}, l_{i-1}) \)
Modeling Marked Point Processes

Forward in time likelihood:

\[ p(t_1, l_1, \ldots, t_n, l_n) = \prod_{i=1}^{n} p(t_i, l_i|h_i) \]

History \( h_i = (t_1, l_1), \ldots, (t_{i-1}, l_{i-1}) \)

Process completely characterized by \textit{conditional intensity functions} \( \lambda_i(t|h) \) for the labels \( l \) \citep{Daley2003}.

\[ \lambda_i(t|h) = \text{expected rate of events labeled } l \text{ at time } t \text{ given history } h \]
Modeling Marked Point Processes

Forward in time likelihood:

\[ p(t_1, l_1, ..., t_n, l_n) = \prod_{i=1}^{n} p(t_i, l_i|h_i) \]

History \( h_i = (t_1, l_1), ..., (t_{i-1}, l_{i-1}) \)

Process completely characterized by \textit{conditional intensity functions} \( \lambda_l(t|h) \) for the labels \( l \) [Daley & Vere-Jones 2003].

\( \lambda_l(t|h) = \) expected rate of events labeled \( l \) at time \( t \) given history \( h \)

Conditional intensity representation:

\[ p(t_i, l_i|h_i) = \lambda_{l_i}(t_i|h_i) \prod_l e^{-\int_{-\infty}^{t_i} \lambda_l(\tau|h_i) d\tau} \]
PCIM

Piecewise-constant Conditional Intensity Model (PCIM) [Gunawardana et al 2011]:

Assumes $\lambda_l(t|h)$ is piecewise constant in $t$
Represents $\lambda_l(t|h)$ for each $l$ by a decision tree
Tree for $\lambda_{\text{RebootFail}}(t|h)$

- $\lambda_{\text{RebootFail}} = 0$
- $\lambda_{\text{RebootFail}} = 0.1$
- $\lambda_{\text{RebootFail}} = 2.5$

- $\text{RebootInit} \in [t - 0:30, t)$
- $\text{RebootFail} \in [t - 0:30, t)$
- $\text{VersionCheckFail} \in [t - 1:00, t - 0:30)$

$\lambda_{\text{RebootFail}}$ values:
- 0
- 0.1
- 2.5
PCIM

Piecewise-constant Conditional Intensity Model (PCIM) [Gunawardana et al 2011]:

Assumes $\lambda_l(t|h)$ is piecewise constant in $t$
Represents $\lambda_l(t|h)$ for each $l$ by a decision tree
PCIM

Piecewise-constant Conditional Intensity Model (PCIM) [Gunawardana et al 2011]:

Assumes $\lambda_l(t|h)$ is piecewise constant in $t$

Represents $\lambda_l(t|h)$ for each $l$ by a decision tree

Parameters have conjugate prior given tree

Learn trees greedily using structural prior and Bayesian model selection

Forecast using importance sampling
Tree for $\lambda_{\text{RebootFail}}(t|h)$
Graphical Representations of PCIMs

Tree for “RebootFail”

depends on past “RebootInit,” “RebootFail,” and “VersionCheckFail”

but not on “VersionCheckSucc” and “RebootSucc.”

(Temporal) conditional independence!!

of intensities, not probabilities.

Temporal conditional independencies can be represented as a directed graph [Didelez2008].

Graph may be cyclic.
Graphical Representations of PCIMs

Tree for “RebootFail”
  depends on past “RebootInit,” “RebootFail,” and “VersionCheckFail”
  but not on “VersionCheckSucc” and “RebootSucc.”

(Temporal) conditional independence!!!
  of intensities, not probabilities.
Graphical Representations of PCIMs

Tree for “RebootFail”

- depends on past “RebootInit,” “RebootFail,” and “VersionCheckFail”
- but not on “VersionCheckSucc” and “RebootSucc.”

(Temporal) conditional independence!!!
- of intensities, not probabilities.

Temporal conditional independencies can be represented as a directed graph [Didelez2008].
- Graph may be cyclic.
PCIMs vs. CTBNs

PCIMs are marked point processes
  Events have piecewise-constant conditional intensities given parents.
  Estimation is simple.
  Forecasting is complex.

CTBNs are Markov marked point processes
  Events (state transitions) have constant conditional intensities given parents: \( \lambda_{x}^{i \to j}(t) = q_{i \to j}(x) \)
  Modeling non-Markov processes requires latent states.
  Estimation is more complex.
  Forecasting simplifies.
PCIMs vs. CTBNs

PCIMs are marked point processes
  Events have piecewise-constant conditional intensities given parents.
  Estimation is simple.
  Forecasting is complex.

CTBNs are Markov marked point processes
PCIMs vs. CTBNs

PCIMs are marked point processes
- Events have piecewise-constant conditional intensities given parents.
- Estimation is simple.
- Forecasting is complex.

CTBNs are Markov marked point processes
- Events (state transitions) have constant conditional intensities given parents: $\lambda_{x:i\to j}(t|h) = q^x_{i\to j}(\text{par}_x(h))$
PCIMs vs. CTBNs

PCIMs are marked point processes
  Events have piecewise-constant conditional intensities given parents.
  Estimation is simple.
  Forecasting is complex.

CTBNs are Markov marked point processes
  Events (state transitions) have constant conditional intensities given parents: $\lambda_{x:i\rightarrow j}(t|h) = q_{i\rightarrow j}^x(\text{par}_x(h))$
  Modeling non-Markov processes requires latent states.
PCIMs vs. CTBNs

PCIMs are marked point processes
Events have piecewise-constant conditional intensities given parents.
Estimation is simple.
Forecasting is complex.

