ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

APPLICATION AND ANALYSIS OF DEEP LEARNING TECHNIQUES ON THE PROBLEM OF DEPTH ESTIMATION FROM A SINGLE IMAGE

M.Sc. THESIS

Alican MERTAN

Department of Computer Engineering

Computer Engineering Programme

JULY 2020

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

APPLICATION AND ANALYSIS OF DEEP LEARNING TECHNIQUES ON THE PROBLEM OF DEPTH ESTIMATION FROM A SINGLE IMAGE

M.Sc. THESIS

Alican MERTAN
(504171543)

Department of Computer Engineering

Computer Engineering Programme

Thesis Advisor: Prof. Dr. Gözde ÜNAL

JULY 2020

ISTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

DERİN ÖĞRENME TEKNİKLERİNİN TEKİL GÖRÜNTÜDEN DERİNLİK TAHMİNİ PROBLEMİ ÜZERİNDE UYGULANMASI VE İNCELENMESİ

YÜKSEK LİSANS TEZİ

Alican MERTAN
(504171543)

Bilgisayar Mühendisliği Anabilim Dalı

Bilgisayar Mühendisliği Programı

Tez Danışmanı: Prof. Dr. Gözde ÜNAL

TEMMUZ 2020

Alican MERTAN, a M.Sc. student of İTU Graduate School of Science Engineering and Technology student ID 504171543, successfully defended the thesis entitled "APPLICATION AND ANALYSIS OF DEEP LEARNING TECHNIQUES ON THE PROBLEM OF DEPTH ESTIMATION FROM A SINGLE IMAGE", which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Prof. Dr. Gözde ÜNAL** İstanbul Technical University	....
Jury Members :	**Prof. Dr. Uluğ BAYAZIT** Istanbul Technical University	....
	Doç. Dr. Mehmet Erkut ERDEM Hacettepe University	....

Date of Submission : 15 June 2020
Date of Defense : 16 July 2020

FOREWORD

This thesis was prepared at the department of Computer Engineering in Istanbul Technical University. The work is based on a project supported by the Scientific and Technological Research Council of Turkey (TÜBİTAK), project no 116E167. I would like to thank to my advisor Prof. Dr. Gozde Unal for her support when it's most needed.

Finally, I wish to thank Dr. Damien Jade Duff for his guidance throughout my whole journey. He has been a true mentor and an idol for me.

July 2020 Alican MERTAN

TABLE OF CONTENTS

Page

ABBREVIATIONS

SIDE	: Single Image Depth Estimation
RGB	: Red Green Blue
2D	: Two Dimensional
3D	: Three Dimensional
MRF	: Markov Random Field
CRF	: Conditional Random Field
CNN	: Convolutional Neural Network
SSIM	: Structural Similarity
DIW	: Depth in the Wild
SIFT	: Scale-invariant Feature Transform
CAD	: Computer Aided Design
Grad-CAM	: Gradient-weighted Class Activation Mapping
ReLU	: Rectified Linear Unit
NDCG	: Normalized Discounted Cumulative Gain
MAP	: Mean Average Precision
AP	: Average Precision
WHDR	: Weighted Human Disagreement Rate
AI	: Artificial Intelligence

LIST OF TABLES

Page

LIST OF FIGURES

Page

APPLICATION AND ANALYSIS OF DEEP LEARNING TECHNIQUES ON THE PROBLEM OF DEPTH ESTIMATION FROM A SINGLE IMAGE

SUMMARY

Depth is a key factor for scene understanding. Depth information allows us to project a scene into 3D. Number of problems such as autonomous driving, object detection, semantic segmentation, virtual and augmented reality, grasping can benefit from depth knowledge of a scene.

In this work, we focus on the problem of depth estimation. Particularly, we are interested in the single image depth estimation (SIDE), where the aim is to estimate pixel-wise depth map, which is the distance of each pixel in meters, for a given RGB image. This is an inherently ambiguous problem because of the fact that same RGB image can be created by infinitely many 3D scenes that varies in scale.

