SEMI-SUPERVISED ITERATIVE TEACHER-STUDENT LEARNING FOR MONOCULAR DEPTH ESTIMATION

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

CEMAL BARIŞKAN SÜVARİ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

FEBRUARY 2021

Approval of the thesis:

SEMI-SUPERVISED ITERATIVE TEACHER-STUDENT LEARNING FOR MONOCULAR DEPTH ESTIMATION

submitted by **CEMAL BARIŞKAN SÜVARİ** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. İlkay Ulusoy
Head of Department, **Electrical and Electronics Engineering** _____

Prof. Dr. Uğur Halıcı
Supervisor, **Electrical and Electronics Engineering, METU** _____

Examining Committee Members:

Prof. Dr. Gözde Bozdağı Akar
Electrical and Electronics Engineering, METU _____

Prof. Dr. Uğur Halıcı
Electrical and Electronics Engineering, METU _____

Prof. Dr. Alptekin Temizel
Graduate School of Informatics, METU _____

Assist. Prof. Dr. Tolga İnan
Electrical and Electronics Engineering, Çankaya University _____

Assist. Prof. Dr. Mehmet Dikmen
Computer Engineering, Başkent University _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Cemal Barışkan SÜVARİ

Signature :

ABSTRACT

SEMI-SUPERVISED ITERATIVE TEACHER-STUDENT LEARNING FOR MONOCULAR DEPTH ESTIMATION

SÜVARİ, Cemal Barışkan
M.S., Department of Electrical and Electronics Engineering
Supervisor: Prof. Dr. Uğur Halıcı

February 2021, 87 pages

Advances in robotics area and autonomous vehicles have increased the need for accurate depth measurements. Depth estimation is one of the oldest problems of computer vision area. While the depth can be estimated by using many methods, finding a cheap and efficient way of doing it was studied for many years. Although, depth measurements using Lidar sensors or RGB-D cameras provides accurate results, due to cost and narrow applicability they are not very effective. On the other hand, using deep learning architectures to estimate depth seems to provide a more efficient, cheaper and robust solution compared to other methods. With the progress in deep learning, monocular depth estimation problem has gained a lot of attention. Recently, representation learning methods showed very promising accuracy results in depth estimation from single images. In this thesis, a deep learning based network architecture is proposed for monocular depth estimation problem. Furthermore, the network is trained with an iterative teacher-student learning framework in a semi-supervised manner. To make student networks generalize better than the teacher network, noise is injected during training of student networks. According to evaluation results our proposed model achieves state-of-the-art accuracy in monocular depth estimation.

Keywords: Deep Learning, Monocular Depth Estimation, EfficientNet, Iterative Teacher-Student Learning

ÖZ

MONOKÜLER DERİNLİK TAHMİNİ İÇİN YARI DENETİMLİ YİNELEMELİ ÖĞRETMEN-ÖĞRENCİ ÖĞRENİMİ

SÜVARİ, Cemal Barışkan
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Uğur Halıcı

Şubat 2021 , 87 sayfa

Robotik alanındaki gelişmeler ve otonom araçlar, doğru derinlik ölçümlerine olan ihtiyacı artırmıştır. Derinlik tahmini, bilgisayar görüşü alanındaki en eski sorunlardan biridir. Derinlik birçok yöntem kullanılarak tahmin edilebilirken, bunu ucuz ve verimli bir şekilde yapmanın yolu uzun yıllar çalışılmıştır. Lidar sensörleri veya RGB-D kameralar kullanılarak yapılan derinlik ölçümleri doğru sonuçlar verse de, maliyet ve dar uygulanabilirlik nedeniyle çok etkili değildir. Öte yandan, derinliği tahmin etmek için derin öğrenme mimarilerinin kullanılması, diğer yöntemlere kıyasla daha verimli, daha ucuz ve sağlam bir çözüm sağlıyor gibi görünmektedir. Derin öğrenmedeki ilerlemeyle birlikte, monoküler derinlik tahmini problemi büyük ilgi gördü. Son zamanlarda, temsil öğrenme yöntemleri, tek görüntülerden derinlik tahmininde çok umut verici doğrulukta sonuçlar göstermektedir. Bu tezde, monoküler derinlik tahmini problemi için derin öğrenme tabanlı bir ağ mimarisi önerilmiştir. Ayrıca ağ, yarı denetimli bir şekilde yinelemeli bir öğretmen-öğrenci öğrenme yapısı ile eğitilmektedir. Öğrenci ağlarını öğretmen ağından daha iyi genelleştirmek için öğrenci ağlarının eğitimi sırasında gürültü enjekte edilmektedir. Değerlendirme sonuçlarına göre, öne-

rilen modelimiz monoküler derinlik tahmininde son teknoloji doğruluğa ulaşmaktadır.

Anahtar Kelimeler: Derin Öğrenme, Tekil Gözlem Derin Tahmini, EfficientNet, Yinelemeli Öğretmen-Öğrenci Öğrenimi

to my beloved wife...

ACKNOWLEDGMENTS

Above all, I would like to thank my supervisor Prof. Dr. Uğur Halıcı for her guidance and support throughout the research.

I would like to thank ASELSAN A.Ş. for giving me the opportunity of continuing my graduate education.

I am thankful to my family for their continuous help and patience throughout my studies and my entire life.

Finally, I would like to thank to my lovely wife Özge for her kind and endless support through my life.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

Lidar	Laser Imaging Detection and Ranging
GT	Ground Truth
CNN	Convolutional Neural Network
ASPP	Atrous Spatial Pyramid Pooling
CT	Computed Tomography
ReLU	Rectified Linear Unit
eLU	Exponential Linear Unit
Abs Rel	Absolute Relative Difference Error
Sq Rel	Squared Relative Difference Error
RMSE	Root Mean Squared Error
RMSE log	Root Mean Squared Error Log

CHAPTER 1

INTRODUCTION

1.1 Motivation and Overview

Depth estimation with a computer based system has been studied by researchers for a long time. In recent years, developments in robotics area and autonomous vehicles have increased the need for accurate and real-time depth measurements. There are also some applications that do not need exact metric depth and a real-time system. For example, 2D-3D converting systems need depth information for a good modeling but relative depth map of the scene would be enough for them to create a model of the scene.

Although early studies of estimating depth through disparity using stereo images yielded satisfactory results, the need for two cameras has led researchers to tackle this problem from different angles. Apart from requiring two cameras to estimate depth, stereo matching is another big challenge in stereo vision systems. After left and right images are rectified it is required to find corresponding edge points and match them in both images to make an accurate depth estimation. Lidar (Laser Imaging Detection and Ranging) on the other hand is undoubtedly one of the most reliable way of measuring depth. It provides enough depth information about the environment but it also has disadvantages as expected. When using a Lidar sensor, misalignment of emitted laser light due to reflective or absorbing surfaces may lead to wrong measurements. Aside from that, sparsity in a depth map captured by a Lidar may become very high. Density in the depth map depends on the number of sensors used in Lidar and the rotational speed in horizontal direction. Density can be increased by increasing the number of Lidar sensors but in return the cost will increase heavily. For the

time being, their prices are high to use widely but they may become cheaper with the developing technologies. Due to the demands of the industry and thanks to the latest advances in deep learning, monocular depth estimation by deep learning systems has become a topic of interest among researchers. Wide applicability is one of the many advantages of a monocular depth predictor.

Many studies have been made by researchers in medical area to investigate the difference in depth perception with a human binocular and monocular vision system. While many researchers have observed that the advantage of a binocular system can be up to hundreds of meters [25], [26], [27], most researchers are agreed on that most efficient distance for the binocular vision system is limited up to around 10 meters. Beyond 10 meters, human vision is essentially monocular. Based on this information, it can be assumed that a monocular depth estimation network can be trained well-enough to make close or even better depth predictions than a human. Surely, there will be some problems to be solved like the need for massive amount of training data, domain adaptation issues, speed of inference etc. If outdoor utilization is planned, there will be environment related problems like bad weather, reduced field of vision etc. Some of these problems can be solved in a monocular depth estimation network easily, while others may raise difficulties. Main goal of our study is to reduce the need for labeled ground truth data with the help of iterative teacher-student learning method while training a monocular depth estimation network.

Different kinds of methodology can be used to train a monocular depth estimation network. Most of the strategies depend on three main learning methods, supervised, self-supervised and semi-supervised learning. Supervised learning methods need to use ground truth depth maps to train the network. If a large enough dataset with ground truth data is available, it would be advantageous to train network in a supervised fashion since it would be able to solve many difficult regions (e.g. textureless regions) with the help of supervision proxy. However, ground truth depth maps are mostly obtained by RGB-D or Lidar cameras and obtaining this kind of data would be costly and challenging. As an alternative, self-supervised methods annihilate the need for ground truth data, but in return the accuracy decreases and the results become less reliable. Semi-supervised methods on the other hand, take advantages of the both sides and present good solutions. Teacher-student learning method presented in this

work is also a semi-supervised method since both labeled and unlabeled datasets are used in training. By using this methodology, the need for massive amount of labeled data can be decreased.

Monocular depth estimation networks perform a kind of representation learning and that's why extracting features from input images successfully is important. For a representation learning problem network architectures consist of two structures in general, an encoder and a decoder. While encoder part extracts features and delivers them to the decoder part, in decoder part these extracted features are processed to represent desired output. Size of the network may determine how good it can extract the features or how much it can learn from a dataset. However, in order to construct a good feature extractor or encoder, solely increasing the size of the model by adding more layers or increasing the channel sizes are not effective ways. On the other hand, compound scaling applied in EfficientNet [17] increases the performance of the encoder while keeping the model size still manageable. Compound scaling is a method applied to the network in order to change its depth, width and resolution in a controlled way. At the time we were working on our study, EfficientNet encoder had not yet been used in any monocular depth estimation network. We wanted to evaluate its performance in a monocular depth estimation network and if satisfactory results were obtained we decided to use it in our model.

1.2 Contributions

There have been many monocular depth estimation networks that are employed in deep learning. As expected in representation learning problems, previous studies have proposed different architectures to adapt their model to the datasets used. Some of these works have also drawn attention to importance of feature extraction. At this point, encoder plays an important role in extracting features in the best manner.

EfficientNet proposed by Tan *et al.* [17] is a convolutional neural network architecture. It is designed and optimized using neural architecture search. Tan *et al.* [17] have used a compound scaling methodology while they design Efficient so its width, depth and resolution can be scaled to different sizes. Due to its structure compact Effi-

cientNet can be used as an encoder in any neural network in order to extract features. Extracted features by EfficientNet can be used in a decoder architecture to solve a representation problem. EfficientNet has achieved a remarkable performance in the studies that it had been used. Tan *et al.* [17]'s proposed architecture outperformed most of the well-known encoder structures in classification and regression related problems.

In this thesis, EfficientNet [17] encoder structure is used in our network architecture. To the best of our knowledge, it had not been used in a monocular depth estimation network previously. We first evaluated its performance by using it in a well-known depth estimation network (Monodepth [10]). We have replaced the encoder structure used in Monodepth with EfficientNet. Then, we have trained this modified network and evaluated its performance. We have observed a performance increase in evaluation results due to used EfficientNet architecture. After obtaining these results we decided to use EfficientNet in our network architecture. For our decoder structure we have investigated the Atrous Spatial Pyramid Pooling (ASPP) method proposed in [28]. It is a structure that is used to capture contextual information at multiple scales in a decoder. We have integrated ASPP structure too into our modified Monodepth network and evaluated its performance. After observing the accuracy increase with the help of ASPP as well, we decided to use it in our decoder structure.

Teacher-student learning methods have been proposed and used in monocular depth estimation networks previously. While, most of the previous methods concentrated on decreasing the size of student network in order to fit it into a smaller area, some others focused on increasing the performance of teacher network with the help of a student network. In deep learning, transferring knowledge from one network to another is called as knowledge distillation. The network that transfers knowledge is named as teacher network and the network that newly trained is named as student network. In our thesis, we used the teacher-student learning methodology proposed in [1]. Proposed method trains a student network iteratively while all student networks and the main teacher network have the same size. To the best of our knowledge, iterative student-teacher learning had not been used in any monocular depth estimation network before. During iterative learning, performance of each student network is expected to outperform previous teacher network. Another important contribution of

this method is that it reduces the size of labeled dataset needed. Acquiring a labeled dataset with ground truth depth maps for a monocular depth estimation network is costly and difficult. Moreover, the noise injected to the system during training in previous methods is data augmentation in general. In fact there are two types of noise used in deep learning applications. One of them is input noise also known as data augmentation methods. Other type of noise is called as model noise also known as regularization methods. In the proposed method, in addition to data augmentation, model noise is also used and injected as dropout and stochastic depth. This kind of aggressive noise injection during training of student networks is expected to increase generalization of trained network and adaption to dataset. With these contributions, effectiveness of iterative student-teacher learning in a monocular depth estimation network is showed. The performance of student networks outperformed the main teacher network and the state-of-the-art accuracy is achieved.

1.3 Thesis Outline

In chapter 1, introduction to monocular depth estimation, possible problems and our solution is given briefly. Motivation of this thesis, outline and the contributions are also presented in this chapter.

In chapter 2, literature on depth estimation problem is reviewed and previous works on monocular depth estimation problem are summarized. These novel works are analyzed in three main titles according to their learning strategies, supervised, unsupervised and semi-supervised. Then, EfficientNet [17] related background information is provided and previous works that incorporate EfficientNet are reviewed. Then, the literature on teacher-student learning is revised. The progress achieved by other researchers in time and studies of student-teacher learning method and the their proposed novel strategies are analyzed. Finally, background information about the KITTI dataset [22] used in training of our models is provided.

In chapter 3, proposed network architecture is described in detail. In Encoder Architecture and Decoder Architecture sections, methods applied in our network are described in detail. Then, in Training Loss section the loss function used in training

is explained. In last, details of the teacher-student learning strategy applied in training is given, along with the followed procedure and the key aspects.

In chapter 4, first, evaluation metrics that are used in determining the performance of a monocular depth estimation network are described. Then, networks that are trained and evaluated are explained. Evaluated performance of our models and the comparison with previous works are given. Deductions made according to our evaluation results are explained. Performance of EfficientNet when used in a monocular depth estimation network is also provided in this chapter.

In chapter 5, a summary of our thesis and the discussion are provided.

CHAPTER 2

RELATED WORK AND BACKGROUND

In this chapter, we review the literature on depth estimation, EfficientNet architecture and teacher-student learning method. We first investigate depth estimation problem and the proposed solutions. Deep learning based previous studies are analyzed according to their learning methods and basic knowledge is provided about these studies. Later, EfficientNet related previous works and the studies that incorporate Teacher-Student Learning Method will be given. Finally, background information about the KITTI dataset [22] used in training and testing will be provided.

2.1 Depth Estimation

Depth of objects in a scene can be estimated or calculated by using passive and active techniques. Passively, depth information can be obtained by 2 methodologies, depth from stereo images and depth from monocular images. On the other hand, active techniques include usage of sensors, RGB-D cameras, structured-light scanners etc. and the depth information is obtained instantaneously. Aim of all these techniques is to help constructing spatial structure of the environment that gives three-dimensional view of the scene. After obtaining depth information, position of the viewer can also be known relative to the surrounding objects.

In computer vision area one of the most used depth calculation method is stereo vision. This computer based passive method date back to 90s [29], [30], [31]. In this approach, stereo cameras/images are used to extract depth information. First, viewing positions of the cameras need to be determined using epipolar geometry. It is a geometrical relation between two views and is used for representing each pixel with

respect to the camera positions as seen on Figure 2.1. In order to calculate depth from disparity, focal lengths and baseline distance between two cameras should be known. Depths in the scene can also be extracted from sequence images of a video. Between consecutive frames of a video, in order to calculate depth of an object, boundaries/edges of it and the camera motion should be detected correctly. However, estimating motion of the camera is a difficult task and the frame rate of the video should be high enough to prevent objects becoming blurry while camera moves. Therefore, this approach does not have a wide-applicability.

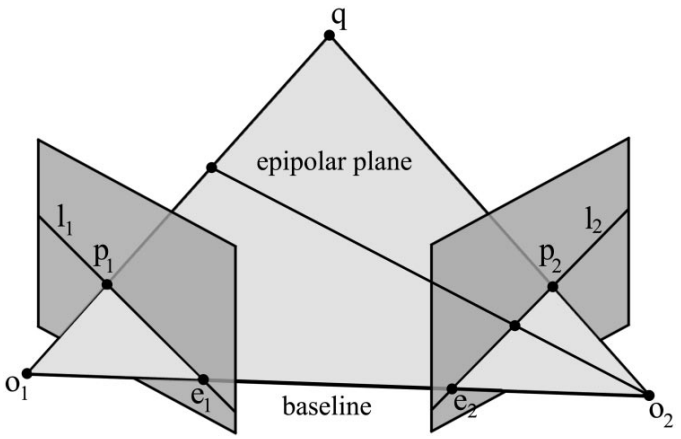


Figure 2.1: Epipolar Geometry [3].

In order to calculate disparity, correspondence (pixel) matching should be done between pixels of both images. For a point in first image, corresponding point in the second image need to be found. Thus, in order to make a good correspondence matching both images should be rectified. Rectification is a process of transforming images such that epipolar lines of the original images match horizontally. Sample images before and after rectification process can be seen on Figure 2.2. Then, along on an epipolar line each pixel in an image is matched with corresponding same pixel in other image using a matching cost function. After pixels are matched, depth can be calculated by using the distance between two cameras and the pixel distance between matched pixels.

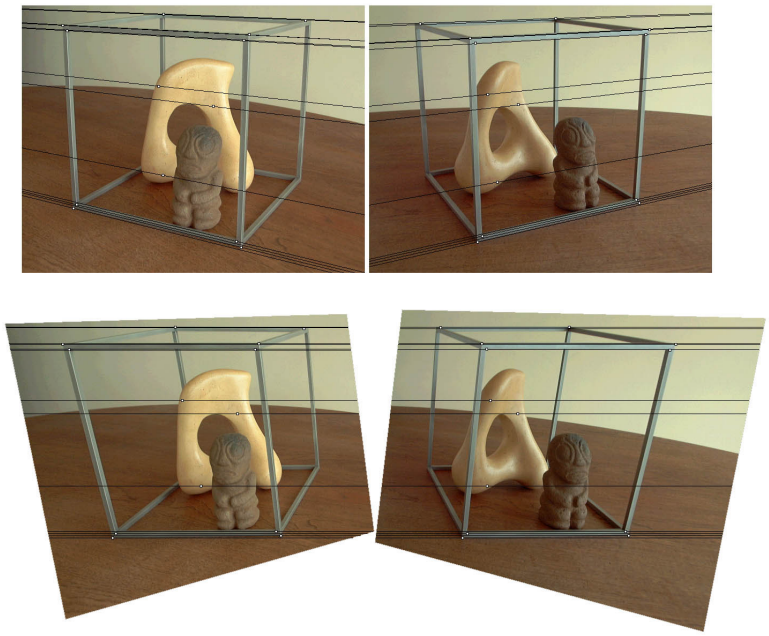
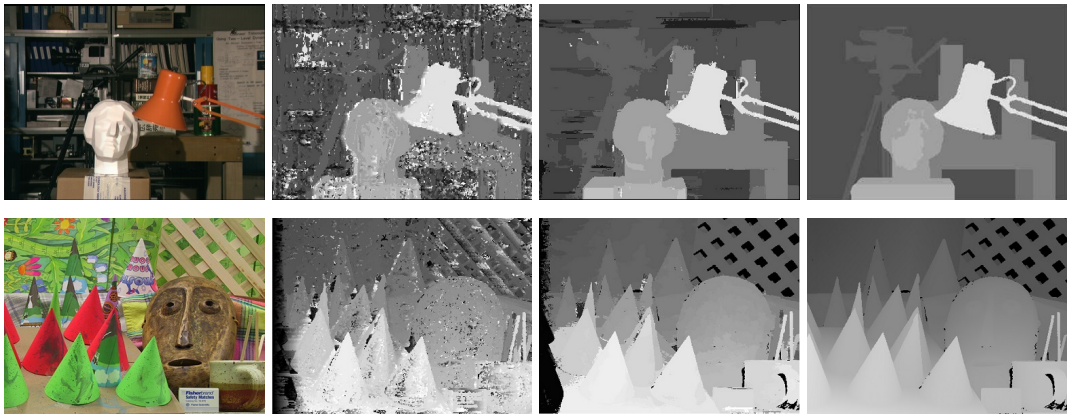


Figure 2.2: Top: Non-rectified Images, Bottom: Rectified Images. [4]

Reflective, transparent, highly textured areas and very smooth regions create the biggest difficulty for stereo matching algorithms. Edge details of an object in one image may be disappeared in second image due to perspective change. If algorithm cannot match these edge points on other image, it might generate an inaccurate depth value at those points and create noise in estimated depth map. Disappearance of object boundaries might also be caused by a fast moving object. Due to direction of movement, the object might become blurred in one of the stereo images. Apart from camera and environment related these problems, algorithm used for depth calculation may also create problems. Matching cost function used in the algorithm may generate false-positive signals for some edges. These falsely matched edges also create inaccurate depth maps. Therefore, post processing techniques are mostly required in stereo vision application in order to eliminate noise and refine depth maps. Most used post processing techniques are median filter, bilateral filter, interpolation etc. In Figure 2.3, calculated disparity maps obtained using stereo vision algorithm are provided along with the refined disparity maps and the ground truth images for comparison.



(a) (b) (c) (d)

Figure 2.3: Original image in (a), raw disparity maps in (b), refined disparity maps in (c) and ground truth maps in (d) [5].

