VIETNAM
WEB
SUMMIT

Developer Circles

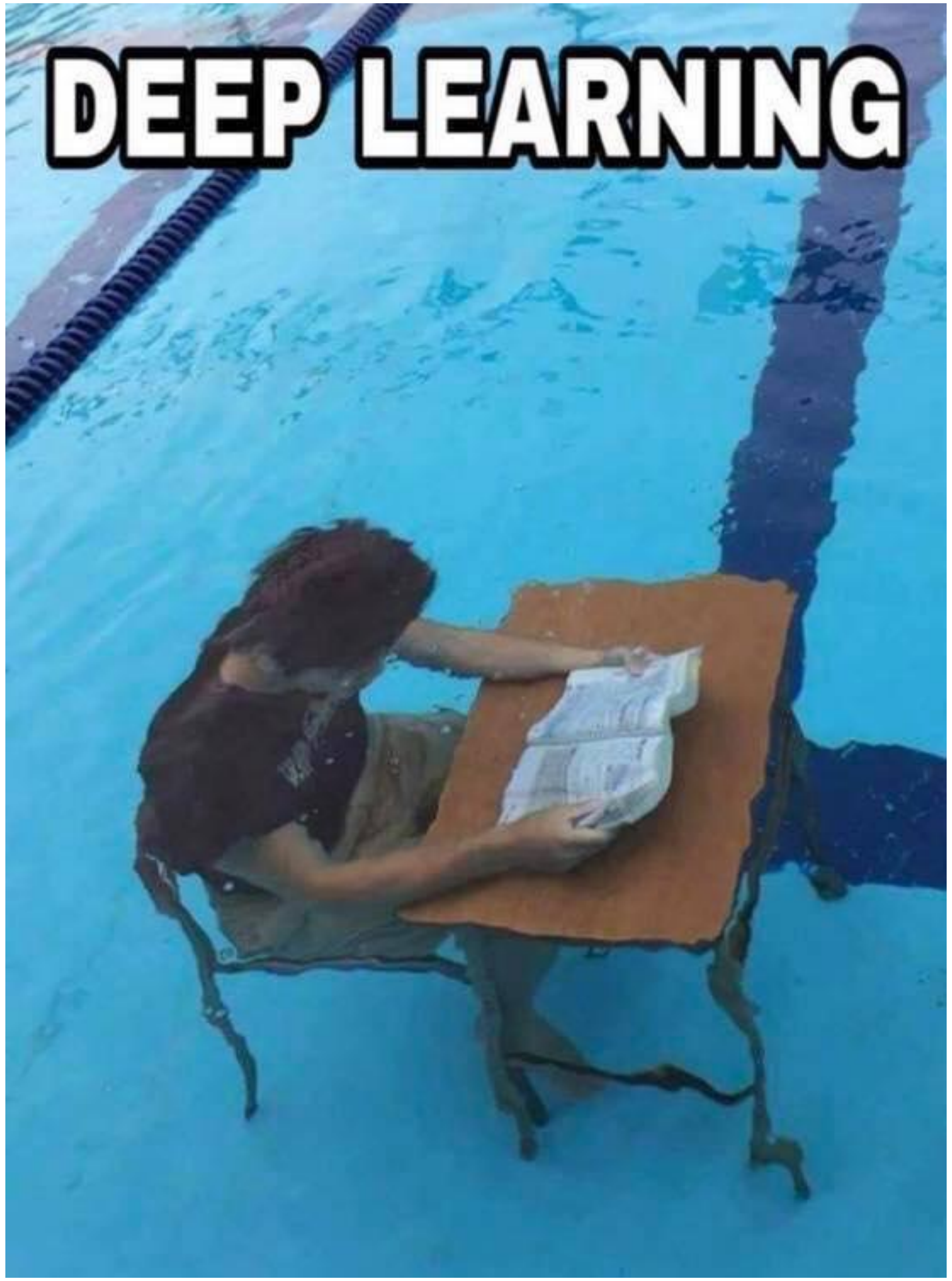
from facebook

Deep learning with PyTorch & applications

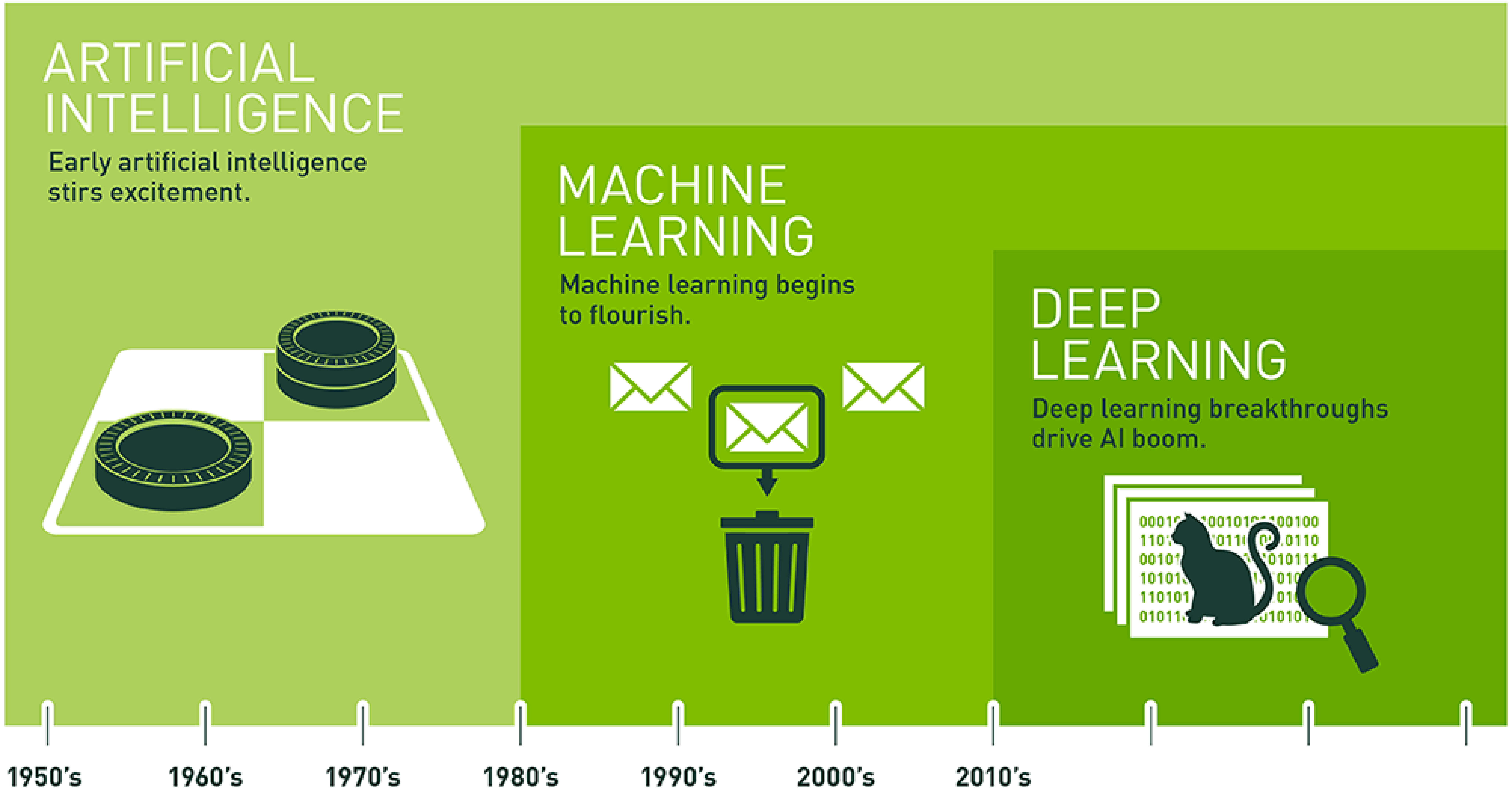
Le Thanh Hung

Community Leader
Facebook Developer Circle Hanoi

facebook for developers


DEEP LEARNING

VIETNAM WEB SUMMIT



Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

Berkeley AI Research Lab, UC Berkeley

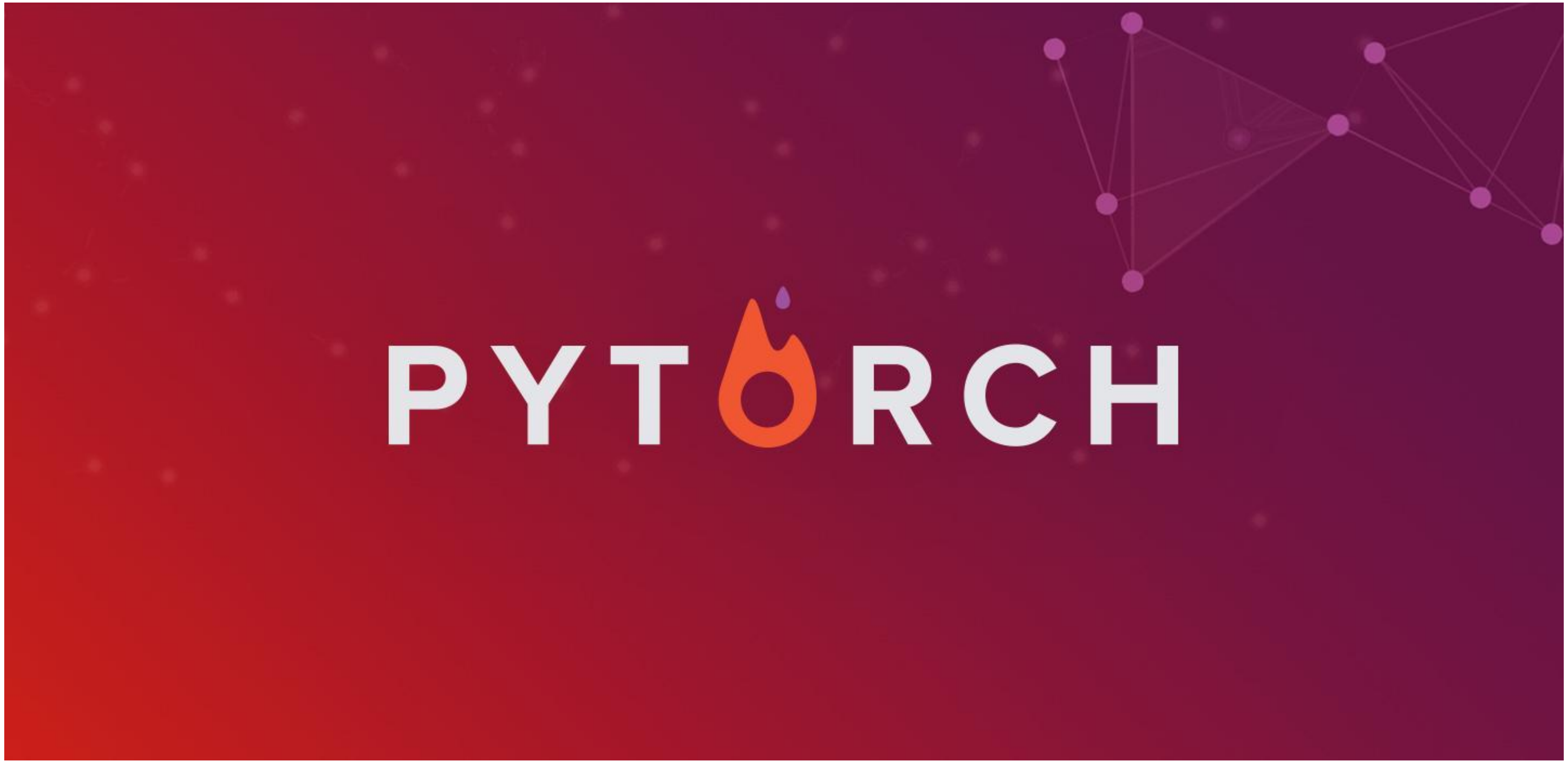
VIETNAM WEB SUMMIT

PYTORCH

- Open source DL Framework
- Only a year ½ old!!!
- Thanks Facebook AI research group!!!
- Tons of Community ❤

CycleGAN and pix2pix in PyTorch

Jun-Yan Zhu and Taesung Park.


PYTORCH

What is PyTorch?

VIETNAM WEB SUMMIT

- automatic differentiation engine
- **Ndarray library with GPU support**
- gradient based optimization package
- Utilities (data loading, etc.)
- Distributed training
- Production-ready C++ runtime

Deep Learning

Reinforcement Learning

Numpy alternative

ndarray library

- np.ndarray <-> torch.Tensor
- 200+ operations, similar to numpy
- very fast acceleration on NVIDIA GPUs

VIETNAM WEB SUMMIT

Numpy

```python
# -*- coding: utf-8 -*-
import numpy as np

# N is batch size; D_in is input dimension;
# H is hidden dimension; D_out is output dimension.
N, D_in, H, D_out = 64, 1000, 100, 10

# Create random input and output data
x = np.random.randn(N, D_in)
y = np.random.randn(N, D_out)

# Randomly initialize weights
w1 = np.random.randn(D_in, H)
w2 = np.random.randn(H, D_out)

learning_rate = 1e-6
for t in range(500):
    # Forward pass: compute predicted y
    h = x.dot(w1)
    h_relu = np.maximum(h, 0)
    y_pred = h_relu.dot(w2)

    # Compute and print loss
    loss = np.square(y_pred - y).sum()
    print(t, loss)

    # Backprop to compute gradients of w1 and w2 with respect to loss
    grad_y_pred = 2.0 * (y_pred - y)
    grad_w2 = h_relu.T.dot(grad_y_pred)
    grad_h_relu = grad_y_pred.dot(w2.T)
    grad_h = grad_h_relu.copy()
