Fred Reiss & Mike Dusenberry
IBM Spark Technology Center
San Francisco

Predicting Breast Cancer Proliferation Scores with Apache SystemML
Let us introduce ourselves.
Who we are

Fred Reiss
Apache SystemML Committer.

2014-present: Chief Architect, IBM Spark Technology Center.

Mike Dusenberry (@dusenberrymw)
Apache SystemML Committer.

2012-2015: M.D. Candidate & Researcher at the Brody School of Medicine
2015-present: Software Engineer, Machine Learning, IBM Spark Technology Center
Founded in 2015.

Location:
Physical: 505 Howard St., San Francisco CA
Web: http://spark.tc Twitter: @apachespark_tc

Mission:
Contribute intellectual and technical capital to the Apache Spark community.
Make the core technology enterprise- and cloud-ready.
Build data science skills to drive intelligence into business applications — http://bigdatauniversity.com

Key statistics:
About 50 developers, co-located with 25 IBM designers.
Major contributions to Apache Spark http://jiras.spark.tc
Apache SystemML is now an Apache Incubator project.
Founding member of UC Berkeley AMPLab and RISE Lab
Member of R Consortium and Scala Center
Spark Technology Center

Founded in 2015.

Location:
Physical: 505 Howard St., San Francisco CA
Web: http://spark.tc  Twitter: @apachespark_tc

Mission:
Contribute intellectual and technical capital to the Apache Spark community.
Make the core technology enterprise- and cloud-ready.
Build data science skills to drive intelligence into business applications — http://bigdatauniversity.com

Key statistics:
About 50 developers, co-located with 25 IBM designers.
Major contributions to Apache Spark http://jiras.spark.tc
Apache SystemML is now an Apache Incubator project.
Founding member of UC Berkeley AMPLab and RISE Lab
Member of R Consortium and Scala Center
Breast Cancer Tumor Proliferation Assessment Challenge 2016 (TUPAC16)

http://tupac.tue-image.nl/
The Dataset:

500 whole-slide images of breast tissue tumors.

Process that produced these images:
1. Biopsy breast tumor.
2. Slice tissue thinly onto slide.
3. Stain slide.
4. Examine slide with microscope.
5. Scan at high resolution.

Tumor proliferation score labels based on mitosis counting by pathologists (M.D., D.O., etc.).
Example Image

2.6 billion pixels

File Size:
657 MB compressed
19.13 GB in 64-bit double format

We have 500 of these!
Example Image

2.6 billion pixels

File Size:
657 MB compressed
19.13 GB in 64-bit double format

We have 500 of these!
Pathologist looks for nuclei characteristics

Example High-Resolution Region

Tiny red rectangle
Tumor Proliferation Score:

Classification of mitotic activity (i.e. tumor growth).

Scores: 1 (best prognosis), 2, 3 (worst prognosis).

One of three components for a histological grading of invasive breast cancer.
## Grading System in Invasive Breast Cancer

<table>
<thead>
<tr>
<th>Score</th>
<th>1</th>
<th>&gt; 2</th>
<th>&gt; 3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A. Tubule formation</strong></td>
<td>&gt;75%</td>
<td>10-75%</td>
<td>&lt; 10%</td>
</tr>
<tr>
<td><strong>B. Mitotic count/HPF (microscope- and field-dependent)</strong></td>
<td>&lt; 7</td>
<td>7-12</td>
<td>&gt;12</td>
</tr>
<tr>
<td><strong>C. Nuclear size and pleomorphism</strong></td>
<td>Near normal; little variation</td>
<td>Slightly enlarged; moderate variation</td>
<td>Markedly enlarged; marked variation</td>
</tr>
</tbody>
</table>

Grade I cancer if total score (A + B + C) is 3-5
Grade II cancer if total score (A + B + C) is 6 or 7
Grade III cancer if total score (A + B + C) is 8 or 9

HPF = high-power field.
Goal:

**Predict** the tumor proliferation score from a whole-slide image.

Only three scores, so treat the problem as a **classification problem**.
State of the Art

Recent published work\textsuperscript{1} uses deep learning with convolutional neural nets to predict the tumor grade.

