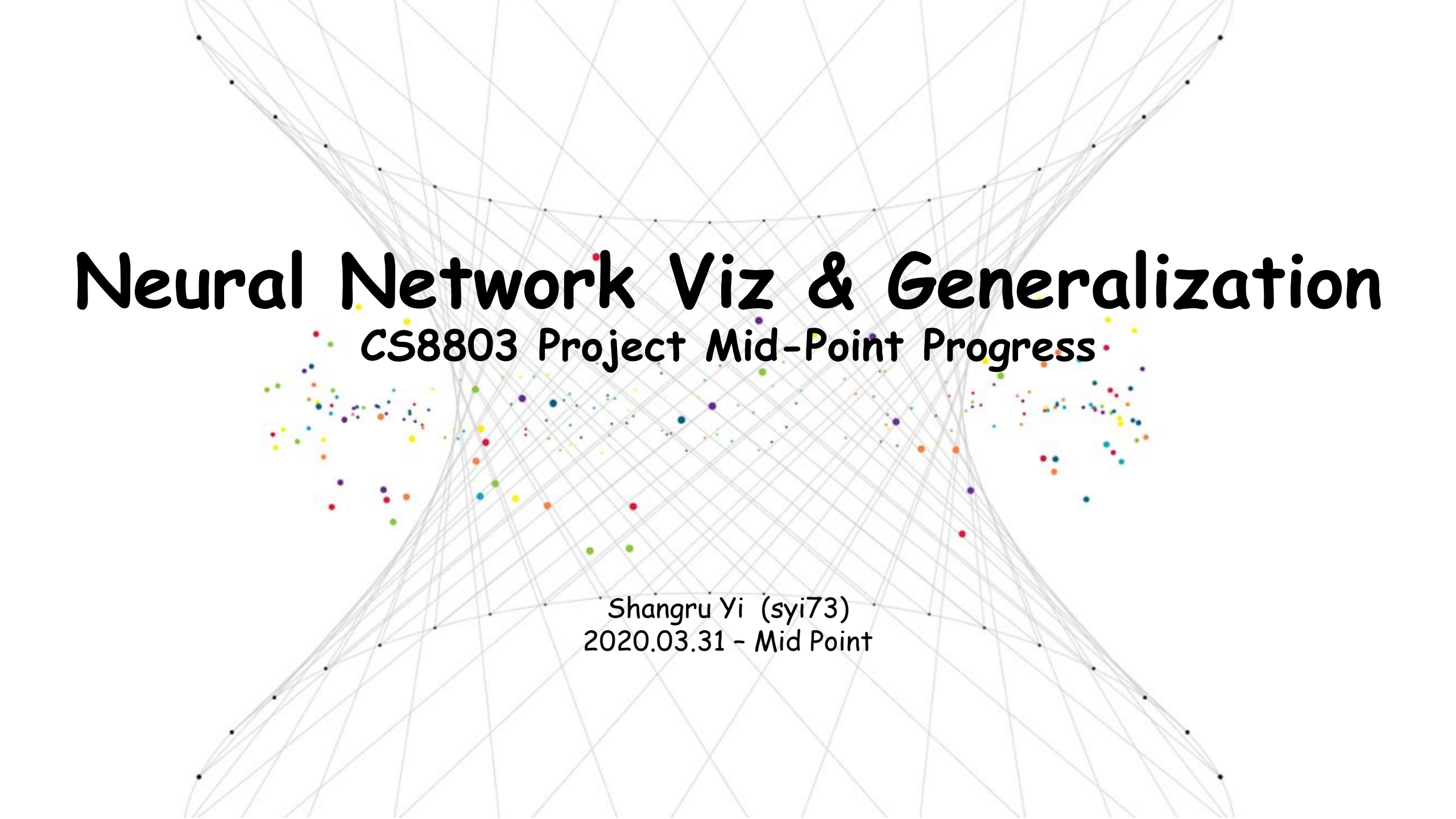
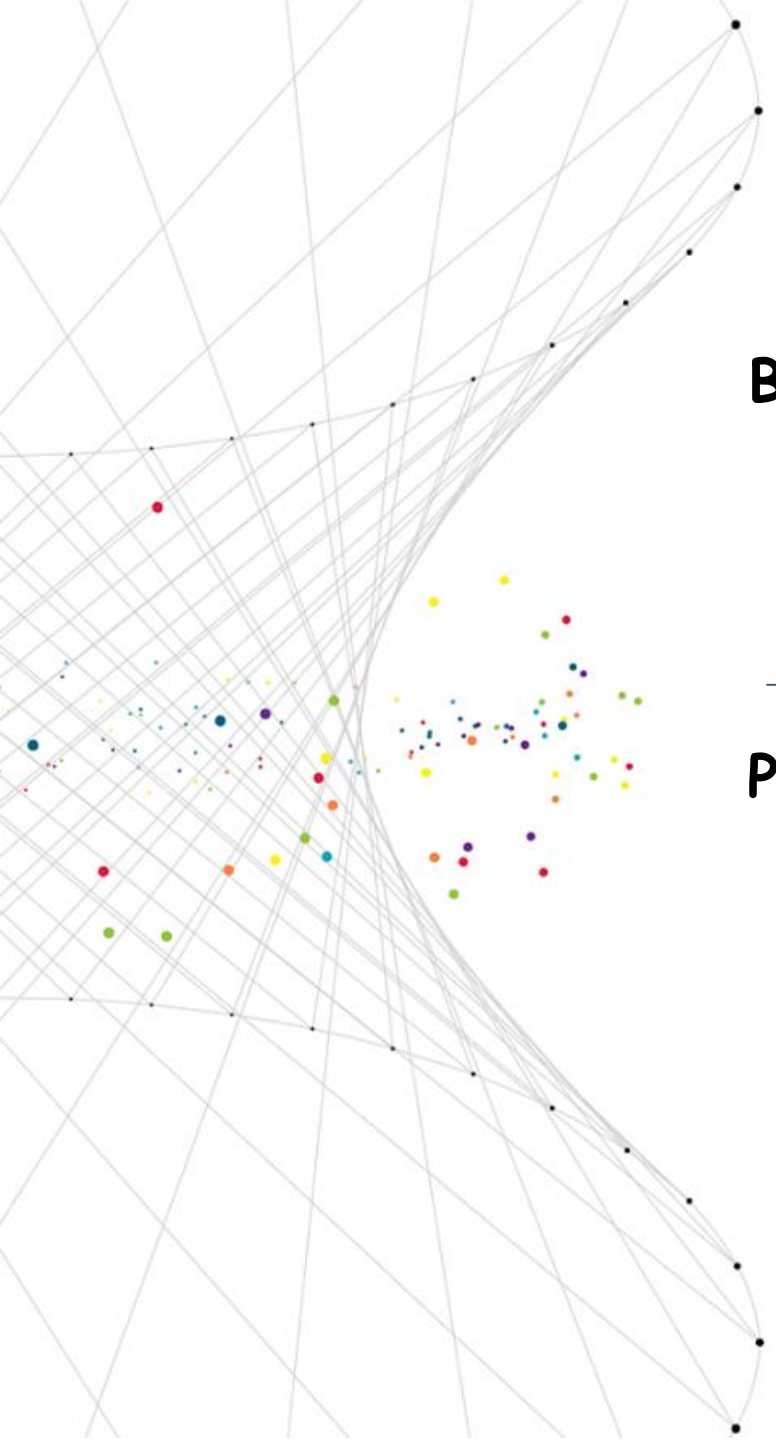


