MISTIQUE: A System to Store and Query Model Intermediates for Model Diagnosis

Presented By:
Nenya Edjah, Bagatur Askaryan, Chi-Ning Chou, and Mitali Bafna
Model Diagnosis
Traditional ML Pipelines

What do my inputs look like after dim reduction?

Which features have the highest magnitude for neighborhood X?

What is the error rate for homes in neighborhood X?
Deep Neural Networks

Which input pixels are the most important?

What do my activation maps look like after this layer?

What do my logits look like for Image-142?
Two Strategies for Getting Intermediates

Store on Disk  vs  Re-run the model

Why is each strategy flawed?
The Solution: MISTIQUE
Pipeline Executor

“make intermediate X”
(e.g. get layer $i$ of model $B$ on examples $\{ex1, ex2\ldots\}$)

Model $B$

Layer $i$

log of intermediate X
### Data Frame

#### Intermediate

<table>
<thead>
<tr>
<th>row_id</th>
<th>feature1</th>
<th>feature2</th>
<th>feature3</th>
</tr>
</thead>
<tbody>
<tr>
<td>ex1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ex2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ex4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ex5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Column Chunk

- row_id | feature1 | feature2 | feature3 |
- ex1     |          |          |          |
- ex2     |          |          |          |
- ...     |          |          |          |
- ex4     |          |          |          |
- ex5     |          |          |          |
- ...     |          |          |          |

- row_id | feature1 | feature2 | feature3 |
- ex1     |          |          |          |
- ex2     |          |          |          |
- ...     |          |          |          |
- ex4     |          |          |          |
- ex5     |          |          |          |
- ...     |          |          |          |
Data Store

log of intermediate X

store X?

query of intermediate Y

Memory

Storage
“get column A of intermediate X”
S: log(specs)  Q: get(keys)

- **Pipeline Executor**
- **Chunk Reader**
- **Data Store**
- **Metadata DB**

Connections:
- S1 → Q1
- S2 → Q3.A
- Q3.B → S3
- Q4 → Q1
- Q2 → Q1
Naive Strategy
Ideas for data storage?
Quantization and Summarization

Real value

Quantized value

Quantization and Summarization

Exact and Approximate De-duplication

≈

1729 times

5 times

Memory

Adaptive Materialization
Exact and Approximate De-duplication

Step 1:

Step 2:
Other ideas for data storage?
Re-run traditional ML pipeline:

\[ \sum_{s=0}^{i} t_{\text{read\_x\_former}}(s) + t_{\text{read\_input}}(s) + t_{\text{exec\_x\_former}}(s) \]
Re-run DNN:

\[ t_{\text{model-load}} + \frac{n \cdot \text{size}(ex)}{\rho} + \frac{n}{b} \sum_{s=0}^{i} t_{fwd}(s, b) \]
Read from Data Store:

\[
\frac{n \cdot \text{size}(ex)}{\rho d}
\]
Storage cost

\[ \gamma = \frac{(t_{i,\text{re-run}} - t_{i,\text{read}}) \cdot n_{\text{query}}(i)}{S(i)} \]
Experimental Setup

TRAD model

DNN model

Intermediates: Model pipeline

Layers of DNN
Read better than rerun

Rerun better than read

TRAD: 2.5X - 390X

DNN: 2.5X for Layer 1
2.5X - 390X for other Layers
Other improvements to MISTIQUE?