What is Deep Learning?

Traditional Machine Learning

Deep Learning
Deep Learning: Machine Learning with Power Tools

Work Faster

Avoid Pitfalls
Image Classification with Traditional ML

1. **Data Cleaning**
2. **Low-Level Features**
3. **High-Level Features**
4. **Dim. Reduction**
5. **Model Training**

Feature Engineering

Data Scientist

Goodness of Fit

Model
Image Classification with **Deep Learning**

- Data Cleaning
- Low-Level Features
- High-Level Features
- Dim. Reduction
- Model Training
- Neural Networks
- Gradient Descent
- Goodness of Fit
- Model
The Overfitting Hazard

In deep learning, the “modeling” step performs many different functions.

- Millions or billions of “model” parameters.
- Often only thousands of labeled examples.
- Overfitting is extremely common.

Avoiding overfitting requires aggressive and creative regularization.

Examples:
- Changing the topology of a neural network.
- Replacing each training example with hundreds of randomly-tweaked copies of itself.
- Fine-tuning from a pretrained model.
Recap: The Classification Problem

<table>
<thead>
<tr>
<th>Score</th>
<th>1</th>
<th>&gt; 2</th>
<th>&gt; 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. Tubule formation</td>
<td>&gt;75%</td>
<td>10-75%</td>
<td>&lt; 10%</td>
</tr>
<tr>
<td>B. Mitotic count/HPF (microscope- and field-dependent)</td>
<td>&lt; 7</td>
<td>7-12</td>
<td>&gt;12</td>
</tr>
<tr>
<td>C. Nuclear size and pleomorphism</td>
<td>Near normal; little variation</td>
<td>Slightly enlarged; moderate variation</td>
<td>Markedly enlarged; marked variation</td>
</tr>
</tbody>
</table>

Grade I cancer if total score (A + B + C) is 3-5
Grade II cancer if total score (A + B + C) is 6 or 7
Grade III cancer if total score (A + B + C) is 8 or 9
HPF = high-power field.
Tools We Used

Apache Spark for data cleaning and preparation

Keras as a high-level front end for deep learning models

TensorFlow as a Keras backend for small-scale experimentation

Apache SystemML as a Keras backend for large-scale training and validation
What is Apache SystemML?

Apache SystemML is a machine learning system for running distributed linear algebra on top of Apache Spark.

Exposes high-level R-like & Python-like languages focused on linear algebra.

APIs for Python, Scala, Java.

Spark Technology Center

Data Cleaning

Low-Level Features

High-Level Features

Dimensional Reduction

Model Training

Neural Network Model Training
Tiling

“Whole-Slide” Image

Image Tiles

Example “Tile” Image (1024x1024x3)

Data Cleaning

Neural Network Model Training

Low-Level Features

High-Level Features

Dim Reduction

Model Training
Filtering

“Tile”

Grayscale

Edge detection

Smoothing w/ morphological ops

Tissue percentage

Keep if >= 90%

Data Cleaning

Neural Network Model Training
Further Preprocessing

Image “Tiles” at 20x resolution

Example Filtered “Tile” Image (1024x1024x3)

Tile “Samples”

