

Machine Learning for CI



Kyra Wulffert

Matthew Treinish

Andrea Frittoli

Open Data Science Conference
London 2019

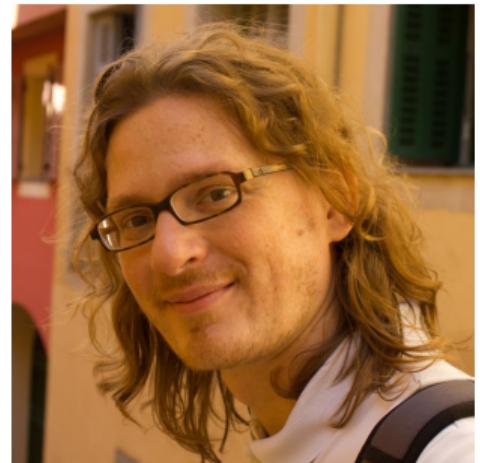
The Team



Kyra Wulffert
kwulffert@gmail.com



Matthew Treinish
mtreinish@kortar.org



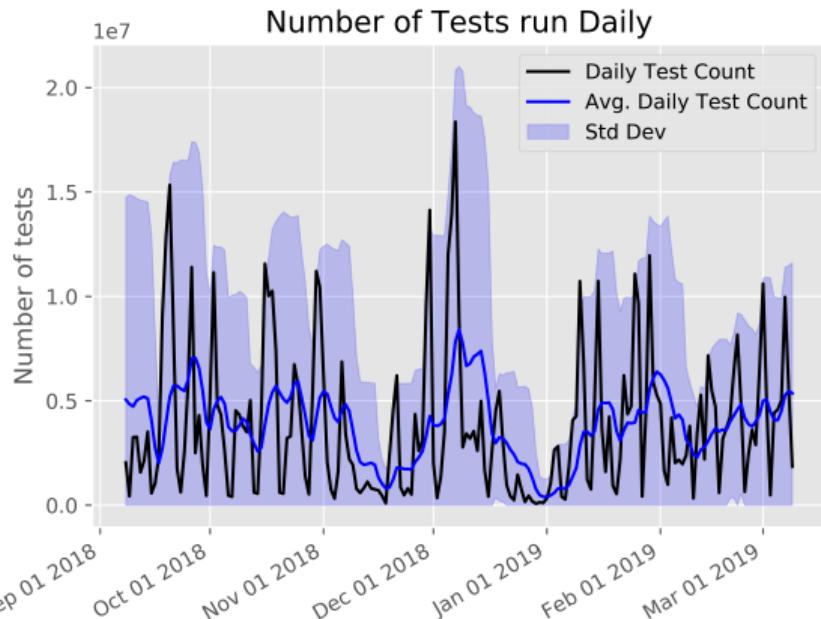
Andrea Frittoli
andrea.frittoli@gmail.com

Data Sourcing



(cc) 0

CI at Scale



Source: subunit2sql-graph dailycount

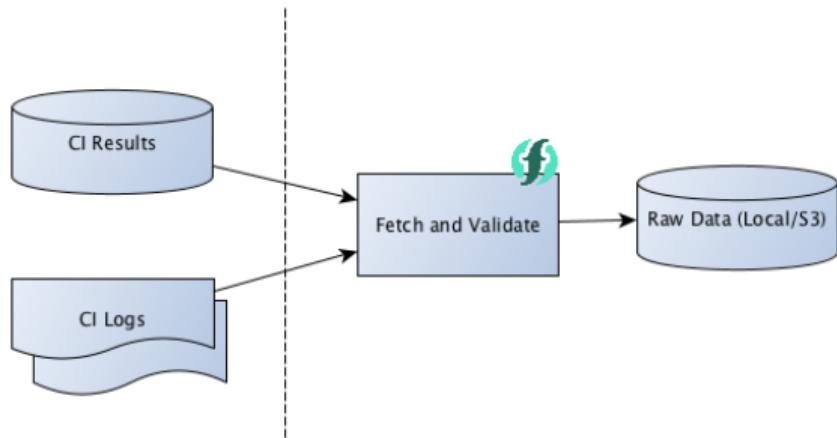
- Continuous Integration
- Continuous Log Data
- Lots of data, little time
- Triaging failures?
- AI to the rescue!

The OpenStack use case

- Integration testing in a VM
- System logs, application logs
- Dstat data
- Gate testing
- Not only OpenStack

Normalized system average load for different examples

Collecting data



- Automation and repeatability
- Light-weight data validation
- Object storage for data
- Periodic Action on OpenWhisk

Data caching diagram

— Data Preparation

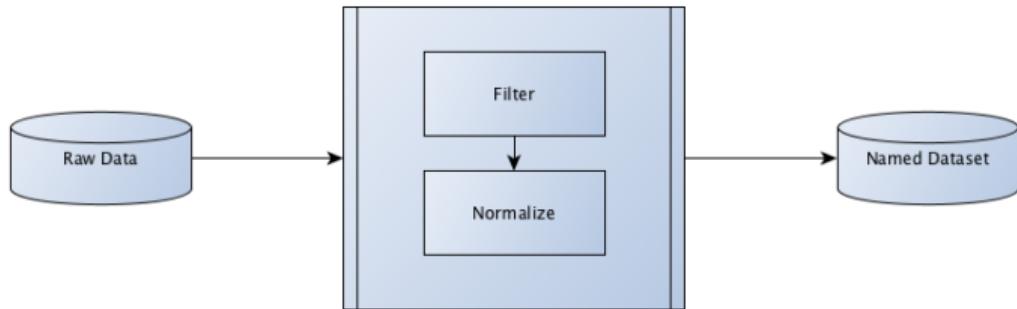


(cc) ①

Experiment Workflow

- **Visualize data**
- **Define a dataset**
- Define an experiment
- Run the training
- Collect results

```
# Build an s3 backed dataset
ciml-build-dataset --dataset cpu-load-1min-dataset \
--build-name tempest-full \
--slicer :2000 \
--sample-interval 10min \
--features-regex "(usr|1min)" \
--class-label status \
--tdt-split 7 0 3 \
--data-path s3://cimlrawdata \
--target-data-path s3://cimldatasets
```