    grad_h[h < 0] = 0
    grad_w1 = x.T.dot(grad_h)

    # Update weights
    w1 -= learning_rate * grad_w1
    w2 -= learning_rate * grad_w2
```

PyTorch

```python
import torch


dtype = torch.FloatTensor
# dtype = torch.cuda.FloatTensor # Uncomment this to run on GPU

# N is batch size; D_in is input dimension;
# H is hidden dimension; D_out is output dimension.
N, D_in, H, D_out = 64, 1000, 100, 10

# Create random input and output data
x = torch.randn(N, D_in).type(dtype)
y = torch.randn(N, D_out).type(dtype)

# Randomly initialize weights
w1 = torch.randn(D_in, H).type(dtype)
w2 = torch.randn(H, D_out).type(dtype)

learning_rate = 1e-6
for t in range(500):
    # Forward pass: compute predicted y
    h = x.mm(w1)
    h_relu = h.clamp(min=0)
    y_pred = h_relu.mm(w2)

    # Compute and print loss
    loss = (y_pred - y).pow(2).sum()
    print(t, loss)

    # Backprop to compute gradients of w1 and w2 with respect to loss
    grad_y_pred = 2.0 * (y_pred - y)
    grad_w2 = h_relu.t().mm(grad_y_pred)
    grad_h_relu = grad_y_pred.mm(w2.t())
    grad_h = grad_h_relu.clone()
    grad_h[h < 0] = 0
    grad_w1 = x.t().mm(grad_h)

    # Update weights using gradient descent
    w1 -= learning_rate * grad_w1
    w2 -= learning_rate * grad_w2
```

Neural Networks

```
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(1, 10, kernel_size=5)
        self.conv2 = nn.Conv2d(10, 20, kernel_size=5)
        self.conv2_drop = nn.Dropout2d()
        self.fc1 = nn.Linear(320, 50)
        self.fc2 = nn.Linear(50, 10)

    def forward(self, x):
        x = F.relu(F.max_pool2d(self.conv1(x), 2))
        x = F.relu(F.max_pool2d(self.conv2_drop(self.conv2(x)), 2))
        x = x.view(-1, 320)
        x = F.relu(self.fc1(x))
        x = F.dropout(x, training=self.training)
        x = self.fc2(x)
        return F.log_softmax(x)

model = Net()
input = Variable(torch.randn(10, 20))
output = model(input)
```

