Ahmedabad
University

CSE641: Computer Vision: Modern Methods And Applications

Report-1

Group 1

Name	Enrollment No.
Dhyey Patel	AU2240054
Malav Modi	AU2240214
Prem Patel	AU2240010

Understanding New Models Which We need to Implement:

Inception:

- The Inception model, also referred to as GoogLeNet, is a deep convolutional neural community (CNN) structure added by means of Google within the paper "Going Deeper with Convolutions" (2014). It became designed to improve computational efficiency and accuracy in photo type responsibilities.
- The core of the structure, Inception module, applies multiple convolutional filters (1x1, 3x3, 5x5) and max pooling in parallel which allows the community to extract multi-scale functions efficiently. 1x1 convolutions are used to lessen the dimensionality which reduces computational fee.
- There are 22 layers which include convolutions, pooling and completely related layers. Two extra auxiliary classifiers at intermediate layers help with gradient propagation throughout schooling. Global Average Pooling reduces parameters and stops overfitting. ReLU activation is used all through the community to introduce non-linearity.
- The inception modules reduce computational value even as retaining excessive accuracy.

MobileNet:

- MobileNet Image Transformer unites MobileNet's low-power operation with transformer macroscale feature acquisitions for delivering effective real-time vision tasks to limited resource platforms. Depthwise separable convolutions in MobileNet decrease both computational requirements and parameter numbers when compared to conventional CNNs. MobileNet models utilize 3x3 convolution kernels while V1 through V3 remain the different versions where V3 features squeeze-and-excitation blocks for enhancing feature representation. Through transformers the model works with self-attention which detects distant relations in pictures but standard CNNs operate with limited local field capabilities. Standard MobileNet transformer designs feature one or more convolutional layers which lead to a transformer encoder block that contains self-attention layers arranged with 4–8 attention heads. This architecture provides three benefits: minimal storage requirements (10MB or less), quick processing speeds and effective operation upon edge devices. The transformer faces two significant limitations: it delivers restricted performance for large ViT models in high-resolution tasks and it may overfit on small datasets because of its minimal weight structure. Such combination of models proves ideal in mobile vision along with robotics and autonomous navigation since they require proper accuracy and efficiency management.

EfficientNet:

- EfficientNet uses superior operational performance and requires minimal computational resources to function successfully when computing power is limited. A system of network scaling through compound adjustment allows better efficiency compared to traditional CNN-based networks. The mobile inverted bottleneck (MBConv) layers from MobileNetV2 allow EfficientNet to extract features at superior capacity levels with reduced model parameter requirements. The network distribution of EfficientNetV2 spans from B0 to B7 with B7 offering better performance and faster operations compared to its preceding versions. The model's main benefits include both low memory usage and fast processing times and computer performance matching the need. Two critical weaknesses exist with the system because it performs poorly on high-resolution images while needing specific adjustments during operation. The mobile vision field and embedded AI along with autonomous navigation benefit from the successful operation of EfficientNet as an accurate and high-speed model.

Understanding Code Given:

There are mainly 4 parts in the code:

1. Image encoder, Text encoder, Point header
2. CLIP model(Combining above 3 blocks + cross entropy loss)
3. Train(I am including data preprocessing in this too)
4. Finding matches between text and image

Goals for next week:

Changing the code and adding all 3 models and running it on Flickr database.
Find the performance change in all 4.
Make the pipeline more efficient.