Example “Sample” Image (256x256x3)
PySpark for Data Preparation

~4 million samples!

```
slides = sc.parallelize(slides_nums)
tile_indices = (slides.flatMap(
    lambda slide: process_slide(slide, folder, training, tile_size, overlap))
tile_indices = tile_indices.repartition(num_partitions)
tile_indices.cache()
tiles = tile_indices.map(lambda tile_index: process_tile_index(tile_index, folder, training))
filtered_tiles = tiles.filter(lambda tile: keep_tile(tile, tile_size, tissue_threshold))
samples = filtered_tiles.flatMap(lambda tile: process_tile(tile, sample_size, grayscale))
if training:
    tumor_score_dict, molecular_score_dict = create_ground_truth_maps(folder)
samples_with_labels = (samples.map(
    lambda tup: (tup[0], tumor_score_dict[tup[0]],
    molecular_score_dict[tup[0]], Vectors.dense(tup[1])))
    df = samples_with_labels.toDF(["slide_num", "tumor_score", "molecular_score", "sample"])
df = df.select(df.slide_num.astype("int"), df.tumor_score.astype("int"),
    df.molecular_score, df["sample"]
```
Deep Learning Model: A Convolutional Neural Network (ResNet topology)

Data Cleaning

Neural Network Model Training

Low-Level Features

High-Level Features

Dim. Red. Model Training

Tumor Proliferation Score
Deep Learning Model: A Convolutional Neural Network (ResNet topology)

1. Replace this
2. Freeze this
3. Train this

Data Cleaning

Neural Network Model Training

Low-Level Features

High-Level Features
Deep Learning Model: A Convolutional Neural Network (ResNet topology)
Setting up the pre-trained Neural Net with Keras

1. Setup & replace classifier

```python
# Create model by replacing classifier of ResNet50 model with new
# classifier specific to the breast cancer problem.
resnet50_base = ResNet50(include_top=False, input_shape=(size, size, channels))
x = Flatten()(resnet50_base.output)
preds = Dense(classes, init=my_init, activation="softmax")(x)
resnet50 = Model(input=resnet50_base.input, output=preds, name="resnet50")
```
Setting up the pre-trained Neural Net with Keras

2. Freeze previous layers

```python
# Freeze all pre-trained ResNet layers.
for layer in resnet50_base.layers:
    layer.trainable = False

# Compile model.
optim = SGD(lr=0.1, momentum=0.9, decay=0.99, nesterov=True)
model.compile(optimizer=optim, loss="categorical_crossentropy",
              loss_weights=[1/num_gpus]*num_gpus, metrics=metrics)
```
# Train new classifier

```python
epochs = 1
hist1 = model.fit_generator(train_generator, samples_per_epoch=train_samples,
                          validation_data=val_generator, nb_val_samples=val_samples,
                          nb_epoch=epochs, class_weight=class_weights, callbacks=callbacks,
                          max_q_size=8, nb_worker=1, pickle_safe=False)
```
Setting up the pre-trained Neural Net with Keras

4. Unfreeze some existing layers

```python
# Unfreeze some subset of the model and compile
for layer in resnet50_base.layers[154:]:
    layer.trainable = True

optim = SGD(lr=0.0001, momentum=0.9)
model.compile(optimizer=optim, loss="categorical_crossentropy",
              loss_weights=[1/num_gpus]*num_gpus, metrics=metrics)
```
Setting up the pre-trained Neural Net with Keras

5. “Fine-tune”

```python
# Fine-tune slowly
initial_epoch = epochs
epochs = initial_epoch + 20
hist2 = model.fit_generator(train_generator, samples_per_epoch=train_samples,
validation_data=val_generator, nb_val_samples=val_samples,
nb_epoch=epochs, initial_epoch=initial_epoch,
class_weight=class_weights, callbacks=callbacks,
max_q_size=8, nb_worker=1, pickle_safe=False)
```
## Running at Scale with SystemML

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Time (seconds)</th>
<th>Time (Hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>54000</td>
<td>15.0</td>
</tr>
<tr>
<td>6</td>
<td>26303</td>
<td>7.3</td>
</tr>
<tr>
<td>9</td>
<td>18078</td>
<td>5.0</td>
</tr>
</tbody>
</table>

### Neural Network Model Training

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Time (seconds)</th>
<th>Time (Hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>54000</td>
<td>15.0</td>
</tr>
<tr>
<td>6</td>
<td>26303</td>
<td>7.3</td>
</tr>
<tr>
<td>9</td>
<td>18078</td>
<td>5.0</td>
</tr>
</tbody>
</table>