Debugging

- PyTorch is a Python extension
- Use your favorite Python debugger
- Use the most popular debugger:

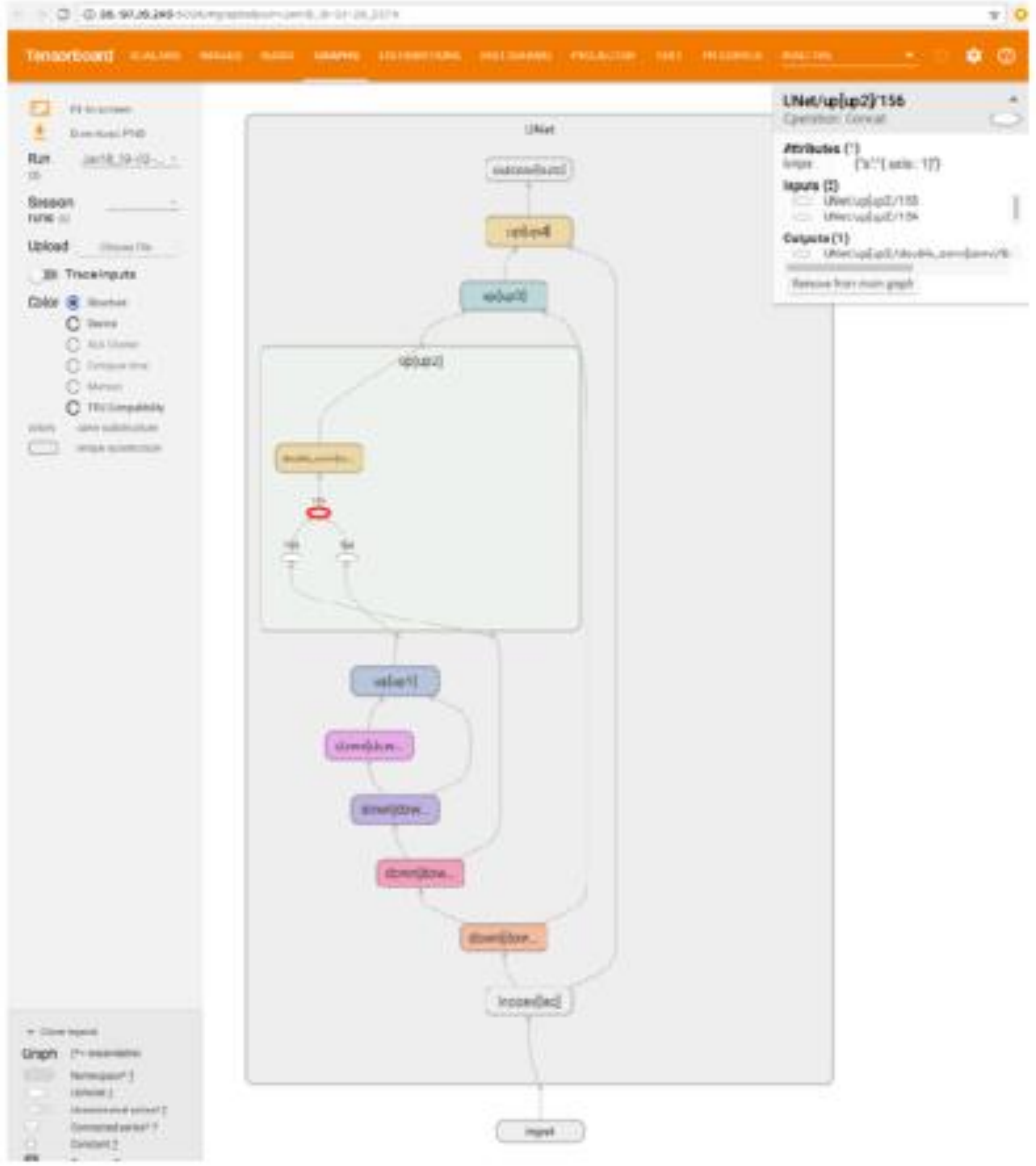
```
print(foo)
```

Visualization

VIETNAM WEB SUMMIT

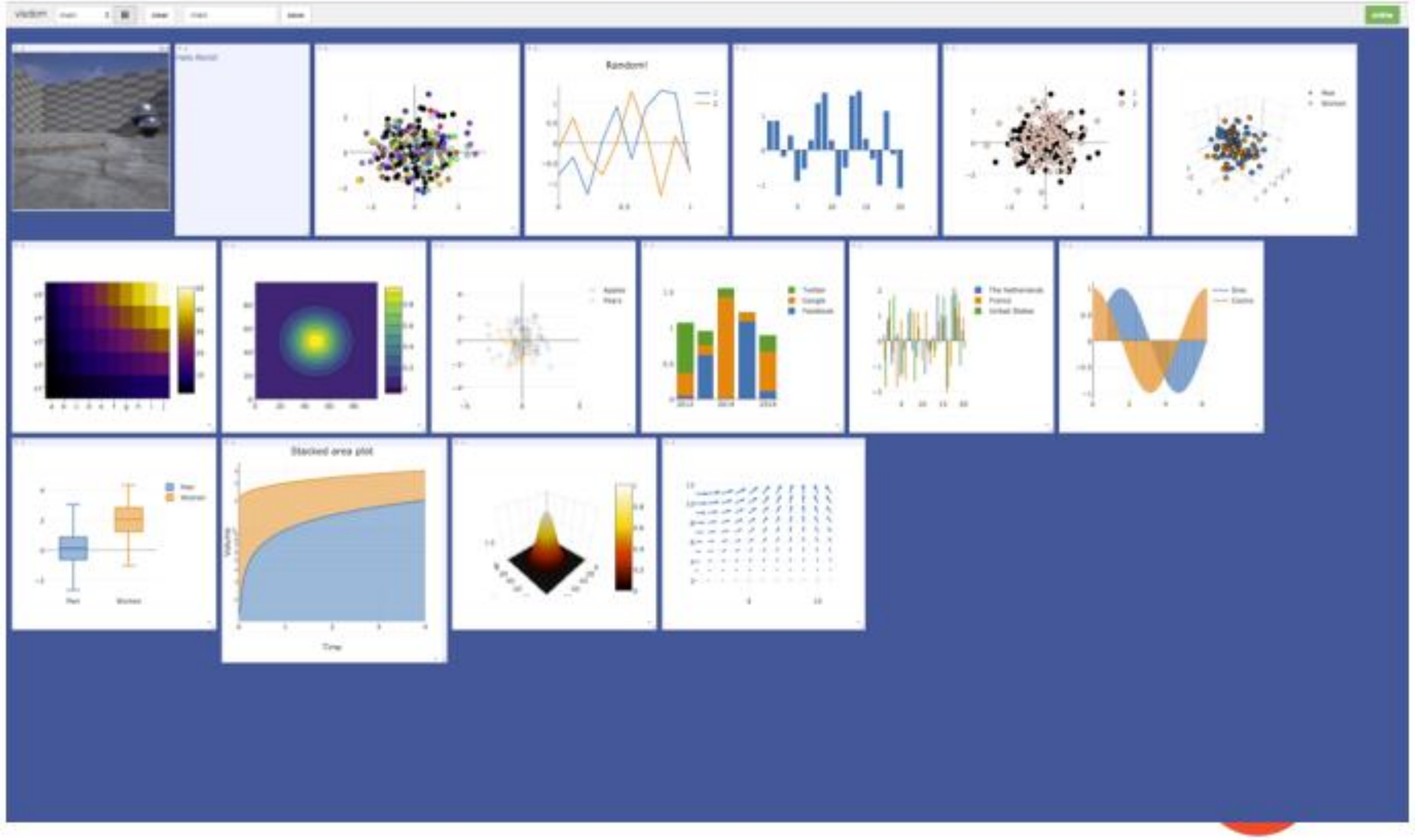
TensorBoard-PyTorch

https://github.com/lanpa/tensorboard-pytorch



Visdom

https://github.com/facebookresearch/visdom



Work items in practice

VIETNAM WEB SUMMIT

Writing Dataset loaders	Building models	Implementing Training loop	Checkpointing models