2.1.1 Monocular Depth Estimation Using Deep Learning

Even though estimation of depth from multiple images has a long history in computer vision area, extracting depth information from single images is rather a new concept in deep learning area. The concept has been started to be investigated thoroughly with the developments in deep learning techniques. One of the biggest problems in deep learning is lack of dataset that fits to the problem which is a considerable problem for monocular depth estimation networks too. Data that can be used in training can be collected by Lidar sensors, RGB-D cameras or stereo vision cameras. However, collecting suitable data is generally costly. Therefore, different learning strategies are developed in order to reduce dependency for dataset.

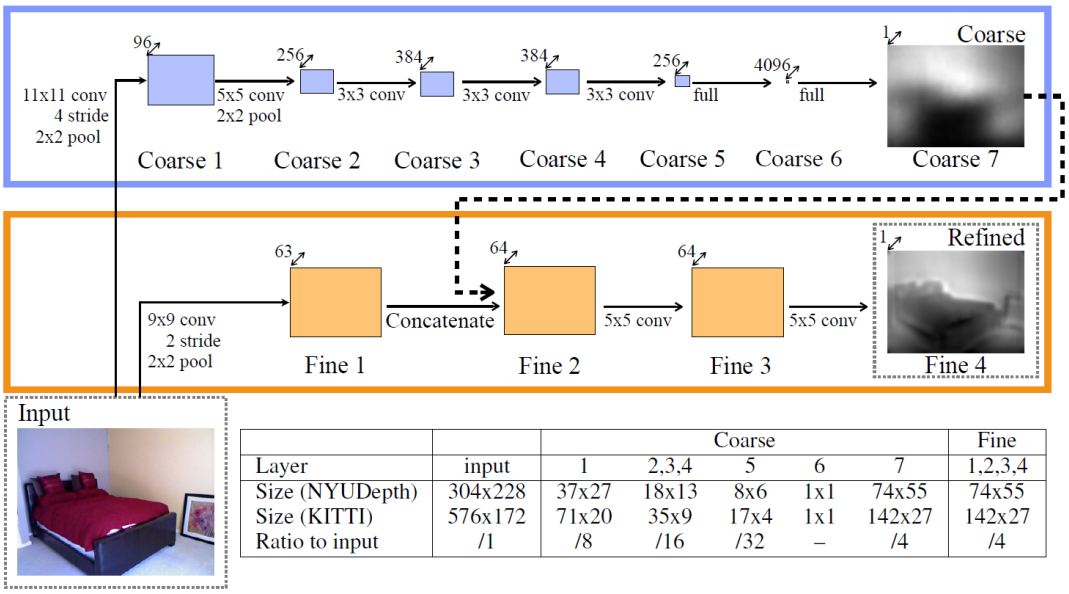
Learning in monocular depth estimation networks can be divided into three main categories according to their strategies:

1. Supervised Learning [6], [32], [7], [8], [33], [34], [35]
2. Unsupervised Leaning [9], [10], [11], [12], [13], [19]
3. Semi-Supervised Learning [14], [15], [16]

2.1.1.1 Supervised Learning

Supervised learning for depth estimation requires pixel-wise ground truth depth information. The method proposed in Eigen *et al.* [6]'s study was one of the first that uses depth information to train a model by deep learning. They state that their CNN-based network that is given on Figure 2.4 comprises of two deep network stacks. In first stack their deep neural network makes depth prediction in coarse scale from an input image. As seen on the Figure 2.4, input image is provided to the both stacks while the output depth map of first stack is also provided to the second stack in order to refine depth map. In second stack the predicted depth map is merged with first layer image features. Task of this stack is to align received coarse depth predictions with the objects in the scene.

In order to calculate error irrespective of the global scale, Eigen *et al.* [6] defined a loss function in log space. They used this scale-invariant error function as training loss to measure difference between predicted depth map and ground truth depth. They claim that using scale-invariant mean squared error function as training loss increases accuracy and improves the sharpness in object edge boundaries.



		Coarse					Fine
Layer	input	1	2,3,4	5	6	7	1,2,3,4
Size (NYUDepth)	304x228	37x27	18x13	8x6	1x1	74x55	74x55
Size (KITTI)	576x172	71x20	35x9	17x4	1x1	142x27	142x27
Ratio to input	/1	/8	/16	/32	–	/4	/4

Figure 2.4: Eigen *et al.* [6]'s network structure

.

After Eigen's study [6] explained above, several methods have been proposed to improve accuracy of estimated depth map. Li *et al.* [32] constructed a deep learning network using Conditional Random Fields for depth map refinement. Similar to [6] they used a two stage network for depth map prediction and refinement. In first stage, super-pixels are obtained from input image and image patches extracted around these super-pixels. Depth map is regressed for these image patches at super-pixel level. Then, in second stage, conditional random fields are used to refine depth map by converting super-pixel depth map to pixel level. By their super-pixel image patch approach, they intended to keep sharper object boundaries.

Additional to ground truth depth supervision some approaches also exploit geometric relationships to extract a better depth map. In [7] Qi *et al.* used two networks to predict depth map and surface normal from single images. Their proposed network architecture is given on Figure 2.5 with sample images. These two networks enable conversion of depth-to-normal and normal-to-depth, and collaboratively increase accuracy of depth map and surface normal. Although their neural network can increase the accuracy of depth maps by this method, for training they require ground truth labeled surface normal which is hard to obtain.

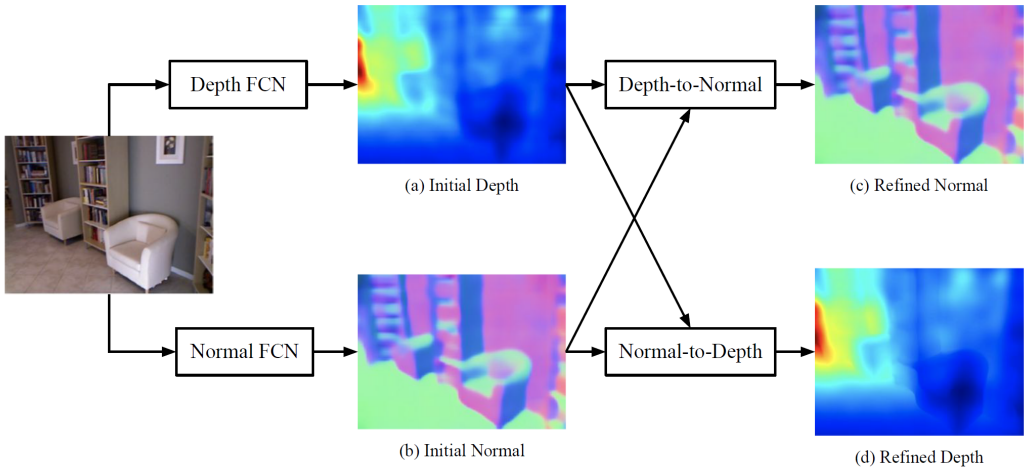


Figure 2.5: GeoNet Network [7].

Ummenhofer *et al.* [8] proposed a network to predict depth by the help of structure from motion (SfM) approach. Two view of the scene is provided as input to create a single depth map in a three stage network structure which is shown on Figure 2.6.

In their study, they argue that a basic encoder-decoder architecture cannot process two input images at the same time. Thus, they proposed an architecture that outputs optical flow, ego-motion and depth map from an image pair. In the first stage, bootstrap net takes image pair as input and calculates depth map and relative pose of the second camera. In the second stage, iterative net has similar structure to bootstrap net, but it processes optical flow and depth map iteratively to increase accuracy. In the final stage, refinement net takes input image and low-resolution predicted depth map as input, and outputs high-resolution depth map by upsampling it to the original resolution. Ummenhofer *et al.* [8] also applied a scale invariant gradient loss in training. This loss term increases smoothness in homogeneous areas and stimulates depth discontinuities in object boundaries.

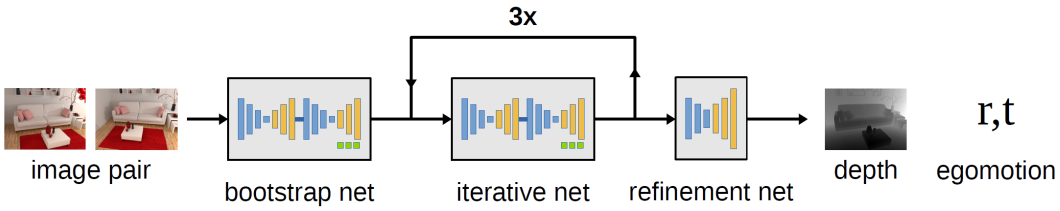


Figure 2.6: Ummenhofer *et al.*'s Depth and Motion Network (DeMoN) [8].

Quality of dataset is as important as the methodology applied in supervised learning systems. Especially in depth prediction very sparse GT (Ground Truth) depth points might limit the learning of the network. Rosa *et al.* [33] drew attention to this problem in their study. They proposed a technique to create denser GT depth maps from sparse Lidar measurements by increasing the valid depth pixels in depth images. They train their model both with sparse GT depth maps and denser GT depth maps and compared the results. The increase they observed in the performance of their network when denser GT depth maps are used is undeniable.

Gan *et al.* [34] also suggested a method to increase quality of GT data while training their network. They use a Stereo Matching Network additionally to predict depth maps from stereo image pairs. Then, they use these generated depth maps to assist Lidar GT depth maps. They also state that in an outdoor scene major depth changes occur in vertical direction. They claim that roads mostly lie on the vertical direction and the faraway objects like sky and mountains are positioned on top of the scene

in general. Based on this assumption they implemented vertical pooling in their network. According to the performance results they share, their proposed techniques yield better results than previous works. However, the assumption they made for vertical direction depth changes cannot be generalized for most cases.

Ranftl *et al.* [35] have proposed an important learning strategy that incorporates multiple datasets in order to increase performance of monocular depth estimation network. They collected ground truth depth maps for multiple datasets and additionally they created their own dataset from 3D movies. In order to create their own dataset from 3D movies they have used stereo matching to infer depth for frames of these movies. They have pointed out to many problems they have encountered during creation of this dataset. Unknown camera parameters, changing resolution, negative and positive disparity values are some of these problems they have dealt with during this process. In order to show generalization ability of their method they have also used zero-shot cross-dataset transfer and evaluated their models on datasets that were not used during training. With the help of their proposed methods for combining multiple datasets, they have achieved high accuracies with their model for monocular depth estimation problem. Another observation they have made during their study is that performance of encoders on ImageNet [2] dataset for object detection problem provides insight for the performance of these encoders on monocular depth estimation problem. This observation was important for us since EfficientNet encoder we choose to use in our networks provides state-of-the-art accuracy results on ImageNet dataset. Using multiple datasets to train their network Ranftl *et al.* [35] created a robust and generalized network. They have also achieved state-of-the-art performance for monocular depth estimation problem and showed the importance of dataset diversity in training once again.

2.1.1.2 Unsupervised Learning

With the increase of layers and trainable parameters in deep neural networks, the need for the train data increases as well mostly. The main problem with the increasing the need of more train data is that ground truth depth maps are difficult to obtain. In this case, unsupervised learning becomes a good option, since unlabeled data might be

rather easy to find.

With the idea of eliminating the need for ground truth depth maps Garg *et al.* [9] became one of the first to propose a method to learn depth in an unsupervised fashion. Their encoder-decoder architecture based model that is shown on Figure 2.7 is trained with stereo image pairs. Left image of the pair is given as input to the network and predicted inverse depth map is obtained as output from the network. The predicted inverse depth map and right image is used to generate target left image by warping. Warping operation simply moves pixels along on horizontal lines. Pixels of far objects move less while pixels of close objects move more. Then the reconstructed left image and the original left image are used together to calculate image reconstruction losses for train loss.

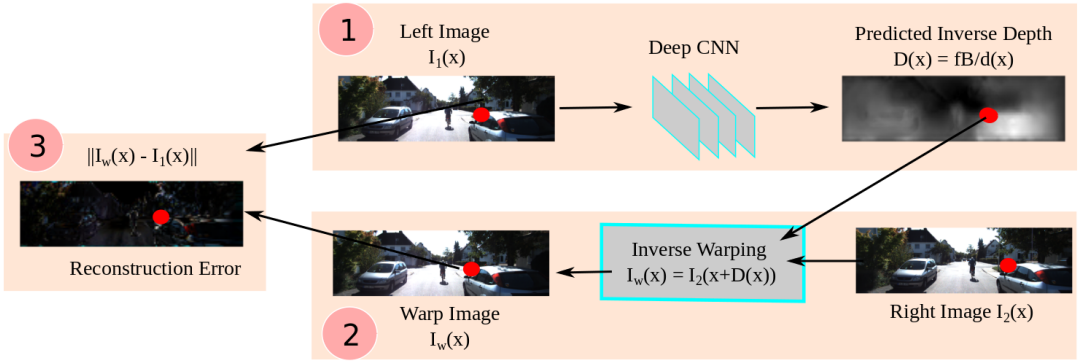


Figure 2.7: Garg *et al.* [9]'s training method.

Godard *et al.* [10], further improved [9] by inserting left-right consistency check into the loss function. Their network outputs two disparity maps (left and rigth) from only left image. Then the left and right images are reconstructed by using these disparity maps. The reconstructed images are compared with the original ones and the differences are noted as train loss. Since they have formulated the depth estimation as an image reconstruction problem, they have selected the loss functions that calculate error between reconstructed and original images. They have used Structural Similarity Index (SSIM), disparity smoothness term and left-right consistency check term as loss functions. SSIM is a function that is used to measure the similarity of two images. It compares two images in terms of luminance, contrast and structure. Disparity smoothness term penalizes depth discontinuities unless there is an image gradient at

regarding area of the image. If image gradient exist at some point, probably there is an object edge and it is not penalized. Left-right consistency check term is calculated by using mean absolute error and it is used for each reconstructed and original pair of images.

In order to warp right image to left image they employ image sampler from spatial transformer network (STN) which uses bilinear sampling. Their left-right consistency check system is shown on Figure 2.8. Bilinear sampler used in the network is locally fully differentiable and according to their evaluation results it improves the overall accuracy of their network. Due to their open-source code, network materials and implementation details, their work has become one of the most referred study in monocular depth estimation field.

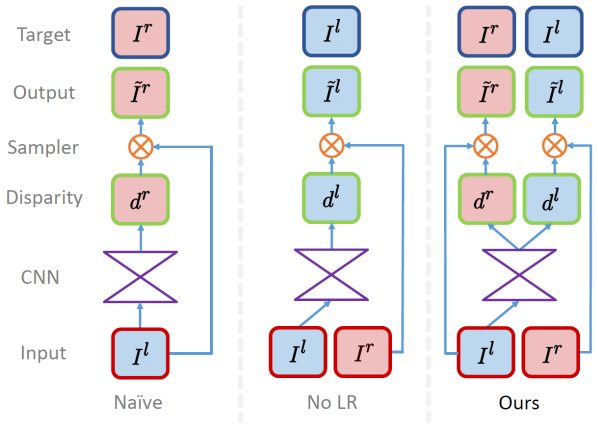


Figure 2.8: Godard *et al.* [10]'s left-right consistency check method.

Inference speed in testing time is also a consideration in depth estimation. For example, fast prediction is essential for autonomous vehicles. Poggi *et al.* [11] improved [10] by simplifying the network architecture and they reported that a remarkable speed in inference time was obtained. They adopted the same strategy for depth prediction problem. However, with their pyramidal encoder structure which reduces the complexity, they have managed to reduce number of parameters to 6% compared to number of parameters used in Godard *et al.* [10]'s network. Their pyramidal network architecture is shown on Figure 2.9. With their simplified model, they can infer depth map even on a Raspberry Pi 3 in 1.7s.

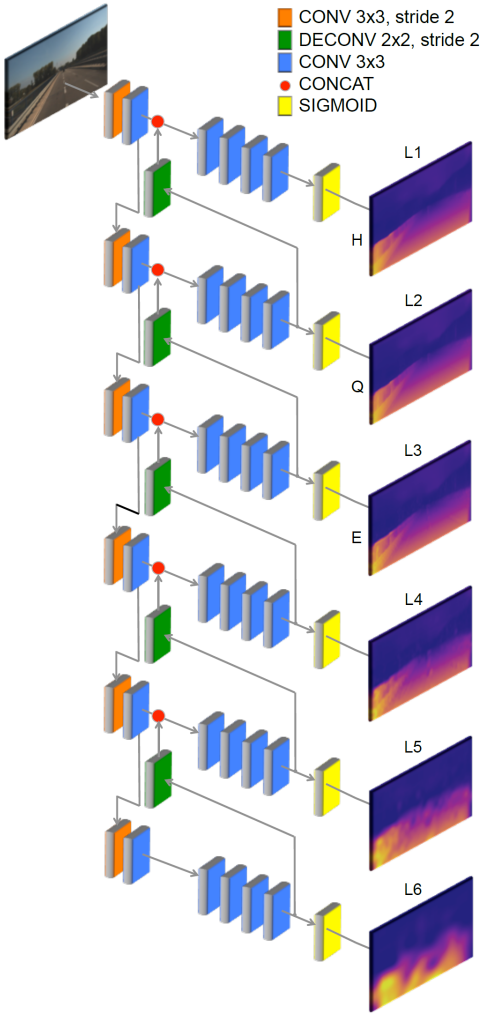


Figure 2.9: Poggi *et al.* [11]'s network structure .

Unsupervised methods investigated until now used stereo images as supervision and their train loss was depending on image reconstruction errors only. Consecutive frames from a video can also be used as supervision in training a depth prediction network. The challenge in this approach is that the camera transformation between consecutive frames must also be predicted, which brings extra complexity to the network.

Zhou *et al.* [12] proposed an architecture to predict depth map and camera pose simultaneously that is shown on Figure 2.10. Three consecutive frames are fed into the network as input. The Pose CNN predicts relative camera poses while the Depth

CNN predicts a depth map from the first image. Then, predicted depth map and other two frames are used with relative camera pose information to reconstruct target image. The image reconstruction stage employs bilinear sampling similar to [10] and similar reconstruction errors are used as train loss. Zhou *et al.* [12]'s proposed method falls behind [10] which might has happened because of two reasons. [12] does not employ left-right consistency check as train loss and this might decrease accuracy. Another possible reason is that [10] uses calibrated stereo pairs, which is absent in [12]'s method and predicting relative camera pose adds another unknown parameter to the architecture.

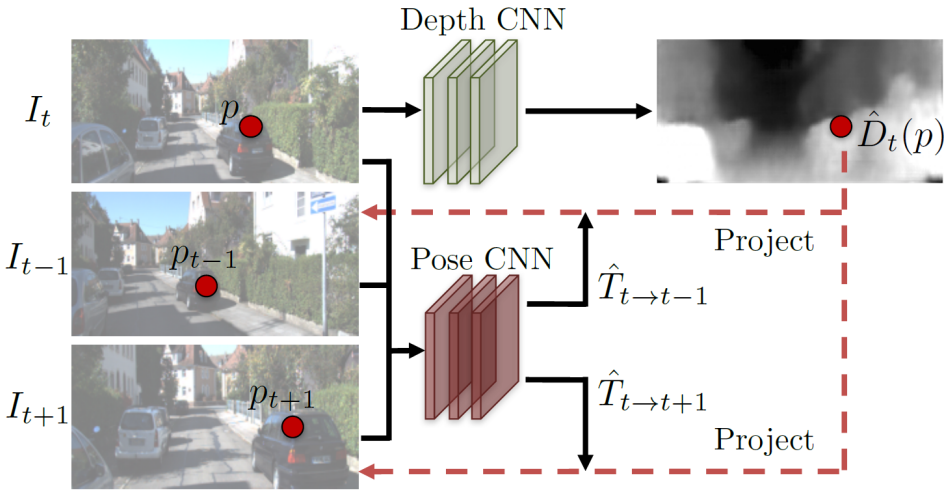


Figure 2.10: Zhou *et al.* Network [12].

There are also some approaches that merge multiple self-supervision methods into one to achieve better accuracy in depth prediction. In their work, Godard *et al.* [13] proposed using video sequences and stereo image supervision together to train their model. They use estimated depth map and estimated relative camera pose to construct other stereo view and adjacent other two frames in the video sequence. Then, image reconstruction errors are calculated using these reconstructed images and the original ones. [13] has some other improvements compared to their previous work [10], in terms of loss functions and network architecture. They have added a pose network to their model to estimate relative camera pose in adjacent frames. Their proposed network architecture and the methods applied in training are shown on Figure 2.11.

In their paper, Godard *et al.* [13] state that with careful choices of appearance losses,

significant improvement in the accuracy can be obtained. One of the main problems in self-supervised methods using video is occluded pixels which are resulted by the camera motion. In a 3 frame sequence, instead of using classical average loss for occluded pixels, they propose to use minimum loss which helps to get non-occluded pixel from amongst previous or next frame. Another improvement they have accomplished compared to their previous work [10] is computing image reconstruction losses in original image scale. In this work, they upsample predicted depth maps to original input image size to compute reconstruction losses instead of computing them on the ambiguous low-resolution images. With this approach they state, texture-copy artifacts are prevented. In this paper, they have reached to the state-of-the art performance in unsupervised learning for monocular depth estimation.