Due to its properties, SIDE problem is tackled with machine learning methods, most successfully with deep learning. General approach is to train a neural network using a data set of RGB images and corresponding ground truth depth maps. We group the existing works under 3 main category based on their main consideration.

The works in the first category aim to solve the SIDE problem without using ground truth data. To achieve this, we see the use of synthetic data or epipolar geometry. The second category consists of works that tackle the SIDE problem in a supervised manner and aim to increase the metric performance. In these works, we see the applications of developments in deep learning techniques, loss functions tailored to the SIDE problem, and multi task learning. In the last category, we group the works that aim to solve the SIDE in the wild problem since the works in the previous categories use limited data sets which contain images coming from particular type of setting such as indoor or outdoor. To achieve this, the SIDE problem is being relaxed to estimation of ordinal relations of pixels, which we refer as relative-SIDE. This relaxation allows diverse data sets to be collected and paves the way for SIDE in the wild.

In this work, we present an analysis of a work, state of the art at the time of analysis, that belongs to the second category. We replicate the work on an indoor data set. To illuminate how the model estimates depth and its performance beyond the metrics, we qualitatively analyze model's response to images that we collected, apply adversarial attack and our variant on Grad-CAM. Our analysis indicates that the model is not able to learn cues that are used by human visual system, instead it exploits very simple patterns specific to the training data set, hence cannot generalize well outside of its data set.

Based on our analysis, we believe SIDE in the wild remains an important challenge. To undertake this challenge, we formulate the relative-SIDE problem as a ranking problem where each pixel is ranked based on the given RGB image. An encoder decoder network model is trained to estimate a score for each pixel to rank them. Contrary of the previous works that uses pairwise losses for training, we investigate

the use of a listwise loss ListMLE, borrowed from ranking literature, as listwise losses claimed to be better than pairwise losses for information retrieval tasks. We show that model trained with our proposed loss achieves comparable performance with state of the art.

Additionally, we propose a new metric that puts more emphasis on pixels that are ranked higher. From the application perspective, having a good estimate for further pixels are not as important as having a good estimate for closer pixels. We believe our proposed metric reflects this important aspect of the SIDE problem. On this metric, our proposed method performs marginally better compared to state of the art.

DERİN ÖĞRENME TEKNİKLERİNİN
TEKİL GÖRÜNTÜDEN DERİNLİK TAHMİNİ PROBLEMİ ÜZERİNDE
UYGULANMASI VE İNCELENMESİ

ÖZET

Derinlik, bir görüntünün çok önemli bilgilerini barındıran bir özelliğidir. Derinlik bilgisi, ait olduğu görüntüde nesne tanıma, anlamsal bölümleme ve görüntü anlamlandırma gibi işlerin doğruluğunu artırmaya yardımcı olabilir. Bunlara ek olarak, artırılmış gerçeklik ve sanal gerçeklik teknolojilerinde kullanılan derinlik farkındalıklı resim düzeltme, 3B modelleme, nesnelerden sakınma, kavrama ve genel olarak robotik işlerinde derinlik bilgisi doğrudan kullanılmaktadır. Son yıllarda büyük ilgi gören otonom araç teknolojileri de görüntünün derinlik bilgisinden faydalanmaktadır.

Çalışmamız, görüntüden derinlik tahmini sorunu üzerinedir. Bilhassa tekil görüntüden derinlik tahmini sorununu ele almaktayız. Tekil görüntüden derinlik tahmini, verilen bir tekil görüntü için yoğun derinlik haritası tahmin etme işidir. Daha da açıkca, verilen tekil görüntüdeki her bir imge noktası için metre cinsinden bir uzaklık bulma işidir.