Dataset preparation diagram

Data Selection

- What is dstat data?
- Experiment reproducibility
- Dataset selection
 - Dstat feature selection
 - Data resolution (down-sampling)

Sample of dstat data

| time | usr | used | writ | 1m |
|---------------------|------|-------------------|-------------------|------|
| 16/02/2019 21:44:52 | 6.1 | $7.36 \cdot 10^8$ | $5.78 \cdot 10^6$ | 0.97 |
| 16/02/2019 21:44:53 | 7.45 | $7.43 \cdot 10^8$ | $3.6 \cdot 10^5$ | 0.97 |
| 16/02/2019 21:44:54 | 4.27 | $7.31 \cdot 10^8$ | $4.01 \cdot 10^5$ | 0.97 |
| 16/02/2019 21:44:55 | 1 | $7.43 \cdot 10^8$ | 4,096 | 0.97 |
| 16/02/2019 21:44:56 | 0.5 | $7.44 \cdot 10^8$ | $1.5 \cdot 10^7$ | 0.97 |
| 16/02/2019 21:44:57 | 1.75 | $7.31 \cdot 10^8$ | 4,096 | 0.97 |
| 16/02/2019 21:44:58 | 0.88 | $7.43 \cdot 10^8$ | 4,096 | 0.9 |
| 16/02/2019 21:44:59 | 1.39 | $7.31 \cdot 10^8$ | $4.51 \cdot 10^5$ | 0.9 |
| 16/02/2019 21:45:00 | 1.01 | $7.44 \cdot 10^8$ | 4,096 | 0.9 |
| 16/02/2019 21:45:01 | 0.75 | $7.46 \cdot 10^8$ | 61,440 | 0.9 |
| 16/02/2019 21:45:02 | 1.26 | $7.31 \cdot 10^8$ | 4,096 | 0.9 |
| 16/02/2019 21:45:03 | 1.13 | $7.44 \cdot 10^8$ | 4,096 | 0.82 |
| 16/02/2019 21:45:04 | 5.77 | $7.77 \cdot 10^8$ | $1.72 \cdot 10^5$ | 0.82 |
| 16/02/2019 21:45:05 | 9.85 | $8.31 \cdot 10^8$ | $4.99 \cdot 10^6$ | 0.82 |
| 16/02/2019 21:45:06 | 3.88 | $8.46 \cdot 10^8$ | $8.25 \cdot 10^7$ | 0.82 |

Data Normalization

– Unrolling

Sample of unrolled data

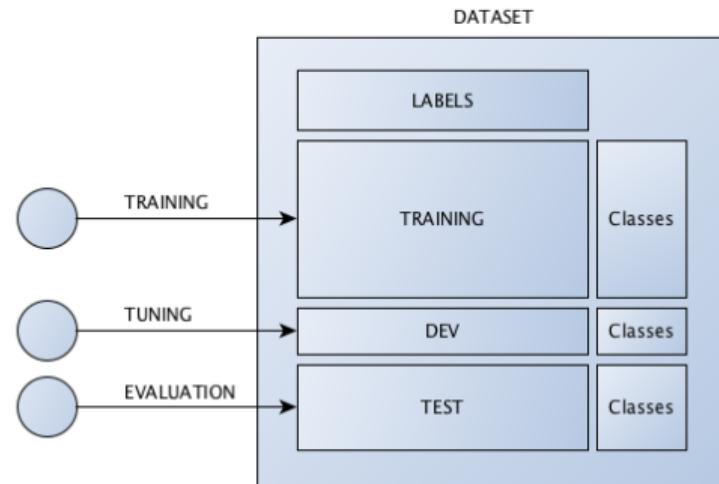
| usr1 | usr2 | usr3 | 1m1 | 1m2 | 1m3 |
|------|------|------|------|------|------|
| 6.1 | 1.75 | 1.26 | 0.97 | 0.97 | 0.9 |
| 5.9 | 1.5 | 3.1 | 0.9 | 0.92 | 0.97 |
| 5.8 | 1.76 | 2.2 | 0.89 | 0.91 | 0.94 |

– Normalizing

Sample of normalized data

| usr1 | usr2 | usr3 | 1m1 | 1m2 | 1m3 |
|------|------|------|------|------|------|
| 0.6 | 0.3 | -0.5 | 0.6 | 0.6 | -0.5 |
| -0.1 | -0.7 | 0.5 | -0.3 | -0.2 | 0.5 |
| -0.4 | 0.3 | 0 | -0.4 | -0.4 | 0 |

Building the dataset



Structure of a dataset

- Split in training, dev, test
- Obtain classes
- Store normalized data on s3
- Input function for training
- Input function for evaluation

Experiment Workflow

- Visualize data
- Define a dataset
- **Define an experiment**
- **Run the training**
- **Collect results**

```
# Define a local experiment
ciml-setup-experiment —experiment dnn—5x100 \
    —estimator tf.estimator.DNNClassifier \
    —hidden-layers 100/100/100/100/100 \
    —steps $(( 2000 / 128 * 500 )) \
    —batch-size 128 \
    —epochs 500 \
    —data-path s3://cimldatasets
```