CTBNs are Markov marked point processes
Events (state transitions) have constant conditional intensities given parents: \( \lambda_{x:i\rightarrow j}(t|h) = q_{i\rightarrow j}^x(par_x(h)) \)
Modeling non-Markov processes requires latent states.
Estimation is more complex.
Forecasting simplifies.
Large Label Sets

When set of possible $l$ is large:

- Need separate tree for each $l$.
- Hard to learn good $\lambda_l(t|h)$ for rare $l$. 

- Idea: Make $\lambda_l(t|h)$ be similar for similar $l$, different for different $l$. 

- Use structured labels with attributes for learning similarity: 
  - RebootFail $\rightarrow$ Reboot:Fail
  - MentalHealth $\rightarrow$ Health&Wellness
Large Label Sets

When set of possible $l$ is large:
   Need separate tree for each $l$.
   Hard to learn good $\lambda_l(t|h)$ for rare $l$.

Idea: Make $\lambda_l(t|h)$ be similar for similar $l$, different for different $l$
Large Label Sets

When set of possible $l$ is large:
   Need separate tree for each $l$.
   Hard to learn good $\lambda_l(t|h)$ for rare $l$.

Idea: Make $\lambda_l(t|h)$ be similar for similar $l$, different for different $l$.

Use structured labels with attributes for learning similarity:
   RebootFail $\rightarrow$ Reboot:Fail
   MentalHealth $\rightarrow$ Health&Wellness:MentalHealth
Large Label Sets

When set of possible $l$ is large:
   Need separate tree for each $l$.
   Hard to learn good $\lambda_l(t|h)$ for rare $l$.

Idea: Make $\lambda_l(t|h)$ be similar for similar $l$, different for different $l$

Use structured labels with attributes for learning similarity:
    RebootFail → Reboot:Fail
    MentalHealth → Health&Wellness:MentalHealth

Attrib 1
Large Label Sets

When set of possible $l$ is large:
   Need separate tree for each $l$.
   Hard to learn good $\lambda_l(t|h)$ for rare $l$.

Idea: Make $\lambda_l(t|h)$ be similar for similar $l$, different for different $l$

Use structured labels with attributes for learning similarity:

RebootFail $\rightarrow$ Reboot:Fail
MentalHealth $\rightarrow$ Health&Wellness:MentalHealth

Attrib 2
Conjoint PCIMs

Represent $\lambda_l(t|h)$ for all $l$ by a single decision tree
Queries label attributes to distinguish dissimilar $l$
\[ \lambda_l = 2.0 \]

\[ \lambda_l = 0.01 \]
\[ l = \lambda_l \]

\[ l = \text{Reboot:*} \]

\[ l = \text{*:Fail} \in [t - 0:30, t) \]

\[ \lambda_l = 2.0 \]

\[ \lambda_l = 0.01 \]

Depends on \( h \) & \( t \), not \( l \)
\[ \lambda_l = 2.0 \]

\[ \lambda_l = 0.01 \]

* : Fail
\[ \in [t - 0:30, t) \]

Depends on \( l \) not \( h \) & \( t \)

\[ l \]
\[ \in [t - 0:30, t) \]

\[ l = \text{Reboot:*} \]

\[ l = \text{*:Fail} \]
\( \lambda_l = 2.0 \)  
\( \lambda_l = 0.01 \)

*Depends on \( l, h & t \).*

- \( *:\text{Fail} \in [t - 0:30, t) \)
- \( l \in [t - 0:30, t) \)
\( \lambda_l = 2.0 \)

\( \lambda_l = 0.01 \)
Model $\lambda_{\text{Rare:Fail}}(t|h)$ even if Rare:Fail unseen in training

$\lambda_l = 2.0$

$\lambda_l = 0.01$
Conjoint PCIMs

Represent $\lambda_l(t|h)$ for all $l$ by a single decision tree
Queries label attributes to distinguish dissimilar $l$

Like PCIM

Has conjugate prior given tree structure
Learn tree using structural prior and Bayesian model selection
Forecast using importance sampling
EXPERIMENTAL RESULTS
Application: Predict whether the user will issue queries in the category “Health & Wellness” in the next week.
Cross Label Dependencies Matter

Application: Predict whether a user **who has never done so** will issue queries in the category “Health & Wellness” in the next week.
C-PCIMs Generalizes From Less Data

Application: Model when users will issue web queries in what categories
C-PCIMs Can Forecast Rare Events

Application:
Predict whether the user will issue queries in the category “Mental Health” in the next day.
(C)PCIMs Allow Diagnosis

\[ \lambda_{\text{SendTech}} = 0 \]

\[ \lambda_{\text{SendTech}} = 0.15 \]

\[ \lambda_{\text{SendTech}} = 0 \]

\[ \lambda_{\text{SendTech}} = 0.75 \]
C-PCIMs Allow Richer Representations

Application: Model which machines will issue which messages when.
Thank You
Modeling User Query Behavior for Advertising

Data: Logs of user query behavior over time
- Train: ~100k queries from ~6k users over 2 months
- Test: ~150k queries from ~6k users over next month
- Test users selected to have at least 2 weeks of data each
- No user overlap in train & test.

Queries mapped to 476 hierarchical categories
- 476 labels
- Attributes: nodes in hierarchy

Task:
- Predict whether a user will issue queries in a target category in a target time period
  (to decide whether to show user corresponding ads in that period)
Modeling Datacenter Machine Behavior

Data: Event logs from a datacenter
  Train \sim 130k events from first two weeks of month
  Test: \sim 170k events from rest of month

221 possible messages \times 71 machines \approx 15k labels
Each machine belongs to 1 of 4 types
Attributes: message, machine, machine type

Task:
  Model dependencies
  Diagnose failures