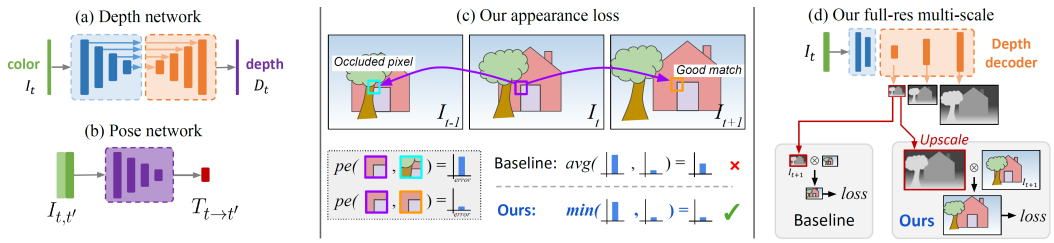


Figure 2.11: Godard *et al.* [13]'s Network Architecture (Monodepth2).

Increase in the accuracy of unsupervised methods has encouraged others to adapt knowledge distillation methods into the monocular depth estimation problem. Pilzer *et al.* [19] adapted an unsupervised monocular depth estimation network to the teacher-student learning framework. Their proposed approach uses stereo image pairs to train a teacher network. As they state, their student network is a sub-network of teacher network, so they call their method more of a self-distillation process. While their teacher network shows promising results, their student network cannot reach to the accuracy of teacher network. Instead, student network is double time faster than the teacher network at inference. Pilzer *et al.* [19]'s methodology is analyzed in detail in Teacher-Student Learning section.

2.1.1.3 Semi-Supervised Learning

In contrast to supervised and unsupervised methods, there are less studies about semi-supervised methods in monocular depth estimation. An approach proposed by Kuznietsov *et al.* [14] uses supervised loss term and unsupervised loss term at the same time during training. Their proposed architecture is shown on Figure 2.12. Predicted inverse depth maps are used to reconstruct left and right images by warping and unsupervised loss term is computed using reconstructed and target images. At the same time supervised loss term is computed using predicted depth maps and ground truth depth maps. They have also evaluated the usage of skip connections from encoder part to decoder part and claimed that using skip connections slightly improves performance of the overall network. Moreover, they stated that estimated depth maps become more detailed when skip connections are used.

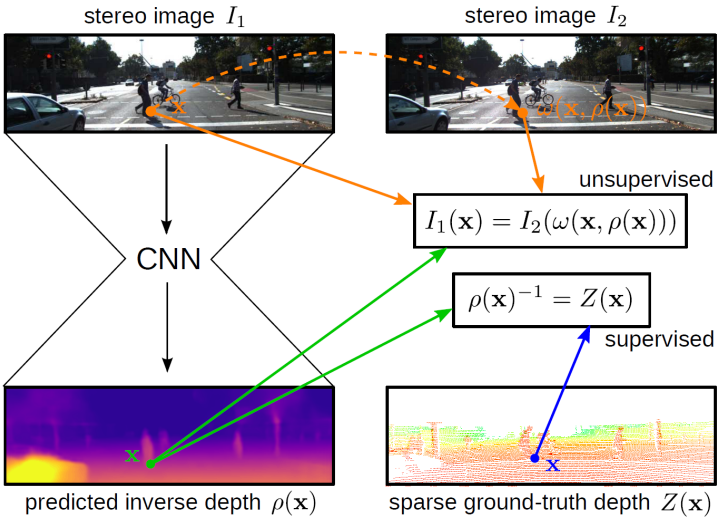


Figure 2.12: Kuznietsov *et al.* [14]'s Network Structure.

Luo *et al.* [15] considered dividing the monocular depth estimation problem into two sub-problems and deal with these problems separately. According to their approach, they expect to decrease the need of network for labeled ground truth depth data. They also stated that using geometric constraints during inference may be possible and can improve the performance even further. Their overall architecture which is given on Figure 2.13 comprise of two sub-networks: View Synthesis Network and Stereo Matching Network. Adapted from Deep3D [36], their View Synthesis Net-

work learns to synthesize right image of stereo pair using left image. In this network only L1 loss is as an unsupervised loss term between original right image and synthesized right image. In Stereo Matching Network, left and synthesized right image are used together in an encoder-decoder architecture pipeline to obtain disparity map. In this network ground truth depth maps are used as supervision in order to compute loss for predicted depth maps. According to their approach two networks are trained separately in first stage, then they are fine-tuned in an end-to-end fashion in second stage.

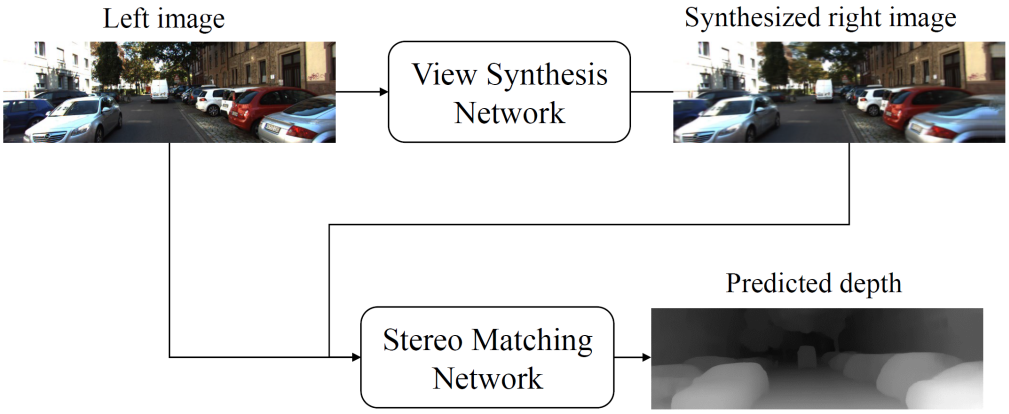


Figure 2.13: Luo *et al.* [15]'s Network Architecture.

Cho *et al.* [16] introduced a new teacher-student learning strategy to train a monocular depth estimation network in a semi-supervised manner. Their proposed method that is shown on Figure 2.14 consists of two stages similar to [15]. In first stage they train a stereo matching network with GT labeled data and call it the teacher network. Then, they allow the teacher network to predict depth from stereo pairs of a massive unlabeled dataset. In the second stage of their strategy, they use these predicted depth maps and unlabeled dataset to train a student network optimized for monocular depth estimation. They have also evaluated the trade-off between the accuracy and the density of pseudo labeled depth maps. The density increases as the pixels in the depth map increase. They measured the accuracy of pseudo labeled depth maps with Abs rel and RMSE error metrics with changing density of the depth maps. They observed that accuracy of the pseudo labeled depth maps increases when density decreases. They interpret this relationship as elimination of inaccurate pixels while the density

decreases. They have also reported that their monocular depth estimation network reaches the best accuracy when the density of pseudo labeled depth maps are around 80 percent. As stated in their paper [16], using more accurate pseudo labeled depth maps does not necessarily yield better results.

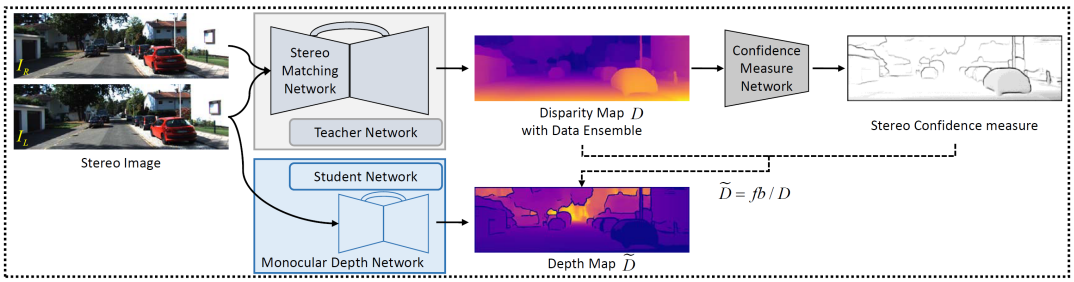


Figure 2.14: Cho *et al.* [16]'s Network

2.2 EfficientNet

This chapter presents background information about EfficientNet model and the analysis of related works that employ EfficientNet. EfficientNet model is a strong tool for deep learning architectures. When proposed by Tan *et. al* [17] in 2019, EfficientNet drew big attention with its accuracy and efficiency. Compared to other Convolution Neural Network based architectures ResNet [37], Xception [38], EfficientNet has achieved state of the art performance in image classification like tasks. They created their baseline network using neural architecture search and with the help of grid search strategy they found appropriate scaling coefficients for each model type. In time, with its small to large scalable models EfficientNet started to meet needs of many architectures by just scaling depth, width and resolution coefficients.

A deep neural networks size can be changed in terms of depth, width and resolution. Depth of a model can simply be considered as the number of layers in the network. Each layer in a deep learning network has convolution based operations and additionally operations like batch normalization, activation function etc. can be added to the layer. As layers in the network increase, the deeper the network becomes. Depth wise scaling is the most popular one among others as known from ResNet [37] models. With increasing layers, models called ResNet18, ResNet50, ResNet100 are

constructed. Width is simply the number of channels used in each convolution layer. Channels of a layer is also known as filters. Input data is encoded into many channels during layer operations and the extracted features are transferred by those channels. Increasing channel sizes in a model may increase number of trainable parameters enormously depending on the convolution operation used in the model. Resolution is simply the size of the input image supplied to the network. In some cases, high resolution images can help detecting small or thin objects more easily at the cost of a larger model size.

Scaling depth, width and resolution were used methods among researchers to achieve better accuracy. At the cost of small increments in model sizes, up-scaling these parameters can provide better results. However, according to [17] up-scaling these parameters individually can provide better results up to a point and after a threshold the performance gain reaches saturation. Compound scaling method as described in [17] is strategically scaling depth, width and resolution of the model all together to achieve a better trade off between model accuracy and size.

To show accuracy increment when depth, width and resolution parameters are scaled individually, ImageNet Top-1 Accuracy vs. Flops graphs are provided in [17]. These graphs are given on Figure 2.15. Accuracy of the model is evaluated after training an image classification network to classify images in ImageNet [2] dataset. ImageNet is a large dataset of images, designed for computer vision tasks. It includes over 14 million images organized into almost 22,000 categories. There are two well-known evaluation metrics for image classification when ImageNet dataset is used, Top-1 accuracy and Top-5 accuracy. Top-1 accuracy is the traditional accuracy, which implies that the networks answer with the highest probability must be expected target answer. If networks answer is not the expected answer, it is considered as a wrong output. Top-5 accuracy implies that, expected target answer must be amongst the 5 highest probability answers given by the network. If the target answer is not amongst these 5 highest probability answers, it is considered as a wrong output. Flops (floating point operations per second), on the other hand, is a measure for model performance. It roughly shows the number of operations required to be done in the model. When depth, width or resolution of the model is increased, number of operations (flops) required in the network increases as it is expected.

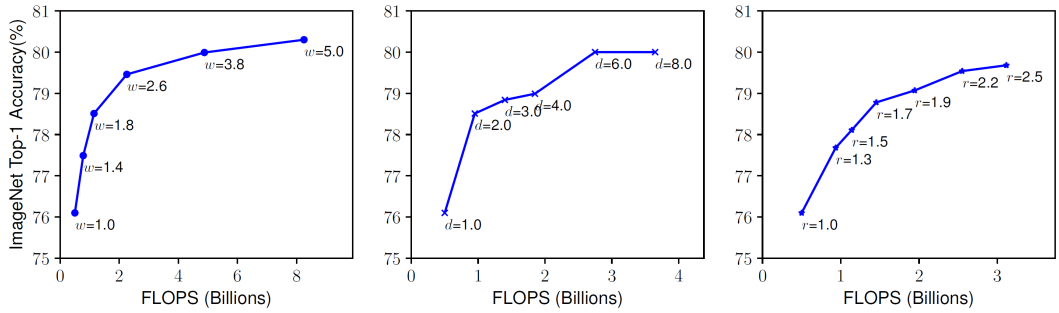


Figure 2.15: Scaling Up a Baseline Model with Different Network Width (w), Depth (d), and Resolution (r) Coefficients [17]

In Figure 2.15, effect of individually scaled width, depth and resolution parameters on the accuracy and flops can be observed. Scaling these three parameters eventually end-up with saturation in accuracy and after some threshold increasing these parameters just decreases the model speed. According to [39], increasing width of model, that is increasing number of channels in the model, helps detecting fine-grained features. However, as discussed in [17], solely increasing width of a shallow network does not provide a huge improvement and network might still have difficulties in detecting fine-grained features.

Increasing depth of model has a similar effect on network in capturing high-level features. Additionally, increasing depth of network helps model to generalize better on unseen datasets. However, as stated in [17], increasing depth of model creates vanishing gradient problem and it becomes difficult to train these networks. In networks with large number of layers, gradient of the loss function approaches to zero when it is calculated for first layers in back propagation operation. This problem is generally caused by some activation functions like sigmoid activation function. Since, sigmoid function projects a large input space into a small one and the derivative of this function is small, gradient of loss function becomes smaller when passes a function like sigmoid in back propagation. As proposed in [17] too, this vanishing gradient problem can also be solved by using skip connections and batch normalization.

Skip connection is the name given to the link between two nonsuccessive layers. With a skip connection, output feature map of a layer is given as input to another layer that

is not adjacent. In the accepting layer two feature maps, one from skip connection and one from previous layer, will be taken as input. These two feature maps are in general combined using concatenation operation and then processed in the accepting layer. These connections between the early layers of the network and the layers at the end help to solve vanishing gradient problem. During back propagation skip connections are used to update weights of earlier layers in the network.

Batch normalization is a set of operations applied to a batch of input data in order to reduce internal covariate shift. When batch normalization is applied, the batch of inputs is simply normalized by subtracting the batch mean and then dividing by the batch's standard deviation. It will be explained in detail in Activation & Batch Normalization section. This operation helps solving vanishing gradient problem by preventing the gradients from getting too large or too small.

Using higher resolution images as input to the model expectedly helps model to capture fine-grained features. As seen in Figure 2.15, after some increment in resolution model accuracy saturates like other parameters. However, increase in resolution exponentially increases flops of the model compared to other parameters.

Based on above studies about parameter scaling, Tan *et. al* [17] reached 2 observations. According to their first observation, upscaling depth, width and resolution of network increases accuracy, but as model gets larger with increment of these parameters, the accuracy increase diminishes. Their second observation states that, in order to get better accuracy, scaling of these parameters need to be done by some specific ratio. Thus, they propose a compound scaling method to scale these parameters for different model types. Their proposed compound scaling method is based on the following formula:

$$\begin{aligned} \text{depth: } d &= \alpha^{\phi} \\ \text{width: } w &= \beta^{\phi} \\ \text{resolution: } r &= \gamma^{\phi} \\ \alpha \cdot \beta^2 \cdot \gamma^2 &\approx 2 \\ \alpha \geq 1, \beta &\geq 1, \gamma \geq 1 \end{aligned} \tag{2.1}$$

ϕ is a compound coefficient, used to scale depth, width and resolution of the network while α, β, γ are constant values, determined by a grid search. In the paper [17], the authors propose a two-step process to determine values of coefficients and constants. In first step, ϕ is fixed to 1 and by using grid search α, β, γ values are determined. For baseline EfficientNet-B0, these values are determined as α=1.2, β=1.1, γ=1.15. In second step, α, β, γ values are fixed and value of ϕ is scaled to get different network structures called EfficientNet-B1 to B7 by using Equation (2.1). Corresponding, depth, width and resolution coefficients of obtained network structures can be seen on table 2.1. According to these obtained coefficient values and repetition, structure of structure of the network is altered and EfficientNet-B0 to B7 are obtained.

Table 2.1: EfficientNet Model Coefficients

	Width Coefficient	Depth Coefficient	Resolution Coefficient
EfficientNet-B0	1.0	1.0	224
EfficientNet-B1	1.0	1.1	240
EfficientNet-B2	1.1	1.2	260
EfficientNet-B3	1.2	1.4	300
EfficientNet-B4	1.4	1.8	380
EfficientNet-B5	1.6	2.2	456
EfficientNet-B6	1.8	2.6	528
EfficientNet-B7	2.0	3.1	600

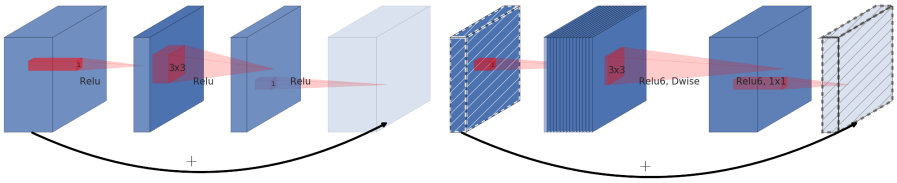
Scaling parameters of the network solely, may not be enough for the best accuracy always. A good network structure is essential to have a good accuracy. Tan *et. al* [17], used Neural Architecture Search to develop a good baseline network structure. Neural Architecture Search (NAS) is an automated design method used for neural networks. It provides the most efficient network design for given needs. While determining network structure they used a specific operational block called MBConv [18] as a main block. MBConv is an Inverted Residual Block proposed in [18] that is optimized for accuracy and consists of several convolution operations. After Neural Architecture

Search, they developed and proposed their baseline network called EfficientNet-B0. Structure of EfficientNet-B0 model can be seen on Table 2.2.

Table 2.2: EfficientNet-B0 baseline network

Stage i	Operator F_i	Resolution H_i x W_i	#Channels C_i	#Layers L_i
1	Conv3x3	224 x 224	32	1
2	MBConv1, k3x3	112 x 112	16	1
3	MBConv6, k3x3	112 x 112	24	2
4	MBConv6, k5x5	56 x 56	40	2
5	MBConv6, k3x3	28 x 28	80	3
6	MBConv6, k5x5	14 x 14	112	3
7	MBConv6, k5x5	14 x 14	192	4
8	MBConv6, k3x3	7 x 7	320	1
9	Conv1x1 & Pooling & FC	7 x 7	1280	1

As stated before MBConv used in EfficientNet is an Inverted Residual Block. It has inverted structure of a Residual Block that is first proposed in ResNet [37]. In neural networks increasing the depth of network by adding more layers does not always make the network train better and reach to a better accuracy but sometimes causes degradation in accuracy. While the model is expected to converge to a higher accuracy, its accuracy saturates and eventually degrades rapidly. He *et. al* [37] claimed that using Residual Blocks helps solving this problem. As seen on Figure 2.16, a residual block has a skip connection and it compresses the data by reducing number of channels and then it expands the data by increasing back the number of channels. With the help of this reduce and increase in the channels, a residual block extracts the useful features more easily. On the other hand, skip connection in the block that connects input to the output prevents loss of data during channel reducement. In the output, compressed and expanded data is added with input data.



(a) Residual Block (b) Inverted Residual Block

Figure 2.16: Structure of Residual and Inverted Residual Block [18]. Thickness of each block shows the relative number of channels.

Inverted residual block follows the reverse of procedure applied in a residual block. First, the data is expanded by increasing number of channels and then it is compressed by decreasing number of channels. Skip connection works in the same way and input data is added to the output data at the end. There are two types of MBConv blocks used in EfficientNet architectures, MBConv1 and MBConv6. The number after the MBConv indicates the expansion size of the block. In MBConv6, the channels are expanded 6 times, while in MBConv1 the channels are not expanded at all.

MBConv blocks in EfficientNet are a version of inverted residual block but have some extra modifications. First, activation functions are changed to Swish activation functions. Structure of swish function is similar to widely-used ReLU and its function is $f(x) = x \cdot sigmoid(x)$. Swish activation function will be explained in detail in Activation & Batch Normalization section with other activation functions. Compared to ReLU, swish activation function shows better performance in deeper networks. Second, a depthwise separable convolution layer is added to the block after the expansion layer. When computational resources are restricted, using depthwise separable convolution layer is more advantageous as they need less resource due to their separable structure.

After the depthwise convolution layer sometimes a squeeze-excitation block is added into the MBConv block. The squeeze-excitation (SE) block proposed in [40], focus on channel relationships in the network. They claim that in a feature map not all channels have the same importance so they must be weighted according to their importance. In a squeeze-excitation block, all channels are reduced to a single value

first. Then, two fully connected layers followed by activation functions adds non-linearity to these values. In the last stage, obtained values are used to weight each corresponding channel by multiplication. According to the [40], SE block improves performance of the network significantly in the exchange for a small computational cost. Structure of MBConv6 block and squeeze-excitation block can be seen on Figure 2.17.