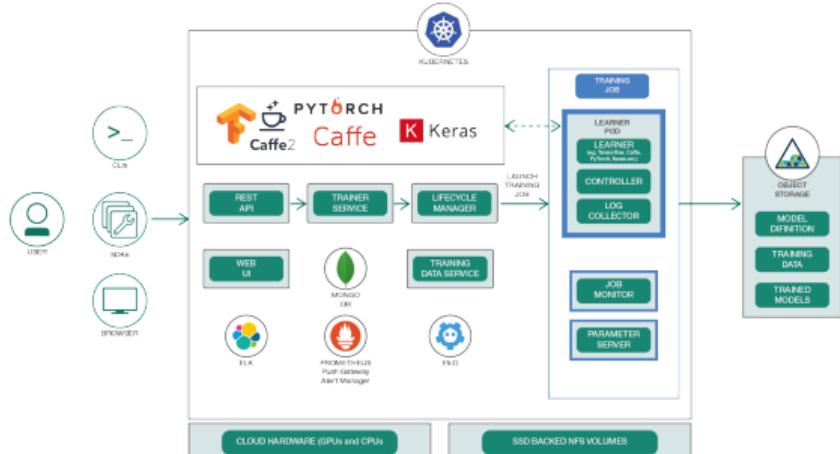
```
# Train the model locally based on the dataset and experiment
# Store the evaluation metrics as a JSON file
ciml-train-model —dataset cpu-load—1min—dataset \
    —experiment dnn—5x100 \
    —data-path s3://cimldatasets

# Train the same model in a FFDL cluster
ffdl_train.sh cpu-load—1min—dataset dnn—5x100
```

Data Pipelines



Training Infrastructure



- TensorFlow Estimator API
- CIML wrapper
- ML framework interchangeable
- Training Options:
 - Run on a local machine
 - Helm deploy CIML, run in containers
 - Submit training jobs to Ffdl
 - Kubeflow

FfDL Architecture - Source: <https://developer.ibm.com/code/>

Prediction

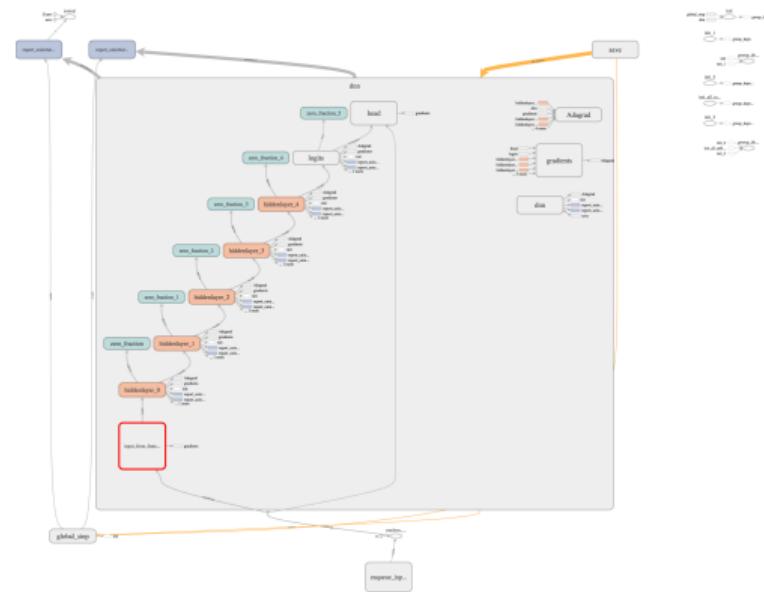
- Event driven: near real time
 - No request to serve the prediction to
 - MQTT Trigger from the CI system
 - CIML produces the prediction
 - Trusted Source: Continuous Training
-
- CIML kubernetes app components:
 - MQTT Client receives events
 - Data module fetches and prepares data
 - TensorFlow wrapper issues the prediction
 - Example: comment back on Gerrit/Github

— Experiments



DNN - Binary Classification

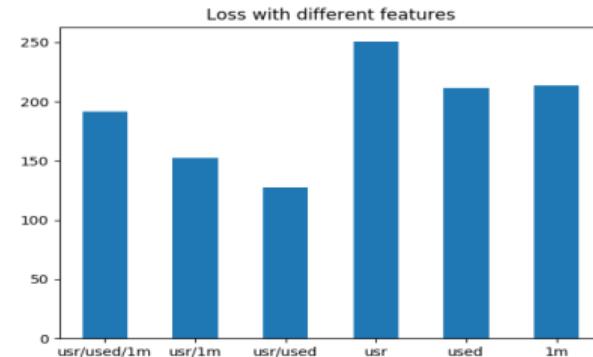
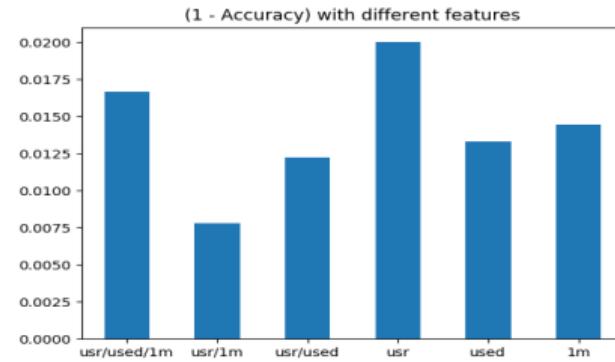
- Classes: Passed or Failed
- Supervised training
- TensorFlow *DNNClassifier*, classes=2
- Dataset:
 - CI Job "tempest-full"
 - Gate pipeline only
 - 3955 examples split in 60% training, 20% dev and 20% test
- Hyper-parameters:
 - Activation function: ReLU
 - Output layer: Sigmoid
 - Optimizer: Adagrad
 - Learning rate (initial): 0.05
 - 5 hidden layers, 100 units per layer
 - Batch Size: 128, Epochs: 500



