# PAPER REVIEW - SOME OF THE NETWORKS FOR SINGLE IMG DEPTH ESTIMATION

Given below is a brief description of the network models used in the SOTA papers in the subtopic of Monocular Depth Estimation or Single Image Depth Estimation.

*Website that was really helpful in finding some papers:-*
*https://paperswithcode.com/task/monocular-depth-estimation*

➔ **Benchmark Analysis of Representative Deep Neural Network Architectures**

- This paper presents various DNNs used in SOTAs that are used in image recognition, and in our case used as the encoder part in depth estimation. It gave some good insight as to what makes a network architecture 'efficient'.

- The accuracy density chart gives a good idea of how efficiently a model makes use of its parameters. Some good candidate networks which we can look into which have lesser computational complexity that **DenseNet-169** are :- **MobileNetv1**, **MobileNetv2**, **DenseNet-121, ResNet-101** based on the accuracy density chart.

- Link for the paper: ***https://arxiv.org/abs/1810.00736***

➔ **High Quality Monocular Depth Estimation via Transfer Learning**

- The 2018 variant/ implementation of this paper made use of the network DenseNet-169(Tensorflow) or DenseNet-161(PyTorch) as the encoder part. There are current community made implementations using smaller encoders ie **MobileNet-v2** which we can look into.

  The github repo for Pytorch implementation is given below:-
  *https://github.com/alinstein/Depth_estimation*

- As we can see even the size of the pre-trained MobileNetv2 model is quite less -> 55MB compared to ~160MB size of DenseNet-169

- The current SOTA paper - AdaBins also uses the DenseNet-169 architecture proposed in this paper and builds on top of it.

➔ Other papers worth considering:-

- Deep Ordinal Regression Network (DORN) - An implementation using **ResNet-101**

- From Big to Small: Multi-Scale Local Planar Guidance for Monocular Depth Estimation - use of **DenseNet-161**

➔ Trial to make cuda compile work:

- Tried building PyTorch from source.
  - Nvidia driver = 460.32.03
  - gcc & g++ - 8.4
  - CUDA Toolkit 11.2
  - cudnn-runtime and cudnn-dev ⟶ 8.1.0
  - magna-cuda 111 from pytorch channel

  Errors: There is an inconsistency in the torchvision lib and I am not able to import it. Will be trying with CUDA 10.2 and build PyTorch from source next time, as some forums mention it to be a more stable version.