AI Is Happening in the Enterprise

How PayPal Boosts Security with Artificial Intelligence
The payments giant keeps fraud losses below industry averages by teaching computers to play detective.
by Michael Morisy

John Deere Adds AI, IoT to Farm Equipment
By: Darryl K. Taft | September 06, 2016

Lowe's in AdWeek: 5 Bleeding-Edge Brands That Are Infusing Retail With Artificial Intelligence
January 03, 2017

Artificial Intelligence: JP Morgan is showing the way...
Jan 31, 2017

Wells Fargo Pushes Into Artificial Intelligence

Wells Fargo has created a team to develop artificial intelligence-
Deep Learning ... Over Large Data Sets

Model Performance by Data Size

- Traditional ML
- Small NN
- Medium NN
- Large NN
### Common Enterprise Use Cases

<table>
<thead>
<tr>
<th>RECOMMENDER SYSTEMS</th>
<th>FRAUD DETECTION</th>
<th>PREDICTIVE MAINTENANCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable more effective suggestions, based on context for individuals, based on a particular objective such as purchase or lifetime value.</td>
<td>Enables real-time detection of events in credit cards and e-banking. Enables fraud prevention, cyber-security and system optimization.</td>
<td>Improves preventative measures &amp; performance with greater accuracy at the asset &amp; component level</td>
</tr>
</tbody>
</table>

#### COMPUTER VISION

- Enables dramatically more accurate visual recognition tasks that include image classification, detection and localization.

#### TEXT AND SPEECH UNDERSTANDING

- Better service and automation for diverse applications such as call center, chat, field service, and medical records.

#### DOCUMENT AUTOMATION

- Enables automation of processes that are human-intensive with higher speed and accuracy with paper or legacy apps.

---

© 2017 Teradata

**Teradata.**
Anti-Fraud in Banking

Business challenges
- To improve fraud detection in business transactions by using deep AI.
- Low fraud detection rates and worrying fraud false positives.
- The bank needed a trusted partner with proven expertise in deep learning and building architectures.

Solution
- Built advanced analytics models and created a blueprint for real-time model scoring.
- Deployed advanced analytics through deep learning.
- Think Big Analytics became a trusted advisor and took a leading role in AI implementation and future projects.

Benefits
- Cost savings of USD millions every month.
- False positives reduced by 50% and detection rate increased by 60%.
- Transaction latency performance of just ~20 mins across 30 million transactions annually.

Tools & Technologies
Fraud Detection with Low False Positives

Deep Learning vs Machine Learning at 1% FP

44% Gain
Fraud Detection vs Human Rules

Deep Learning

Rules Engine

Deep Learning vs Rules Engine at 9% FP

40% Gain
Document Automation in Banking

Business challenge

▪ Labor intensive and time consuming document handling process.
▪ Facilitating industrialization and process automation to maximize resource utilization, improve productivity and save cost.

Solution

▪ Creating an Operations Center of Excellence (CSO), including the use of AI.
▪ Using Machine Learning and Deep Learning to automate the document handling process.
▪ Building a cognitive platform that allows the reuse of capabilities for different use cases.

Benefits

▪ Able to classify the documents with 0.99 accuracy.
▪ Reduced time and cost by process automation with Machine and Deep Learning.
▪ Providing a platform to work on improving the efficiency of other use cases.

Tools & Technologies
Document Automation Learnings

Validation
- Several rules and checks were required to improve the output from OCR.
- Depending on the entity, this step can be time consuming, its a trade-off between speed vs. accuracy

Process Documents
- Several approaches were considered
- Quality of documents was a challenge and we recommend improving the image quality for better performance

Classification
- Classic Machine Learning was enough to achieve great results for document type classification

Recognition
- Out of the different OCR tested (Abby, Google, Microsoft and Tesseract), Microsoft and Tesseract had the best results and both were included in the final architecture for the PoC, both combined with several NLP approaches
- Given the quality of certain documents, OCR results were sometimes incorrect

Detection
- Deep Learning approach for image detection seems to work well but requires more annotated data then available during PoC, minimum 1K per class is our recommendation
- Alternative OCR + NLP is also viable but requires NLP and words matching which is a more sensitive approach to different document types than the DL approach
Enterprise DL Characteristics