Neural Network Viz & Generalization

CS8803 Project Mid-Point Progress

Shangru Yi (syi73)

2020.03.31 – Mid Point



Background & Motivation

- NN Model Abstraction
- Heterogeneous Programming

State-of-Art

- TensorBoard
- Azure Machine Learning Studio
- Skyline (MLSys 20')

Proposed Solutions

- High-level Abstraction
- Formatted Model Representation
- State-of-art Combination

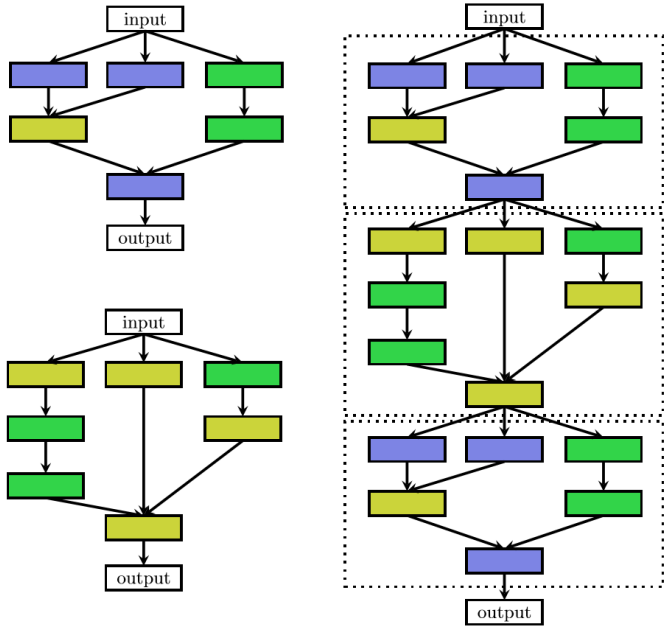
Project Timeline

- Implementation Timeline
- Status Update & On-going
- Possible Evaluation (Generality & Easiness)

Problem Introduction

"Motivated by hand-crafted architectures consisting of repeated motifs, lots of paper in Neural Architecture Search are proposed to search for such motifs, dubbed cells or blocks. The final architecture is built by stacking these cells in a predefined manner – with search space reduction, more data adaption, useful strategy."

