Deep Analytics Pipeline: Components

Large E2E analytics pipelines have 2 types of components

Classical SQL operations
Filter, Project, Join, Aggregate etc. that have deterministic outputs
Used in data preprocessing, feature construction etc.

Machine Learning (ML) operations
May output scores/probabilities (e.g. P(user to buy | history) = 0.13)
Some of which use only the SQL operators (e.g. Naïve Bayes)
Others require new patterns
  Iterate on the same data multiple times till some convergence criterion of an optimization function (e.g. SGD)
  Communicate summaries to other nodes (e.g. aggregation tree) in every iteration
  Fault awareness/tolerance
Training a ML model may be able to trade off some amount of faults (data/node unavailability) with time/budget/quality
Scoring is applying the trained ML model
  Expressible using SQL operators for many common ML algorithms
User Interest Profiles

Figure 6 in *Scalable Distributed Inference of Dynamic User Interests for Behavioral Targeting*, [Amr Ahmed](#) et al. Proceedings of KDD.
Entities in internet eco-system

Content
(pages, blogs etc.)

Search Engine

Searches

User

Browses

Content
(pages, blogs etc.)

User

Interacts

Content/Display Advertising

Queries

Content
(pages, blogs etc.)

Queries

Content
(pages, blogs etc.)

Ads
(Text, Display etc.)

Content/Display Advertising

Search Advertising

User

Interacts

Content
(pages, blogs etc.)

Ad

Searches

User

User

User

User
Data characteristics

Different data types
  - Boolean, categorical, ordinal, numeric, text, graph

Dense, or Sparse

Long tail distribution (e.g. Zipfian) is default in almost every dimension
  - e.g. # of queries per user

From: Figure 1c in Anatomy of the long Tail, Goel et al., WSDM 2010
User Interest Modeling Pipeline: Data

Volume

Typical portals have a few MM – few 100 MM users per day
100 MM --10’s of billion activities (content views/clicks, search queries/clicks, ad views/clicks etc.)

Variety

- Web server logs
- Search query logs
- Ad server logs
- Click (content and ads) logs
- Social network logs (IM, Mail graph etc.)

Dimensions

- User data (demographic, geographic)
- Content metadata (title, content, category etc.)
- Ads metadata (text, graphics, category etc.)

Velocity

- Logs rolled up every few minutes and processed in latencies of few minutes, hours, days, week
- Model training in batch mode (few hours)
- Model scoring (applying the models) using both
  - Batch logs data (every hour, day, week)
  - Real time streaming events (a few seconds from user event to profile update)
User Interest Modeling Pipeline

Modeling pipeline to model user interests

Typical objectives
Content clicks
Response to an ad (clicks/conversions, message retention etc.)

Main components to train, score and evaluate models
Data Generation
  - Data Acquisition
  - Feature and Target Generation
  - Feature Target join
Model Training
Offline Scoring and Evaluation
Batch scoring and upload to online serving
User Interest Modeling: Offline Training and Evaluation

1. Data Acquisition
   1a. Data Acquisition
   1b. Target Generation
   1c. Feature Target Join

2. Model Training
   2. Model Training

3. Model Scoring
   3. Model Scoring

4. Evaluation
   4. Evaluation

Evaluation Phase

User event history

ML Model
Feature and Target Windows in Training

Query

Visit page on Finance

Time

T₀

Interest event

Moving Window

Feature Window

Target Window

9
Batch Scoring Components

1a. Data Acquisition

User event history

1b. Feature generation

Features

3. Model Scoring

Scores

ML Model

Online Scoring & Serving
Overview of Batch User Interest Modeling Pipeline

**Feature/Target Generation**
1. Join event x dimension
2. Group by User
3. Filter events
4. Aggregate

**ML Modeling Engine**
1. Join Features/Targets
2. Sample target/non-targets
3. Train ML model

**Scoring and Evaluation**
1. Join features by model weights to get scores
2. Group by scores and targets
3. Evaluation

- **Dimensions**
- **User events**
- **Targets**
- **Features**
- **Models**
- **Scores, Reports**

HDFS

Online Scoring and Serving
Benchmark proposal

Data
- Simulator to generate user events data sampled from appropriate distributions for each event type

Problems
- Scoring

Create features
- Tag/categorize each event
- RFM (Recency, Frequency, Monetary value) features for each category by event type

Produce a score for each category,
- Dot product (join) of weights with feature in each category and aggregate