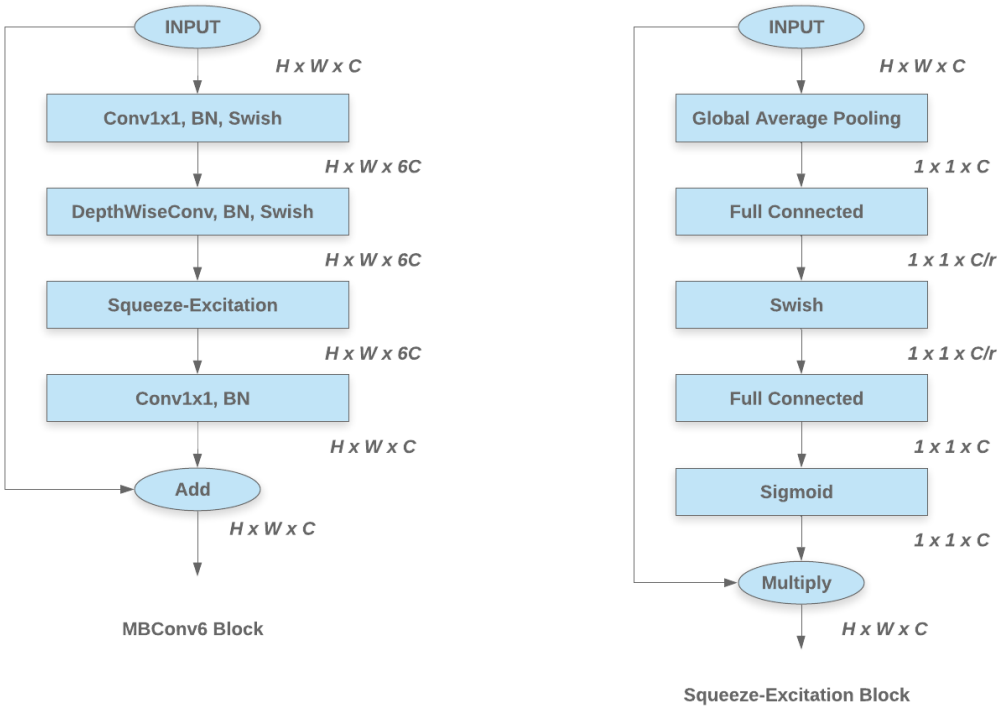


Figure 2.17: MBConv6 Block and Squeeze-Excitation Block

Since it is published, studies that use EfficientNet model family proposed by Tan *et. al* [17] have increased unexpectedly fast. Liebl *et. al* [41] pointed out the challenge in separation of text from non-text content in optical character recognition (OCR) of historical documents. They approached to the problem like a segmentation problem and they trained their models using supervised learning. EfficientNet-B1 and B2 models were employed in their studies and they indicated that EfficientNet employed network gives one of the best results among 11 other networks. Another computer vision related study was carried out by Zoph *et. al* [42] about semantic segmentation task. Their network architecture with EfficientNet outruns state-of-the-art DeepLabV3+

[43] architecture in PASCAL VOC 2012 segmentation dataset [44]. Moreover, they achieved this with EfficientNet-B7 model used with one third of FLOPs compared to DeepLabV3+ [43].

Image classification is one of the most studied computer vision task in deep learning. Noisy Student Training method proposed by Xie *et. al* [1], improves image classification accuracy of EfficientNet Networks on ImageNet Dataset [2]. The proposed training method is a semi-supervised learning method called Teacher-Student learning. The training process consists of two steps. Firstly, an EfficientNet based network is trained on ImageNet [2] with labeled images as a teacher network. Then, 300M unlabeled images are provided to the network and pseudo labels of these images are taken as output from teacher network. In second step, another EfficientNet based network is trained with labeled and pseudo labeled images as a student network. At this stage, two types of noise is additionally included into the process which are input noise and model noise. Both of them is used to make student network generalize better. Input noise is added to the training as data augmentation. On the other hand, regularization methods like dropout and stochastic depth are known as model noise and they are also injected to the training process of student network. The proposed Noisy Student Training process improves self-training and increases EfficientNet's accuracy in ImageNet image classification task by noticeably. The Teacher-Student learning process and the details of this paper will be provided in Teacher-Student Learning Section.

HigherHRNet [45] is an human pose estimation network proposed by Cheng *et. al*. Using high resolution future pyramids in their network they have achieved the state-of-the-art accuracy for human pose estimation problem. Neff *et. al* [46] proposed merging HigherHRNet [45] with EfficientNet [17], in order to decrease computational load of the network and increase accuracy in human pose estimation task. They used the advantage of scalable architecture of EfficientNet models from B0 to B7. They state that, by scaling EfficientNet models, they managed to set up lightweight networks for human pose estimation while maintaining accuracy competitive with other state-of-the-art models.

Tan *et. al* [47] have modified EfficientNet for Object Detection task and introduced

EfficientDet network architecture. Similar to EfficientNet, EfficientDet architecture is also based on scalable depth, width and resolution parameters. They imply that with their network structure, accuracy obtained by other state-of-the-art models is achieved in object detection with 4x – 9x smaller network sizes.

Object detection architectures are used even in diagnosis of certain diseases from medical images. Anwar *et. al* [48] have studied diagnosis of COVID-19 from CT (Computed Tomography) scan images using an EfficientNet based network structure. By using COVID and non-COVID CT scan images for supervised learning, they approached to the problem like an object detection problem and achieved almost 90% accuracy in detection of infected lungs from CT scan images.

2.3 Activation & Batch Normalization

There are many activation functions that can be used in deep neural networks based on the problem need to be solved. Depending on the choice of activation function, performance and accuracy of the overall architecture can be increased. In deep learning studies one of the most used activation function is Rectified Linear Unit (ReLU). As given in the equation 2.2 ReLU has non-negative activation so mean of activation is always above zero and it creates an undesirable bias for next layers in the network. If mean activations of layers in the network are around zero learning process might become faster and convergence time will be decreased.

ReLU Activation Function:

$$f(x) = max(0, x) \tag{2.2}$$

Unlike Rectified Linear Unit (ReLU), Exponential Linear Unit (eLU) can generate negative outputs as given in equation 2.3. It is differentiable and continuous at all points. Compared to ReLU it might be slower but compared to other non-saturating activation functions, eLU still provides speed advantage during training time. As provided in equation 2.3, eLU function consists of a parameter α that is used to control when function saturates for negative input values.

eLU Activation Function:

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha(e^x - 1) & \text{if } x < 0 \end{cases} \tag{2.3}$$

Similar to eLU, Sigmoid is a continuous and differentiable activation function. However, it has a disadvantage that it creates vanishing gradient problem if it is used too much in hidden layers of the network. Sigmoid projects a large input space into a small one so gradient of loss function becomes smaller as it passes a Sigmoid function in back propagation. However, this problem can be eliminated in general if skip connections are used. Formula of Sigmoid Activation Function is provided in equation 2.4. Its outputs are bounded in the range of (0,1).

Sigmoid Activation Function:

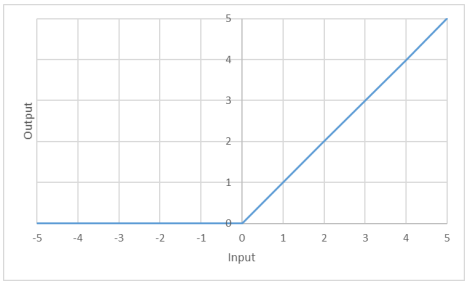
$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.4}$$

Swish Activation Function is similar to ReLU in terms of structure but it is continuous and differential at all points. As explained before in EfficientNet section of Chapter 2, it is used in EfficientNet structure. In equation 2.5 formula of Swish Activation Function is given.

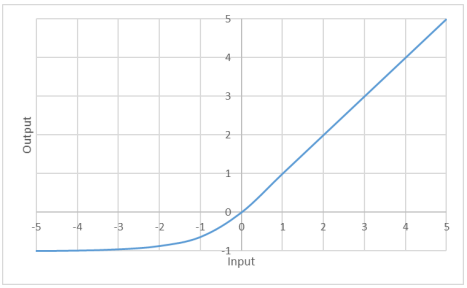
Swish Activation Function:

$$f(x) = x \cdot sigmoid(x) \tag{2.5}$$

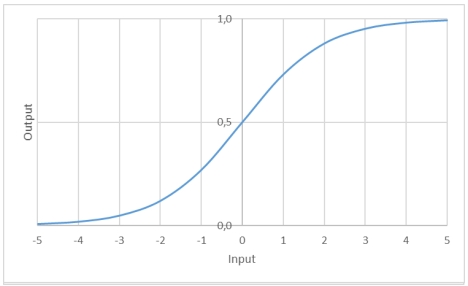
Graph of all four activation functions are provided in the Figure 2.18 for comparison.



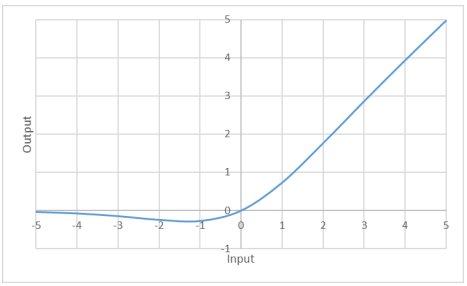
(a) ReLU Activation Function



(b) eLU Activation Function



(c) Sigmoid Activation Function



(d) Swish Activation Function

Figure 2.18: Activation Functions

As explained before in EfficientNet section, Batch normalization helps solving vanishing gradient problem. If the variance of data transferred from one layer to another is too large, during back propagation some gradients calculated for weight update would be so small while some of them is very large. These small gradients will eventually vanish before they reach to earlier layers. If Batch normalization is applied, calculated gradients will have small variance and the vanishing gradient problem might be solved.

Besides its effect on the vanishing gradient problem, Batch normalization also accelerates training by standardizing the data in a batch between layers. Formula of Batch normalization is provided in equation 2.6. There are two trainable parameters of Batch normalization operation, γ and β. During back propagation these parameters are updated and the normalization applied to the batch after an activation layer in forward propagation is reversed by using only these two parameters. ϵ is a small

constant added to the variance in order to avoid dividing by zero.

$$
\begin{aligned}
\mu_B &= \frac{1}{m}\sum_{i=1}^{m} x_i \quad \text{//Batch Mean} \\
\sigma_B^2 &= \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_B)^2 \quad \text{//Batch Variance} \\
\hat{x}_i &= \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \quad \text{//Normalize} \\
y_i &= \gamma \hat{x}_i + \beta \qquad \text{//Scale and Shift}
\end{aligned}
\tag{2.6}
$$

2.4 Teacher-Student Learning

In deep learning applications lack of training data and the limited computational resources are the biggest motivations in developing different techniques. With this motivation and due to it's wide applicability, teacher-student learning methods appear in many different studies in deep learning area. Teacher-student learning approaches are often called as knowledge distillation. In general, the smaller of the two networks is called as student and it is trained by the supervisory signals that come from the larger network. These supervisory signals might be gradient required for the weight update of student network or any kind of data that can be used in training of student network. Size of student network can be small or same compared to teacher network. If a network that is lighter than the original network but has similar accuracy is required, then the student network can be built smaller than the teacher network.

Bucilua *et. al* [49] were one of the first to propose a method to train a smaller network with the help of a larger original network. Motivation of their work was to create a compressed model of their original model to use it in computational power or storage space limited applications. They called their method as model compression and did not refer their large and small networks as teacher and student but the procedure applied in their work lead the way for teacher-student learning methods. The idea behind their method was to use a largely trained model to create pseudo labels for a pseudo dataset. Then, by using these pseudo dataset and labels created, they trained a smaller network to mimic original large network.

Later Hinton *et. al* [50] popularized the idea by using knowledge distillation description for a similar process, transferring knowledge from a large model to a small model. In their method which is also known as vanilla knowledge distillation, teacher model is much larger than the student model. With the help of supervisory data coming from teacher model, student model tries to generate same outputs or soft labels of teacher model. Supervisory data from the teacher does not need to be taken from the output necessarily. Instead it can be intermediate features that can guide the training of student network. In our case supervisory data predicted by teacher network is the depth map and these depth maps are used to train our student network. These predicted depth maps are called pseudo labels in our work.

With the improvements in teacher-student learning frameworks, more methods have been proposed in the deep learning area. Even some approaches have reached to the state-of-the art performance results while dealing with monocular depth estimation problem. The teacher-student approach proposed by Pilzer *et al.* [19] has a novel self-supervised architecture. They do not use any supervisory signal from outside of the network. Instead, they use stereo image pairs as input to the network and calculate reconstruction errors in loss function similar to [9] and [10] in order to train their networks. Novelty in their study is that they train their student and teacher networks in a self-supervised fashion.

In Pilzer *et al.* [19]'s architecture, teacher network and student network which is a sub-network of teacher are trained together as seen on Figure 2.19. In their method, student network makes a disparity map estimation for an input image first. Then, opposite view image is reconstructed using input image and the predicted disparity map. Then, the overall network, called Teacher, calculates reconstruction errors and implements inconsistency checks for the reconstructed image. Calculated loss is used by teacher to train itself along with student network.

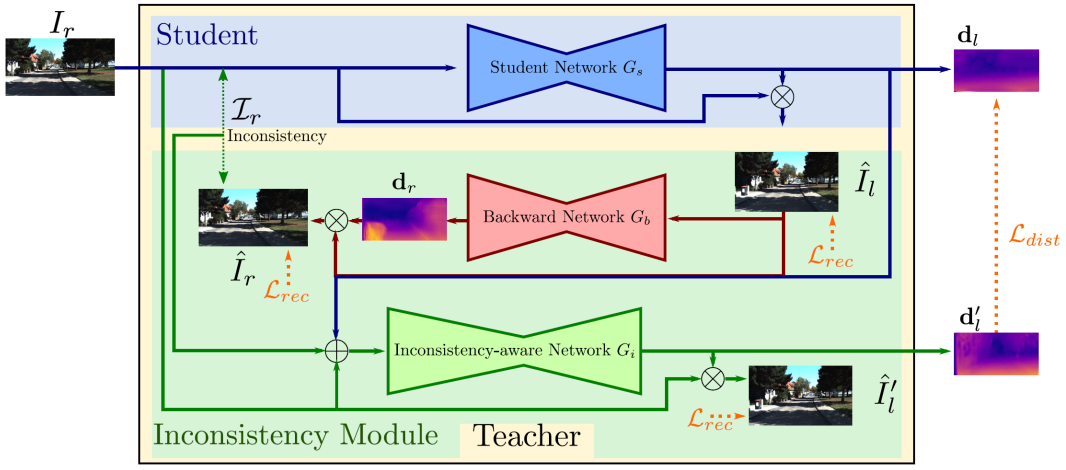


Figure 2.19: Pilzer *et al.* [19]'s network structure.

In Pilzer *et al.* [19]'s approach, since training of the student by the teacher is distillation of knowledge from overall network to a sub-network of all structure, they called it self-distillation. As stated before in Unsupervised Learning section, according to their evaluation results, their student network shows worser performance than the teacher and this would be expected since they use student network to improve performance of teacher.

In Cho *et al.* [16]'s proposed method teacher and student networks have different structures. Both of them predicts depth maps but they do it by using different methodologies. Their teacher network is a Stereo Matching Network and uses stereo image pairs to predict a depth map while student is a shallow network that predicts depth by using single images. In other words, their student network is a monocular depth estimating network while the teacher is not. Additional to auxiliary information provided by teacher network they also provide stereo confidence maps as supervision to the student network during training to prevent the student from using inaccurately predicted depth maps. They mainly focused on domain adaptation in their work but they also proved that using pseudo ground truth depth maps to train a student network can provide good results.

Ye *et al.* [20] proposed a method to train a student network by using more than one teacher network, which they call knowledge amalgamation. They have two pre-

trained teacher networks for scene parsing (semantic segmentation) and depth estimation. Both these networks and the student network have same architecture. Basically their method depends on exchanging pre-trained network blocks between teacher networks and the student. They called this process as knowledge amalgamation.

In Ye *et al.* [20]'s architecture, student network's decoder part splits into two at some point to output both scene parsing and depth estimation results as seen on Figure 2.20. The main objective here is to train an encoder to extract both depth and scene parsing features at the same time. During training of the student network they try to match extracted features and final predictions with the teacher networks outputs. They stated that, according to their results, student network surpasses the teacher networks in their own area of specialties.

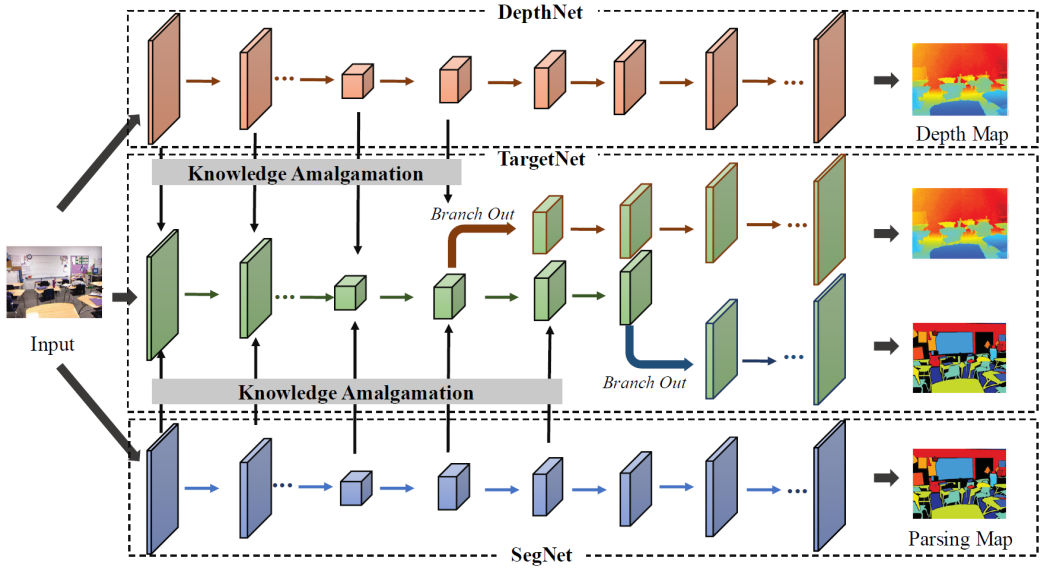


Figure 2.20: Ye *et al.* [20]'s network structure.

Similar to monocular depth estimation, semantic segmentation is one of the most studied problem of deep learning area. Many methods including teacher-student learning procedures [51], [52], [21] have been applied to reach state-of-the-art performance in this topic. Chen *et al.* [21] focused on the problem of inadequate number of ground truth labeled datasets and how hard it is to obtain them. Their solution includes a labeled small dataset and an unlabeled dataset in training of a network. They proposed to train a teacher network with labeled images and then use this network to

predict labels of an unlabeled dataset. Then in the last stage, they use both labeled and pseudo labeled dataset in training of a student network. Their student and teacher network has a shared encoder, which enables student network to learn from features that are extracted by teacher network. In their work Chen *et al.* [21] investigated the amount of labeled data needed for their student network to pass performance of fully supervised teacher network. They show that when same amount of labeled data is used in training of both teacher and student, at some point with the help of pseudo labeled dataset, student network can pass the performance of teacher network.

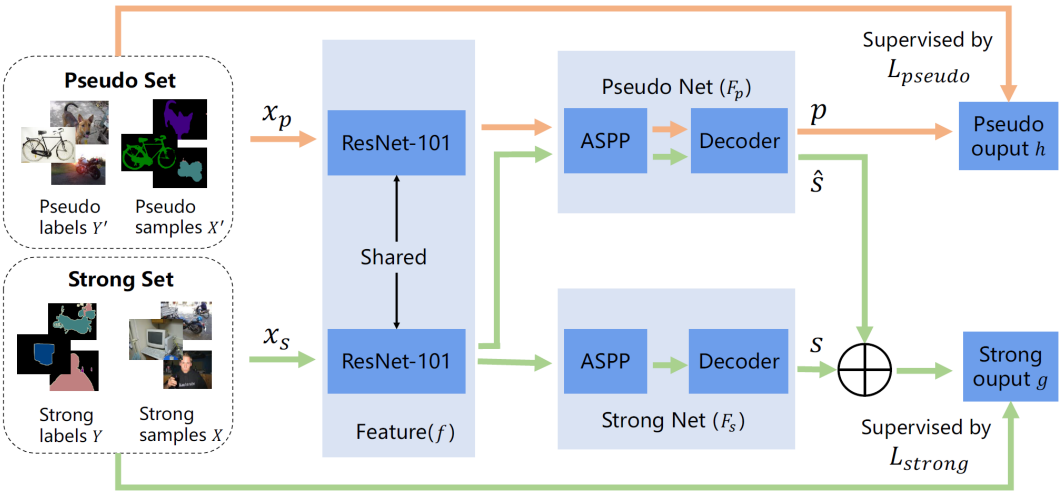


Figure 2.21: Chen *et al.*'s network structure. Image is taken from [21]

In most of the teacher-student learning methods, student network is smaller than the teacher network and the main objective is training a student network that can mimic the teacher network. As mentioned in EfficientNet section, Xie *et. al* [1] brought a new perspective to the teacher-student learning methods. They proposed using a student network that is equal-or-larger than the teacher network and adding both input noise and model noise to the training process of the student. Main difference of their work from previous teacher-student learning methods is that they add the noise very aggressively to make student network outperform the teacher and generalize much better than it.

In Xie *et. al* [1]'s approach, noise is injected by three methods. Data augmentation is applied during training as input noise. On the other hand, Dropout [53] and stochastic

depth [54] are applied as model noise. Model noise methods are also known regularization methods. Data augmentation is a well-known and commonly used input noise method in deep learning. During training altering input data is called data augmentation and generalization of the network for unseen input data is expected by using this method. If input data given to the network is an image, then properties like color, brightness, saturation and angle of it can be changed before it is fed into network. Moreover, it can be stretched or cropped.

Dropout is another commonly used method in deep learning that is proposed by Srivastava *et. al* [53]. Dropout is an operation applied to the layers. According to a probability rate during training some randomly chosen units are not taken into account while calculating the layer output. Those chosen units are called dropped and they are ignored during that forward pass of the network. This forces network to extract required features with less number of units. Most important expectation from the dropout operation is to reduce over fitting of the network.

Stochastic depth process is proposed by Huang *et. al* [54] and the goal is same with the dropout process. While dropout removes units during training stochastic depth removes layers and replaces them with identity functions. This enables the network to become shorter during training and train in fewer time. It also prevents overfitting and makes the network generalize better on unseen data. All these three noise elements are only used in training and not used in testing.