Python + PyTorch - an environment to do all of this

Interfacing with environments	Building optimizers	Dealing with GPUs	Building Baselines

+ tools for making your model production-ready

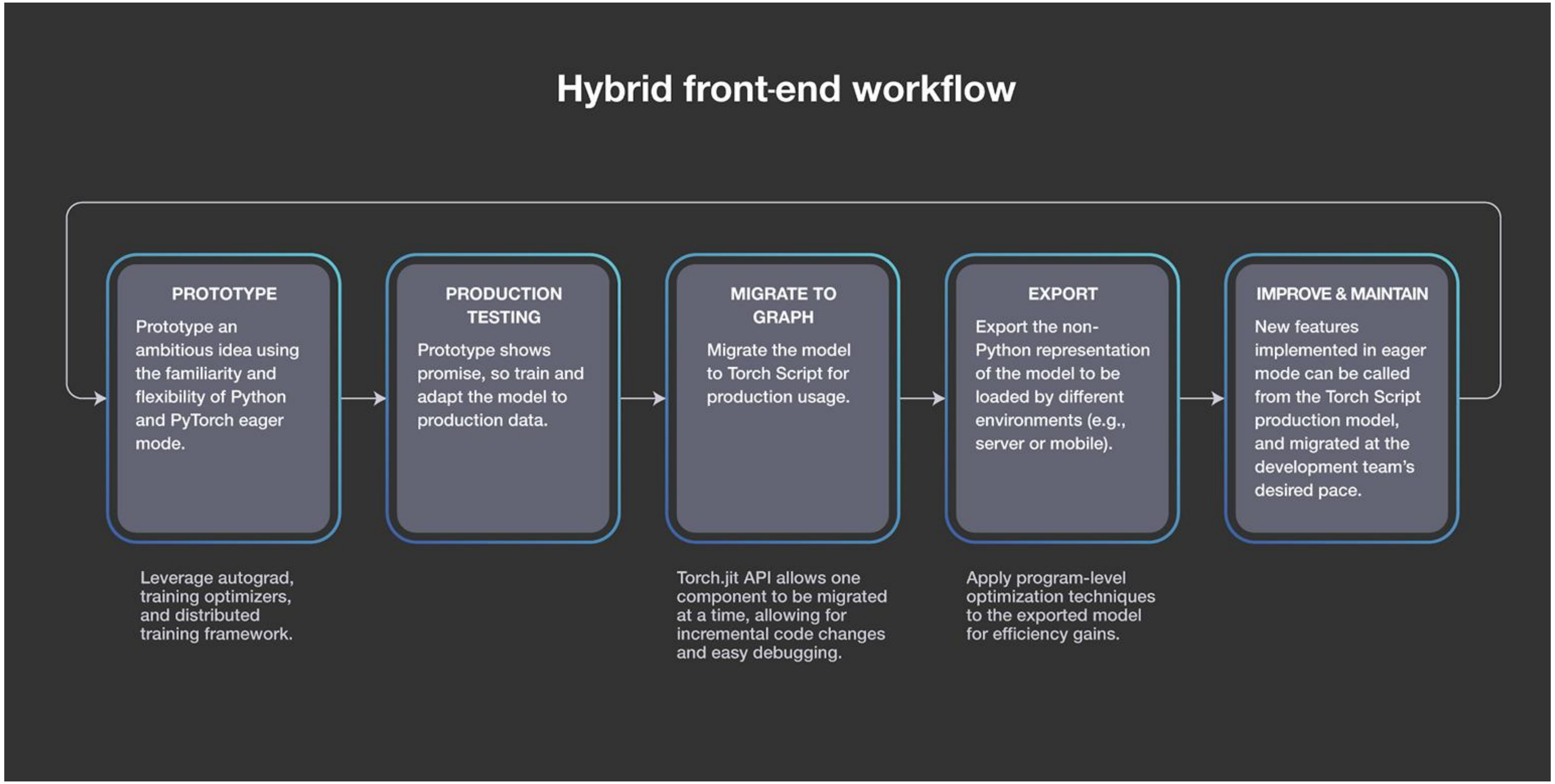
Distributed training	Performance optimization	Deployment for inference

DEVELOPER EFFICIENCY

- Debuggability
- Interactivity
- Simplicity
- Intuitiveness

INFRASTRUCTURE EFFICIENCY

- Efficiency
- Reliability
- Scalability
- Cross Platform



Hybrid front-end workflow
PROTOTYPE
Prototype an ambitious idea using the familiarity and flexibility of Python and PyTorch eager mode.
PRODUCTION TESTING
Prototype shows promise, so train and adapt the model to production data.
MIGRATE TO GRAPH
Migrate the model to Torch Script for production usage.
EXPORT
Export the non-Python representation of the model to be loaded by different environments (e.g., server or mobile).
IMPROVE & MAINTAIN
New features implemented in eager mode can be called from the Torch Script production model, and migrated at the development team's desired pace.
Leverage autograd, training optimizers, and distributed training framework.
Torch.jit API allows one component to be migrated at a time, allowing for incremental code changes and easy debugging.
Apply program-level optimization techniques to the exported model for efficiency gains.