- Neural Architecture Search: A Survey



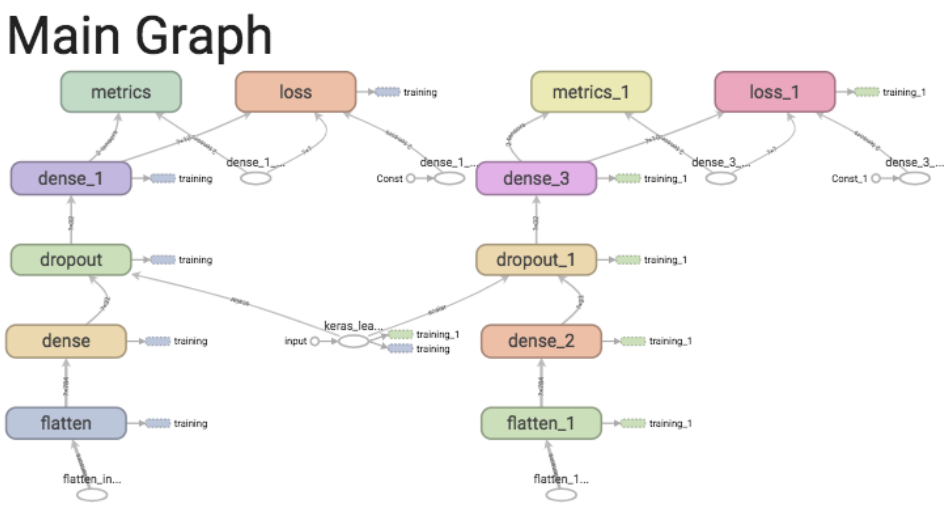
What can we improve?

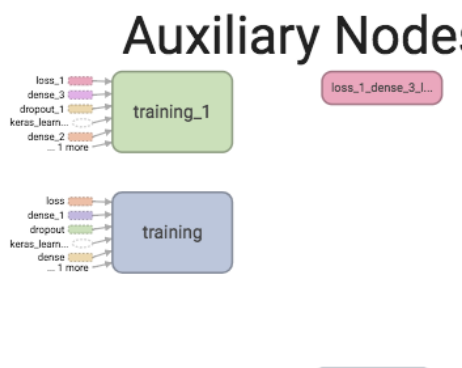
- Abstraction
- Code Reusage
- Readable Structed Programming
- Combination with Current Platforms

State-of-Art: TensorBoard

Machine learning invariably involves understanding key metrics such as loss and how they change as training progresses. These metrics can help to monitor the training, e.g. checking overfitting if the training is too long.

1. Graphs dashboard is a powerful tool for examining TensorFlow model, i.e. viewing a conceptual graph of model's structure and ensure it matches the intended design.
2. Scalars Dashboard allows one to visualize these metrics using a simple API with little effort.

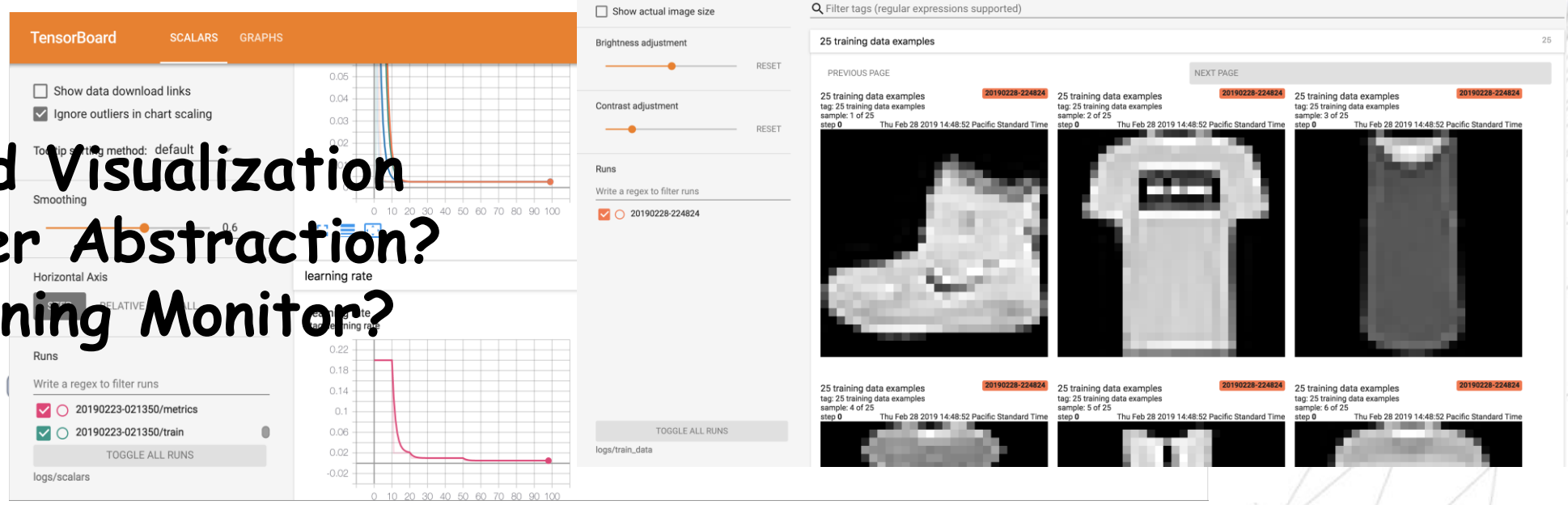




State-of-Art: TensorBoard

Machine learning invariably involves understanding key metrics such as loss and how they change as training progresses. These metrics can help to monitor the training, e.g. checking overfitting if the training is too long.