As mentioned before, Xie *et. al* [1] has a two-step training process. Teacher network is trained with a labeled dataset without injecting any noise. Then, this trained network is used for creating pseudo labels for an unlabeled dataset. In second stage, student network is trained with labeled and pseudo labeled datasets by injecting noise into the training. However, their method does not ends at this stage. Their method's another difference from previous works is that they propose training more than one student network in an iterative way. When they train first student, it becomes their new teacher and gives pseudo labels to the unlabeled dataset again. With labeled dataset and updated pseudo labeled dataset they train another student. They claim that, this iterative learning strategy can continue until accuracy reaches at top or over-fitting starts. Their proposed iterative learning process is given in Figure 2.22.

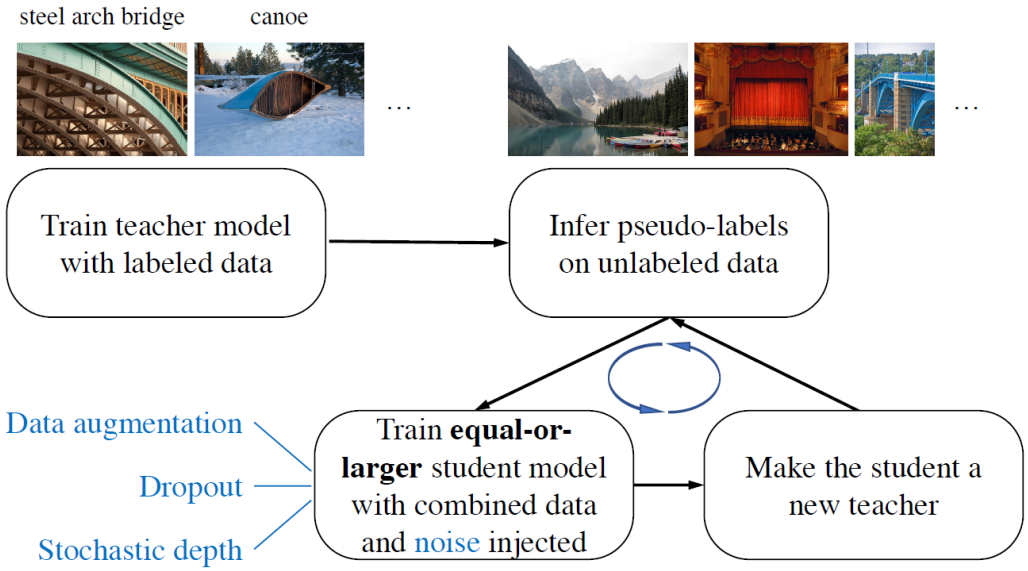


Figure 2.22: Xie *et al.* [1]'s iterative learning method.

They show the effectiveness of iterative training and share the acccuracy results of their image classification network tested on ImageNet Dataset [2]. During three iterative learning process their network continuously increases its accuracy. They have also observed that their student network trained on labeled and pseudo labeled datasets outperforms the network that is pretrained on unlabeled dataset and finetuned on labeled dataset. This comparison reveals the effectiveness of training on both labeled and pseudo labeled dataset jointly.

In their study Xie *et. al* [1] also drew attention to the ratio of unlabeled data to labeled data used in training of student networks. They claim that if the number of images in unlabeled dataset increases compared to labeled dataset, student networks can generalize better. In order to increase performance of third student network they increase the ratio of images in unlabeled and labeled dataset from 14:1 to 28:1 in third iteration. One of their findings is using a large ratio between unlabeled and labeled data number increases accuracy of the student network and its generalization ability as well. They also suggest that joint training on both datasets provides much better results than training separately or fine-tuning on any of them.

Xie *et. al* [1] evaluated the effect of injecting noise into the network as well. Since

they train student networks with a much larger dataset compared to teacher network, they thought performance increase in the student networks may not be resulted from the injected noise but it might be a result of increasing the number of training data. Therefore they have trained and evaluated their networks with and without noise in order to observe effectiveness of noise in performance. They have used EfficientNet-B5 model size for their teacher and student networks. They state that, in order to save time they did not use iterative training and same number of data is used for labeled and unlabeled datasets. Their evaluation results are given on Table 2.3. They have trained their networks on ImageNet Dataset [2] for image classification problem. Original EfficientNet-B5 network accuracy result is also provided on the Table 2.3 for comparison.

Table 2.3: Effect of injecting noise into the training [1]. Evaluation results on ImageNet Dataset [2] for image classification problem.

Model Name	Top-1 Accuracy
Original EfficientNet-B5	83.3%
Noisy Student Training (B5)	83.9%
Student w/o Aug	83.6%
Student w/o Aug, SD, Dropout	83.2%
Teacher w. Aug, SD, Dropout	83.7%

With their findings and their proposed noisy iterative learning process, Xie *et. al* [1] made a great contribution to the deep learning literature. In this thesis, we create a monocular depth estimation network and train it similar to Xie *et. al*'s method, teacher-student iterative learning.

2.5 Dataset

KITTI Dataset [22] contains several images from outdoor scenes captured while driving with a car. For each scene there are images taken from different angles of the car

together with depth maps obtained by Lidar depth sensor. These captured images and depth maps are synchronized and calibrated by publishers. There are a total of 42,382 stereo image pairs from 61 scenes in KITTI dataset. Images in the dataset generally have 1245x375 pixels in size. All these image pairs have raw depth maps obtained by Lidar but not all of these depth maps are reliable to use. Some of these images and depth maps still have calibration faults, measurement errors etc. Lidar measurements are sparse that is depth maps in the dataset does not have depth measurement for each pixel corresponding to the camera image. Only 5% of the pixels in input image has depth values. Moreover, the angle of view of Lidar is narrow in vertical direction and ground truth depth maps only include depth measurements in the two third of the overall image and one third of the depth map is empty from top.

There are also KITTI 2015 stereo dataset [24] that provides highly detailed 200 ground truth depth maps for specifically selected 200 images. There are left and right views so there are actually a total of 400 images. Due to low frame rate of the laser scanner depth of moving objects are not scanned precisely. In order to solve this problem, Menze *et al.* [24] proposed a method. They first recovered the static background of the scene. After estimating ego-motion of the moving objects, using optical flow they have calculated the precise depth of these objects. Then, they have re-inserted calculated depth of these moving objects into the ground truth depth maps. Since, these 200 image-ground truth pairs are highly detailed, they are used for comparison of our modified Monodepth model to the original Monodepth model in the early stages of our work.

According to utilizable data, there are some splits of KITTI proposed by researchers that work on monocular depth estimation problem. Eigen *et al.* [6] have split the KITTI dataset into two for training and testing purposes and the images they have selected from KITTI dataset was named as Eigen Split of KITTI. Their training set includes 23,488 images from different 32 scenes and testing set includes 697 images from remaining 29 scenes. The KITTI dataset also includes opposite (right camera) view of these 23,488 images. In later works, [22] published annotated depth maps of KITTI dataset which includes denser depth measurements compared to initial sparse measurements. 652 images from Eigen test split was provided in annotated depth maps and in the evaluation stage of their model performance many researchers that

work on depth estimation problem have trained their networks on Eigen train split and evaluated their models on Eigen test split. In a short time, Eigen train and test splits have become a fair comparison benchmark for researchers. Some images and corresponding depth maps from KITTI dataset can be seen on Figure 2.23. Since lidar depth measurements are sparse as seen in the images on second row, we have applied an interpolation procedure to make them more perceivable. These dense ground truth images are shown on last row.

Figure 2.23: KITTI Dataset [22] sample images. From top to bottom: Sample Image, Sparse Ground Truth Depth Map, Dense Ground Truth Depth Map

In our work, we have used Eigen train split of KITTI dataset to train our models and evaluated them on Eigen test split in order to fairly compare our results with other studies. As explained before, there are also right camera views of each image included in Eigen train set. So, there are approximately 46,000 images with ground truth depth maps that can be used in training. For our teacher-student learning method, we use some of these images with ground truth to train our teacher model in a supervised manner and use rest of them to predict depth maps. Overall training procedure will be explained in Teacher-Student Learning section of Chapter 3.

CHAPTER 3

PROPOSED METHOD

In this chapter, proposed network architecture will be presented in detail. First, overview of encoder and decoder parts of the network architecture and the training losses used in the training will be explained. Later, the iterative teacher-student learning procedure that has been followed during training of our models will be explained in detail.

3.1 Network Architecture

Most of neural networks consist of three main parts, encoder, decoder and training loss. Encoder structure is composed of neural layers which calculates convolutional operations, pooling operations etc. Input to an encoder can be any type of data, an image, a recorded voice even a written text while the output of it is a feature map/tensor. Encoder part as its name signifies, encodes the data or features given in the input data and provides it to decoder part. Decoder part on the other hand, takes the encoded feature maps and gives the best closest match to intended output. Output of decoder can be a reconstructed image, a probability distribution, a depth map or any type of desired data. The proximity of obtained output to the desired output is calculated by training loss. Overall network model tries to minimize this training loss to get desired output by changing weights used in layers of encoder and decoder parts. The change that will be applied to weight values are calculated using training loss with gradient operations and applied to the layers through back-propagation. Our proposed overall network structure is given in Figure 3.1

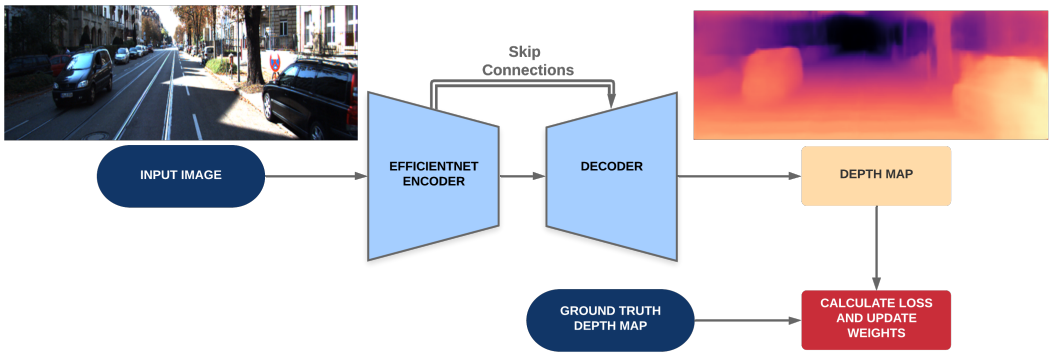


Figure 3.1: Our proposed network structure.

3.1.1 Endoder Architecture

VGG [55], ResNet [37] and Xception [38] are well known and mostly used encoders in deep learning area. Performance of these structures in most deep learning studies are non-negligible. On the other hand, there are other proposed encoder architectures like EfficientNet [17] that show success on the applied problems. EfficientNet has showed a great success on Image Classification problem of ImageNet [2] and drew attention since then. However, EfficientNet is a rather new concept in deep learning field and the problems that it has been applied are still few. To the best of our knowledge, during this study, it was not applied to the monocular depth estimation problem yet. Therefore, considering its high performance on the problems it is applied we proposed a model including EfficientNet as encoder and tested it on depth estimation problem.

Since EfficientNet was not used in any monocular depth estimation networks previously, we wanted to see it's performance before we proceed further on our model. Therefore, we have decided to choose a well-performed open-source monocular depth estimation network for our comparison. Godard *et al.* [10] have proposed a depth estimation model trained in an unsupervised fashion. Their model was one of the most referenced paper in depth estimation area since they have shared every single file they have used, all trained models and all hyper-parameter values. Because of their open-source content, we have chosen Monodepth [10] to evaluate EfficientNet's

performance.

Godard *et al.* [10] have used ResNet and VGG as encoders in their model called Monodepth. Their encoder and decoder part are two different blocks. Therefore, we have replaced their encoder block with the EfficientNet-B7 encoder. For a fair comparison we did not used any pre-trained EfficientNet model parameters in our network. Monodepth's unsupervised learning depends on using stereo image pairs during training. For both input images, their model outputs two disparity maps of provided scenes. Then by using opposite view disparity maps and original input images, other views are reconstructed by spatial transformation. In other words, by using right image and left disparity map, left image is reconstructed and by using left image and right disparity map, right image is reconstructed. In the Figure 3.2, only one part of the reconstruction is given for simplicity.

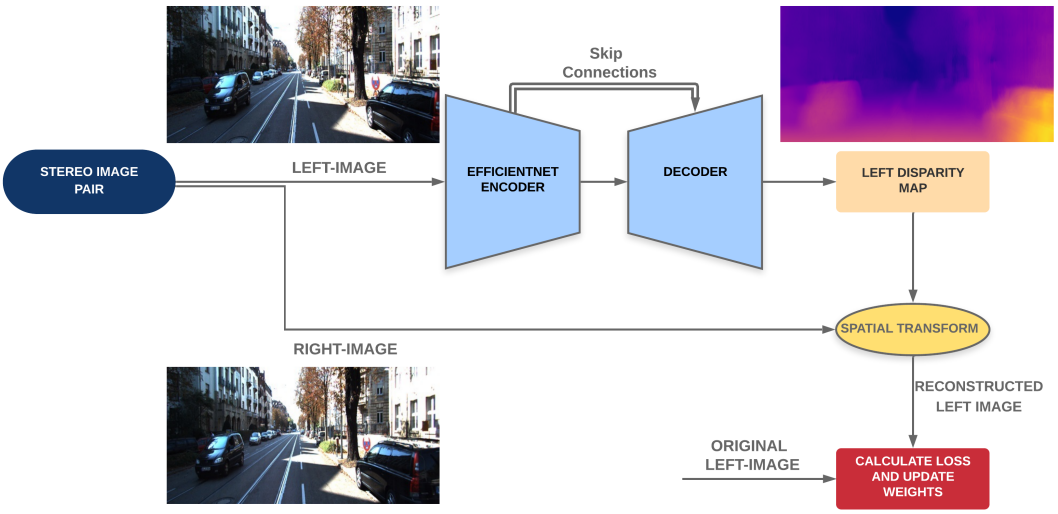


Figure 3.2: Modified Monodepth network structure. Encoder part is replaced with EfficientNet

After modified Monodepth network is tested, evaluation results were good as expected and accuracy of the model was increased compared to original Monodepth. In fact, the results were better than expected. Monodepth was trained on full training set of KITTI dataset while we could only train it on 1/3 of KITTI dataset due to resource limitations. Since, two input images are used during training, required computational resource was heavy. Although, our model with EfficientNet encoder was trained with

a smaller subset of KITTI dataset, the evaluation results were much better than the original Monodepth model. Details of these results will be given in Experimental Results chapter.

3.1.2 Decoder Architecture

Objective of encoder part is to extract features from an input data and transfer them to decoder part, while in this part, objective is combining these features to produce a meaningful output. In our case, since depth estimation is a representation problem and aim of our decoder part is to recover depth map for a given input image. In encoder part, input image is reduced to 1/32 of its original size with pooling operations. While width and height of input image is reduced, to prevent data loss channels of the feature map are also increased. In an RGB image channel size is 3, while in our model, encoder outputs a feature map with 2560 channels. Extracted features are stored in these 2560 channels and given to the decoder part. In a monocular depth estimation network, decoder part is responsible for processing the extracted features to get desired output while recovering the original size of the input image at the output.

Main operations used in decoder part are in general convolutional operations, resizing operations and concatenation of skip connections obtained from previous layers. Resizing can be applied by image resizing methods or upsampling functions that includes convolution operation. Although, there are not exact results for our particular case, using upsampling and deconvolution functions that has trainable parameters inside, may cause checker board artifacts in the output image as mentioned in [56]. Therefore, we have preferred to use image resizing functions while upscaling our feature map in the decoding phase.

As explained before, skip connections help dealing with vanishing gradient problem. Therefore, we have used skip connections between our encoder part and decoder part. Long skip connections from encoder part to decoder part transfers low-level features extracted in earlier layers and ensures flowing of maximum feature information between two architectures. These connections also help to recover spatial information better during up-sampling. We get skip feature maps from 5 different step of encoder part so each one of them has different sizes in width and height. These skip feature

maps are concatenated with the same size feature maps at the decoder part. Batch normalization is also used to deal with vanishing gradient problem. Besides it accelerates training by normalizing the batch of inputs. Due to these reasons we have decided to use Batch normalization too in our decoder part.

As explained before in Activation & Batch Normalization section of Chapter 2, Exponential Linear Unit (eLU) can generate negative outputs and compared to activation functions, eLU provides speed advantage during training time. Due to these reasons we have decided to use eLU activation function in our hidden layers of decoder part. As given before in equation 2.3, eLU activation function has a parameter α that determines when function saturates for negative input values. In our case we have used the suggested default value of 1 for α.

Sigmoid activation functions output is bounded in the range of (0,1). In general it is used to predict a probability in last layer of deep neural networks. We have decided to use Sigmoid Activation Function in the last layer of our decoder part in order to make it generate relative depth values for each pixel. After a relative depth map is obtained at the last layer of our network we scale it to the range of 0 to 80 meters as it is the range of depth maps in KITTI dataset.

There are other alternative techniques to use in decoding stage of the networks. One of them is Atrous Spatial Pyramid Pooling (ASSP), proposed by Chen *et al.* [28] in DeepLab network structure which is a well-known high performance image segmentation network. Atrous convolution (Dilated convolution) is a method applied to enchance field-of-view of filters. Dilation rate is a parameter that is used to define enlargement of a filter. When atrous convolution with dilation rate (r) is applied in a layer, (r-1) zeros are put between each elements of filter. For example, if a convolution with a 3x3 filter and dilation rate 2 is going to applied in a layer, then filter size becomes 5x5. In ASPP, first atrous convolutions are applied at different dilation rates to the same input. This operation helps obtaining useful context information at multiple-scales. Then outputs of these atrous convolution operations are concatenated and 1x1 convolution is applied to obtain final feature map. Using ASPP enables an enhanced field-of-view in the network and incorporate spatially distributed features. According to the results Chen *et al.* [28] shared, using ASPP in their model brought

a significant performance gain.

Since [28] is published, different types of ASPP is proposed in deep learning field. One of these types was proposed by Artacho *et al.* [23] for semantic segmentation problem, called Waterfall Atrous Spatial Pyramid Pooling. Their proposed approach differs from the original ASPP by its structure. In original ASPP, all atrous convolution operations are parallel or cascaded, while, in Artacho *et al.*'s proposed method atrous convolution operations are both parallel and cascaded. When operations are parallel, they all use the same input data which has the most channels. However, when operations are cascaded output of an operation has less channels and the next operation is required to process less number of channels but in return they are lacking the larger FOV. Artacho *et al.* [23] claimed that their approach requires less number of parameters and it is more memory efficient while it provides better performance. Their proposed ASPP structure is given in the Figure 3.3. Our proposed model has a similar structure to waterfall ASPP, but we do not use average pooling operation which is not used in original ASPP as well.

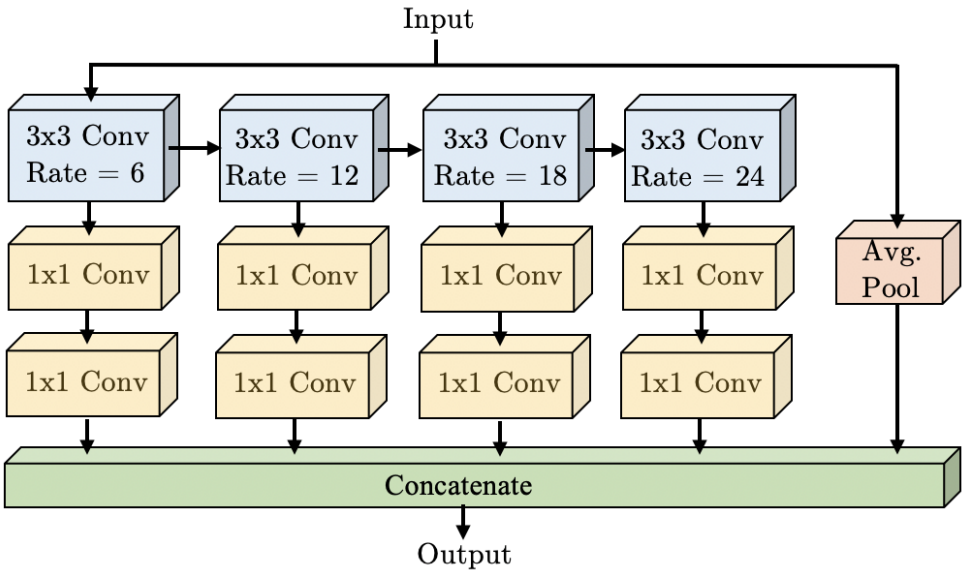


Figure 3.3: Waterfall Atrous Spatial Pyramid Pooling(ASPP) structure [23].

Before we proceed further and start training our model with iterative teacher-student learning mechanism we wanted to observe performance increase of our ASPP structure. While, we were training our modified Monodepth network to observe Effi-

cientNet performance, we have also included ASPP and trained two models with and without using ASPP. We have observed a significant accuracy gain in the evaluation results of model trained with ASPP. Test results of these trained models will be given in Experimental Results chapter. After testing performance of ASPP structure, we have decided to add it into our proposed model.