VIETNAM WEB SUMMIT

PYTORCH ECOSYSTEM

Tools, Libraries and Datasets to enable cutting-edge AI development

PyTorchVision

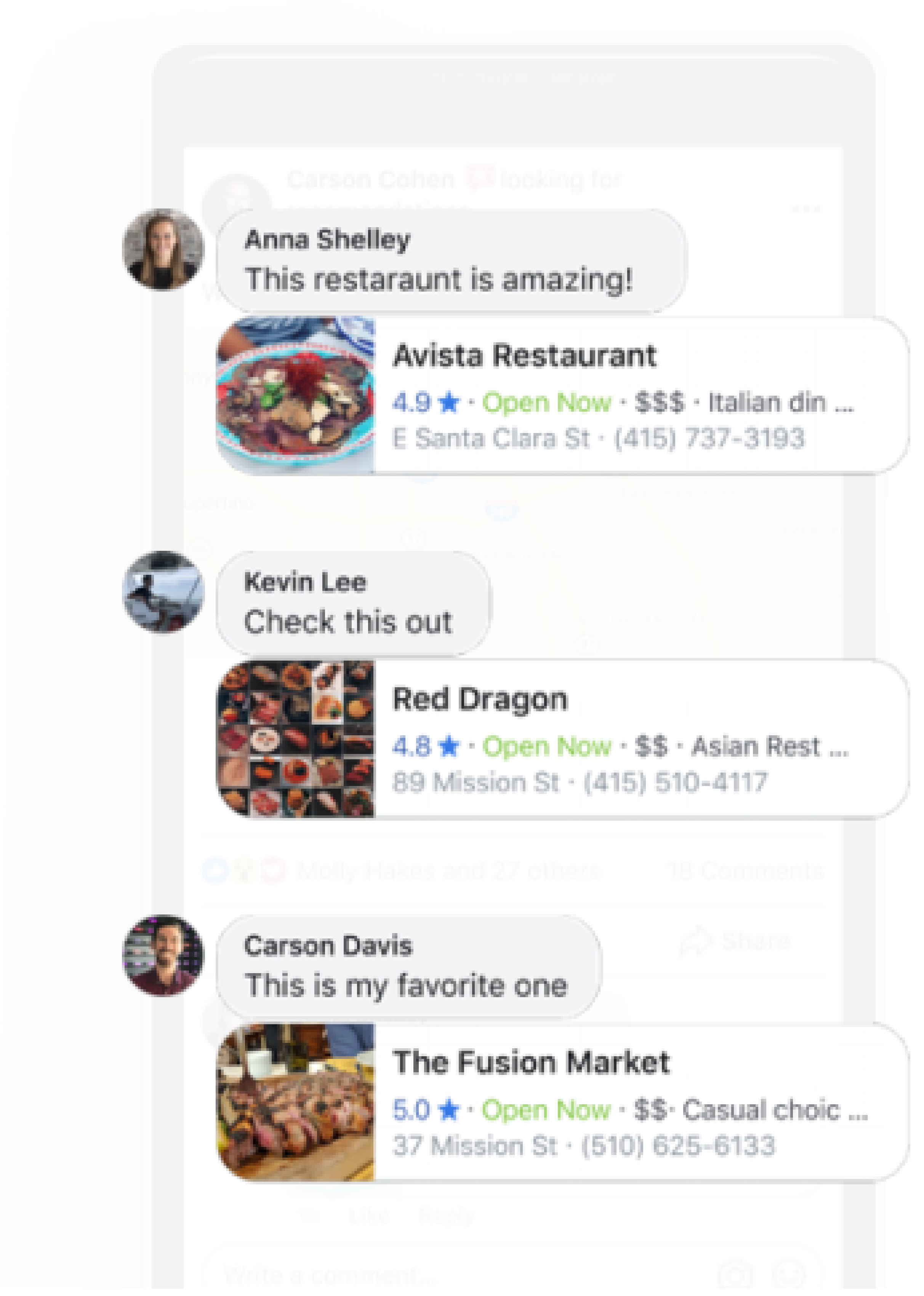
PyTorchReasoning

PyTorchLanguage

PyTorchSpeech

PyTorch Tools

SOCIAL RECOMMENDATIONS

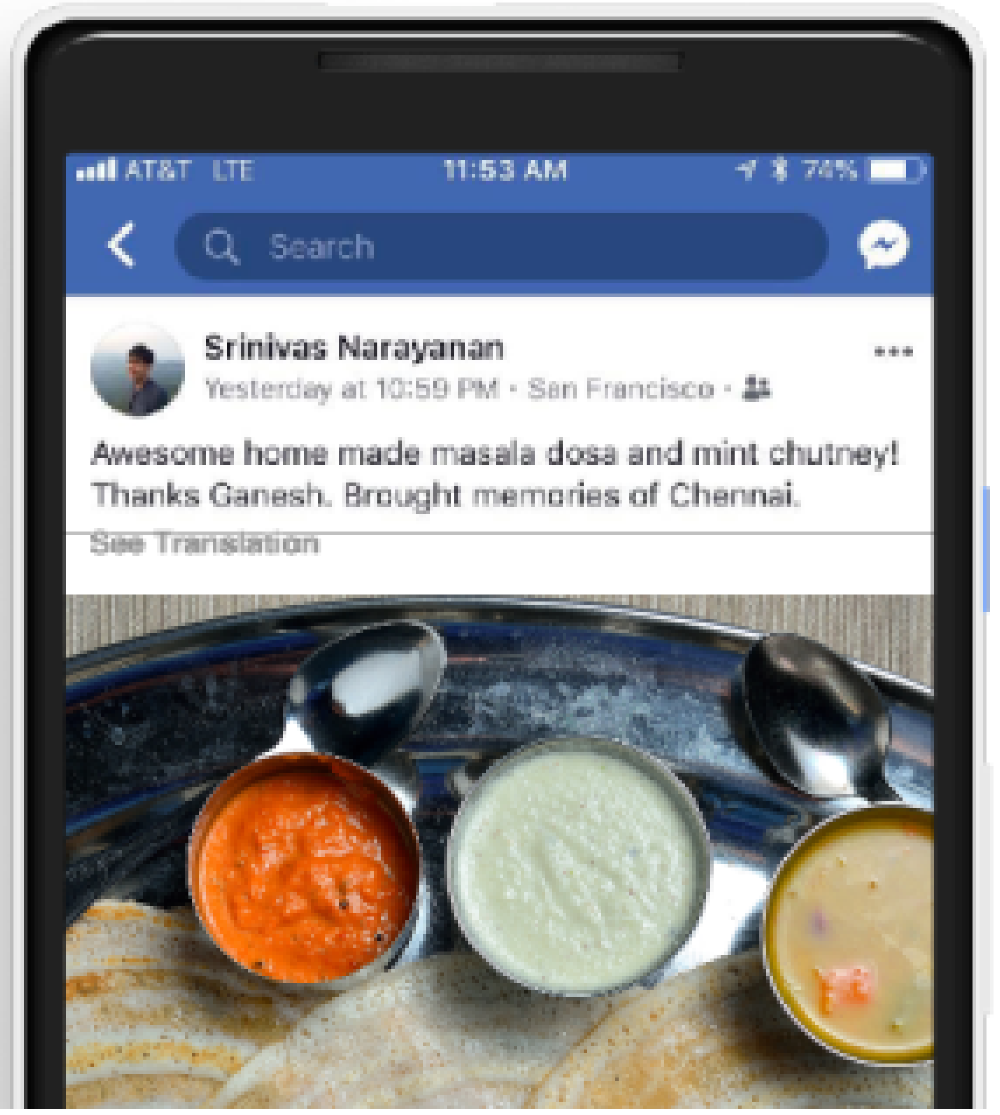


VIETNAM WEB SUMMIT

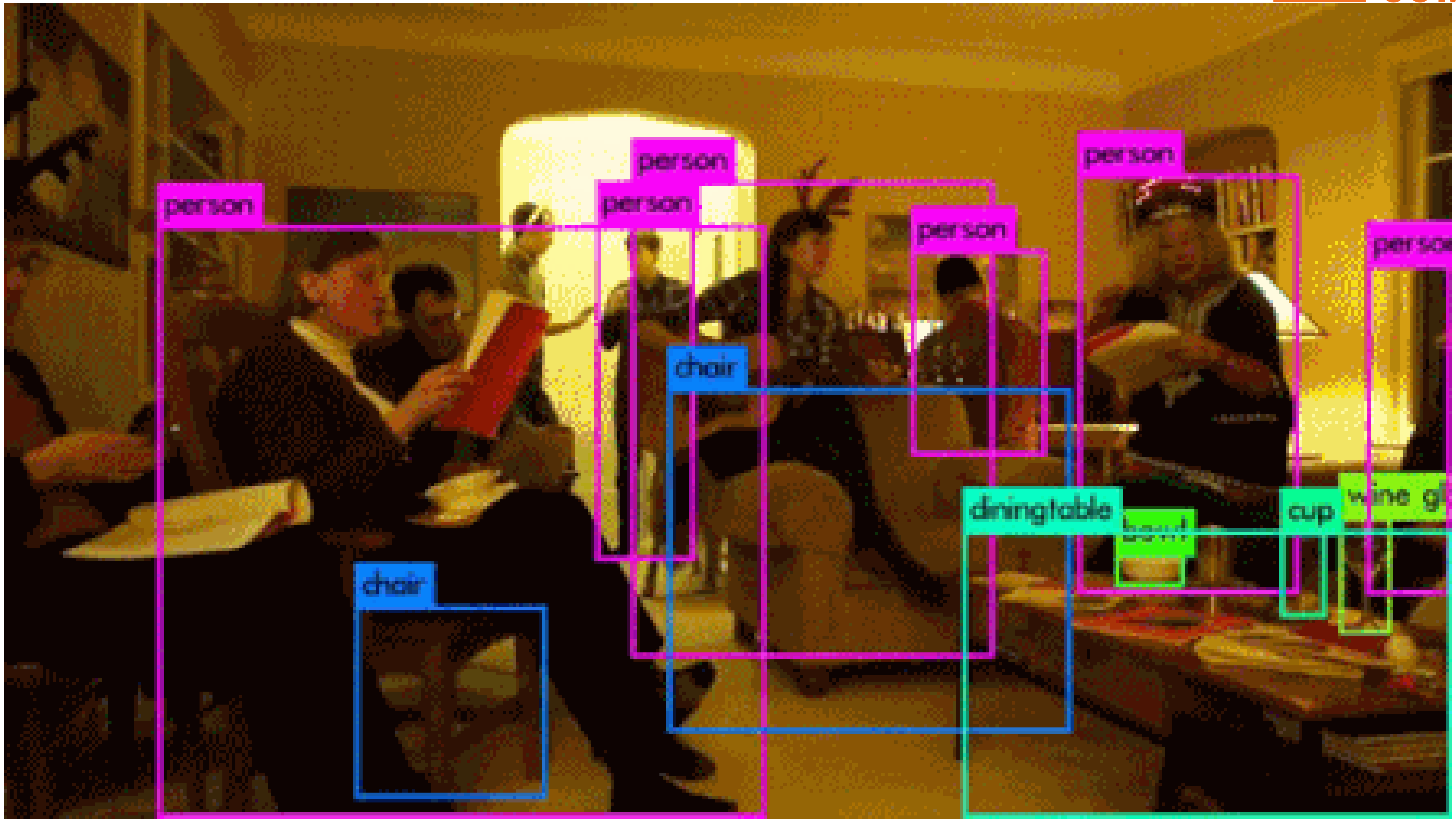
ENHANCING EXISTING PRODUCTS

SOCIAL RECOMMENDATIONS

MACHINE TRANSLATIONS



- 45+: Languages supported fc
- 2K+: Translation directions
- 6B+: Translation impressions