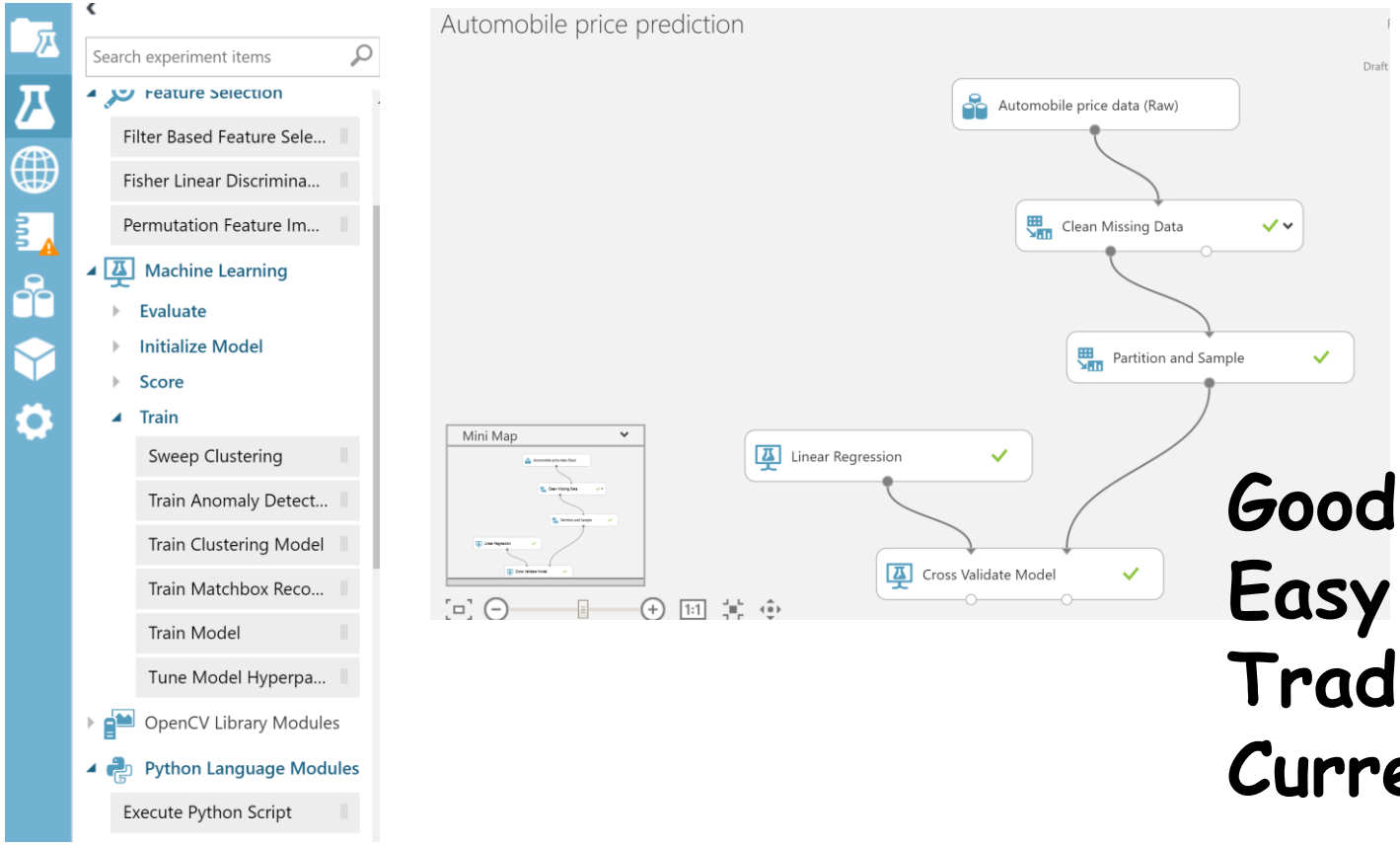
1. Graphs dashboard is a powerful tool for examining TensorFlow model, i.e. viewing a conceptual graph of model's structure and ensure it matches the intended design.
2. Scalars Dashboard allows one to visualize these metrics using a simple API with little effort.



Good Visualization
Layer Abstraction?
Training Monitor?