Network Graph - Source: TensorBoard

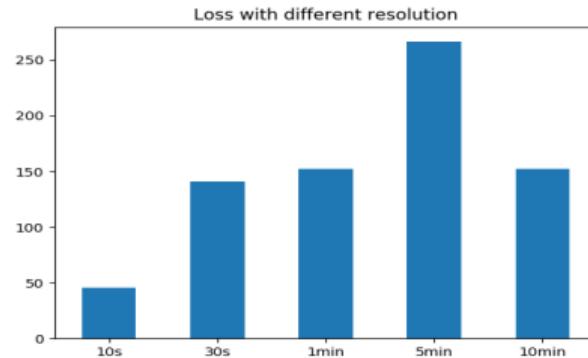
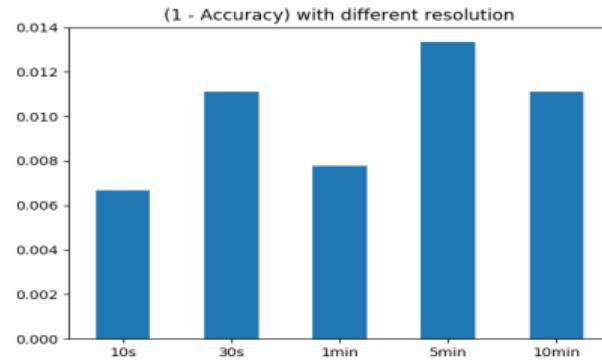
DNN - Binary Classification

- Selecting the best feature set
- Primary metric: accuracy
- Aim for lower loss, caveat: overfitting
- Key:
 - **usr**: User CPU
 - **used**: Used Memory
 - **1m**: System Load - 1min Average
 - Data Resolution: **1min**
 - Source: TensorFlow evaluation
- Winner: (**usr, 1m**) tuple
- Accuracy achieved: **0.992**
- 6 mistakes on a 791 test set



DNN - Binary Classification

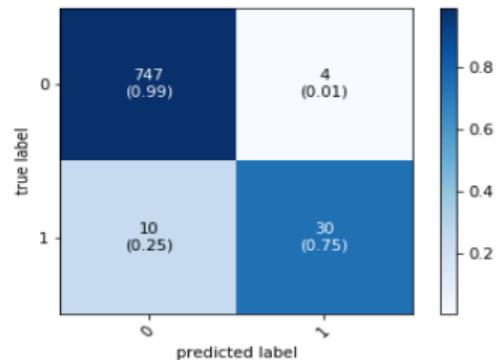
- Selecting the data resolution
- Primary metric: accuracy
- Aim for lower loss, caveat: overfitting
- Note: careful with NaN after down-sampling
- Key:
 - Original data frequency: 1s
 - x-axis: new sampling rate
 - Features: (**usr**, **1m**)
 - Source: TensorFlow evaluation
- Winner: **10s**
- Accuracy achieved: **0.993**
- 6 mistakes on a 791 test set



DNN - Binary Classification - Metrics report

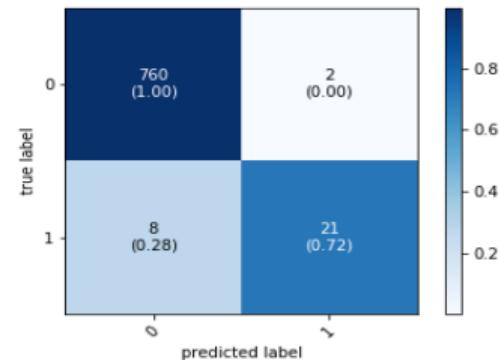
Metrics report for usr 1m and 1min

| status | precision | recall | f1-score | support |
|----------|-----------|--------|----------|---------|
| passed | 0.99 | 0.99 | 0.99 | 751 |
| failed | 0.88 | 0.75 | 0.81 | 40 |
| accuracy | | | 0.98 | 791 |



Metrics report for usr 1m and 10s

| status | precision | recall | f1-score | support |
|----------|-----------|--------|----------|---------|
| passed | 0.99 | 1 | 0.99 | 762 |
| failed | 0.91 | 0.72 | 0.81 | 29 |
| accuracy | | | 0.99 | 791 |



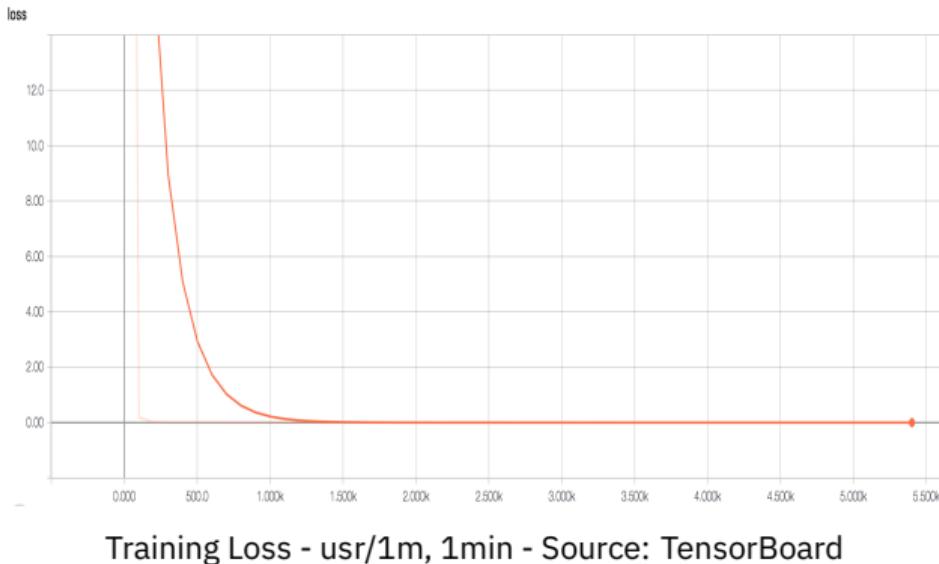
Changing test job

| metric | tempest-full | tempest-full-py3 |
|----------------------|--------------|------------------|
| accuracy | 0.994 | 0.953 |
| loss | 47.176 | 86.873 |
| auc_precision_recall | 0.949 | 0.555 |