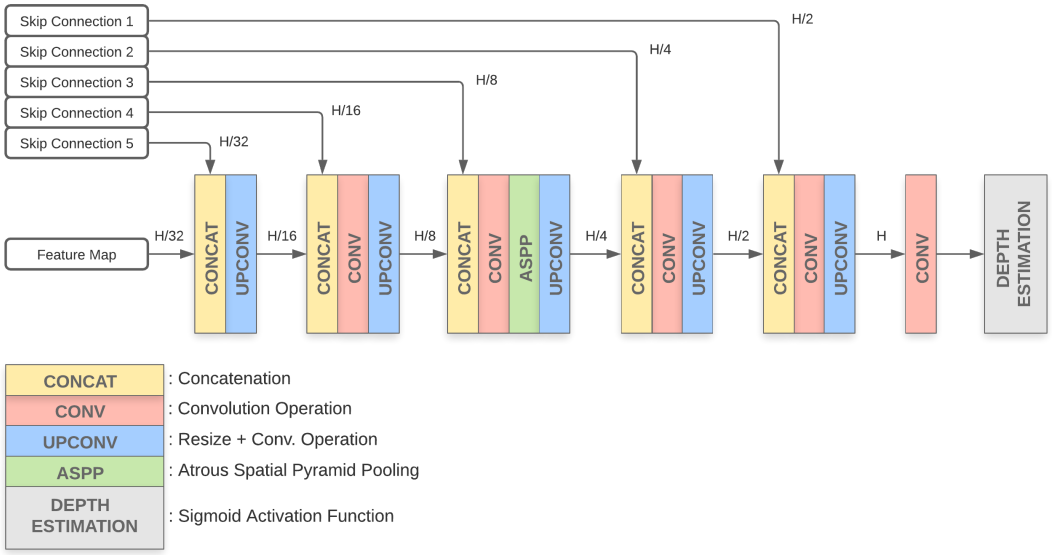


Figure 3.4: Decoder structure used in our network.

Overall structure of decoder part can be seen on Figure 3.4. Feature map and skip connections are obtained from encoder of our network. Concat operation is concatenation of two feature maps without loss of any information. Conv block includes a standard convolution operation. Upconv block incorporates a resize operation which doubles the size of input followed by a convolution operation. At the end of decoder part, a sigmoid activation function is used to estimate depth of every pixel in the output image. In training, ground truth is used to calculate error of estimated depth map that will be mentioned in Training Loss section.

3.1.3 Training Loss

In monocular depth estimation networks pixel-wise regression is applied during training. Difference between the expected and the predicted values are called as error and

the model tries to minimize this error by using an optimization algorithm and optimizes itself. Most known optimization algorithm is stochastic gradient descent but through time additional to it many alternative optimization algorithms have been proposed. One of them is adam optimizer that is proposed by Kingma *et al.* [57]. Adam optimizer have proved its reliability and efficiency in many works it is used. Therefore, we use adam optimizer in our model as well. Apart from optimizer, choosing a training loss function is also important.

Training loss or error in monocular depth estimation networks is chosen based on the learning method used. Unsupervised learning methods incorporate reconstruction losses while in supervised methods deviation between the ground truth depth map and the predicted depth map is used. Semi-supervised methods might be using both of them together with auxiliary geometric knowledge.

In supervised learning methods of monocular depth estimation most preferred loss functions are L1 and L2 loss. They give the absolute and squared difference between the predicted and expected values respectively as shown in equations 3.1 and (3.2)

$$L1\ Loss = \sum_{i=1}^{n} |y_{true} - y_{predicted}| \tag{3.1}$$

$$L2\ Loss = \sum_{i=1}^{n} (y_{true} - y_{predicted})^2 \tag{3.2}$$

Similar to optimization algorithm there are many proposed loss functions that work well for specific tasks. For monocular depth estimation problem Eigen *et al.* [6] proposed a scale invariant error function to be used in a supervised learning method. Their proposed loss function handles the difference between expected and predicted depth values in log space in order to solve scale-problem. Their proposed loss function can be seen on equation (3.4). Lee *et al.* [58] presented a modified version of Eigen's loss function. They claimed that with a higher λ convergence of the model improves and the final performance increases. Their proposed loss function can be

seen on equation (3.5)

$$d_i = log(y_{true_i}) - log(y_{predicted_i}) \tag{3.3}$$

$$Loss = \frac{1}{n}\sum_{i=1}^{n} {d_i}^2 - \frac{\lambda}{n^2}\left(\sum_{i=1}^{n} d_i\right)^2 \tag{3.4}$$

$$Loss = \frac{1}{n}\sum_{i=1}^{n} {d_i}^2 - \frac{1}{n^2}\left(\sum_{i=1}^{n} d_i\right)^2 + (1-\lambda)\frac{1}{n^2}\left(\sum_{i=1}^{n} d_i\right)^2 \tag{3.5}$$

In our learning process, we used the loss function proposed by [58]. In our case, expected depth values are depth value pixels that are available in the ground truth map. Since Lidar depth measurements are sparse in the depth map, zero valued pixels in the ground truth depth maps are not taken into account in loss calculation.

In early stages of our work, while we were evaluating performance of EfficientNet and ASPP, we have trained the modified Monodepth model in an unsupervised manner like proposed by Godard textitet al. [10] in their original work. As explained before their unsupervised learning model were using stereo images to create other view with the help of depth maps. Then, reconstructed left-right images are used with original left-right images to calculate training losses. Their training losses depend on reconstruction losses like SSIM (Structural Similarity Index Measure). In order to make a fair comparison we have used the same training loss functions to train our modified Monodepth model.

3.2 Teacher-Student Learning Method

Teacher-Student learning mechanism has been applied in many studies in deep learning area. While the concept of training of a student network with the help of an teacher network is same, many different methods have been used during training. Increasing or decreasing size of student with respect to teacher network, embedding student network into the teacher network are just some methods. When Xie *et. al* [1] outperformed the state of art image classification perfomances on ImageNet [2], noisy

student learning method applied by them added a new perspective to the literature of deep learning. There are some key points about their research that are provided in their paper one-by-one. One of them is having a student model size same or larger than the teacher leads to better results. Joint training of student model on labeled and unlabeled datasets achieves a higher accuracy compared to separately training on both datasets. We have applied their iterative teacher-student learning procedure into the monocular depth estimation problem which is not applied before to the best of our knowledge. While training our models we paid attention to their findings to achieve a higher accuracy.

As explained before we have used the Eigen split of KITTI in training of our network which is seen on Figure 3.1. After some eliminations, we had a total of 40,526 images that we could use in training. We have used 5,790 of these images with ground truth depth maps and called them labeled dataset, while 34,736 of them is used without ground truth depth maps and called as unlabeled dataset. Thus, the ratio between labeled and unlabeled dataset has become as 1:6. Number of images used in each network is given on Table 3.1.

Table 3.1: Number of images used in training.

Model Name	# of Labeled Images	# of Pseudo Labeled Images	# of Total Images	Ratio
Teacher Network	5,790	-	5,790	-
Student Network	5,790	34,736	40,526	1:6

After splitting the dataset and building our teacher network with EfficientNet we trained it with labeled dataset without injecting any noise. Contrary to some other methods we have not used a pre-trained model for our teacher network and trained it from the scratch. After training of teacher network is completed, we made it to predict depth maps for our unlabeled dataset. We called the depth map predictions of teacher network for unlabeled dataset as pseudo labels. With the predictions of teacher network, we had a total of 40,526 images with depth maps. We then created our student network again without a pre-trained model. Student network's size is same with the teacher network. In the training of student network both input noise and model noise

is injected. Input noise is injected as data augmentation. Model noise methods are also known as regularization methods and they are injected as dropout and stochastic depth as explained before. With the injection of such noise, the generalization and the performance of student network is expected to outperform teacher network. First student network is trained with both labeled and pseudo labeled datasets and after training is completed it becomes the new teacher. As a new teacher our old student network makes depth map predictions for the 34,736 images of unlabeled dataset. Then, these new pseudo labels are replaced with the old pseudo labels. As iterative learning continues, we created our second student network and trained it with the labeled and new pseudo labeled dataset. At this point we have stopped training since we had enough data to evaluate our results. We had one old teacher model and two student models one of which became a teacher afterwards. Our whole simplified training process is given below. Training hyper-parameters and evaluation results of trained model will be given in Experimental Results chapter.

1. Train a teacher network by using labeled dataset
2. Make depth map predictions for unlabeled dataset using teacher network
3. Train a student network by using labeled and pseudo labeled dataset
4. Trained student model becomes the new teacher
5. Make new depth map predictions for unlabeled dataset using new teacher network
6. Train a student network by using labeled and new pseudo labeled dataset
7. Process is completed

CHAPTER 4

EXPERIMENTAL RESULTS

4.1 Performance Criteria

In order to evaluate performance of a monocular depth estimation network, there are some metrics used in the literature. Commonly accepted these metrics can be classified into two sub-classes as error (Abs Rel, Sq Rel, RMSE, log RMS) and accuracy metrics. In error functions lower values are better while in accuracy higher values are better. Formulations of these evaluation metrics are given as follows:

Absolute Relative Difference Error (Abs Rel) :

$$\frac{1}{N}\sum_{i=1}^{N}\frac{|y_i - \hat{y}_i|}{y_i} \tag{4.1}$$

Squared Relative Difference Error (Sq Rel) :

$$\frac{1}{N}\sum_{i=1}^{N}\frac{||y_i - \hat{y}_i||^2}{y_i} \tag{4.2}$$

Root Mean Squared Error (RMSE) :

$$\sqrt{\frac{1}{N}\sum_{i=1}^{N}||y_i - \hat{y}_i||^2} \tag{4.3}$$

Root Mean Squared Error Log (RMSE log) :

$$\sqrt{\frac{1}{N}\sum_{i=1}^{N}||\log y_i - \log \hat{y}_i||^2} \tag{4.4}$$

Threshold :

$$\% \text{ of } y_i \text{ s.t. } \max(\frac{y_i}{\hat{y}_i}, \frac{\hat{y}_i}{y_i}) = \delta < threshold \tag{4.5}$$

y_i and $\hat{y}_i$ denotes the ground truth depth value and predicted depth value of corresponding pixel. N is the number pixels evaluated in a depth image. Since ground truth depth maps are sparse and not every pixel has a depth value. These zero valued pixels are not taken into account when evaluation is being done. The accuracy is described by the percentage of the relative predicted depth value within threshold 1.25^j where j = 1, 2, 3. A higher δ indicates a better accuracy. These 5 evaluation metrics are the most used when evaluating performance of a monocular depth estimation model on KITTI dataset.

In addition to pixel sparsity in depth maps obtained by Lidar sensors, these depth maps have one more problem. Due to placement on top of a car, Lidar sensors scan the environment horizontally. However, in vertical direction their view of angle is limited since they do not move vertically. Due to this reason, depth information in ground truth depth maps of KITTI dataset is limited from top and bottom. On each depth map, about 30 percent from the top and 10 percent from the bottom of the image are blank. These parts of the depth map images are cropped and the rest of the depth map is used for evaluation.

4.2 Test Networks

In section 3.1, architecture of used models are explained. In order to use EfficientNet in our model we were required to evaluate its performance in a monocular depth estimation network since it had not been utilized previously in a depth estimation study. With this motivation we created the modified Monodepth model by replacing the encoder used in Monodepth with EfficientNet encoder. We called this network architecture as "Modified Monodepth Eff.". Then, we integrated ASPP module into the decoder part of Monodepth to evaluate its performance too and called it as "Modified Monodepth Eff.&ASPP". Evaluation results of original Monodepth model will be provided in section 4.4 for comparison. Original Monodepth has VGG network encoder integrated and for training of their model KITTI [22] dataset is used. Details

of this network are given in Table 4.1

Table 4.1: Test networks for evaluation of EfficientNet Encoder and ASPP Module.

Model Name	Train Method	Encoder	ASPP	Training Dataset
Monodepth [10]	Unsupervised	VGG	No	KITTI
Modified Monodepth Eff.	Unsupervised	EfficientNet-B7	No	KITTI
Modified Monodepth Eff.&ASPP	Unsupervised	EfficientNet-B7	Yes	KITTI

After deciding to use EfficientNet and ASPP, we have created our own network architecture which was described in Section 3.1. Since EfficientNet provides scalability for network with compound scaling methodology, we created three teacher networks using EfficientNet-B5, B6 and B7. We trained these networks in a supervised fashion on our selected labeled dataset that is explained in Section 3.2 and evaluated their performances in order to decide which model we can use as a teacher network in iterative teacher-student learning. Noise is not injected during training of these teacher networks. Details of these networks are given on Table 4.2.

Table 4.2: Test networks for evaluation of different EfficientNet model sizes.

					EfficientNet Model Coefficients		
Model Name	Train Method	Encoder	ASPP	Noise	Width	Depth	Resolution
Teacher-B5	Supervised	EfficientNet-B5	Yes	No	1.6	2.2	456
Teacher-B6	Supervised	EfficientNet-B6	Yes	No	1.8	2.6	528
Teacher-B7	Supervised	EfficientNet-B7	Yes	No	2.0	3.1	600

According to procedure of teacher-student learning method, we trained our teacher network in a supervised manner without injecting noise as described in 3.2. We used ground truth labeled dataset to train our teacher network. After teacher network is trained it predicted the depth maps for unlabeled dataset. Then, first student network was trained with the labeled and pseudo labeled dataset that is predicted by the teacher. After first student became the teacher, second student was trained with the

labeled and pseudo labeled dataset that is predicted again by the new teacher. Details of these 3 networks are given in Table 4.3

Table 4.3: Teacher and student test networks. Both Labeled and Pseudo Labeled datasets are parts of Eigen training split of KITTI dataset as described before.

Model Name	Train Method	Encoder	ASPP	Training Dataset
Teacher Network	Supervised	EfficientNet-B7	Yes	Labeled
Student-1 Network	Supervised	EfficientNet-B7	Yes	Labeled + Pseudo Labeled
Student-2 Network	Supervised	EfficientNet-B7	Yes	Labeled + Pseudo Labeled

In order to investigate effects of noise injection more comprehensively, we have also trained teacher networks with noise injected. We have used the Teacher-B5, B6 and B7 networks explained before and evaluated their performances in order to observe effect of noise injection for different scale networks. After training these networks with noise, we chose "Noisy-Teacher-B7" to become a teacher to predict depth maps for unlabeled dataset. Then, with two iterations we have trained two student networks. By applying iterative teacher-student learning procedure here too, we expect to observe how injecting noise to the teacher affects student networks. Details of these networks are given in Table 4.4

Table 4.4: Test networks for evaluation of noise injection to the teacher network during training.

					EfficientNet Model Coeff.		
Model Name	Train Method	Encoder	ASPP	Noise	Width	Depth	Resolution
Noisy-Teacher-B5	Supervised	EfficientNet-B5	Yes	No	1.6	2.2	456
Noisy-Teacher-B6	Supervised	EfficientNet-B6	Yes	No	1.8	2.6	528
Noisy-Teacher-B7	Supervised	EfficientNet-B7	Yes	No	2.0	3.1	600
Student-1 with Noisy Teacher-B7	Supervised	EfficientNet-B7	Yes	No	2.0	3.1	600
Student-2 with Noisy Teacher-B7	Supervised	EfficientNet-B7	Yes	No	2.0	3.1	600

4.3 Implementation Settings

Training of Modified Monodepth models were done with the hyper-parameter settings proposed and applied by Godard *et al.* [10]. They have trained their Monodepth model for 50 epochs with a batch size of 8. Due to system resource limitations, we have trained our Modified Monodepth models with a batch size of 4. As proposed by [10], initial learning rate was 10^{-4} for the first 30 epochs and then learning rate was halved for each 10 epochs. Adam optimizer was used for applying gradient operations to trainable parameters with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. No pre-trained network was used while training Modified Monodepth models.

All our teacher and student models were trained for 50 epochs separately. Each of them was trained from scratch without using any pre-trained network. Our teacher networks take 9 hours to train for 50 epochs. Each of our student networks take 50 hours to train for 50 epochs.

In training of all our models, learning rate was 10^{-4} at the beginning. Similar to training of [10] and many other monocular depth estimation studies, learning rate was kept same for 30 epochs and then for each 10 epoch it was halved. Batch size was 4 during training. Adam optimizer was also used in training of teacher and student networks with same hyper-parameters before. Input images provided for training had size 256x512 pixels that means each training image from KITTI dataset was resized to this dimension and then provided to the network.

4.4 Performance Evaluation

In this section, performance of EfficientNet encoder and ASPP module will be given and be compared with Monodepth [10]'s results. Test results of our monocular depth estimation teacher and student networks will be provided and the effect of iterative teacher-student learning method will be explained. Experimental results will be compared with previous works on monocular depth estimation problem and discussed.

4.4.1 Evaluation of EfficientNet & ASPP

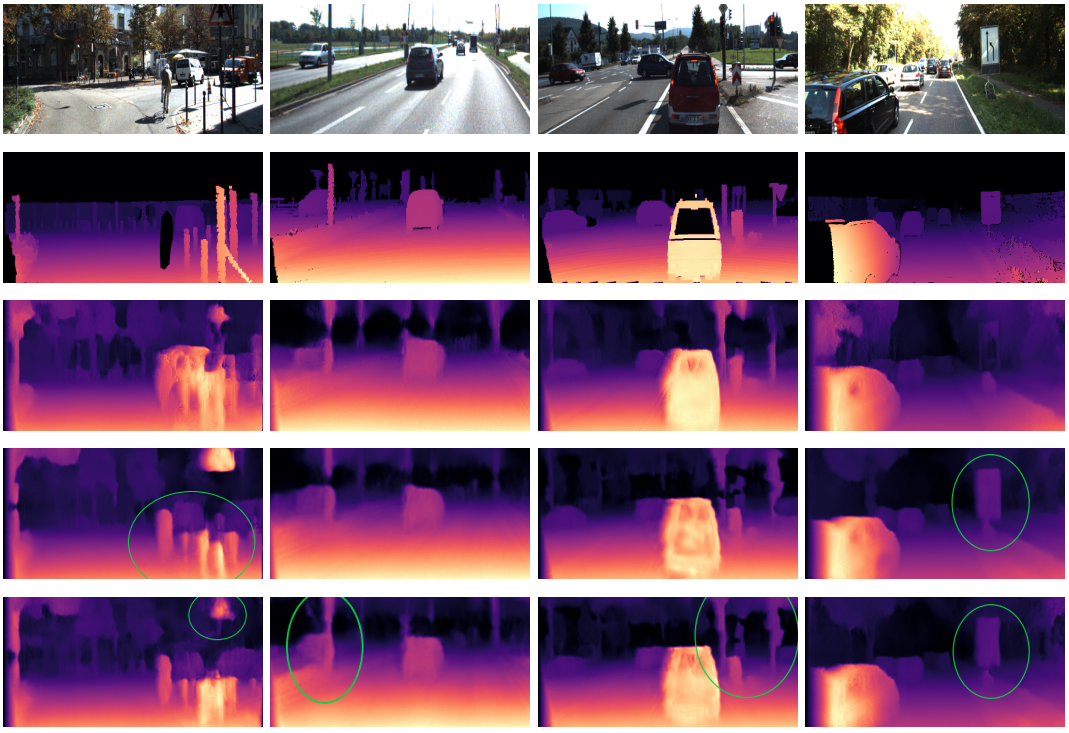
As explained before our Modified Monodepth model was trained for 50 epochs by using stereo images in an unsupervised manner same with the original Monodepth model [10]. After training is completed, we tested our network on KITTI 2015 stereo dataset [24] which has 200 highly detailed ground truth depth maps. Comparison of test results on this highly detailed dataset instead of sparse ground truth depth maps would give more information about the efficiencies of EfficientNet and ASPP. Performance results of our models are provided in Table 4.5 along with the original Monodepth model.

Table 4.5: Evaluation Results of EfficientNet Encoder and ASPP Module.

		Error (Lower is better)				Accuracy (Higher is better)		
Model Name	Training Dataset	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Monodepth [10] with VGG	KITTI	0.124	1.388	6.125	0.217	0.841	0.936	0.975
Modified Monodepth Eff.	KITTI	0.109	1.355	5.453	0.187	0.885	0.956	0.982
Modified Monodepth Eff.&ASPP	KITTI	0.102	1.095	5.353	0.180	0.888	0.956	0.982

According to results given in Table 4.5 we can see that EfficientNet replaced with VGG provides a significant error drop in absolute relative difference and RMS error. Another large error drop is accomplished in the square relative error when the ASPP module used in decoder part.

Hyper-parameters used in training of our Modified Monodepth models are in fact optimized for original Monodepth with VGG encoder. Godard *et al.* [10] have optimized their network and training hyper-parameters in order to obtain best results for their Monodepth model. We evaluated our Modified Monodepth models after training them using same hyper-parameters. Therefore, we can say that hyper-parameters that we used in training of our modified Monodepth models might not be optimized for a network with EfficientNet encoder and ASPP. Hyper-parameters can be optimized further using grid search and a better performance can be achieved by these networks.



(a) (b) (c) (d)

Figure 4.1: Qualitative comparison of predicted depth maps. Images are taken from KITTI 2015 stereo dataset [24]. From top to bottom, sample image, ground truth depth map, prediction of 'Original Monodepth', 'Modified Monodepth Eff.' and 'Modified Monodepth Eff.&ASPP'.

Figure 4.1 presents the comparison of predicted depth maps of Original Monodepth and our modified Monodepth models. Improvements made in depth predictions are marked with green circles. As seen on the predicted depth maps of Image (a), model with Efficientnet shows better performance on realizing distance between poles. On the other hand, ASPP recognizes triangular shapes more successfully. In some cases, even if features for thin objects are extracted by encoders, ordinary upsampling operations and simple decoders might not be enough to process these features and detect these thin objects. Especially in Images (b) and (c), effectiveness of ASPP in realizing thin objects is clear.

Success of EfficientNet in extracting features can also be observed in Image (d) of Figure 4.1. While, the traffic sign in Image (d) is not recognized by Original Mon-

odepth it is detected when the encoder is replaced with EfficientNet. After observing these evaluation results, we have decided to use both EfficientNet and ASPP in our proposed model architecture.