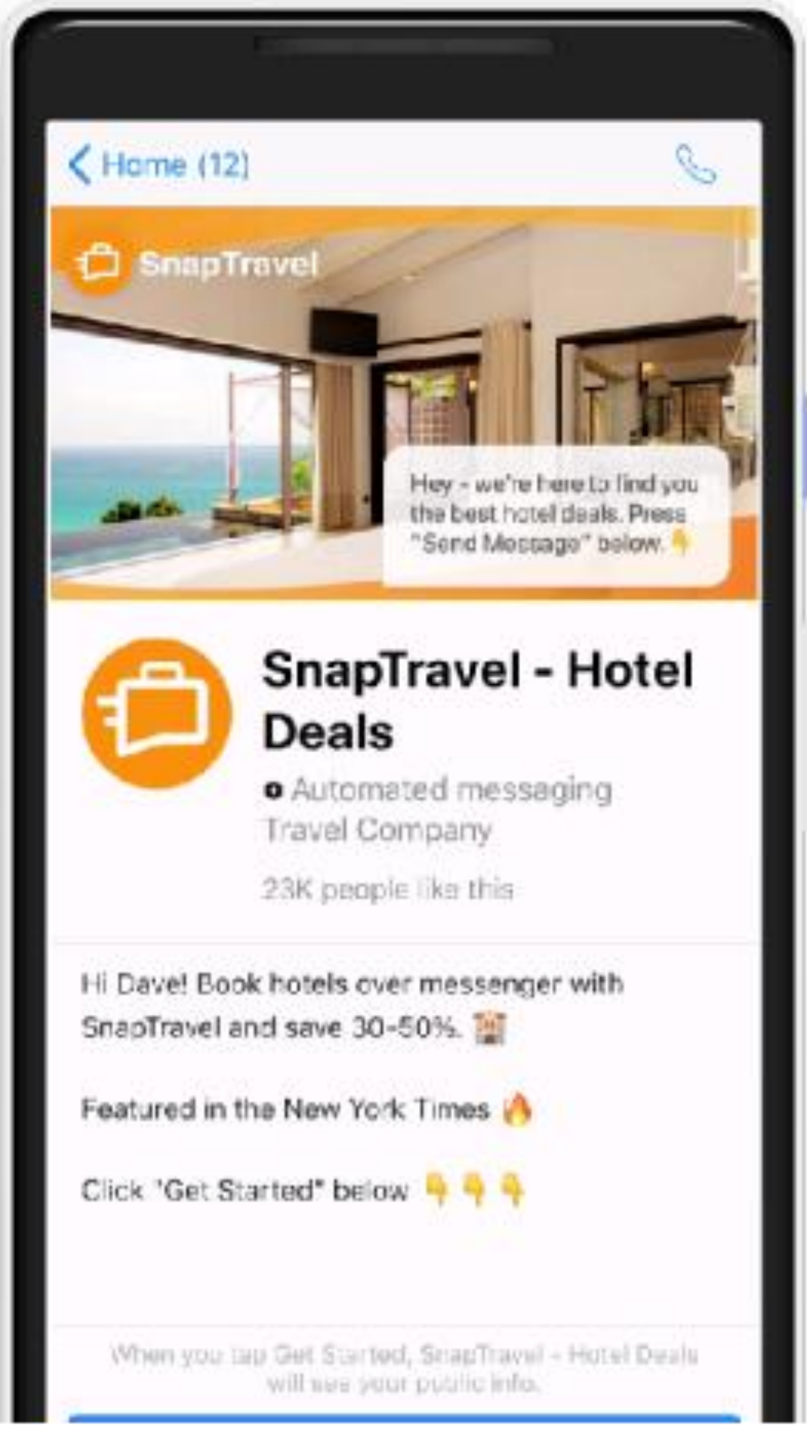


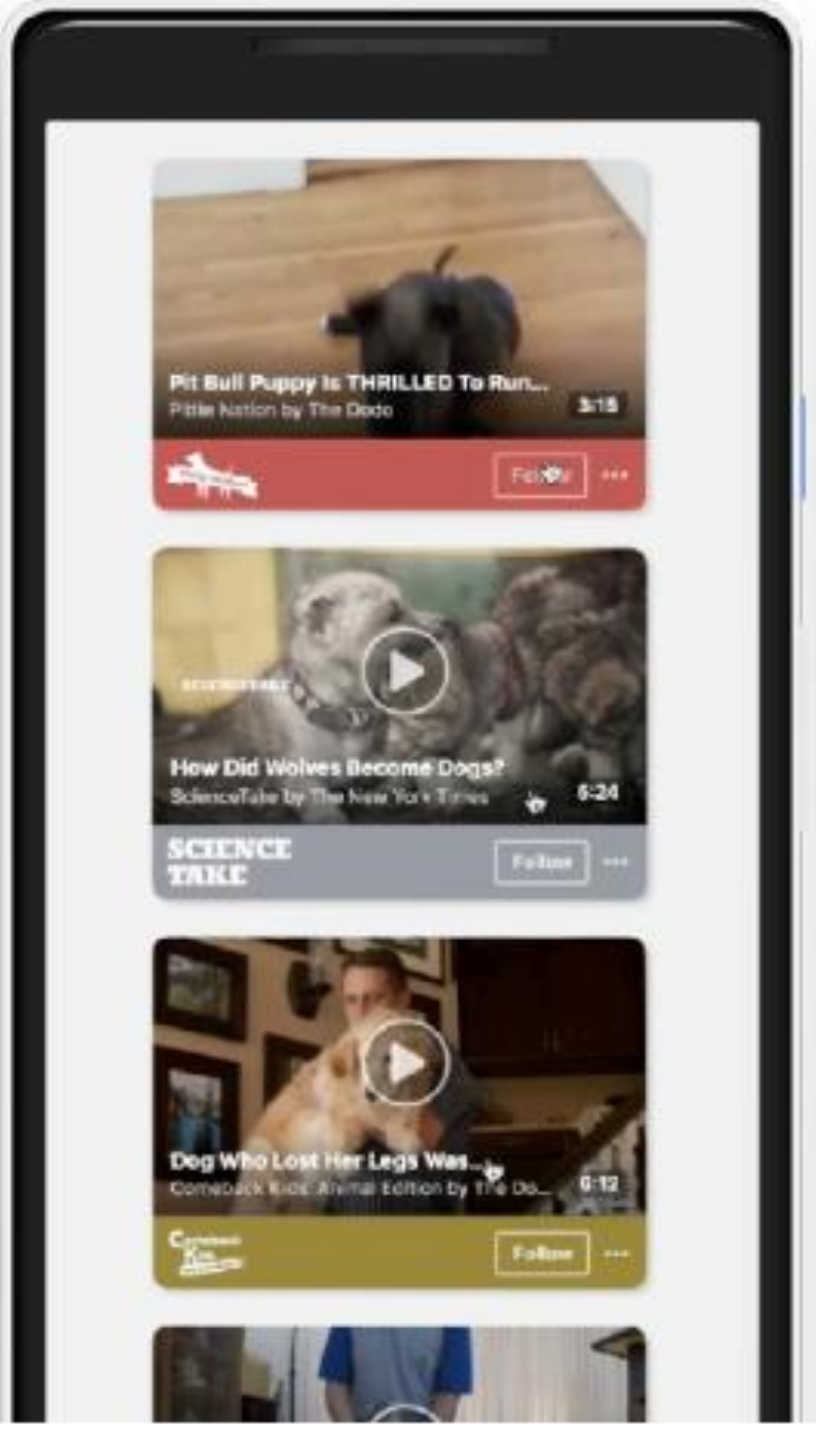
YOLO v3 PYTORCH

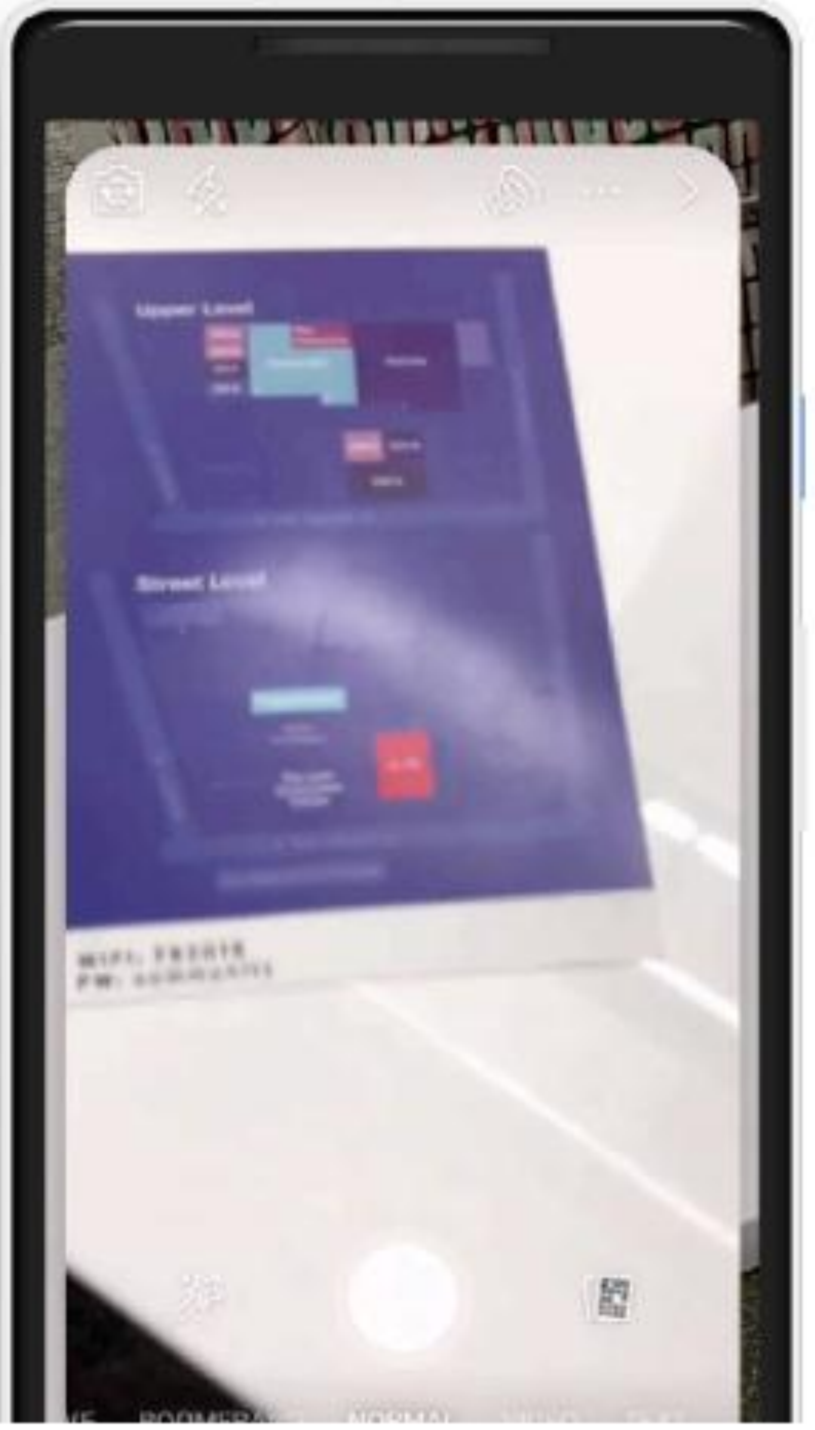
VIETNAM
WEB
SUMMIT

POWERING NEW EXPERIENCES

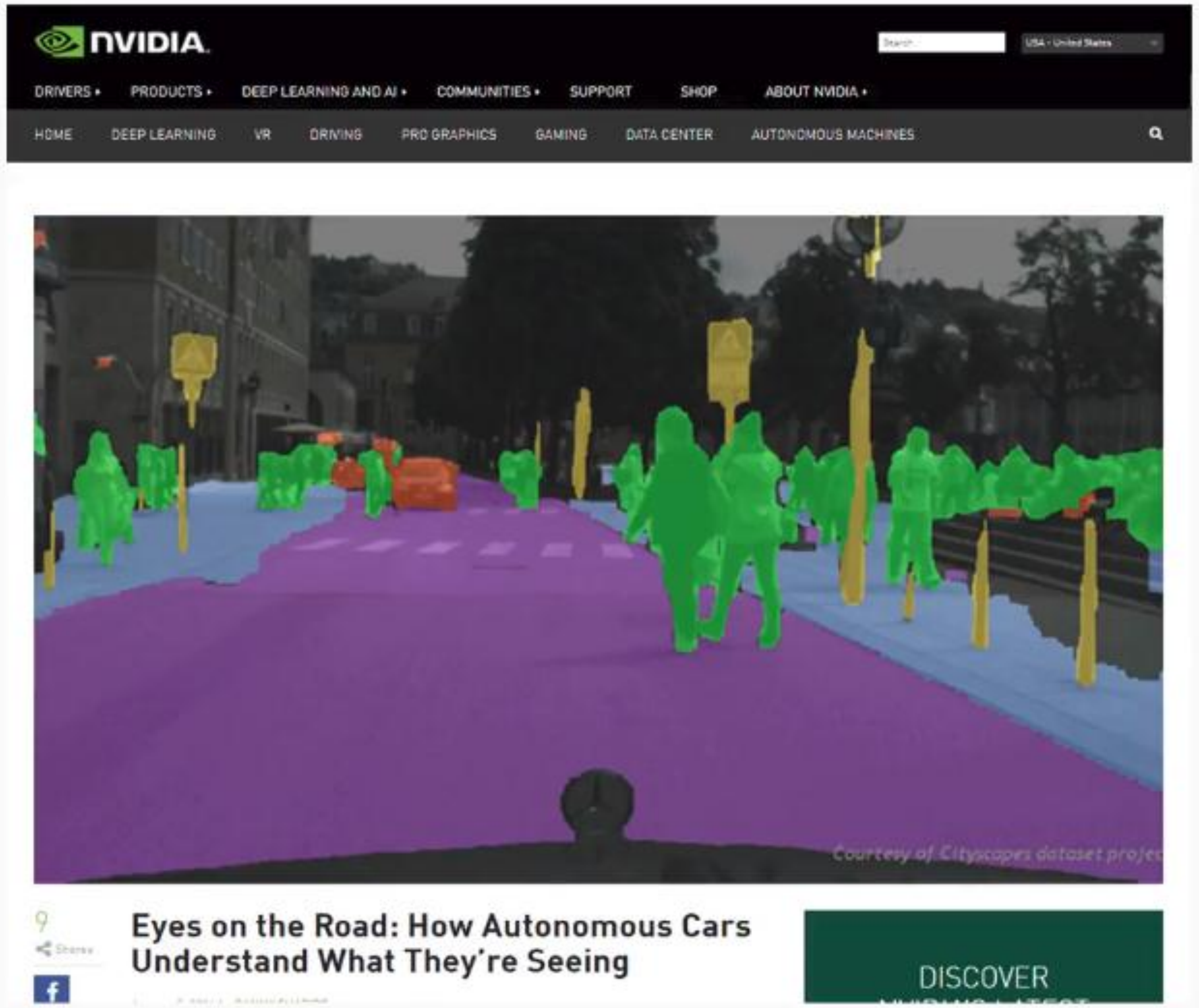
BOTS & ASSISTANTS	GENERATED CONTENT	AR EFFECTS	VR HARDWARE







Use Cases: Self Driving Cars



Camera

- Cheap
- Highest resolution
- Huge data = deep learning
- Human brains use similar sensor technology for driving
- Bad at depth estimation
- Not good in extreme weather

For the full updated list of references visit: https://selfdrivingcars.mit.edu/references

MIT 6.S094: Deep Learning for Self-Driving Cars https://selfdrivingcars.mit.edu Lex Fridman lex.mit.edu January 2018

Ecosystem

- Pix2PixHD

https://github.com/NVIDIA/pix2pixHD

Input labels

Synthesized image

VIETNAM WEB SUMMIT

Andrej Karpathy

@karpathy

Director of AI at Tesla. Previously a Research Scientist at OpenAI, and CS PhD student at Stanford. I like to train Deep Neural Nets on large datasets.