- Time series
- Long tail distributions
- Complex correlations
- Structured data
Deep Learning is a Lot More than ML Code

Source: Hidden Technical Debt in Machine Learning Systems
Sculley et. al. Google
Agenda

Overview

Development Challenges

Production Challenges

Conclusions
Model Training Can Be Costly

<table>
<thead>
<tr>
<th>Batch Size</th>
<th>epochs</th>
<th>Top-1 Accuracy</th>
<th>hardware</th>
<th>cost ($)</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>256</td>
<td>100</td>
<td>58.7%</td>
<td>8-core CPU + K20 GPU</td>
<td>3,000</td>
<td>144h</td>
</tr>
<tr>
<td>512</td>
<td>100</td>
<td>58.8%</td>
<td>1 DGX station</td>
<td>129,000</td>
<td>6h 10m</td>
</tr>
<tr>
<td>4096</td>
<td>100</td>
<td>58.4%</td>
<td>1 DGX station</td>
<td>129,000</td>
<td>2h 19m</td>
</tr>
<tr>
<td>32K</td>
<td>100</td>
<td>58.5%</td>
<td>512 KNLS</td>
<td>1.2 million</td>
<td>24m</td>
</tr>
</tbody>
</table>

Table 7: The speed and hardware cost for training AlexNet.
For batch size=32K, we changed local response norm in AlexNet to batch norm.

<table>
<thead>
<tr>
<th>Batch Size</th>
<th>epochs</th>
<th>Top-1 Accuracy</th>
<th>hardware</th>
<th>cost ($)</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>256</td>
<td>90</td>
<td>73.0%</td>
<td>1 DGX station</td>
<td>129,000</td>
<td>21h</td>
</tr>
<tr>
<td>8192</td>
<td>90</td>
<td>72.7%</td>
<td>1 DGX station</td>
<td>129,000</td>
<td>21h</td>
</tr>
<tr>
<td>8192</td>
<td>90</td>
<td>72.7%</td>
<td>32 DGX stations</td>
<td>4.1 million</td>
<td>1h</td>
</tr>
<tr>
<td>32K</td>
<td>90</td>
<td>72.4%</td>
<td>512 KNLS</td>
<td>1.2 million</td>
<td>1h</td>
</tr>
</tbody>
</table>

Table 8: The speed and hardware cost for training ResNet50. We did not use data augmentation.
# Model Training Challenges

<table>
<thead>
<tr>
<th>Long Training Duration</th>
<th>From Research to Production Code</th>
<th>Scaling Execution</th>
<th>Parameter Optimization</th>
<th>Technical Challenges</th>
</tr>
</thead>
<tbody>
<tr>
<td>Many iterations to tune even simple models: model parameters and data prep</td>
<td>Academic quality open source code for latest research (if even available) often needs refactoring / rewrite</td>
<td>Scaling the execution to multiple nodes and GPUs not straightforward Ongoing updates in production, monitor quality</td>
<td>Manual optimization of model parameter settings inefficient</td>
<td>Incorrect handling of categorical features when training may confuse the network Class imbalance common, limits scale of data, realistic synthetic data</td>
</tr>
</tbody>
</table>
# Model Training Learnings

<table>
<thead>
<tr>
<th>Automated Model Search</th>
<th>Reproducible Results</th>
<th>Transfer Learning</th>
<th>Robust Enterprise Connectors</th>
<th>Model Debugging</th>
<th>Technical Resolutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automated model search is better than human effort: grid, random, Bayesian, DRL, e.g., YellowFin</td>
<td>Data set versioning and model management to ensure reproducibility of results</td>
<td>For some domains: faster training, reduce required training data volume</td>
<td>Data warehouse integration</td>
<td>Visualization tools</td>
<td>Categorical features w/ 1-Hot-Encoding, hashes, embeddings</td>
</tr>
<tr>
<td>Continuous updates, production quality</td>
<td></td>
<td></td>
<td>Secure data lake integration</td>
<td>Boost confidence and insight with Model Interpretation</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Integrated data e.g., virtual query</td>
<td></td>
<td>Weighted loss functions, oversampling, blocking</td>
</tr>
</tbody>
</table>
Model Visualization
# Data Prep Challenges