State-of-Art: Azure Machine Learning Studio

Azure Machine Learning Studio is a GUI-based integrated development environment for constructing and operationalizing Machine Learning workflow on Azure. It is a collaborative, drag-and-drop tool that one can use to build, test, and deploy predictive analytics solutions on various data.



Good Generalization
Easy Drag and Place
Traditional ML?
Current Platforms?

State-of-Art:Skyline (MLSys 20')

Skyline is a tool used with Atom to profile, visualize, and debug the training performance of PyTorch neural networks.

```python
import torch.nn as nn


class Model(nn.Module):
    def __init__(self):
        super().__init__()
        self.conv = nn.Conv2d(in_channels=3, out_channels=6, kernel_size=3)
        self.linear = nn.Linear(in_features=387096, out_features=10)

    def forward(self, input):
        out = self.conv(input)
        return self.linear(out.view(-1, 387096))
```

- `skyline_model_provider`
- `skyline_input_provider`
- `skyline_iteration_provider`

```python
import torch
import torch.nn as nn

from my_project.model import Model


class ModelWithLoss(nn.Module):
    def __init__(self):
        super().__init__()
        self.model = Model()
        self.loss_fn = nn.CrossEntropyLoss()

    def forward(self, input, target):
        output = self.model(input)
        return self.loss_fn(output, target)


def skyline_model_provider():
    # Return a GPU-based instance of our model (that returns a loss)
    return ModelWithLoss().cuda()


def skyline_input_provider(batch_size=32):
    # Return GPU-based inputs for our model
    return (
        torch.randn(batch_size, 3, 256, 256).cuda(),
        torch.randint(low=0, high=9, size=(batch_size,)).cuda(),
    )
def skyline_iteration_provider(model):
    # Return a callable that executes one training iteration
    optimizer = torch.optim.SGD(model.parameters(), lr=1e-3)
    def iteration(*inputs):
        optimizer.zero_grad()
        out = model(*inputs)
        out.backward()
        optimizer.step()
    return iteration
```

Binding with PyTorch
On-training Metrics Vis?
Model Structure Vis?

What is Expected?

State-of-art Summary

- Low level generalization & abstraction
- On-training metrics monitoring - visualization
- Not combination with current mainstream platforms
- Traditional machine learning (Not NN)
- Lack of model structure visualization

Falsifiable Hypothesis

- Visualized programming for neural network and user interactions are achievable.
- A unified representation format for cross-platform is possible
- Above-layer abstraction can speedup model construction and pattern re-usage.
- Higher level debugging with possible fine-grained hardware assignment can be solved in an easy way with visualization.
- Combination with current in-progress training visualization can bring more convenience.

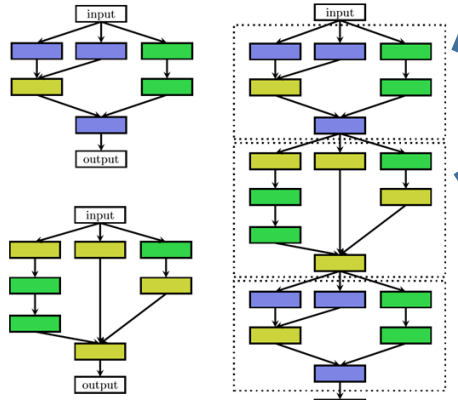
What is Expected?

State-of-art Summary

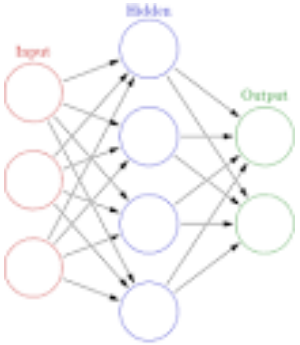
- Low level generalization & abstraction
- On-training metrics monitoring - visualization
- Not combination with current mainstream platforms
- Traditional machine learning (Not NN)
- Lack of model structure visualization

Keras

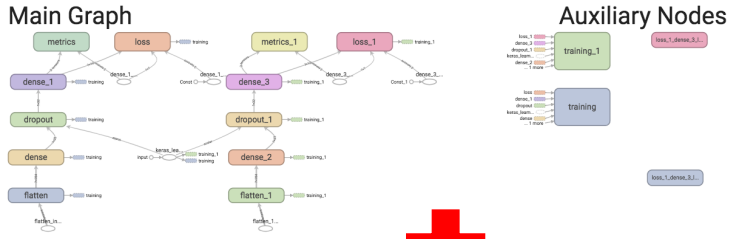
PYTORCH



What we expect?