![Graph showing processing rate vs. number of nodes]

- **Processing Rate**: 1 GPU
Tying It All Together

“Whole-Slide” Image

Image Tiles

Example Filtered “Tile” Image

Tile Samples

Example “Sample” Image

ConvNet:

Tumor Proliferation Score

ConvNet:
Initial Results

Confusion Matrix over **Training** Data:

<table>
<thead>
<tr>
<th>Predicted Value</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1573992</td>
<td>19295</td>
<td>23736</td>
</tr>
<tr>
<td>2</td>
<td>18225</td>
<td>880976</td>
<td>12084</td>
</tr>
<tr>
<td>3</td>
<td>14565</td>
<td>10953</td>
<td>1055774</td>
</tr>
</tbody>
</table>

Confusion Matrix over **Validation** Data:

<table>
<thead>
<tr>
<th>Predicted Value</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual Value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>244949</td>
<td>82977</td>
<td>100850</td>
</tr>
<tr>
<td>2</td>
<td>63894</td>
<td>32571</td>
<td>80108</td>
</tr>
<tr>
<td>3</td>
<td>101243</td>
<td>51265</td>
<td>103583</td>
</tr>
</tbody>
</table>
Initial Results

Chance of predicting 1 or 2 when actual score is 1
Training: 98.5%
Validation: 76.5%

Chance of predicting 2 or 3 when actual score is 3
Training: 98.6%
Validation: 60.4%

Overall chance of predicting a 1 or 3 score within ±1:
Training: 98.5%
Validation: 68.4%

Overfitting
Problem #1: Imperfect Filtering

**Problem**: Examining the dataset showed a large number of samples containing extraneous tissue that would not be used by a pathologist.

- Adipose (fat), noisy stroma, etc.

**Solution**: More aggressive filtering by adding optical density (OD) checks, which is a measure of light absorbance.

- Only keep tissue above a certain mean OD threshold (after morphological smoothing).

Change in accuracy:

- Training: 98.5% ➔ 97.9%
- Validation: 68.4% ➔ 71.3%

Less overfitting, but still overfitting…
Problem #2: Inconsistent Staining

Slide preparation (amounts of each stain, handling, etc.) can result in visual differences between slides using the same H&E stain. Since our samples come from 500 slides, this can lead to overfitting.
Stain Normalization

Stain Normalization

Eigendecomposition, projection, extraction of stain vectors, and recreation of slide image.

tl;dr: Image $\rightarrow$ linear algebra $\rightarrow$ stain-normalized image
Stain Normalization

Change in accuracy:
Training: 97.9% ➞ 90.1%
Validation: 71.3% ➞ 73.4%

Regularization by data cleaning…
Result Quality Progression

Accuracy on Training Set vs Accuracy on Validation Set

Accuracy on Training Set: 0.6 - 1.0
Accuracy on Validation Set: 0.6 - 0.7
Summary

Deep Learning = ML w/ power tools

Domain adaption is difficult.

Although neural network models take care of a large number of previously hand-engineered tasks, they still require guidance.

SystemML allows one to take a model + available hardware and automatically scale it up.

Check out our code!


Collaborate with us on the ML/DL, systems, etc.!
Thank You!

Thanks to the Apache SystemML Team!

Matthias Boehm
Rich Bowen
Joseph Bradley
Michael Dusenberry
Deron Eriksson
Alexandre Evfimievski
Faraz Makari
Nakul Jindal
Holden Karau
Xiangrui Meng
Madison Myers
Niketan Pansare
Berthold Reinwald
Fred Reiss
Luciano Resende
Felix Schueler
Prithviraj Sen
Henry Saputra
Arvind Surve
Shirish Tatikonda
DB Tsai
Glenn Weidner
Patrick Wendell
Imran Younus
Reynold Xin