Tekil görüntüden derinlik tahmini sorununu ilginç ve zorlu kılan şey, bu sorunun özünde bulunan belirsizliktir. Aralarında yalnızca ölçek farkı bulunan sonsuz sayıdaki 3B sahne, aynı 2B görüntüyü oluşturabilir. Bu durum, RGB görüntüleri ile derinlik haritaları arasında bire çok ilişkisinin varlığını işaret eder. Öyleyse bu durumda, yapay olarak geliştirilmiş görü sistemlerine kıyasla kalite ve genelleme kabiliyetleri bakımından büyük bir üstünlük sağlayan insan görü sistemi tekil görüntüden derinlik tahminini nasıl başarmaktadır? Bu sorunun cevabı, insan görü sisteminin kullandığı ipuçlarında yatmaktadır.

İnsanlar tekil görüntüden derinlik tahmini yapmak için, statik tekil ipuçlarından faydalanmaktadır. Bu ipuçları yedi tanedir ve ilki üst üste oturma ipucudur. Bir nesne, diğer bir nesneyi kısmen örttüğünde görülür ve kısmen örtülmüş nesnenin örten nesneye kıyasla daha uzak olduğu anlaşılır. İkinci ipucu perspektif ipucudur. Birbirine paralel çizgilerin uzaklık arttıkça yakınlaşmış gözükmeleri durumudur. Bu ipucu ile ilişkili olarak iki ayrı ipucu daha bulunmaktadır. Bunların ilki büyüklük ipucudur. Aynı nesne, uzaklığı ile ters orantılı olarak retinamızda farklı büyüklüklerde görüntü oluşturmaktadır. Gerçek büyüklüğü hakkında fikir sahibi olduğumuz nesneler için, retinamızda oluşturdukları görüntünün büyüklüğü ile kıyaslayarak derinlik tahmini yaparız. Perspektif ile ilişkili olan diğer ipucu ise doku eğimidir. Bu ipucunu, eğimli bir yüzeye baktığımızda yüzeyin dokusunun uzaklıkla birlikte daha yoğun gözükmesi olarak gözlemleyebiliriz. Bir diğer statik tekil ipucu atmosferik ipucudur. Atmosferde bulunan partiküllerin etkisi ile, nesneler bizden uzaklaştıkça daha bulanık ve mavimtırak görünürler. Bunların yanısıra, ışık ve gölge örüntüleri de tekil görüntüden derinlik tahmini yaparken kullandığımız ipuçları arasındadır. Birbiri üzerine gölge düşüren nesneler veya nesnenin yüzeyine ekli gölgeler derinlik algısı yaratır. Son olarak, ufuk çizgisine yakın nesneler daha uzak görünürler ve buna yükseklik ipucu adı verilir.

Bu ipuçları arasında en önemlisi büyüklük ipucudur. İnsanlar olarak bizlerin günlük hayatta sıkça gördüğümüz nesnelerin büyüklükleri hakkında gerçeğe yakın tahminleri vardır. Çevremizdeki dünyaya bakıp 2B görüntüler gözlemlediğimizde, görü sistemimiz önsel bilgilerimizi kullanarak bu 2B görüntüyü oluşturacak 3B sahneyi kolayca tahmin eder.

Bütün bu bilgiler ışığında, insanların öğrenilmiş önsel bilgiyi kullanarak istatiksel bir şekilde derinliği thamin ettiği sonucuna varırız. Bu sonuç aynı zamanda tekil görüntüden derinlik tahmini sorunu alanında yapılan araştırmalarında gidişatına yansımıştır. Bir çok çalışma, istatiksel yöntemler kullanarak tekil görüntüden derinlik tahmini sorununu ele almıştır.

Tekil görüntüden derinlik tahmini sorununu çözmek için kullanılan yöntemlerin başında makine öğrenmesi yöntemleri gelmektedir. Bunlardan en başarılısı ise, son yıllarda bilgisayarlı görü alanında birçok soruna başarı ile uygulanmış derin öğrenme yöntemleridir. Derin öğrenme yöntemlerinin genel yaklaşımı, RGB görüntüleri ve bunlara karşılık gelen gerçek referans değer derinlik haritalarından oluşan bir veri kümesi ile sinir ağları eğitmektir.

