DeepLite

Real-Time Deep Learning Framework for Neighborhood Analysis

Duy H. Ho, Raj Marri, Sirisha Rella, Yuguung Lee
University of Missouri – Kansas City
Mobile Deep Learning

Current deep neural architectures for mobile devices:

- MobileNet-V1 (small model size and complexity)
- ShuffelNet
- MobileNet-V2 (optimized architecture)
- NasNet Mobile (based on reinforcement learning)
- MobileNet-V3 (AutoML included)
Real-time Deep learning

DL for Human activity recognition (HAR):
- Home behavior analysis
- Gait analysis
- Gesture recognition

Mobile Real-time DL:
- Face recognition
- Emotion recognition
- Speech recognition
Motivation

- Most of the current technologies are very generic
- Current Mobile Deep learning solutions are restricted to optimized models for IOS and Android
- Current solutions are not sufficient for Real world deep learning
- Real world applications need more comprehensive solution
- Comprehensive model can be too big for edge computing
Deep learning at Edge

Challenges
- Resource constraint compared to Cloud
- Low Inference speed
- Model optimization
- Deployment Architecture
- Dynamic Real time data
- Data security
Importance of Neighborhood Analysis

- Crime rate prediction
- Mental health
- Water quality detection
- Real estate assessments
- Diveristy assessment
- Ecological imbalance detection
Literature Review

- Gebru et al. estimated the demographic makeup of neighborhoods in the United States.
- Apte et al. analyzed environmental exposures worldwide by utilizing DL and Google Street View (GSV).
- Helbich et al. utilized GSV images and DL to extract metrics of green and blue space and inversely associated with depressive symptoms among the elderly.
DeepLite

- Real-time deep learning for real-world applications
- Architecture which seamlessly works with the cloud for deploying the DL models
- Inference network for distributed and collaborative inferencing capability of models on the edge
- Dynamic inferencing based on user context
Platform technologies

- Docker (popular container technology)
- Kubernetes (container orchestration engine)
- AWS EC2 instance
- AWS Elastic Kubernetes service
- S3 buckets
- Nvidia Jetson Nano
- Ansible
DeepLite Architecture

- Automated trained model deployment from Cloud to Edge through Ansible
- On-the-fly Containerisation of Deep Neural Network (DNN) models
- AWS Elastic Kubernetes Service to host cloud containerised DNN models
- Nvidia Jetson Nano as Edge device to host DeepLite platform

Fig. 1: DeepLite Architecture
Inference Network

- Models are deployed as microservices
- Distributed and collaborative inferencing
- Rest API enabled communication between models
- Configurable inference context
- Inference designed specific to the neighborhood study
- Inference on real-time streaming data

Fig. 2: Inference Network
**NeighborNet Mobile Interface**

- Users can navigate to their preferred location with a street view Google map.
- System will automatically display information relevant to the selected area.
- Returns multiple street view images of the neighborhood, important nearby places.
- Returns sentiment metrics (age, diversity, greenery, safety, traffic conditions, and recreational level).
- Provides more comprehensive analytical understanding of the neighborhoods.
Deep Learning Models

- **House type model**: Detects house or an apartment
- **House age model**: Detects house age classification of Ages 0-20, 21-40, 40-60
- **Greenery level**: Detects level of green space of landscape with trees, shrubs
- **Recreational model**: Detects park, swimming pool, gym, parking
- **Traffic model**: Detects car, traffic light, traffic sign

Fig. 4: NeighborNets Object Detection Models
House Type Model

- The dataset that is fed to the network is annotated using the Supervisely tool
- Implemented these models through the Gluon/MXNet framework
- The model is validated with the real-time images that are streamed from the GSV Static API and the images from Google
House Age model

- This model with 3 classes (house ages of 0-20, 21-40, 41-60)
- A total of 540 images, 450 images for training and 90 images for validation.
- TensorFlow framework for classifying the age of houses based on the images
- The average accuracy of the validation was 73%
Greenery model

- A higher score indicates a higher chance of an eco-friendly community
- Two classes: tree and lawn, which are clear signs of greenness in the neighborhood
- Trained with the Faster R-CNN network
- The average mAP of the object detection was reported as 69%.
# Inference network models

## TABLE I: NeighborNets Inference Network Models

<table>
<thead>
<tr>
<th>Type</th>
<th>Model</th>
<th>Framework</th>
<th>Categories</th>
<th>Size(mb)</th>
<th>Accuracy</th>
<th>Cloud Runtime (ms)</th>
<th>Edge Speed (FPS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type Detection</td>
<td>Faster R-CNN</td>
<td>Gluon/MXNet</td>
<td>house, apartment, commercial</td>
<td>33</td>
<td>(mAP) 85.13%</td>
<td>34.42</td>
<td>35-40</td>
</tr>
<tr>
<td>Age Classification</td>
<td>Inception V3</td>
<td>TensorFlow</td>
<td>house age classification of Ages 0-20, 21-40, 40-60</td>
<td>177</td>
<td>(Acc) 73%</td>
<td>90</td>
<td>10-15</td>
</tr>
<tr>
<td>Greenery Level</td>
<td>Faster R-CNN</td>
<td>Gluon/MXNet</td>
<td>Level of green space of landscape with trees, shrubs or bushes</td>
<td>130</td>
<td>(mAP) 76.02%</td>
<td>46.41</td>
<td>35-40</td>
</tr>
<tr>
<td>Recreation</td>
<td>Resnet 50</td>
<td>TensorFlow</td>
<td>park, swimming pool, gym, parking</td>
<td>92</td>
<td>(Acc) 77%</td>
<td>3000</td>
<td>-</td>
</tr>
<tr>
<td>Traffic [33]</td>
<td>Faster R-CNN</td>
<td>Caffe</td>
<td>car, traffic light, traffic sign</td>
<td>-</td>
<td>(mAP) 69%</td>
<td>31.88</td>
<td>35-40</td>
</tr>
</tbody>
</table>

* Cloud: Google Colab, Edge: NVIDIA Jetson Nano, FPS: Inference Performance in Images per Seconds, -: under development
Jetson Nano Edge Inference

- Models are containerised using docker and orchestrated using kubernetes
- Models are optimised for TensorRT inference engine
- Ansible automates the deployment of models to the edge after training in the Cloud
- Models are deployed in a dynamic service mesh
- Ambas dor is used to define the routing rules between the services
- Context aware side cars are deployed in a pod with every model container for dynamic context aware setting
Conclusion

- Real world neighborhood assessment is subjected many conditions
- Microservices like architecture can provide better accuracy and context aware inferencing at edge for the neighborhood
- Containerisation provides fast deployment and easy packaging
- Domain expert has the flexibility in defining the inference networking
- In future, we want to adopt less resource intensive platform edge like K3s rather than K8s or Kubernetes
Thank You