4.4.2 Evaluation of EfficientNet Model Sizes

In order to investigate how changing encoder size alters the evaluation results of a neural network in monocular depth estimation problem, we used the compound scaling ability of EfficientNet. We designed three teacher networks using B5, B6 and B7 model sizes of EfficientNet and trained them on our labeled dataset for 50 epochs. At this stage we did not inject any noise to the networks during training. Evaluation results of these three networks are provided in Table 4.6.

Table 4.6: Evaluation results of different EfficientNet model sizes.

		Error (Lower is better)				Accuracy (Higher is better)		
Model Name	Encoder Type	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Teacher-B5	EfficientNet-B5	0.126	0.823	4.843	0.188	0.841	0.955	0.985
Teacher-B6	EfficientNet-B6	0.115	0.718	4.522	0.173	0.864	0.962	0.988
Teacher-B7	EfficientNet-B7	0.110	0.671	4.349	0.165	0.876	0.965	0.990

EfficienNet model B7 is the largest network that is proposed by *et. al* [17]. Evaluation results in Table 4.6 clearly shows that, using this largest model provides a significant performance increase in the network. This would have been expected, since a larger network can adapt itself to a dataset with a higher accuracy unless it overfits to the dataset. As seen from the results, performance continuously increases from EfficientNet B5 to B7. We can see that the accuracy and the performance is not saturated yet. If there was a larger EfficientNet model and it was used in our network it could have provided better results. However, for the time being EfficientNet model B7 is the largest network that is proposed by *et. al* [17]. Therefore, considering our computational resource limitations and the high performance of B7, we have decided to use model B7 in our teacher and student networks.

4.4.3 Evaluation of Iterative Teacher-Student Learning

Performance of our proposed architecture and the effect of iterative teacher-student learning can be best compared with other monocular depth estimation networks that are trained by using the same dataset. We present here the performance of previous works that are trained and tested on KITTI dataset in 3 classes according to their training method. In fact, unsupervised methods cannot be exactly compared to semi-supervised and supervised methods since they output a relative depth map for the objects in the scene. In order to get exact metric depth map from a model, metric depth information or camera intrinsic parameters should be provided to the model. Yet, we provide performance results of unsupervised methods here to show some state-of-the-art results in literature of monocular depth estimation.

All models provided in Tables 4.7, 4.8 and 4.9 are trained and tested on Eigen splits of KITTI dataset. In Table 4.7, evaluation results of previous unsupervised methods are provided. In Table 4.8, evaluation results of previous supervised methods are provided along with our teacher networks that are trained in a supervised fashion. In Table 4.9, evaluation results of previous semi-supervised methods are provided along with our student networks that are trained in a semi-supervised fashion.

Table 4.7: Evaluation results of Unsupervised trained networks. Best results are in **bold** and second best results are underlined (Stereo: Stereo images are used in training. Video: Consecutive frames of a video are used in training.)

			Error (Lower is better)				Accuracy (Higher is better)		
Year	Model Name	Train Data Type	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
2017	Godard *et al.* [10]	Stereo	0.148	1.344	5.927	0.247	0.803	0.922	0.964
2018	Poggi *et al.* [11]	Stereo	0.153	1.363	6.030	0.252	0.789	0.918	0.963
2019	Casser *et al.* [59]	Video	0.109	0.825	4.750	0.187	0.874	0.958	0.983
2019	Li *et al.* [60]	Video	0.150	1.127	5.564	0.229	0.823	0.936	0.974
2019	Pilzer *et al.* [19](Student)	Stereo	0.142	1.231	5.785	0.239	0.795	0.924	0.968
2019	Pilzer *et al.* [19](Teacher)	Stereo	0.098	0.831	4.656	0.202	0.882	0.948	0.973
2019	Godard *et al.* [13]	Video + Stereo	**0.080**	**0.466**	**3.681**	**0.127**	**0.926**	**0.985**	**0.995**
2020	Wang *et al.* [61]	Video + Stereo	0.101	0.725	4.360	0.179	0.898	0.965	0.983
2020	Patil *et al.* [62]	Video	0.111	0.821	4.650	0.187	0.883	0.961	0.982

Unsupervised learning methods when trained by both stereo images and video sequences show promising results. The main problem with video sequences is the estimation of change in relative position of the camera between consequtive frames. However, this problem mostly be solved by the auxiliary information obtained from stereo images. By merging the depth information from both input data, more reliable depth maps and higher performances are can be obtained as seen on Table 4.7 from results of [13] and [61].

Table 4.8: Evaluation results of Supervised trained networks. Best results are in **bold** and second best results are underlined. (GT: Ground truth depth maps are used in training. SMN: Stereo matching network is used in training for auxiliary depth information.)

			Error (Lower is better)				Accuracy (Higher is better)		
Year	Model Name	Train Data Type	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
2014	Eigen *et al.* [6]	GT	0.190	1.515	7.156	0.270	0.692	0.899	0.967
2018	Gan *et al.* [34]	GT + SMN	**0.098**	0.666	**3.933**	0.173	**0.890**	0.964	0.985
2019	Rosa *et al.* [33]	GT	0.123	**0.641**	4.524	0.199	0.881	**0.966**	0.986
2020	Fang *et al.* [63]	GT	0.105	-	4.486	0.183	0.873	0.960	0.984
	Ours (Teacher)	GT(1/6 of KITTI)	0.110	0.671	4.349	**0.165**	0.876	0.965	**0.990**

As mentioned before our proposed model for teacher network was trained by 1/6 of Eigen train split of KITTI dataset in a supervised manner. Although, less training data was used, the evaluation results of our teacher model are still comparable to the results of state-of-the-art methods as seen from Table 4.8.

Loss function used in training plays an important role for the performance of network. Sometimes even used loss functions can be inferred from the evaluation results. For example, when L1 loss function provided in equation 3.1 is used as training loss function, model tries to minimize relative distance between predicted and expected depth values. This results in a low valued absolute relative error in evaluation of the model. However, when squared relative error and RMS error of the model is compared with similar models, their values might still be relatively high. By using this knowledge, we can compare our models given in Table 4.9 with the model proposed by Yang *et al.* [65]. Loss function used in [65] includes L1 loss and according to their

Table 4.9: Evaluation results of Semi-Supervised trained networks. Best results are in **bold** and second best results are underlined. (GT: Ground truth depth maps are used in training. SGM: Semi-Global Matching used in training for obtaining depth map from stereo images.)

			Error (Lower is better)				Accuracy (Higher is better)		
Year	Model Name	Train Data Type	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
2017	Kuznietsov *et al.* [14]	Stereo + GT	0.113	0.741	4.621	0.189	0.862	0.960	0.986
2018	Guo *et al.* [64]	Stereo + GT	0.105	0.717	4.422	0.183	0.874	0.959	0.983
2018	Yang *et al.* [65]	Stereo + GT	0.097	0.734	4.424	0.187	0.888	0.958	0.980
2019	Tosi *et al.* [66]	Stereo + SGM	0.111	0.867	4.714	0.199	0.864	0.954	0.979
2019	Amiri *et al.* [67]	Stereo + GT	**0.096**	0.552	3.995	0.152	**0.892**	0.972	0.992
2019	Cho *et al.* [16]	Stereo + GT	0.099	0.748	4.599	0.183	0.880	0.959	0.983
	Ours (Student-1)	Unlabeled + GT	0.102	0.551	3.979	0.151	0.889	0.975	**0.993**
	Ours (Student-2)	Unlabeled + GT	0.100	**0.525**	**3.930**	**0.149**	**0.892**	**0.976**	**0.993**

evaluation results, their absolute relative error is lower than our model. However, when compared with other evaluation metrics, both of our student models and even our teacher model given in Table 4.8 shows a better performance due to our scale-invariant loss function presented in equation 3.1.

As seen on Table 4.9, our Student-1 network achieves a state-of-the-art performance when trained with labeled and pseudo labeled datasets. One of the most important advantage we had here was that our teacher model was also trained well compared to other works presented in Table 4.8. Of course the effects of EfficientNet encoder and ASPP module used in our network were undeniable. With a poorly trained model, predicted depth maps of the unlabeled dataset would be poor and expected performance increase in the student model would not be occurred. That's why we have evaluated it's performance first with state-of-the-art performances of previous works. After obtained satisfactory results we continued predicting depth maps of unlabeled dataset and training student.

Table 4.10: Evaluation results of Teacher and Student networks

		Error (Lower is better)				Accuracy (Higher is better)		
Model Name	Train Data Type	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Teacher	GT(1/6 of KITTI)	0.110	0.671	4.349	0.165	0.876	0.965	0.990
Student-1	Unlabeled + GT	0.102	0.551	3.979	0.151	0.889	0.975	0.993
Student-2	Unlabeled + GT	0.100	0.525	3.930	0.149	0.892	0.976	0.993

After training and evaluating our Student-2 network we have observed the performance increase compared to Student-1 as seen on Table 4.10. Increase in performance results was relatively small compared to the difference between teacher and Student-1 network. That would be expected, since teacher network was trained with a small labeled dataset only. On the other hand, even though it was small, the performance increase in Student-2 network was good enough to prove effectiveness of iterative teacher-student learning methodology.

Student-1 and Student-2 networks were trained with a same amount of training data and the depth maps were similar. There were improvements in the pseudo labeled dataset predicted by Student-1 when compared to the depth maps predicted by teacher network. Therefore, this improvement in the training data resulted in a performance increase in Student-2 networks evaluation results.

Both Student-1 and Student-2 networks were trained without pre-trained weights. One of the reasons behind this implementation was that when student model uses pre-trained weights from previous network, it might get stuck in a local optima. Another reason was possibility of overfitting of the model. To prevent these from happening, training the network from scratch sometimes becomes a better option and that was the motivation of us for training from scratch.

Aside from the performance increase in metric evaluation results, improvement from teacher to first and second student networks can be observed in predicted depth maps. We selected 9 different test images from Eigen test split. These 9 images are taken in unique environments. Since original GT images taken by Lidar are sparse, we have applied interpolation to understand them better while comparing with our models'

results. These 9 test images, their GT depth maps and predicted depth maps by our models are provided in Figures 4.2, 4.3 and 4.4.

After we made our models to predict depth maps of these 9 test images, we observed the improvement in recognizing people. As seen in Figure 4.2, while the teacher model has difficulties in detecting each individual person, student-1 and student-2 models detects these people more successfully. Moreover, vehicle shapes are also recognized by student networks more nicely.

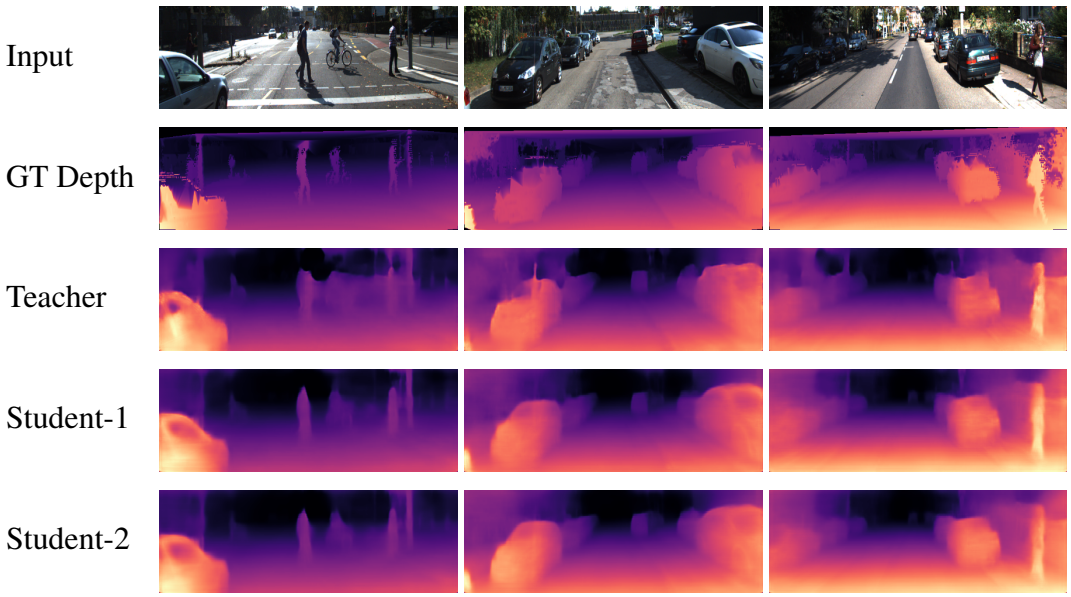


Figure 4.2: Qualitative comparison of predicted depth maps. Images are taken from Eigen Test Split of KITTI Dataset [22].

Test images that made teacher network to show worst performance are shown in Figure 4.3 and Figure 4.4. Due to lack of training data, teacher network cannot recognize the shapes of cars properly. Apart from this, most of the problematic areas have windows of cars. This is one of the biggest problems of predicting depth maps by deep learning models. Lidar depth maps are generally not continuous and reliable on glass surfaces which causes the deep learning models to get confused. That's why our teacher network cannot make proper prediction in these regions. At this point advantage of iterative teacher-student learning method steps in and the predictions made by student networks these regions become more appropriate. In predicted depth maps of student-2 network, car windows are almost continuous and shapes of cars are pre-

dicted correctly.

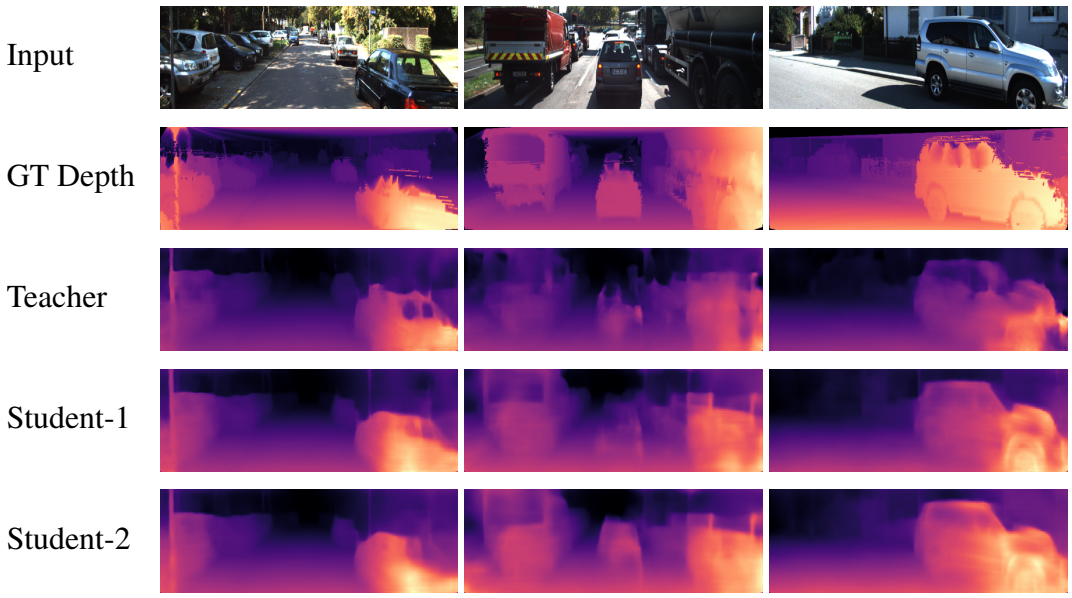


Figure 4.3: Qualitative comparison of predicted depth maps. Images are taken from Eigen Test Split of KITTI Dataset [22].

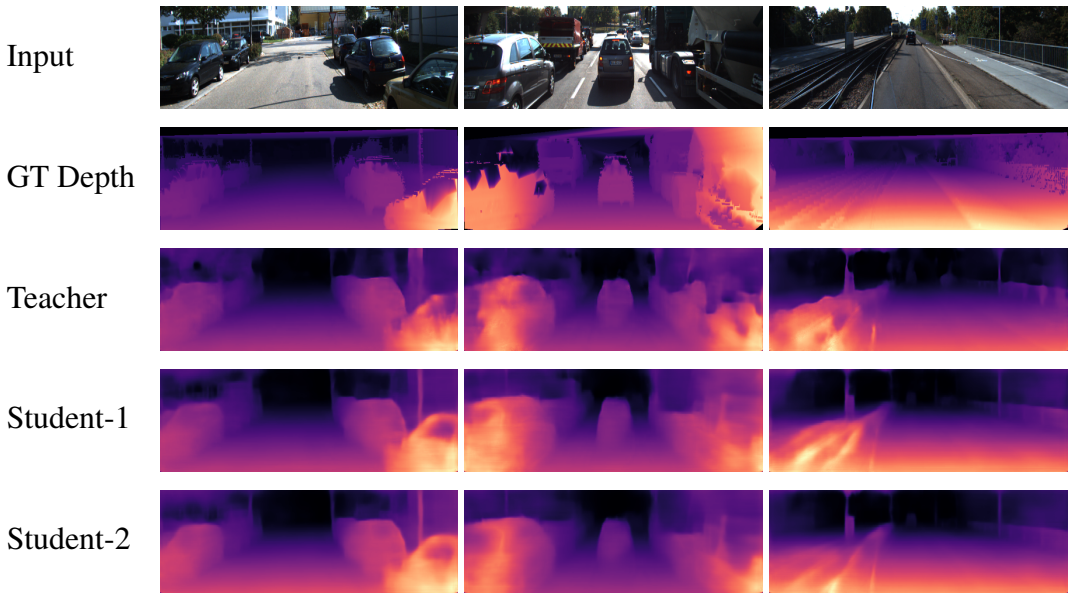


Figure 4.4: Qualitative comparison of predicted depth maps. Images are taken from Eigen Test Split of KITTI Dataset [22].

Another visible problem was the detection of railway tracks as seen on the rightmost image in Figure 4.4. Even though it was not supposed to be sensed as an object, rail-

way tracks was recognized by teacher network as an object and a false depth value was predicted. On the other hand, as a progress in student networks this wrongly recognized area becomes corrected and in student-2 network it almost disappears. These results also show that the noise injected during training of students to make them generalize better has helped our student networks. We can say that the generalization of student networks compared to the teacher network is achieved.

As observed in Figures 4.2, 4.3 and 4.4, provided training data plays an important role in model performance. Due to sparsity of GT depth maps and lack of training data, in some regions teacher network performs poorly as expected. When student-1 network is trained with labeled and pseudo labeled datasets, it outperforms the teacher network and shows an important improvement in predicted depth maps.

With the help of performance increase in student-2 network, pseudo labeled dataset becomes more accurate when predicted again by new teacher compared to the performance of old teacher network. In last stage, when student-2 network is trained with more accurate pseudo labeled dataset and labeled dataset, it outperforms both the teacher and the student-1 network. Effectiveness of iterative teacher-student learning method in monocular depth estimation problem is discovered with these results.

4.4.4 Evaluation of Noise Injection During Training

At this stage, we trained three teacher models by injecting noise during training. Data augmentation, Stochastic Depth and Dropout are applied to the networks. Similar to previous teacher training processes, all models are trained on our labeled dataset for 50 epochs and tested on Eigen test split. Evaluation results of these three networks are provided in Table 4.11 along with our standard teacher networks for comparison.

Table 4.11: Evaluation results of Noise Injection During Training.

		Error (Lower is better)				Accuracy (Higher is better)		
Model Name	Encoder Type	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Teacher-B5	EfficientNet-B5	0.126	0.823	4.843	0.188	0.841	0.955	0.985
Noisy-Teacher-B5	EfficientNet-B5	0.104	0.578	4.016	0.154	0.885	0.973	0.993
Teacher-B6	EfficientNet-B6	0.116	0.729	4.547	0.174	0.865	0.962	0.988
Noisy-Teacher-B6	EfficientNet-B6	0.102	0.566	3.945	0.151	0.889	0.974	0.993
Teacher-B7	EfficientNet-B7	0.110	0.671	4.349	0.165	0.876	0.965	0.990
Noisy-Teacher-B7	EfficientNet-B7	0.101	0.555	3.925	0.149	0.891	0.976	0.993

After evaluating noise injected three models we observed surprising results. Injecting noise aggressively to the teacher networks increases the performance results significantly. According to these results, we can claim that noise injection makes the monocular depth estimation networks generalize better.

In table 4.11, we can see that performance of small networks like B5 and B6 increases relatively more compared to a larger network B7. At this point, our decoder part might be limiting the performance increase of Teacher-B7 and Noisy-Teacher-B7 models. However, in Teacher-B5 and Teacher-B6 models, since the limit performance is not reached, injection of noise makes the networks generalize better and help them to increase their performances further.

Performance of these networks might also be limited by the number of train samples used during training. If a larger dataset is used in training of non-noisy teacher networks, their performance can increase further and reach to the performance of noisy-teacher networks.

After obtaining these results, we have decided to train student networks with two iterations by using the Noisy-Teacher-B7 as our teacher network. For training student networks, procedure of iterative teacher-student learning is followed. Noise is also injected to the student networks during training. Evaluation results of these noisy-teacher and student networks are provided in Table 4.12 along with our standard teacher and student networks for comparison. All models provided on Table 4.12 have same encoder model size, EfficientNet-B7.