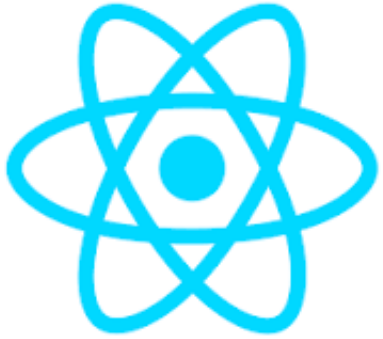


- Visualized programming
- Unified representation format
- Above-layer abstraction
- Higher level debugging
- Hardware assignment
- Binding with current platforms – with possible adjustments
- Cross Platforms



Skyline

Proposed Solutions



PYTORCH

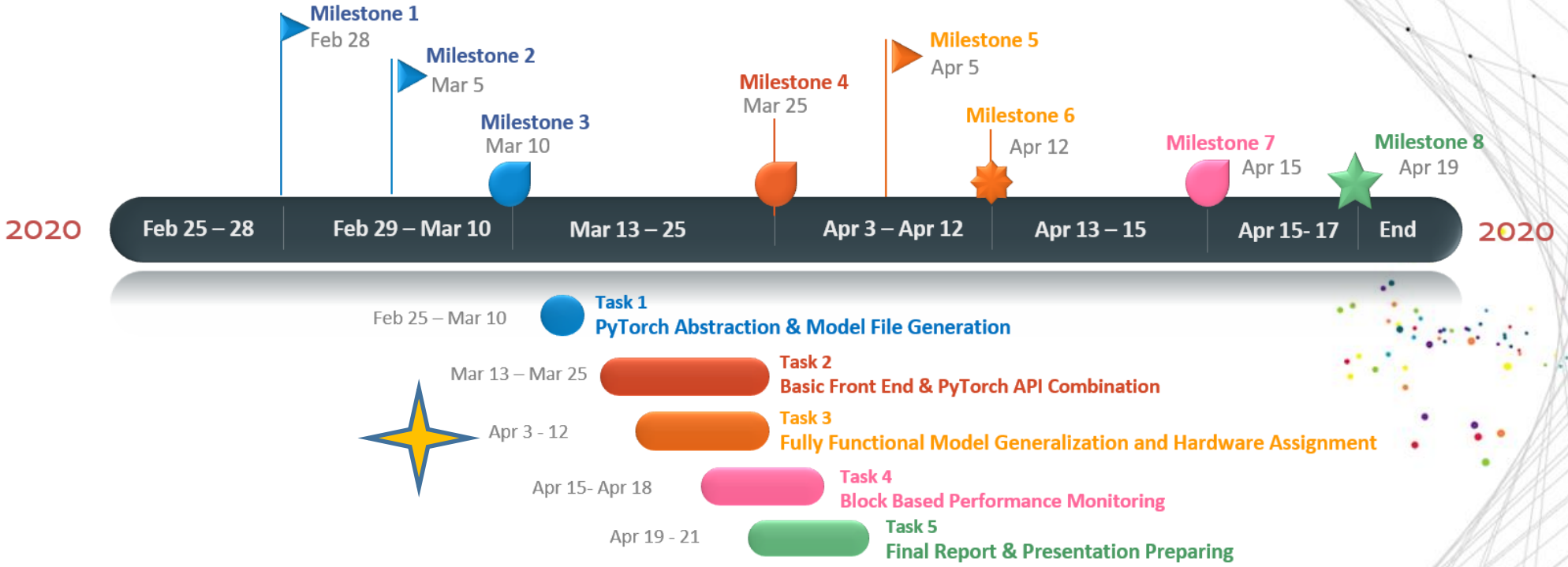
Skyline

- React + D3
  - Front end visualization
  - User interactions
  - Input & output
  - Bind PyTorch function and argument
  - Parse parameter to formatted model file and readable programming file
  - Parse exported model files for further changes
- PyTorch + Skyline
  - On-training monitoring
  - Parsing generated model
  - API provider
  - Modify Skyline for block supports (if we can do that, otherwise use torch.cuda)