Training

For each category,
- Create Targets (e.g. ad-click events in category) at time window from “t” to “t+delta_t” for each category
- Create Features at some time “t” for each category
- Join Targets with Features
- Train a linear machine learning model for each category with these features (e.g. liblinear SVM, or Vowpal Wabbit)
- Score on target/feature events from a different time window, for a different set of users
- Evaluate by comparing the score with the ad click/no-click events in the future, using standard ML metrics
  - Area under the curve
Benchmark proposal: Data simulator

For each user
  Pick # of sessions
For each session
  Pick # of events
For each event
  • Pick timestamp
  • Pick category of event
  • Pick event type from possible event types (page view, search, search result click, search ad view, search ad click, display ad view, display ad click)
  • Generate event attributes (timestamp, ID etc)
  • With some probability, generate event attributes based on the user profile at that time
benchmark proposal: Data simulator

Each event attribute within a session depends either

1. On the value of the attribute in the previous event
   - For example, if the query is in category C, the subsequent query is likely to be in C or a related category a well
2. There is a finite chance a new category event is started

Pick attributes from an appropriate distribution, e.g.

- Draw counts, time differences from a Poisson with a given mean
- Draw event categories are drawn from a multinomial distribution (optionally biased with the user profile)
- Add long tail skew over and top of these distributions – present in all dimensions in real data
## Data Acquisition

<table>
<thead>
<tr>
<th>User</th>
<th>Time</th>
<th>Event</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>U₁</td>
<td>T₀</td>
<td>visited autos.msn.com</td>
<td>Web server logs</td>
</tr>
<tr>
<td>U₁</td>
<td>T₁</td>
<td>searched for “car insurance”</td>
<td>Search logs</td>
</tr>
<tr>
<td>U₁</td>
<td>T₂</td>
<td>browsed stock quotes</td>
<td>Web server logs</td>
</tr>
<tr>
<td>U₁</td>
<td>T₃</td>
<td>saw an ad for “discount brokerage”, but did not click</td>
<td>Ad logs</td>
</tr>
<tr>
<td>U₁</td>
<td>T₄</td>
<td>checked Mail</td>
<td>Web server logs</td>
</tr>
<tr>
<td>U₁</td>
<td>T₅</td>
<td>clicked on an ad for “auto insurance”</td>
<td>Ad logs, click server logs</td>
</tr>
</tbody>
</table>
## Benchmark proposal: Volume

Classes as outlined by Milind Bhandarkar in WBDB-4

<table>
<thead>
<tr>
<th>Class</th>
<th># Unique Users</th>
<th># Unique Pages</th>
<th># Unique Queries</th>
<th># Unique SearchAds</th>
<th># Unique DisplayAds</th>
<th>Social Graph Edges</th>
<th>Average # of Events per user in time window</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tiny</td>
<td>100k</td>
<td>100</td>
<td>1k</td>
<td>3k</td>
<td>300</td>
<td>300k</td>
<td>10</td>
</tr>
<tr>
<td>Small</td>
<td>1M</td>
<td>1k</td>
<td>10k</td>
<td>10k</td>
<td>1k</td>
<td>5M</td>
<td>30</td>
</tr>
<tr>
<td>Medium</td>
<td>10M</td>
<td>5k</td>
<td>500k</td>
<td>500k</td>
<td>5k</td>
<td>100M</td>
<td>100</td>
</tr>
<tr>
<td>Large</td>
<td>100M</td>
<td>20k</td>
<td>2M</td>
<td>2.5M</td>
<td>10k</td>
<td>1B</td>
<td>1000</td>
</tr>
<tr>
<td>Huge</td>
<td>1B</td>
<td>50k</td>
<td>10M</td>
<td>10M</td>
<td>50k</td>
<td>100B</td>
<td>1000</td>
</tr>
</tbody>
</table>
**Benchmark proposal: Variety**

Fact (user event) data generators for the following events using appropriate distributions:

1. Browsing page views
   - userID, ts, geo-location, pageID, list of <page_contentID,position>
2. Search queries (see links at: [http://jeffhuang.com/search_query_logs.html](http://jeffhuang.com/search_query_logs.html))
   - userID, ts, geo-location, queryID, ranked list of <pageIDs, list of <page_contentID>>
3. Ads – search and display ads – associated with queries and page views
   - userID, ts, geo-location, pageID, list of <ad_ID,position>
4. Clicks – sprinkle on top of these – with delays from views
   - userID, ts, <ad_ID,position>
5. Social graph data – start with Twitter user/follower graph public data
   - [http://bickson.blogspot.com/2012/03/interesting-twitter-dataset.html](http://bickson.blogspot.com/2012/03/interesting-twitter-dataset.html)