Table 4.12: Evaluation results of Teacher and Student networks

		Error (Lower is better)				Accuracy (Higher is better)		
Model Name	Train Data Type	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Teacher	GT(1/6 of KITTI)	0.110	0.671	4.349	0.165	0.876	0.965	0.990
Student-1	Unlabeled + GT	0.102	0.551	3.979	0.151	0.889	0.975	0.993
Student-2	Unlabeled + GT	0.100	0.525	3.930	0.149	0.892	0.976	0.993
Noisy-Teacher	GT(1/6 of KITTI)	0.101	0.555	3.925	0.149	0.891	0.976	0.993
Student-1 with Noisy-Teacher	Unlabeled + GT	0.099	0.525	3.820	0.146	0.894	0.977	0.994
Student-2 with Noisy-Teacher	Unlabeled + GT	0.099	0.524	3.819	0.146	0.894	0.977	0.994

According to the results on Table 4.12, noisy-teacher already had better evaluation results compared to standard teacher network that is trained without noise. However, using iterative teacher-student learning can even increase the performance further. "Student-1 with Noisy-Teacher" network trained after the "Noisy-Teacher" network outperforms the previous evaluation results. However, a significant performance increase is not observed for "Student-2 with Noisy-Teacher" network. This shows us that the performance of network has saturated. At this point, in order to increase performance one solution might be increasing the number of unlabeled data samples. Another solution might be increasing the noise injected to the network during training. More aggressively injecting noise might help the Student-2 network to generalize much better.

4.4.5 Other Results

During training of our teacher and student networks we have saved training loss values to investigate learning capability of our teacher and student networks. Loss values are saved after each epoch. Epoch Number vs. Training Loss graphs of our three networks are given on Figure 4.5.

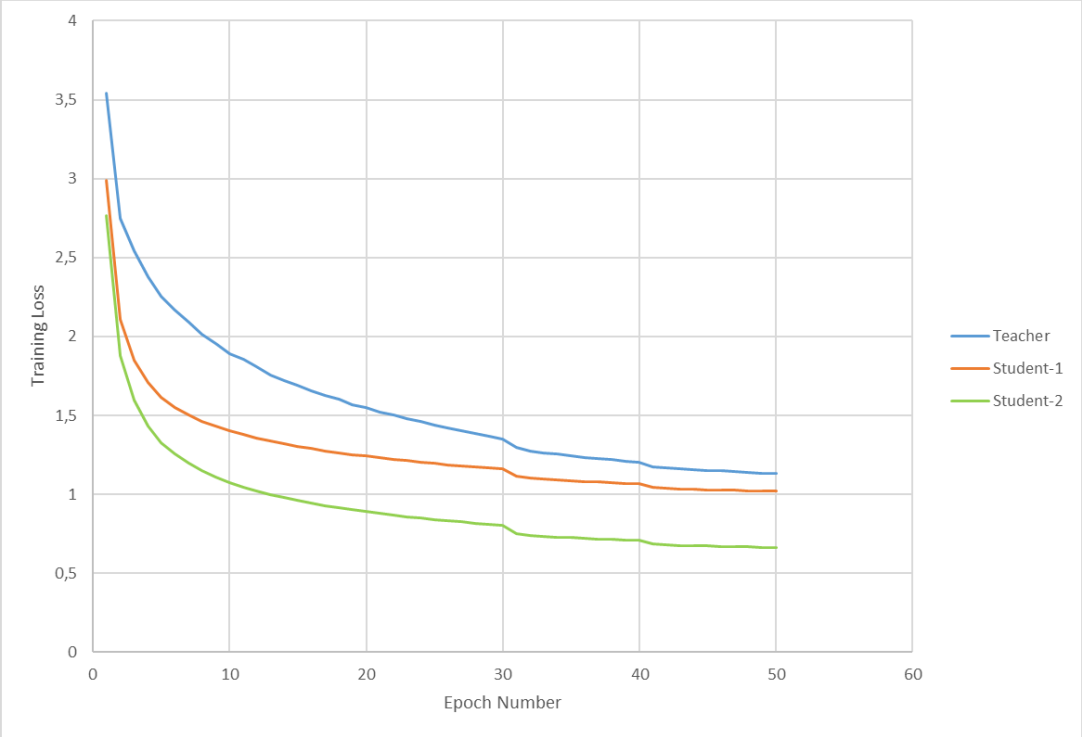
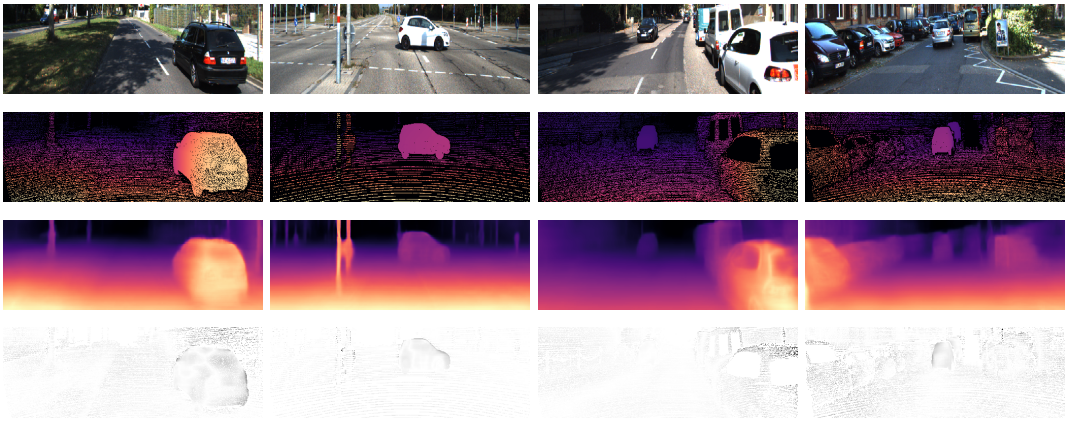


Figure 4.5: Epoch Number vs. Training Loss

As seen on Figure 4.5, our Student-1 network converged to a fewer loss value than the teacher. On the other hand, Student-2 network converged to a much fewer loss value than the Student-1 network. We think, increasing the dataset by adding pseudo labeled dataset in training of Student-1 network made it converge to a lesser loss value. However, in training of Student-2 network the number of data used was same with the Student-1 networks training, yet Student-2 network converged to a much lesser loss value. No pre-trained weights were used in training of these networks. Therefore, we believe the reason behind Student-2 network having a much less training loss compared to others is that pseudo labeled dataset used in training was predicted by Student-1 network. At that point Student-1 network already had a high performance and so the predicted depth maps by Student-1 network were more accurate than the Teacher networks predictions. Using a more accurate pseudo labeled dataset in training of Student-2 Network ensured it to converge to a lesser training loss value.

We have computed error maps for some predictions to examine whether our Student-2 network performs better on near objects or distant objects. Error map is calculated

through absolute pixel difference between ground truth and predicted depth maps. Predictions are made on KITTI 2015 stereo dataset [24] which has more highly detailed ground truth depth maps. Original images, GT images, predicted depth maps and error maps are provided on Figure 4.6.



(a) (b) (c) (d)

Figure 4.6: Qualitative comparison of predicted depth maps by Student-2 network with ground truth depth maps in terms of error. Images are taken from KITTI 2015 stereo dataset [24]. From top to bottom, sample image, ground truth depth map, prediction of Student-2 network, error map.

After calculating error maps, we observed that there is not a big difference between far and near estimations of depth. As seen on error map of image 'c', if object is very close to the camera some distortions might be encountered. However, in general faulty estimations occur on object boundaries as it is expected.

In order to observe effect of epoch number in training, we have evaluated performance of three models for our Student-1 network. Evaluations are made with Student-2 network models trained for 30, 40 and 50 epochs. Results are given on Table 4.13.

Table 4.13: Evaluation results of Student-1 network for different epoch numbers.

		Error (Lower is better)				Accuracy (Higher is better)		
Model Name	Epoch Number	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Student-1	30	0.100	0.527	3.861	0.147	0.890	0.976	0.994
Student-1	40	0.099	0.526	3.854	0.146	0.892	0.977	0.994
Student-1	50	0.099	0.525	3.820	0.146	0.894	0.977	0.994

As observed on Table 4.13, Student-1 network reaches to a high accuracy even before 30 epochs. Small improvements occur on RMS error but mostly other error values and accuracy values does not change much. Although performance does not change much, the network does not overfit to the dataset and its accuracy does not decrease as most of the deep learning networks. This is achieved as noise is aggressively injected during training.

CHAPTER 5

CONCLUSION

A deep learning model can learn to estimate depth from monocular images with the help of representation learning. This could be done with a good network structure and an appropriate dataset suitable for the method used in learning. The aim of this thesis is to design a successful auto-encoder structure and train it with teacher-student learning method to achieve state-of-the-art accuracy. For that purpose, we start with analyzing the networks structures and learning methods used in previous works. Since monocular depth estimation problem is solved by representation learning, the feature extraction by encoder structure plays an important role in networks performance. EfficientNet encoder [17] designed using neural architecture search, has shown an outstanding performance in previous works in deep learning area. Therefore, we have decided using EfficientNet encoder structure in our network architecture. We first evaluated its performance by using it in a well-known model Monodepth [10] since it was not used in a monocular depth estimation network before. We showed with our experiments that using EfficientNet in a depth estimation network increases performance. We have also evaluated the performance of ASPP (Atrous Spatial Pymarid Pooling) [28] and showed that using it in decoder structure increases performance as well. After these evaluations we have created our own network structure by using EfficientNet and ASPP.

In second stage of our work, we focused on the learning method used in our experiments. We first built 3 different size teacher networks by using EfficientNet-B5, B6 and B7 encoder models and trained them. We built these different size networks with the help of EfficientNet compound scaling strategy. We evaluated their performances in order to decide which model size we should use as our original teacher

network. We observed that EfficientNet-B7 model performed much better compared to B5 and B6 models as it was expected in the first place. Since EfficientNet-B7 network has a larger network size it can learn and generalize better. After we have evaluated B5 and B6 model sizes we observed that they cannot reach to the performance of B7. However, when considered in terms of speed, EfficientNet-B7 trains slower compared to others. After deciding to use EfficientNet-B7 model, we started training with iterative teacher-student learning method. Our first teacher network was trained in a supervised fashion with a small labeled dataset taken from KITTI dataset [22]. Even though it was trained with a small dataset, our first evaluation results showed that compared to the state-of-the-art supervised learning networks our proposed network structure performed close and even better in some cases. Then we used our teacher network to predict depth for an unlabeled large dataset and called this dataset as pseudo labeled dataset. Our first student network was trained with labeled and pseudo labeled datasets. In an iterative way, we used our first student network to predict depth for pseudo labeled dataset and trained our second student network with labeled and pseudo labeled datasets again. During training process of student networks noise was aggressively injected to the models to make them generalize and perform better than teacher network as proposed by Xie *et. al* [1]. Through exhaustive experiments we proved that a depth estimation model trained with iterative teacher-student learning method can increase its performance incredibly. Moreover, our tests conducted on sample images showed that in confusing areas of the scene, student networks learn to recognize objects better than teacher network even though they are trained with same labeled dataset.

In third stage of our work, we trained three different size teacher networks again by injecting noise this time. Then we compared our noise injected teacher network in order to investigate effect of noise injection during training. We have observed a good performance increase thanks to noise injection. We took Noisy-Teacher-B7 network to become our teacher again for the iterative teacher-student learning. Following the same procedure applied before, we trained two student networks. But this time, our teacher network was noisy as well. Then, we evaluated the performances of student networks compared them to the previous student networks. There was a performance increase compared to teacher network and previous student networks but it was small.

Apparently, injecting noise to the teacher made the student networks saturate eventually and they have reached to the top performance.

After evaluating all our models, our experiments showed that by using a small ground truth labeled dataset, state-of-the-art performance results can be achieved with a monocular depth estimation network if iterative teacher-student learning method is used in training. The key practical benefit of proposed method is that it could enable the decrease in need for massive labeled dataset in the monocular depth estimation problem.

There are several studies left for future work in this thesis. First of all, transfer learning may be applied to our networks in order to evaluate them on different datasets. While KITTI dataset includes outdoor street scenes mostly, an indoor or a different dataset can also be used to train our networks from scratch. Moreover, our study reveals that using noisy teacher-student learning, a solid student networks can be trained even with a small set of ground truth labeled dataset. With the inspiration of the results we share in this thesis, a network for a different representation problem can also be trained. In the future, in order to increase accuracy and reliability of our network and decrease its inference time, network architecture may be improved. Further, monocular depth estimation networks can even be used in autonomous vehicles if high reliability is achieved. But they should also have the ability to output real-time depth maps.

REFERENCES

[1] Q. Xie, E. Hovy, M.-T. Luong, and Q. V. Le, "Self-training with noisy student improves imagenet classification," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695, 2020.

[2] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, pp. 211–252, 2015.

[3] Y. Yekutieli, R. Mitelman, B. Hochner, and T. Flash, "Analyzing octopus movements using three-dimensional reconstruction," *Journal of neurophysiology*, vol. 98, pp. 1775–90, 10 2007.

[4] C. Loop and Zhengyou Zhang, "Computing rectifying homographies for stereo vision," in *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, vol. 1, pp. 125–131, 1999.

[5] G. d. S. Vieira, F. A. A. M. N. Soares, G. T. Laureano, R. T. Parreira, J. C. Ferreira, and R. Salvini, "Disparity map adjustment: a post-processing technique," in *2018 IEEE Symposium on Computers and Communications (ISCC)*, pp. 580–585, 2018.

[6] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *NIPS*, p. 2366–2374, 2014.

[7] X. Qi, R. Liao, Z. Liu, R. Urtasun, and J. Jia, "Geonet: Geometric neural network for joint depth and surface normal estimation," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 283–291, 2018.

[8] B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy, and T. Brox, "Demon: Depth and motion network for learning monocular stereo,"

2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5622–5631, 2017.

[9] R. Garg, B. V. Kumar, G. Carneiro, and I. Reid, "Unsupervised cnn for single view depth estimation: Geometry to the rescue," in *ECCV*, pp. 740–756, 2016.

[10] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *CVPR*, pp. 6602–6611, 2017.

[11] M. Poggi, F. Aleotti, F. Tosi, and S. Mattoccia, "Towards real-time unsupervised monocular depth estimation on cpu," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5848–5854, 2018.

[12] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *CVPR*, pp. 6612–6619, 2017.

[13] C. Godard, O. M. Aodha, and G. Brostow, "Digging into self-supervised monocular depth estimation," *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3827–3837, 2019.

[14] Y. Kuznietsov, J. Stückler, and B. Leibe, "Semi-supervised deep learning for monocular depth map prediction," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2215–2223, 2017.

[15] Y. Luo, J. Ren, M. Lin, J. Pang, W. Sun, H. Li, and L. Lin, "Single view stereo matching," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 155–163, 2018.

[16] J. Cho, D. Min, Y. Kim, and K. Sohn, "A large rgb-d dataset for semi-supervised monocular depth estimation," *ArXiv*, vol. abs/1904.10230, 2019.

[17] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *ArXiv*, vol. abs/1905.11946, 2019.

[18] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, 2018.

[19] A. Pilzer, S. Lathuilière, N. Sebe, and E. Ricci, "Refine and distill: Exploiting cycle-inconsistency and knowledge distillation for unsupervised monocular

depth estimation," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9760–9769, 2019.

[20] J. Ye, Y. Ji, X. Wang, K. Ou, D. Tao, and M. Song, "Student becoming the master: Knowledge amalgamation for joint scene parsing, depth estimation, and more," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2824–2833, 2019.

[21] Z. Chen, R. Zhang, G. Zhang, Z. Ma, and T. Lei, "Digging into pseudo label: A low-budget approach for semi-supervised semantic segmentation," *IEEE Access*, vol. 8, pp. 41830–41837, 2020.

[22] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3354–3361, 2012.

[23] B. Artacho and A. Savakis, "Waterfall atrous spatial pooling architecture for efficient semantic segmentation," *Sensors*, vol. 19, p. 5361, Dec 2019.

[24] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3061–3070, 2015.

[25] R. S. Allison, B. J. Gillam, and E. Vecellio, "Binocular depth discrimination and estimation beyond interaction space," *Journal of Vision*, vol. 9, no. 1, p. 10–10, 2009.

[26] S. Palmisano, B. Gillam, D. G. Govan, R. S. Allison, and J. M. Harris, "Stereoscopic perception of real depths at large distances," *Journal of Vision*, vol. 10, pp. 19–19, 06 2010.

[27] A. Glennerster, B. Rogers, and M. Bradshaw, "Stereoscopic depth constancy depends on the subject's task," *Vision Research*, vol. 36, no. 21, pp. 3441 – 3456, 1996.

[28] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, pp. 834–848, 2018.

[29] M. Okutomi and T. Kanade, "A multiple-baseline stereo," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 4, pp. 353–363, 1993.

[30] Y. Boykov, O. Veksler, and R. Zabih, "A variable window approach to early vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 12, pp. 1283–1294, 1998.

[31] S. Birchfield and C. Tomasi, "Depth discontinuities by pixel-to-pixel stereo," in *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, pp. 1073–1080, 1998.

[32] Bo Li, Chunhua Shen, Yuchao Dai, A. van den Hengel, and Mingyi He, "Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1119–1127, 2015.

[33] N. Rosa, V. Guizilini, and V. Grassi, "Sparse-to-continuous: Enhancing monocular depth estimation using occupancy maps," *2019 19th International Conference on Advanced Robotics (ICAR)*, pp. 793–800, Dec 2019.

[34] Y. Gan, X. Xu, W. Sun, and L. Lin, "Monocular depth estimation with affinity, vertical pooling, and label enhancement," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[35] R. Ranftl, K. Lasinger, D. Hafner, and V. Koltun, "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PP, pp. 1–1, 08 2020.

[36] J. Xie, R. B. Girshick, and A. Farhadi, "Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks," in *ECCV*, pp. 842–857, 2016.

[37] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[38] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1800–1807, 2017.

[39] S. Zagoruyko and N. Komodakis, "Wide residual networks," *ArXiv*, vol. abs/1605.07146, 2016.

[40] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-excitation networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, pp. 2011–2023, 2020.

[41] B. Liebl and M. Burghardt, "An evaluation of dnn architectures for page segmentation of historical newspapers," *ArXiv*, vol. abs/2004.07317, 2020.

[42] B. Zoph, G. Ghiasi, T.-Y. Lin, Y. Cui, H. Liu, E. D. Cubuk, and Q. V. Le, "Rethinking pre-training and self-training," *ArXiv*, vol. abs/2006.06882, 2020.

[43] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," *ArXiv*, vol. abs/1802.02611, 2018.

[44] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, pp. 303–338, June 2010.

[45] B. Cheng, B. Xiao, J. Wang, H. Shi, T. Huang, and L. Zhang, "Higherhrnet: Scale-aware representation learning for bottom-up human pose estimation," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5385–5394, 2020.

[46] C. L. Neff, A. Sheth, S. Furgurson, and H. Tabkhi, "Efficienthrnet: Efficient scaling for lightweight high-resolution multi-person pose estimation," *ArXiv*, vol. abs/2007.08090, 2020.

[47] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10778–10787, 2020.

[48] T. Anwar and S. Zakir, "Deep learning based diagnosis of covid-19 using chest ct-scan images," in *2020 IEEE 23rd International Multitopic Conference (INMIC)*, pp. 1–5, 2020.

[49] C. Bucila, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *KDD '06*, vol. 2006, pp. 535–541, 08 2006.

[50] G. E. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *ArXiv*, vol. abs/1503.02531, 2015.

[51] J. Xie, B. Shuai, J. Hu, J. Lin, and W. Zheng, "Improving fast segmentation with teacher-student learning," in *BMVC*, 2018.

[52] M. Bellver, A. Salvador, J. Torres, and X. G. i Nieto, "Budget-aware semi-supervised semantic and instance segmentation," in *CVPR Workshops*, 2019.

[53] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014.

[54] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, "Deep networks with stochastic depth," in *ECCV*, pp. 646–661, 2016.

[55] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2015.

[56] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, vol. 1, 10 2016.

[57] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2015.

[58] J. H. Lee, M. Han, D. W. Ko, and I. H. Suh, "From big to small: Multi-scale local planar guidance for monocular depth estimation," *ArXiv*, vol. abs/1907.10326, 2019.

[59] V. Casser, S. Pirk, R. Mahjourian, and A. Angelova, "Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 8001–8008, 07 2019.

[60] S. Li, F. Xue, X. Wang, Z. Yan, and H. Zha, "Sequential adversarial learning for self-supervised deep visual odometry," *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2851–2860, 2019.

[61] J. Wang, G. Zhang, Z. Wu, X. Li, and L. Liu, "Self-supervised joint learning framework of depth estimation via implicit cues," *ArXiv*, vol. abs/2006.09876, 2020.

[62] V. Patil, W. V. Gansbeke, D. Dai, and L. V. Gool, "Don't forget the past: Recurrent depth estimation from monocular video," *IEEE Robotics and Automation Letters*, pp. 6813–6820, 2020.

[63] Z. Fang, X. Chen, Y. Chen, and L. Van Gool, "Towards good practice for cnn-based monocular depth estimation," in *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1080–1089, 2020.

[64] X. Guo, H. Li, S. Yi, J. Ren, and X. Wang, "Learning monocular depth by distilling cross-domain stereo networks," *ArXiv*, vol. abs/1808.06586, 2018.

[65] N. Yang, R. Wang, J. Stückler, and D. Cremers, "Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry," in *ECCV*, p. 835–852, 2018.

[66] F. Tosi, F. Aleotti, M. Poggi, and S. Mattoccia, "Learning monocular depth estimation infusing traditional stereo knowledge," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9791–9801, 2019.

[67] A. J. Amiri, S. Yan Loo, and H. Zhang, "Semi-supervised monocular depth estimation with left-right consistency using deep neural network," in *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, p. 602–607, IEEE Press, 2019.