Model-Driven Processing in Sensor Networks

Jun Yang
CPS 296.1, Spring 2007
Sensor Data Processing
With contents from C. Guestrin

Model-driven processing

- Amol Deshpande, Carlos Guestrin, Samuel Madden, Joseph M. Hellerstein, and Wei Hong. "Model-Driven Data Acquisition in Sensor Networks." *International Conference on Very Large Data Bases*, 2003

Analogy: sensor network as a DB

- TinyDB
- SQL-style query
- Declarative interface:
  - Sensor nets are not just for PhDs
  - Decrease deployment time

Limitations?

- Data representation/interpretation
  - Plain SQL on raw data may give misleading results
    - Sensors readings do not represent “truth”
  - Plain SQL is awkward
    - No convenient way to handle missing data
- Efficiency
  - Every node must wake up at every time step
    - For `SELECT *` (collect all), every node transmits to the root
  - Doesn’t take advantage of data correlation in a principled way

Correlation in sensor data

- Natural consequence of continuous physical phenomena + dense network
- Correlation across time
  - History of readings → info about future readings
- Correlation across space
  - One sensor’s readings → info about others’ readings
- Correlation across modalities
  - One attribute (e.g., light) value → info about another attribute (e.g., temperature) value

Announcements (Feb. 6)

- The first round of presentations finalized
  - Watch your email inbox today
- Readings for Thursday: more uses of models—Ken and snapshot queries (no reviews)
- Reading for next Tuesday (review): directed diffusion
- Course project milestone 1: March 1
Model-driven data acquisition

Advantages
- Use of prior knowledge about correlations
- Observe fewer/cheaper attributes
  - Avoid not only transmission but also acquisition
  - What's the caveat?
- Solution to missing data
- Incorporation of new observations in knowledge
- Reuse of information among queries and over time

Working with probabilistic models
- Learn joint distribution $p(X_1, \ldots, X_n)$ from historical data
- Example query: know $X_2$ within $\pm \varepsilon$ with prob. at least $1-\delta$
  - Marginalize: $p(x_2) = \int p(x_1, x_2) \, dx_1$
  - Compute mean: $\mu_2 = \int_2 p(x_2) \, dx_2$
  - Compute confidence: $P(X_2 \in [\mu_2-\varepsilon, \mu_2+\varepsilon]) = \int_{\mu_2-\varepsilon}^{\mu_2+\varepsilon} p(x_2) \, dx_2$
    - If it's at least $1-\delta$, return $\mu_2$
    - What if it's not?

Working with probabilistic models (cont’d)
- Example query cont’d
  - Acquire the value of $X_1$, and exploit correlation to better estimate/bound $X_2$
  - Posterior distribution:
    - $p(x_2 | X_1 = 18) = p(18, x_2) / p(X_1 = 18)$
    - Compute new mean and confidence based on this distribution
  - If new confidence is good enough, return new mean
  - If not, acquire more attributes and condition further on these observations (in the worst case, acquire $X_2$ itself)

Dynamic models
- Assume Markovian transition model: $p(X_t | x^{t-1})$, learned from historical data
  - Joint distribution at time $t-1$: $p(x_t^{-1} | o_1^{t-1}, \ldots, t-1)$
  - Apply transitional model: $p(x_t | o_1^{t-1}, \ldots, t-1) = \int p(x_t | x^{t-1}) p(x^{t-1} | o_1^{1-t-1}) \, dx^{t-1}$
    - Typically adds more uncertainty
    - Make new observations $o_t$ and further condition $p(x_t | o_1^{1-t})$ on $o_t$ to get $p(x_t | o_1^{1-t})$
      - Typically reduces uncertainty
    - Repeat
Supported queries

- Value query: value of $X_i \pm \epsilon$ with prob. at least $1-\delta$
- Range query: value of $X_i \in [a, b]$ with prob. at least $1-\delta$ or no more than $\delta$
  - Compute $\int p(x) \, dx$
- Aggregation query: average of all $n$ readings within $\pm \epsilon$ with prob. at least $1-\delta$
  - $p(Y = y) = \int p(x_1, \ldots, x_n) \mathbf{1}[\sum x_i/n = y] \, dx_1 \ldots dx_n$
  - Requires solutions to integrals
    - In general requires numerical integration or sampling
    - For “nice” distributions (e.g., Gaussian), sometimes can compute in closed-form

Query optimization

- Which readings shall we acquire?
- How do we collect them?
- Utility?
  - Query-driven, model-based: How much does it help us resolve remaining uncertainty?
- Cost?
  - Acquisition
  - Transmission

Choosing a plan

- Example query: $X_i \in [a, b]$ with prob. $\geq 1 - \delta$
  - Benefit of observing $\mathcal{O} = \{0, \ldots, n\}$
    - $R(\mathcal{O}) = \max\{P(X_i \in [a, b] | o), 1-P(X_i \notin [a, b] | o)\}$
  - But since we don’t know $o$, we settle for expected benefit: $R(\mathcal{O}) = \int p(o) R(o) \, do$
- Optimization problem
  - Minimize $\sum_{o \in \{0, \ldots, n\}} Cost(\mathcal{O})$ such that $R(\mathcal{O}) \geq 1 - \delta$
  - Exhaustive search
  - Greedy heuristic: next to acquire is the reading with the highest benefit/cost ratio
    - Note that benefit changes as more readings are acquired

Network and query plan

- Assume quasi-static network topology
- Plan collects a subset $\mathcal{O}$ of sensor readings
  - Using a path that starts and ends at the root and visits all nodes in $\mathcal{O}$
    - Why not a tree?

Experimental results

- Redwood tree and Intel lab
- Learned models from data
  - Learned a different transition model for each hour of the day (domain knowledge)

Cost vs. confidence
Approximate range queries

- Confidence set at 95%

![Graph showing approximate range queries]

Comparison with competitors

- Where did approximate caching lose big?

![Comparison with competitors graph]

Discussion

- Finally, a reality check for the DB approach to sensors!
- How much do you trust your model?
  - What if it isn’t Gaussian after all?
  - Since we have assumed Gaussian in deciding what not to acquire, would the decision reinforce our (false) assumption?
  - What if the goal is to learn the model instead?
    - How would this change the utility of observations?
- How dynamic/adaptive is this approach?
- How do you measure utility in more complex situations?
  - MIN/MAX Multiple queries?
- Outliers—can we really avoid acquisition?
- Expensive to optimize at the root for every epoch
- Is further compression worthwhile on the tour?