- Train with "tempest-full"
- Evaluating with "tempest-full-py3"
 - Similar setup, uses python3
 - It does not include swift and swift tests
 - 600 examples evaluation set
- Dataset and training setup:
 - Features: (usr, 1m)
 - Resolution: 1min
 - Same hyper-parameters

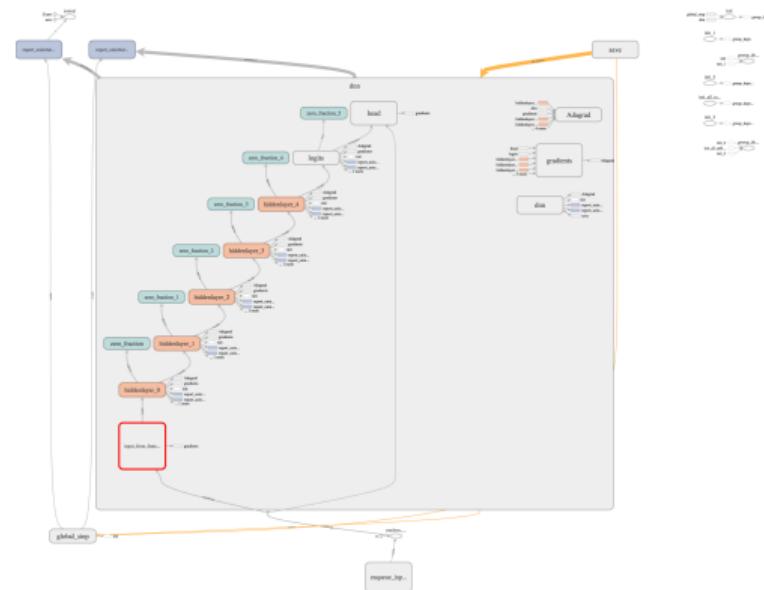
Binary Classification - Summary

- Features: User CPU and 1min Load Avg
- Resolution: 10s best, 1 minute may be enough
- High accuracy: **0.993**
- High precision, recall and F1-score
- A trained model might be applicable to similar CI jobs



DNN - Multi Class

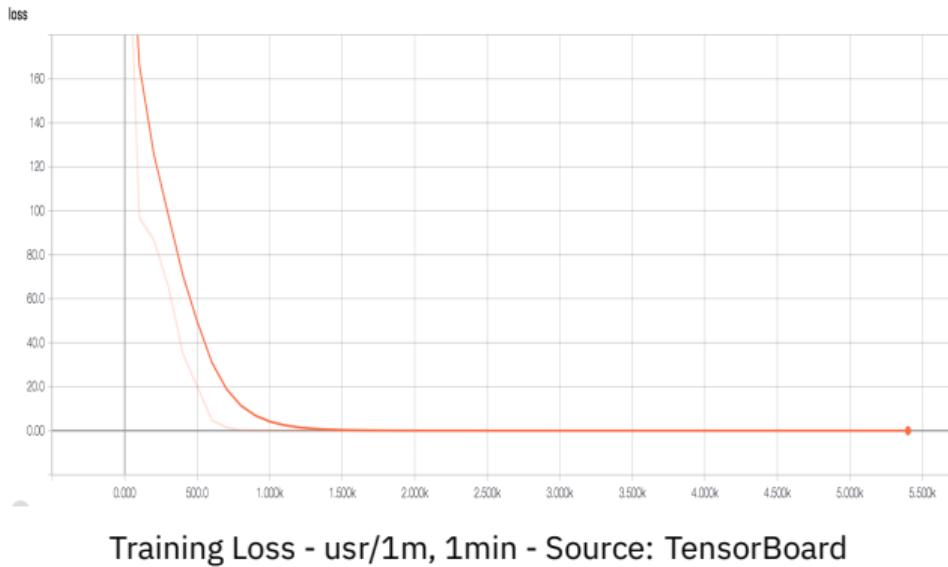
- Classes: Hosting Cloud Provider
- Supervised training
- TensorFlow *DNNClassifier*, classes=**10**
- Dataset:
 - CI Job "tempest-full"
 - Gate pipeline only
 - 3000 examples, 2100 training, 900 test
- Hyper-parameters:
 - Activation function: ReLU
 - Output layer: Sigmoid
 - Optimizer: Adagrad
 - Learning rate (initial): 0.05
 - 5 hidden layers, 100 units per layer
 - Batch Size: 128, Epochs: 500



Network Graph - Source: TensorBoard

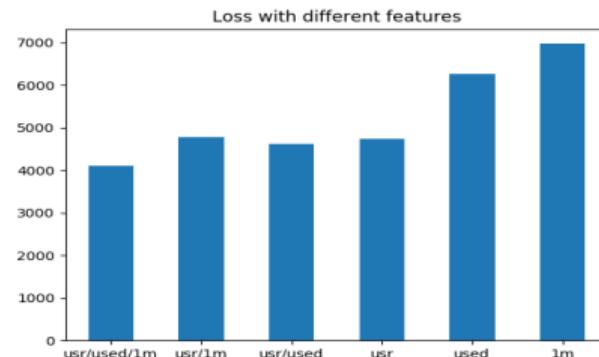
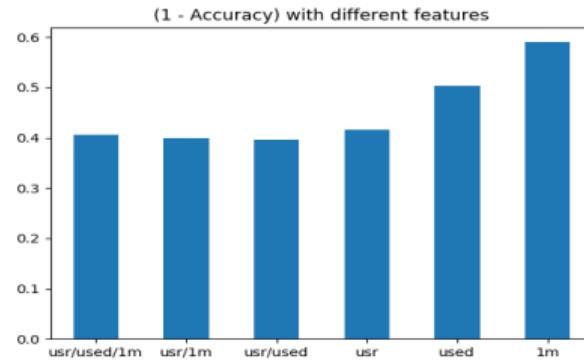
DNN - Multi Class

- Features: (usr, 1m)
- Resolution: 1min
- Loss converges, but...
- Evaluation accuracy achieved:
0.601
- Not good!