Stanford

cs.stanford.edu/~karpathy/

Joined April 2009

Andrej Karpathy

@karpathy

Following

I've been using PyTorch a few months now and I've never felt better. I have more energy. My skin is clearer. My eye sight has improved.

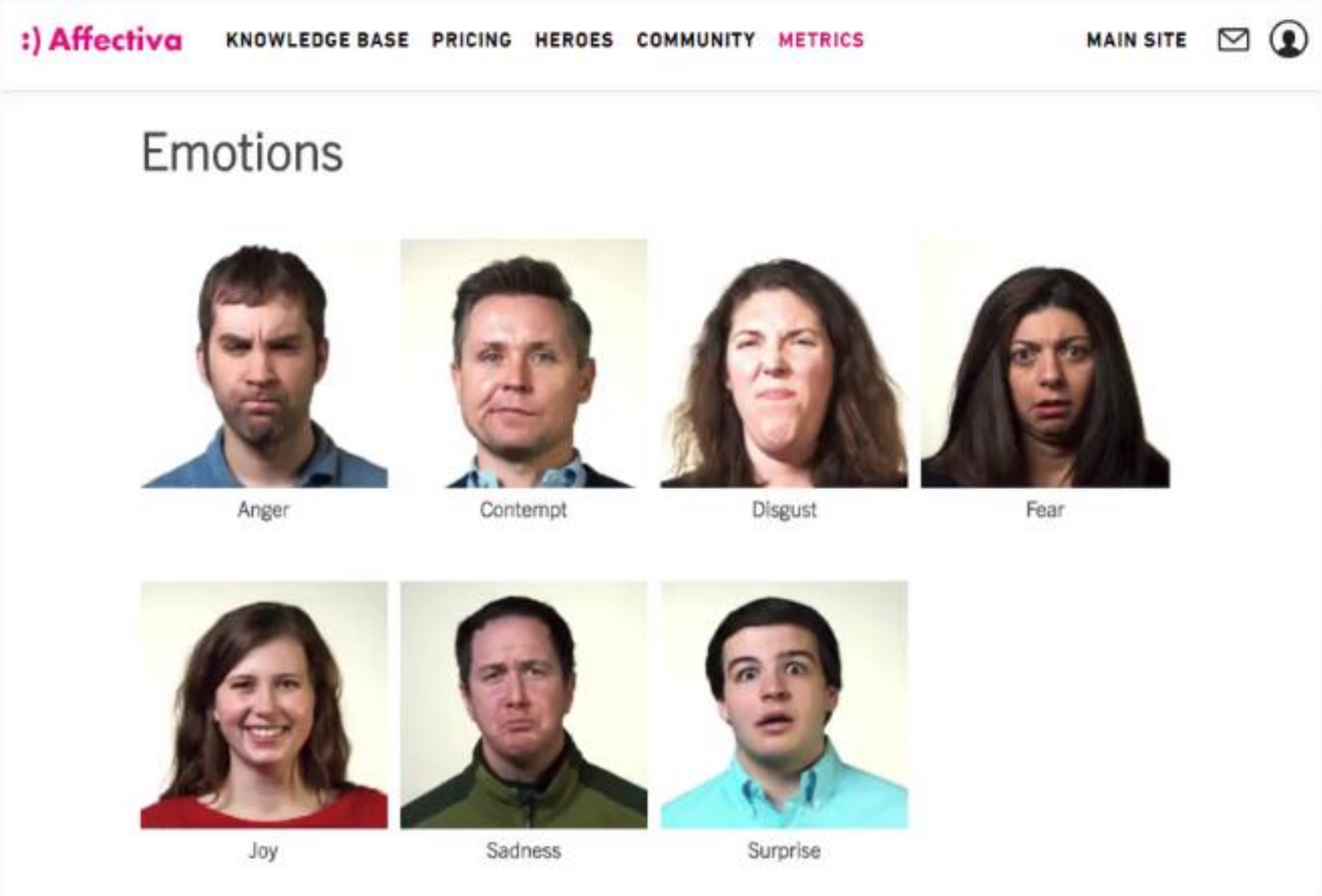
11:56 AM - 26 May 2017

384 Retweets **1,519** Likes

33 384 1.5K

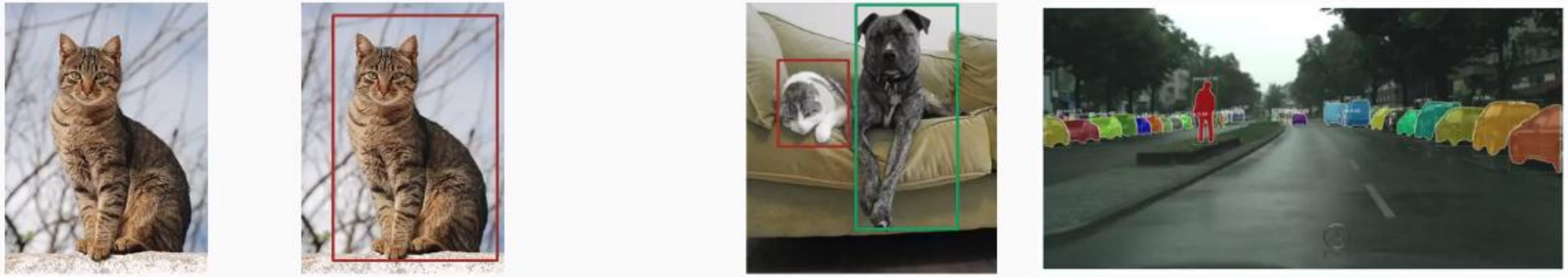
Use Cases: Emotion Detection

VIETNAM WEB SUMMIT



Tasks in Computer Vision

Classification | **Classification + Localization** | **Object Detection** | **Instance Segmentation**



CAT | CAT | CAT, DOG | HUMAN, CAR

Single Object | Multiple Objects

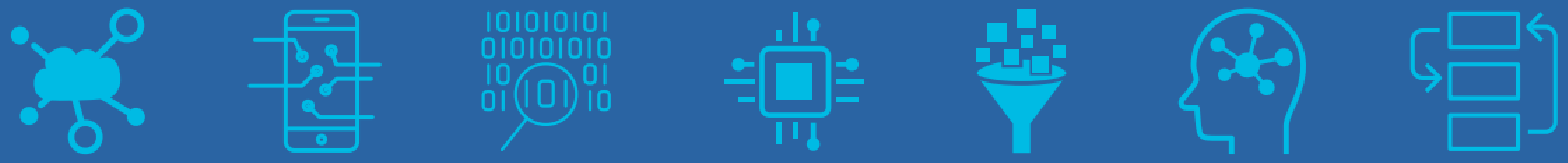
Fig: Tasks in Computer Vision (Image Credit: Wikimedia commons, Mask RCNN)

https://commons.wikimedia.org/wiki/File:Trillium_Poncho_cat_dog.jpg
https://en.wikipedia.org/wiki/Tabby_cat#/media/File:Cat_November_2010-1a.jpg
https://arxiv.org/abs/1703.06870

VIETNAM WEB SUMMIT

facebook for developers

AI Masterclass



VIETNAM WEB SUMMIT

Join us: https://www.facebook.com/groups/DevCHanoi/

Developer Circles

from facebook

VIETNAM
WEB
SUMMIT

Thank You