Literatürdeki çalışmaları, ana fikirlerini baz alarak kümeleyebiliriz. Alandaki her çalışma, tekil görüntüden derinlik tahmini sorununa farklı bir açıdan yaklaşmakta, varolan çalışmaların çeşitli eksiklerini kapatmaya çalışmaktadır. Bazı çalışmalar, kullanılan derin öğrenme yöntemlerinde gerçekleşen yeni gelişmeleri tekil görüntüden derinlik tahmini sorununa uygulayarak veya çoklu görevden faydalanarak metrik başarımı artırmaya çalışırken, diğer çalışmalar etiketlenmiş veriye olan ihtiyacı bitirme, daha iyi 3B yapılar elde etme, kaotik gerçek yaşam koşullarında çalışma gibi farklı sorunları aşmaya çalışmaktadır. Biz literatürdeki çalışmaları üç farklı kümeye ayırdık.

İlk küme, tekil görüntüden derinlik tahmini sorununu etiketlenmiş veriye ihtiyaç duymadan çözmeye çalışan çalışmalardan oluşmaktadır. Burada iki ana yaklaşım bulunmaktadır. Bunların ilki sentetik veri kullanarak zahmetli gerçek referans değer toplama işini otomatik olarak sentetik veri üretimi ile ekarte eden yaklaşımdır. Diğeri ise epipolar geometriden yararlanarak gözetimsiz öğrenme yöntemleri geliştiren yaklaşımdır.

İkinci küme, ana hedefi metrik başarımı artırmak olan gözetimli öğrenme çalışmalarından oluşmaktadır. Bu çalışmalar da iki ana başlık altında incelenebilir. İlki, sinir ağlarının mimarilerini, kullanılan yitim fonksiyonlarını v.b. geliştirerek başarımı artırmaya çalışan çalışmalardan oluşurken, ikincisi çeşitli farklı görevleri çoklu görev çatısı altısında birleştiren çalışmalardan oluşmaktadır.

Son küme, tekil görüntüden derinlik tahmin sorununu kaotik gerçek yaşam koşullarında çözmeyi hedefleyen çalışmalardan oluşmaktadır. Önceki kümelerdeki çalışmalar, yalnızca iç mekan ya da yalnızca dış mekan gibi kısıtlı çevrelerden toplanmış veri kümeleri üzerinde çalışmaktadır. Kaotik gerçek yaşam koşullarında çalışabilmek için, tekil görüntüden derinlik tahmini sorunu imge noktalarının sırasal ilişkilerinin tahmini sorununa indirgenerek rahatlatılmıştır ve rahatlatılmış sorun nispi derinlik tahmini olarak adlandırılmıştır. Bu rahatlatma, çeşitliliği yüksek veri kümelerinin toplanmasına olanak vererek kaotik gerçek yaşam koşullarında tekil görüntüden derinlik tahmini sorunun çözümünün yolunu açmıştır.

Bu çalışmada, araştırmanın yapıldığı tarihte güncel en iyi sonuçları alan ikinci kümeye ait bir çalışmanın analizini sunuyoruz. Bu çalışma bir iç mekan veri kümesinde tekrarlanmıştır. Modelin derinliği nasıl tahmin ettiği ve konvansiyonel metriklerin ötesinde nasıl performans gösterdiği araştırılmıştır. Bu hedefle, toplamış olduğumuz çeşitli görüntülere modelin verdiği cevaplar niteliksel olarak incelenmiş, çekişmeli saldırı ve Grad-CAM yönteminin bizim tarafımızdan değiştirilmiş bir varyantı uygulanmıştır. Yaptığımız analiz işaret etmektedir ki, model insan görü sistemi tarafından kullanılan ipuçlarını öğrenememiş, bunun yerine veri kümesinde bulunan basit örüntülerden faydalanmaya çalışmaktadır ve bu durum modelin genelleme kabiliyetini olumsuz etkilemektedir.