<table>
<thead>
<tr>
<th>Source Data Quality</th>
<th>Data Integration</th>
<th>Time Series</th>
<th>Reproduc-ability</th>
<th>Feature Quality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incomplete, stale, artificial data risks</td>
<td>High value and volume separate</td>
<td>Sliding windows complex</td>
<td>For train &amp; test data</td>
<td>Complex pipelines, subtle errors not easy to detect</td>
</tr>
<tr>
<td>Lack of visibility</td>
<td>Structured data access</td>
<td>Aggregation of history</td>
<td>Handling data updates &amp; schema evolution</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Data security not optional</td>
<td>Indepen-dent series</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Data Prep Learnings

<table>
<thead>
<tr>
<th>Production Pipelines</th>
<th>Integrated Big Data</th>
<th>Temporal and Time Series Support</th>
<th>Analytics Ops</th>
</tr>
</thead>
<tbody>
<tr>
<td>Governance, quality, integration w/ monitoring for data feeds and features</td>
<td>In-memory data frames isn’t the right answer</td>
<td>Temporal SQL simplicity</td>
<td>Regression test and controlled deployments for data pipelines and feature prep code</td>
</tr>
<tr>
<td>Scalable joins and functions over data sets simplify, enable traceability</td>
<td>Virtual or physical warehouse is simplest</td>
<td>Time series simplified w/ SQL windows</td>
<td>Database and monitoring of pipelines for governance and traceability of data sets</td>
</tr>
<tr>
<td></td>
<td>Alternative is replication into a common environment from multiple sources</td>
<td>Independent time series: parallel LSTMs or merged time series w/ nulls</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

© 2017 Teradata
### Time Series Window SQL Example

```sql
SELECT territory, smonth, sales, AVG(sales) OVER (PARTITION BY territory ORDER BY smonth ROWS 2 PRECEDING)
FROM sales_history;
```

<table>
<thead>
<tr>
<th>territory</th>
<th>smonth</th>
<th>sales</th>
<th>Moving Avg(sales)</th>
</tr>
</thead>
<tbody>
<tr>
<td>East</td>
<td>199810</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>East</td>
<td>199811</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>East</td>
<td>199812</td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td>East</td>
<td>199901</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>East</td>
<td>199902</td>
<td>10</td>
<td>9</td>
</tr>
</tbody>
</table>
Temporal SQL Example

SEQUENCED QUERY—TEMPORAL JOIN
A laptop (150,101) was found to be broken due to shock. Find out other items that were on the same route (same truck) at the same time.

WITH TEMPORAL SUPPORT
VALIDTIME
SELECT b.item_id, b.item_serial_num
FROM objectlocation a, objectlocation b
WHERE a.item_id = 150
AND a.item_serial_num = 101
AND a.location = 'Route to Store R'
AND b.location = a.location

WITHOUT TEMPORAL SUPPORT WITH STATE TABLE
SELECT b.item_id, b.item_serial_num, BEGIN(b.validtime), END(b.validtime)
FROM state_objectlocation a, state_objectlocation b
WHERE a.item_id = 150 and a.item_serial_num = 101 and a.location = 'Route to Store R'
AND b.location = a.location
AND BEGIN(a.validtime) <= BEGIN(b.validtime)
AND END(a.validtime) >= END(b.validtime)
UNION ALL
SELECT b.item_id, b.item_serial_num, BEGIN(b.validtime), END(b.validtime)
FROM state_objectlocation a, state_objectlocation b
WHERE a.item_id = 150 and a.item_serial_num = 101 and a.location = 'Route to Store R'
AND b.location = a.location
AND BEGIN(a.validtime) <= BEGIN(b.validtime)
AND END(a.validtime) >= END(b.validtime)
UNION ALL
SELECT b.item_id, b.item_serial_num, BEGIN(b.validtime), END(b.validtime)
FROM state_objectlocation a, state_objectlocation b
WHERE a.item_id = 150 and a.item_serial_num = 101 and a.location = 'Route to Store R'
AND b.location = a.location
AND BEGIN(a.validtime) <= BEGIN(b.validtime)
AND END(a.validtime) >= END(b.validtime)
UNION ALL
SELECT b.item_id, b.item_serial_num, BEGIN(b.validtime), END(b.validtime)
FROM state_objectlocation a, state_objectlocation b
WHERE a.item_id = 150 and a.item_serial_num = 101 and a.location = 'Route to Store R'
AND b.location = a.location
AND BEGIN(a.validtime) <= BEGIN(b.validtime)
AND END(a.validtime) >= END(b.validtime)
# Model Management Challenges