Project Timeline

- Previous mainly for Tensorflow and Keras

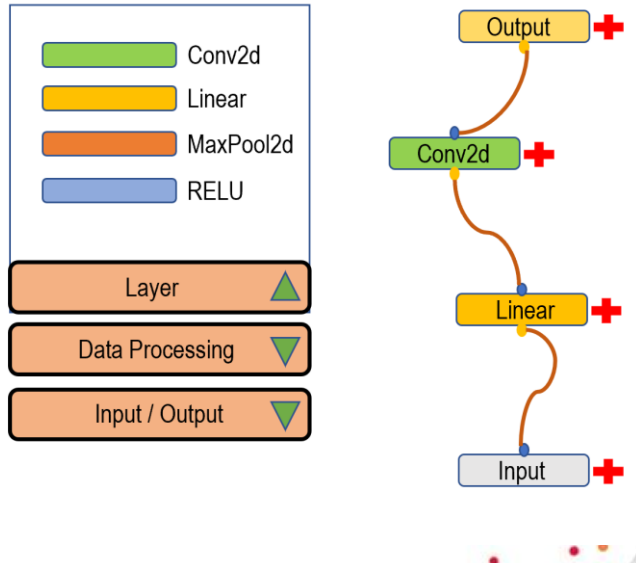
Project Timeline

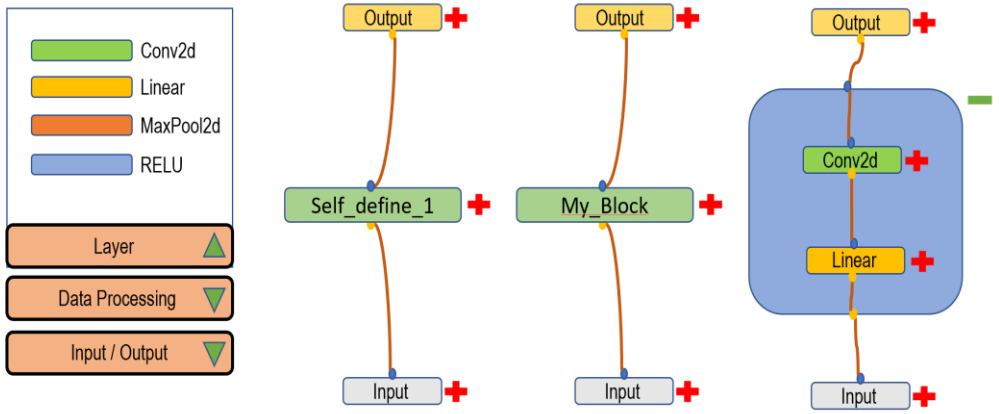


- Not Tensorflow
- Add possible supports for Skyline or …

Status Update & On-going

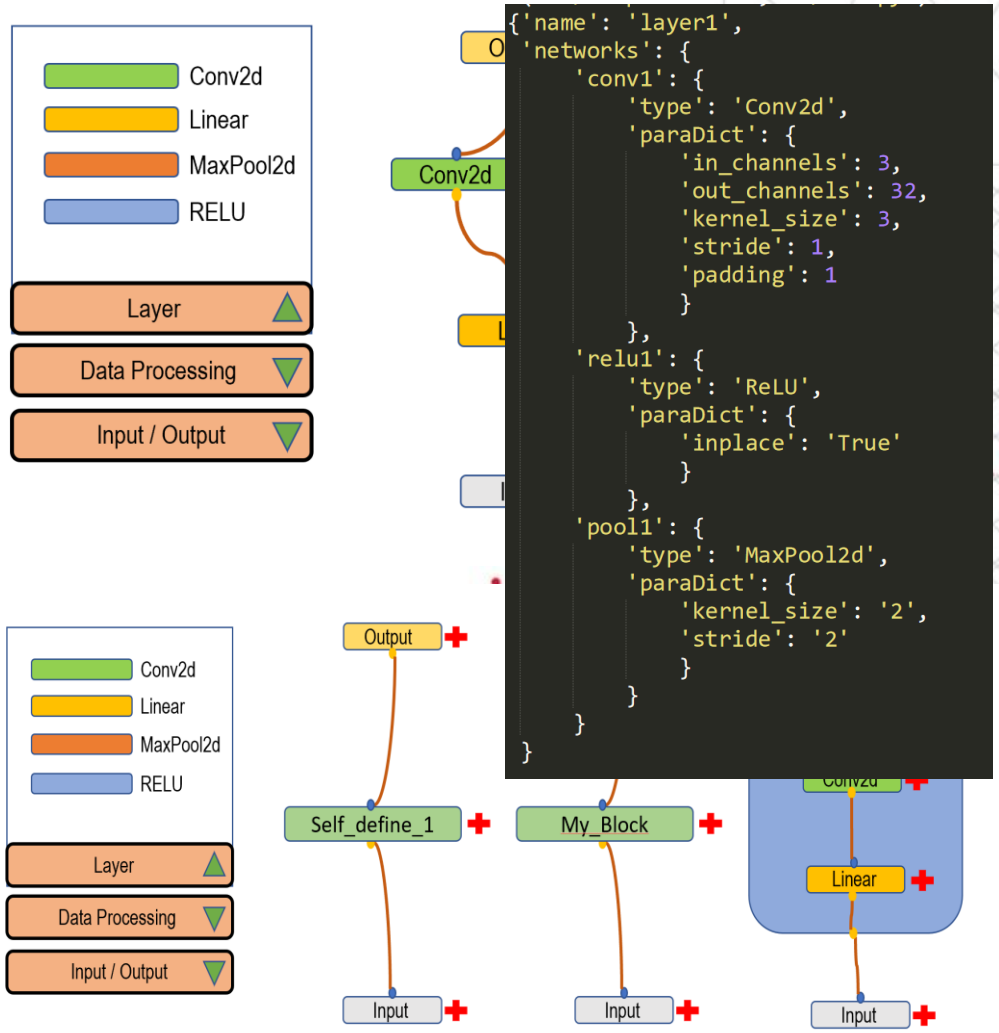
- Front end visualization (basic)
- User interactions – layer construction
- Input & output – finish address binding
- Bind PyTorch function and argument – refining
- Parse parameter to formatted model file and readable programming file – ongoing
- On-training monitoring (Skyline)
- Parsing generated model – JSON
- Modify Skyline for blocks monitoring – reading ongoing