Dimension data:

1. (Multiple/Hierarchical) Tags/categories for each unique pageID, queryID, adID etc
2. Pick a distribution from say, freebase?
3. Text/Graphics of ads
4. User Demographic/geo attributes, declared interest profiles
Benchmark proposal: Problem

L Scoring
All users must be scored in every category
Must be fault tolerant

L Training
Can train on:
- Single node if the sampled down data fits in a single node
- Distributed training (can down sample or ignore some data on failures) as long as ML metrics are reported on full out of sample, out of time data.
- However, must score and evaluate on all data
Benchmark proposal: Learning curve of ML models

Generalization performance vs. Data volume (or # of iterations in some algos.)

- Simple ML model
- Complex ML model
Performance: Metrics

Polish performance

Distribution over a number of simulated datasets

Each component

End to end pipeline

Need both fast mean performance, *and* low variance in performance end to end

Cross validation estimates of ML metrics (Area under the curve, Precision/Recall) on training data

Valuation metrics on out of sample, out of time data

Up the space of model quality, performance and time/budget e.g.

- How good are the models for a fixed budget within a maximum time?
- Optimize the model quality for a fixed budget irrespective of the time?
- Optimize the model quality within a fixed time etc.?
- A cost function specifying the relative weights of quality, budget and time etc.?
Candidate Problems and Metrics

End to End pipeline comprising the following:
dimensions

quality of learner

performance

data properties
Quality

Outputs of most ML models are probabilities
- Classification: e.g. Probability of class given features
- Regression: e.g. Mean value of output given features
- Clustering: e.g. Cluster membership probability given features

How well does the model metric reflect the business objectives?

A metric optimized by machine learning algorithms
- Likelihood, Mis-classification error, cross entropy, Root mean square, Hinge loss, NDCG etc.

A metric used to evaluate the “generalization” performance of learned algorithm on unseen data
- Global metrics
  - Area under the ROC or Precision/Recall curves
- Operating point metrics
  - Accuracy, Precision, Recall, True/False positives, True/False negatives, sensitivity, specificity etc.

A metric relevant to business objectives
- Lift, Enrichment rate, ROI, Maximum return under practical constraints on human involvement etc.

A limited number of evaluations to prevent learning the evaluation data
Performance

Dimensions

Wall-clock time

Fault tolerance and recovery
• Must handle failures in all components

Sensitive to resource usage/costs

Computing, Storage and Communication

Map the space of Quality, Performance and Resource, e.g.

- How good are the models for a fixed budget within a maximum time?
- Optimize the model quality for a fixed budget irrespective of the time?
- Optimize the model quality within a fixed time etc.
- Cost function specifying the relative weights of quality, budget and time etc.
Large dimensionality of possible user activities, yet a typical user has a sparse activity vector. Possible values of events change over time.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Possible Values</th>
<th>Typical values per user</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pages</td>
<td>~ MM</td>
<td>10 – 100</td>
</tr>
<tr>
<td>Queries</td>
<td>~ 100s of MM</td>
<td>Few</td>
</tr>
<tr>
<td>Ads</td>
<td>~ 100s of thousands</td>
<td>10s</td>
</tr>
</tbody>
</table>
Data Acquisition

- Tag and Transform
  - Categorization
  - Topic
  - ....

- Project relevant event attributes

- Filter irrelevant events

- Map Operations

Event Feeds
- User event
- User event
- User event

HDFS

Normalized Events (NE)
## Data Acquisition

Output:
Single normalized feed containing all events for all users per time period

<table>
<thead>
<tr>
<th>User</th>
<th>Time</th>
<th>Event</th>
<th>Tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>U₁</td>
<td>T₀</td>
<td>Content browsing</td>
<td>Autos, Mercedes Benz</td>
</tr>
<tr>
<td>U₂</td>
<td>T₂</td>
<td>Search query</td>
<td>Category: Auto Insurance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td></td>
<td></td>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>U₂₃</td>
<td>T₂₃</td>
<td>Mail usage</td>
<td>Drop event</td>
</tr>
<tr>
<td>U₃₆</td>
<td>T₃₆</td>
<td>Ad click</td>
<td>Category: Auto Insurance</td>
</tr>
</tbody>
</table>
Feature and Target Generation

Features:
Summaries of user activities over a time window
Aggregates, Moving averages, Rates etc. over moving time windows
Support online updates to existing features