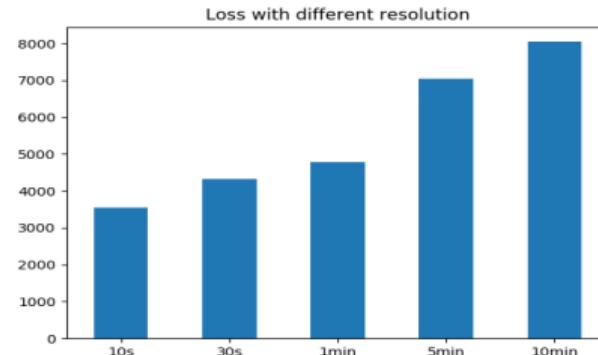
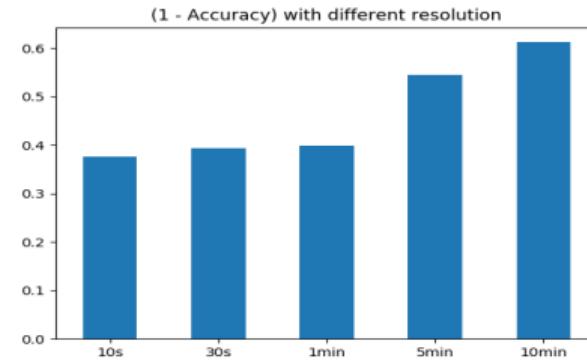
Multi Class - Different Features

- Try different combinations of features
- Primary metric: accuracy
- Aim for lower loss, caveat: overfitting
- Key:
 - **usr**: User CPU
 - **used**: Used Memory
 - **1m**: System Load - 1min Average
 - Data Resolution: **1min**
 - Source: TensorFlow evaluation output
- No real improvement
- Best accuracy achieved: **0.603**
- Adding Disk I/O or process data does not help either



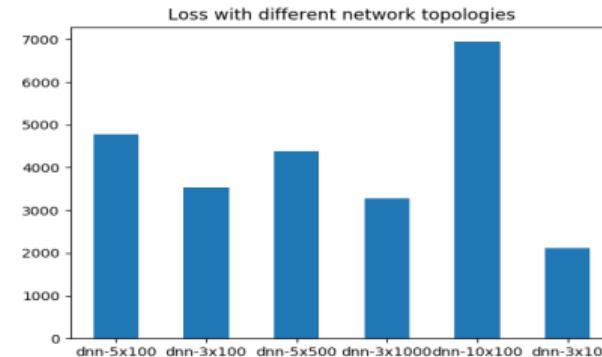
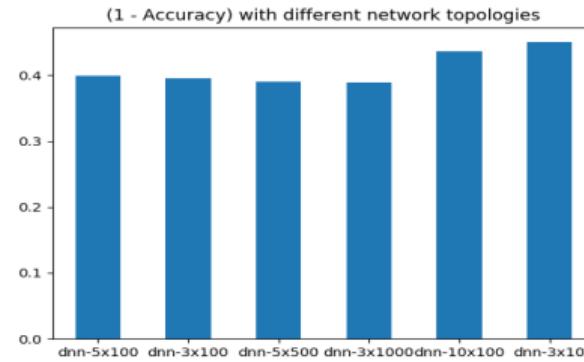
Multi Class - Changing Resolution

- Trying to change the data resolution
- Primary metric: accuracy
- Aim for lower loss, caveat: overfitting
- Key:
 - Original data frequency: 1s
 - x-axis: new sampling rate
 - Features: (**usr, 1m**)
 - Source: TensorFlow evaluation
- No real improvement
- Best accuracy achieved: **0.624**



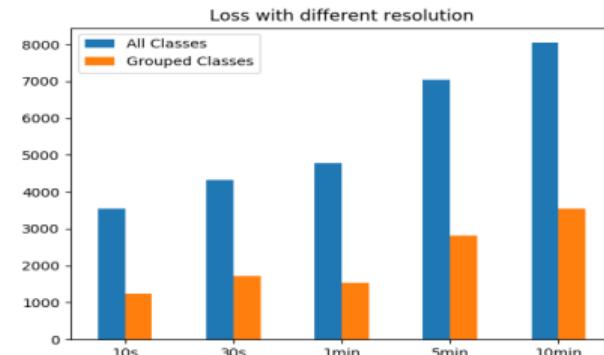
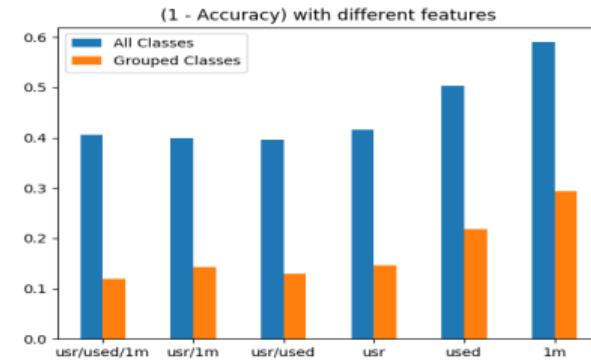
Multi Class - Network topology

- Trying to change the network depth
- Trying to change number of units per layer
- Primary metric: accuracy
- Aim for lower loss, caveat: overfitting
- Key:
 - x-axis: units and hidden layers
 - Features: (**usr**, **1m**)
 - Resolution: **1min**
 - Source: TensorFlow evaluation
- No real improvement
- Best accuracy achieved: **0.668**