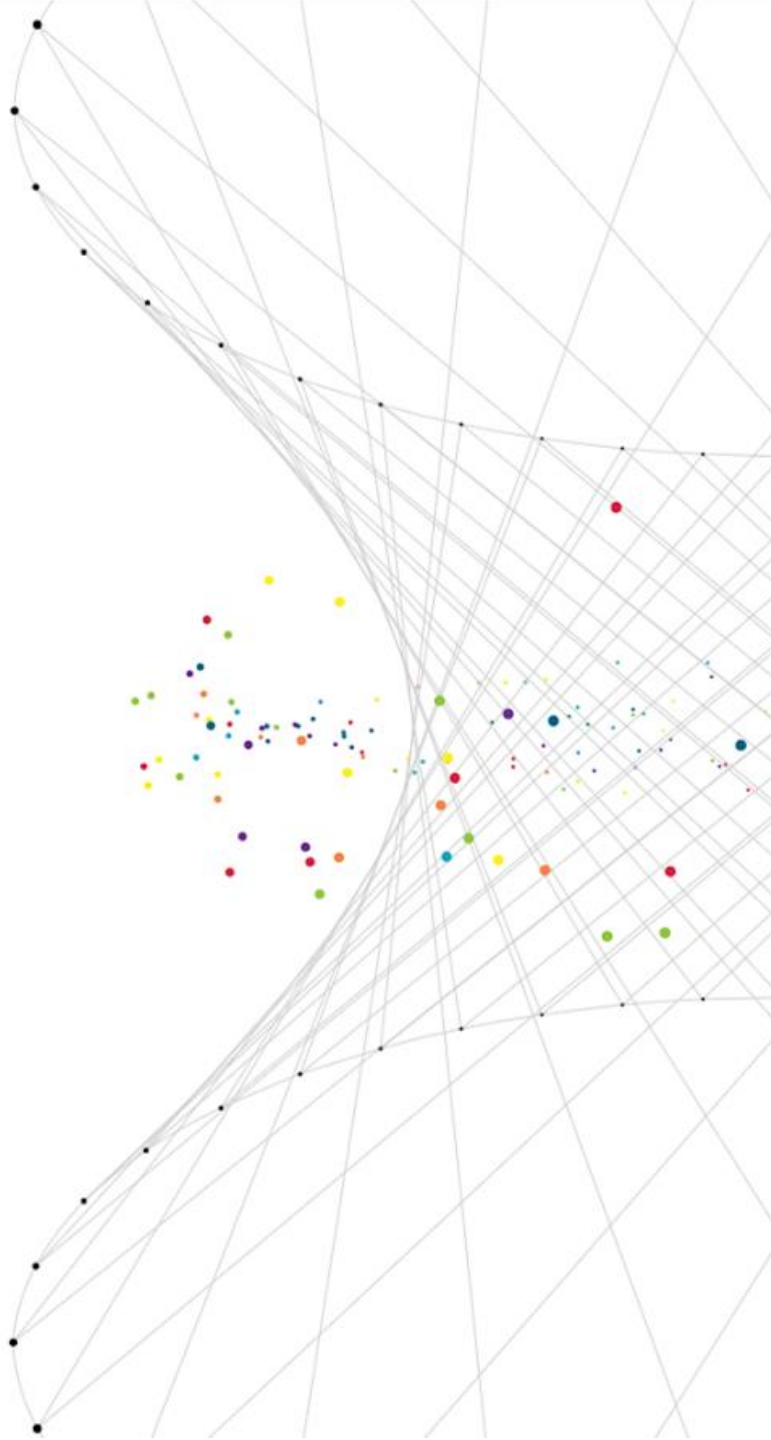
Status Update & On-going

- Front end visualization (basic)
- User interactions – layer construction
- Input & output – finish address binding
- Bind PyTorch function and argument – refining
- Parse parameter to formatted model file and readable programming file – ongoing
- On-training monitoring (Skyline)
- Parsing generated model – JSON
- Modify Skyline for blocks monitoring – reading ongoing



Possible Evaluations

- Network structure visualization
- Blocks performance monitoring (not whole model)
- Readability of generated programming file
- User interactions
- ……



Reference:

Azure machine learning studio. https://docs.microsoft.com/en-us/azure/machine-learning/studio. Accessed: 2020-03-25

Tensorboard. https://www.tensorflow.org/tensorboard/graphs. Accessed: 2020-03-25.

Skyline. https://github.com/geoffxy/skyline-atom. Accessed: 2020-03-25.

Paszke, Adam, et al. "PyTorch: An imperative style, high-performance deep learning library." Advances in Neural Information Processing Systems. 2019.

Elsken, Thomas, Jan Hendrik Metzen, and Frank Hutter. "Neural architecture search: A survey." arXiv preprint arXiv:1808.05377 (2018).