Targets:
Constructed in the offline model training phase
Typically user actions in the future time period indicating interest
- Clicks/Click-through rates on ads and content
- Site and page visits
- Conversion events
  - Purchases, Quote requests etc.
  - Sign-ups to newsletters, Registrations etc.
Feature Generation

<table>
<thead>
<tr>
<th>U₁</th>
<th>T₀</th>
<th>Content browsing</th>
<th>Autos, Mercedes Benz</th>
</tr>
</thead>
<tbody>
<tr>
<td>U₁</td>
<td>T₂</td>
<td>Search query</td>
<td>Category: Auto Insurance</td>
</tr>
<tr>
<td>U₁</td>
<td>T₃</td>
<td>Click on search result</td>
<td>Category: Insurance premiums</td>
</tr>
<tr>
<td>U₁</td>
<td>T₄</td>
<td>Ad click</td>
<td>Category: Auto Insurance</td>
</tr>
</tbody>
</table>

All events for U₁

Reduce 1

Reduce 2

Summaries over user event history

Aggregates within window
Time and event weighted averages
Event rates

Map 1

Map 2

Map 3

All events for U₂

U₁, Event 1
U₂, Event 2
U₁, Event 2
U₂, Event 3
U₂, Event 1

Aggregate Normalized events

NE 1  
NE 4
NE 7
NE 2
NE 5
NE 8
NE 3
NE 6
NE 9

HDFS  
Feature Set
Training Features and Targets

Low target rates

Typical response rates are in the range of 0.01% ~ 1%

Many users have no interest activities in the target window

First construct the targets

Compute the feature vector only for users with targets

Reduces the need for computing features for users without target actions

Owes stratified sampling of users with different target and feature attributes
Modeling Algorithms

- Regressions
  - Different flavors: Linear, Logistic, Poisson etc.
  - Constraints on weights
  - Different regularizations: $L_1$ and $L_2$
- Decision trees
  - Used for both regression and ranking problems
  - Boosted trees
- Naïve Bayes
- Support vector machines
  - Commonly used in text classification, query categorization etc.
- Online learning algorithms
Modeling Algorithms

Maximum Entropy modeling
Log-linear link function.
Classification problems in large dimensional, sparse features
Constrained Random Fields
Sequence labeling and named-entity recognition problems
Some algorithms are not easy to implement in MR framework
ain one model per node.
Each node can model one response
Scoring and Evaluation

- Dynamically weight model training phase to features from Feature generation component
- Mapper operations only
- Nino* equation editor
- Embedded compiler can compile arbitrary scoring equations.
- Can also embed any class invoked during scoring
- Can modify features on the fly before scoring
- Evaluation metrics
- Sort by scores and compute metrics in reducer
- Precision vs. Recall curve
- Lift charts

* http://docs.codehaus.org/display/JANINO/Home
<table>
<thead>
<tr>
<th>Component</th>
<th>Data Processed</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Acquisition</td>
<td>~ 1 Tb per time period</td>
<td>2 – 3 hours</td>
</tr>
<tr>
<td>Generate Features and Targets</td>
<td>~ 1 Tb * Size of feature window</td>
<td>4 - 6 hours</td>
</tr>
<tr>
<td>Model Training</td>
<td>~ 50 - 100 Gb</td>
<td>1 – 2 hours for 100’s of models</td>
</tr>
<tr>
<td>Scoring</td>
<td>~ 500 Gb</td>
<td>1 hour</td>
</tr>
</tbody>
</table>
Current challenges

- Limited size of name-node
  - File and block meta-data in HDFS is in RAM on name-node
  - On name-node with 64Gb RAM
    - ~ 100 million file blocks and 60 million files
  - Upper limit of 4000 node limit cluster
  - Adding more reducers leads to a large number of small files

- Copying data in/out of HDFS
  - Limited by external file system read/write rates

- High latency for small jobs
  - Overhead to set up may be large for small jobs
practical considerations

Reduce amount of data transfer from mapper to reducer
There is still disk write/read in going from mapper to reducer
- Mapper output = Reducer input files can become large
- Can run out of disk space for intermediate storage

Project a subset of relevant attributes in mapper to send to reducer
Use combiners
Compress intermediate data

Distribution of keys
Reducer can become a bottleneck for common keys
Use Partitioner to control distribution of map records to reducers
E.g. distribute mapper records with common keys across multiple reducers in a round robin manner
practical considerations

Dedicated partitioning of data
Multiple files helps parallelism, but hit name-node limits
Smaller number of files keeps name-node happy but at the expense of parallelism

Thus, ideal for distributed computing algorithms requiring communications (e.g. distributed decision processes)

MPI on top of the cluster for communication