<table>
<thead>
<tr>
<th>Model Scale</th>
<th>Experiment Scale</th>
<th>Performance Analysis</th>
<th>Into Production</th>
</tr>
</thead>
<tbody>
<tr>
<td>O(1k) models in enterprise</td>
<td>O(10k) experiments trying new features, approaches</td>
<td>Valid simulations Repeatable data sets</td>
<td>Live tests A/B test methodology Deploy &amp; rollback</td>
</tr>
</tbody>
</table>
Model Management: Capabilities to Support

- **Auditability**
- **Continuous Monitoring**
- **Automated Retraining**

**Model Management**
- Champion Challenger
- Performance Data Store
- Feature Store
- Model Metadata Store
- Model Store

**Sources**
- Validation
- Historic
- Stream

**Continuous Integration**
- Auto Deploy

**Production Scoring**

**Production Data Output**
Agenda

Overview

Development Challenges

Production Challenges

Conclusions
Model Serving

TYPICAL CONDITIONS

• Models typically fit in a single GPU’s RAM
• Models will require between 5 million and 100 million FLOPS per inference
• Models have fixed size inputs and outputs
• Models may depend on conflicting versions of libraries

REQUIREMENTS

• Online scoring of ~100,000 inferences/second (1 GB/s)
• Online inference responses required within 5 to 100 ms
• Up to 100s of model instances concurrently
• QoS
  • Enable model ensembling
  • Support of conflicting versions of libraries
• High availability
• High throughput
• Recovery after failure (fault tolerant)
• Realtime or batch data integration

• Difficulty to deploy and scale multi-GPU multi-node infrastructures
Model Serving - Demo

[Image of the Model Serving - Demo interface]

---

Synthetic Fraud
Fill in the form to test features for fraud

- Model Name *
- Model Version *
- Request Type *
- Data *

Score

Image Classification
Submit images to be scored

- Model Name *
- Model Version *
- Request Type *

Select or drop files

Score
## Model Interpretation Challenge

<table>
<thead>
<tr>
<th>Black Boxes</th>
<th>Regulations</th>
<th>Complex Domains</th>
<th>Development Integration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Humans need confidence</td>
<td>Auditability</td>
<td>Explanations beyond simple perturbation</td>
<td>Explanations span from development to production</td>
</tr>
<tr>
<td>Trust required to act</td>
<td>Compliant explanations, e.g., EU’s General Data Protection Regulation (GDPR)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

© 2017 Teradata
Model Interpretation Learnings

• Application of LIME - Locally Interpretable Model Explanation research

• Generative models for more complex domains promising
# Production Data Integration Challenges

<table>
<thead>
<tr>
<th>Real-time Feeds</th>
<th>Batch Feeds</th>
<th>Rapid Iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>High availability, reliability</td>
<td>Massive scale support</td>
<td>Feature prep needs to be in sync w/ training</td>
</tr>
<tr>
<td>Streaming data</td>
<td>Integration with data pipelines, storage</td>
<td>Same logic, latency</td>
</tr>
<tr>
<td>Low latency feature calc (e.g., time series)</td>
<td>Application integration</td>
<td>Forward compatibility</td>
</tr>
<tr>
<td>Application integration</td>
<td>Application &amp; data change</td>
<td></td>
</tr>
</tbody>
</table>