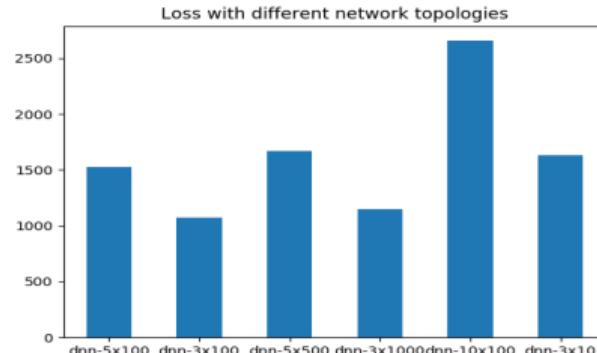
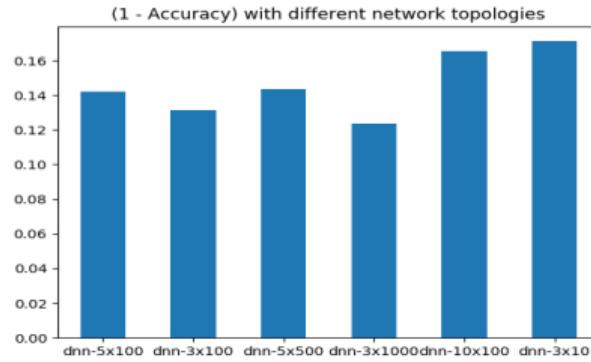
Multi Class - Reducing the number of classes

- Reducing the number of classes
 - Different regions from a Cloud Operator
 - Consider as a single class
 - New number of classes is **7**
- Experiments:
 - Train with different feature sets
 - Train with different resolutions
 - Source: TensorFlow evaluation
- Significant improvement!
- Best accuracy achieved: **0.902**
- What does that mean?



Multi Class - Tuning network topology

- Tuning network topology
- Experiments:
 - x-axis: units and hidden layers
 - Features: (**usr, 1m**)
 - Resolution: **1min**
- Some improvement
- Winner: 3x100. Accuracy: **0.925**



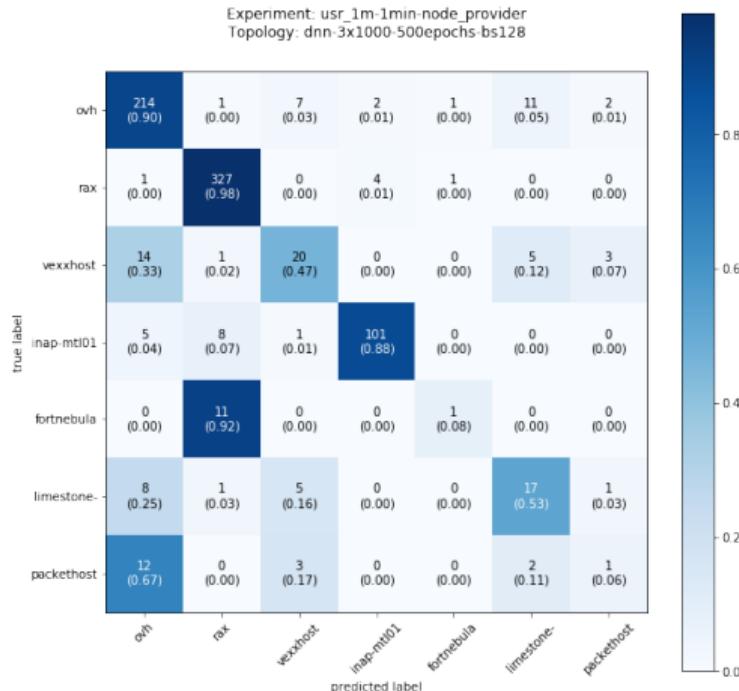
Multi Class - Metrics report

– Experiment:

- Dataset: 3955 examples split in 60% training, 20% dev and 20% test
- Features: User CPU and Load Avg
- Resolution: 1 minute
- Hyperparameters: 3 hidden layers, 1000 units each
- Cloud providers: 7

Metrics report for usr 1m and 1min

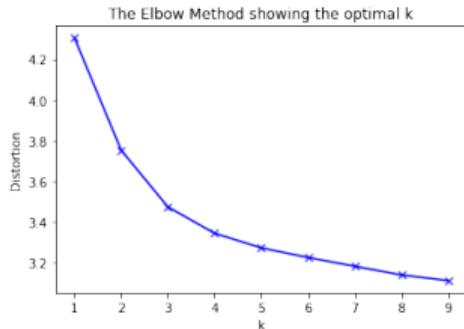
| provider | precision | recall | f1-score | support |
|------------|-----------|-------------------|-------------------|---------|
| rax | 0.94 | 0.98 | 0.96 | 333 |
| inap-mtl01 | 0.94 | 0.88 | 0.91 | 115 |
| ovh | 0.84 | 0.9 | 0.87 | 238 |
| vexxhost | 0.56 | 0.47 | 0.51 | 43 |
| limestone | 0.49 | 0.53 | 0.51 | 32 |
| fortnebula | 0.33 | $8 \cdot 10^{-2}$ | 0.13 | 12 |
| packethost | 0.14 | $6 \cdot 10^{-2}$ | $8 \cdot 10^{-2}$ | 18 |
| accuracy | | 0.86 | | 791 |



Multi Class - Clustering

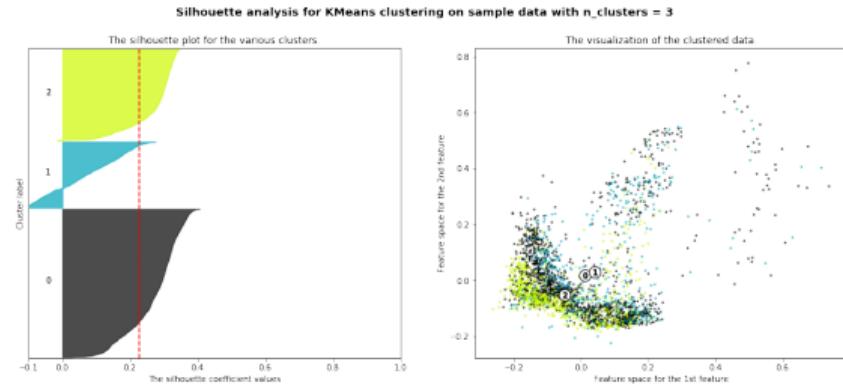
- Unsupervised learning on multiclass dataset
- Dataset: features CPU, Avg.load; resolution 10s
- Algorithm: kmeans
- Number of clusters based on elbow method and silhouette score
- Recommended number of clusters: 3 to 4
- Fits with the number of providers with most examples in dataset

| class | provider | examples |
|-------|------------|----------|
| 2 | rax | 1,208 |
| 6 | inap-mtl01 | 807 |
| 1 | ovh | 377 |
| 0 | vexxhost | 173 |
| 5 | limestone | 120 |
| 3 | packethost | 57 |
| 4 | fortnebula | 31 |

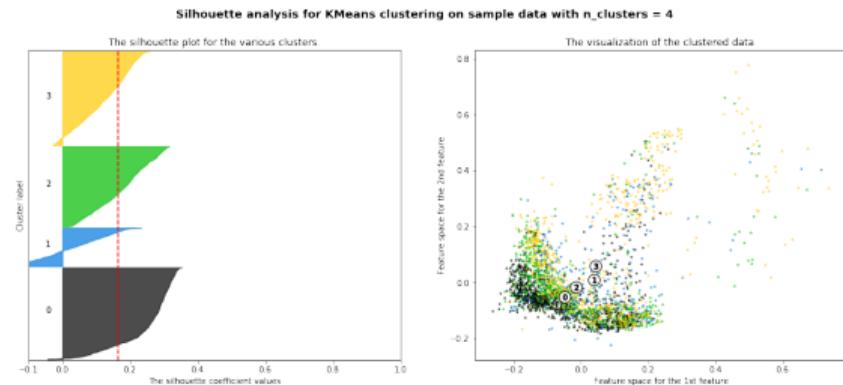


Multi Class - Clustering

| cluster | classes | example | pclass | pcluster |
|---------|---------|---------|--------|----------|
| 0 | 1 | 374 | 99.2 | 27.8 |
| | 2 | 795 | 65.8 | 59.2 |
| | 3 | 47 | 82.5 | 3.5 |
| 1 | 5 | 68 | 56.7 | 11.3 |
| 2 | 6 | 768 | 95.2 | 92.6 |



| cluster | class | example | pclass | pcluster |
|---------|-------|---------|--------|----------|
| 0 | 6 | 771 | 95.5 | 92.3 |
| 1 | 5 | 64 | 53.3 | 25.4 |
| 2 | 1 | 349 | 92.6 | 47.7 |
| 3 | 2 | 772 | 63.9 | 90.4 |



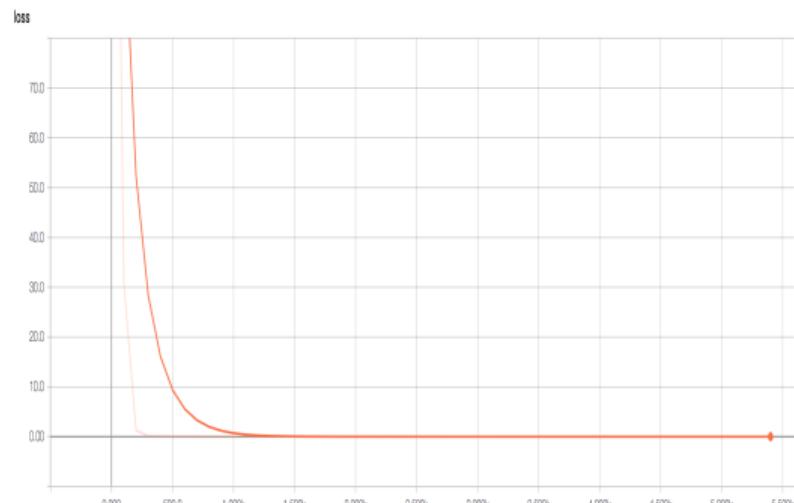
Multi Class - Changing test job

| metric | tempest-full | tempest-full-py3 |
|--------------|--------------|------------------|
| accuracy | 0.925 | 0.775 |
| average_loss | 0.978 | 3.271 |
| loss | 586.713 | 1,962.447 |

- Train with "tempest-full"
- Evaluating with "tempest-full-py3"
 - Similar setup, uses python3
 - It does not include swift and swift tests
 - 600 examples evaluation set
- Dataset and training setup:
 - Features: (usr, 1m)
 - Resolution: 1min
 - Same hyper-parameters (dnn-3x100)

Multi Class - Summary

- User CPU and 1min Load Avg
- Resolution: 1 minute is enough
- Hyperparameters: 3 hidden layers, 100 units each
- Reasonable accuracy: **0.925**
- Classification of new or less used cloud providers fails
- System data can be used to cluster heavily used cloud providers
- Clustering of CP inap is accurate, what is special in this CP?
- A trained model is not applicable to similar CI jobs



Training Loss - usr/1m, 1min, dnn3x100 - Source: TensorBoard

Conclusions

Conclusions

- Collect data
- Know your data
- Work with cloud tools
- Able to confirm that system load plays a role in failures
- Load profiles are consistent across regions in our cloud providers

Future Work

- Look at adapting techniques for new models with different data
- Human curated dataset for supervised training
- Research clustering techniques for unsupervised training
- Explore job portability
- Use the pass/fail data to detect degradation in CI Jobs

Questions?

- This talk: https://github.com/afrittoli/cimpl_talk
- CIML: <https://github.com/mtreinish/cimpl>

Thank You!