© 2017 Teradata
Serving Data Integration Approach

Model performance monitoring

Model Management

Model Training

Batch Data Preparation

Inference Operational Store

Inference Results & Model Interpretation

Inference Platform

Model Scoring

Real Time Data Preparation

Data Warehouse

Operational Data

Scoring

Input

Model Features

Data

Data

Data

Data

Data

Model Features

Model Features

Model Features

Models

Deploy Model

Model Metrics

Model Metrics

Models
### Database Serving Demo

<table>
<thead>
<tr>
<th></th>
<th>v24</th>
<th>v25</th>
<th>v26</th>
<th>v27</th>
<th>v28</th>
<th>amount</th>
<th>class</th>
<th>score</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>81.85</td>
<td>-0.33850</td>
<td>0.92513</td>
<td>0.14326</td>
<td>0.09419</td>
<td>0.35953</td>
<td>0.04621</td>
<td>0.08370</td>
</tr>
<tr>
<td>11</td>
<td>70247</td>
<td>0.50051</td>
<td>0.12087</td>
<td>0.02136</td>
<td>0.12497</td>
<td>0.81357</td>
<td>0.06252</td>
<td>0.01133</td>
</tr>
<tr>
<td>12</td>
<td>262866</td>
<td>0.69671</td>
<td>0.00585</td>
<td>0.17480</td>
<td>-0.49202</td>
<td>0.89534</td>
<td>0.04274</td>
<td>0.00132</td>
</tr>
<tr>
<td>13</td>
<td>95638</td>
<td>0.39230</td>
<td>0.11345</td>
<td>0.03358</td>
<td>-0.35499</td>
<td>0.89159</td>
<td>0.02641</td>
<td>0.00099</td>
</tr>
<tr>
<td>14</td>
<td>998586</td>
<td>0.10474</td>
<td>0.27525</td>
<td>0.01129</td>
<td>0.10049</td>
<td>0.51361</td>
<td>0.02143</td>
<td>0.12150</td>
</tr>
<tr>
<td>15</td>
<td>60136</td>
<td>0.02831</td>
<td>0.13238</td>
<td>0.10572</td>
<td>-0.23746</td>
<td>0.74289</td>
<td>0.06015</td>
<td>0.27505</td>
</tr>
<tr>
<td>16</td>
<td>48308</td>
<td>-0.60583</td>
<td>0.78815</td>
<td>0.16517</td>
<td>-0.03912</td>
<td>0.45354</td>
<td>0.08706</td>
<td>0.12939</td>
</tr>
<tr>
<td>17</td>
<td>90422</td>
<td>0.10375</td>
<td>0.33102</td>
<td>0.319</td>
<td>-0.38205</td>
<td>0.74113</td>
<td>0.09280</td>
<td>0.01705</td>
</tr>
<tr>
<td>18</td>
<td>91897</td>
<td>-0.88858</td>
<td>0.63209</td>
<td>0.49716</td>
<td>0.04900</td>
<td>0.67286</td>
<td>0.07062</td>
<td>0.12939</td>
</tr>
<tr>
<td>19</td>
<td>219</td>
<td>0.04211</td>
<td>0.89699</td>
<td>0.15172</td>
<td>0.48130</td>
<td>0.32895</td>
<td>0.39025</td>
<td>0.04954</td>
</tr>
<tr>
<td>20</td>
<td>21129</td>
<td>0.29818</td>
<td>0.66276</td>
<td>0.56085</td>
<td>0.32905</td>
<td>0.99342</td>
<td>0.02284</td>
<td>0.00760</td>
</tr>
<tr>
<td>21</td>
<td>903795</td>
<td>-0.30412</td>
<td>0.50100</td>
<td>0.20544</td>
<td>0.07200</td>
<td>0.10631</td>
<td>0.08655</td>
<td>0.06349</td>
</tr>
<tr>
<td>22</td>
<td>745289</td>
<td>-1.37186</td>
<td>0.29453</td>
<td>0.38889</td>
<td>0.39081</td>
<td>0.68543</td>
<td>0.15170</td>
<td>0.01137</td>
</tr>
<tr>
<td>23</td>
<td>60433</td>
<td>0.37374</td>
<td>0.36254</td>
<td>0.36875</td>
<td>0.39905</td>
<td>0.50275</td>
<td>0.01141</td>
<td>0.00644</td>
</tr>
</tbody>
</table>
Need for Repeatable Infrastructure

- Version Control
- Workflow management
- Continuous Integration
- Model Management
- Application Management
- Data Science Workbench
- Production Data Systems
- Data Lab
- Curated Pipelines
- Data Sources
Conclusions

Deep Learning realizing value in enterprise
Builds on foundations of integrated data and analytics ops
Leverage new compute, frameworks at scale
Investments in people, technology, data science and operating model