Yaptığımız analizden yola çıkarak, tekil görüntüden derinlik tahmini sorununda kaotik gerçek yaşam koşullarında çalışma sorununun halen aşılması gereken önemli bir zorluk olarak bizi beklediğine inanıyoruz. Bu sorunu ele almak için, nispi derinlik tahmini sorununu, her imge noktasının verilen görüntüye göre sıralandığı bir sıralama sorunu olarak formülize ettik. Bir gizyazar gizçözer sinir ağı modelini, imge noktalarını sıralamak için kullanılmak üzere bir puan tahmin etmesi için eğittik. Önceki ikili yitim fonksiyonu kullanarak gerçeklenmiş olan çalışmaların aksine, sıralama literatüründe ikili yitim fonksiyonlarına kıyasla daha iyi sonuçlar verdiği raporlanan listesel yitim fonksiyonu ListMLE fonksiyonunun kullanımını araştırdık. Önerdiğimiz yitim fonksiyonu ile eğitilen modelin güncel olan en iyi durum ile karşılaştırılabilir başarımlar aldığını gösterdik.

Ek olarak, daha yukarıda sıralanmış imge noktalarını vurgulayan bir metrik önerdik. Uygulama açısından düşünüldüğünde, yukarıda sıralanması gereken imge noktalarını doğru şekilde sıralamanın, aşağıda sıralanması gereken imge noktalarına kıyasla daha önemlidir. Önerdiğimiz metrik, tekil görüntüden derinlik tahmini sorunun bu yönünü yansıtmaktadır. Bu metrikde, önerdiğimiz yöntem güncel olan en iyi durumdan marjinal olarak daha iyi sonuçlar vermiştir.

Ek olarak, daha yukarıda sıralanmış imge noktalarını vurgulayan bir metrik önerdik. Uygulama açısından düşünüldüğünde, yukarıda sıralanması gereken imge noktalarının doğru şekilde sıralanması, aşağıda sıralanması gereken imge noktalarına kıyasla daha önemlidir. Önerdiğimiz metrik, tekil görüntüden derinlik tahmini sorunun bu yönünü yansıtmaktadır. Bu metrikde, önerdiğimiz yöntem güncel olan en iyi durumdan marjinal olarak daha iyi sonuçlar vermiştir.

1. INTRODUCTION

Depth estimation from a single image (SIDE, short for Single Image Depth Estimation) is the task of estimating a dense depth map for a given single RGB image. More specifically, for each pixel in the given RGB image, one needs to estimate a metric depth value. An example of an input image and corresponding depth map can be seen in **Figure 1.1.** Here the colors in the depth map correspond to the depth of that pixel: blueish means the pixel is closer to us and reddish means the pixel is further away from us.

Figure 1.1 : Input RGB image and the depth map estimated by the neural network of Fu et al. [2].

What makes the SIDE problem interesting and challenging is its inherent ambiguity. Endless number of different 3D scenes can result in the same 2D image. This suggests that there is a one to many mapping from RGB images to depth maps. If this is the case, how do human beings, whose visual systems highly surpass artificially created visual systems in terms of quality and generalization, estimate the depth from monocular images? The answer to this question lies in the cues humans use to do SIDE.

For estimating the depth from a single image, the human visual system is the most superior system in terms of quality and generalization. Foley and Maitlin [3] catalogue the known pictorial (static) monocular cues used by human beings to estimate depth from a single image. There are seven such static cues that we can use to estimate the depth from a static single image. The first cue is occlusion which happens when one object partially covers another one. The partially covered object is considered to be farther away. The second cue is called perspective. We can observe this by looking at parallel lines which appear to meet in the distance. There are two other cues that are related to the perspective. One of them is size cue. Same object can have different sizes on the retinal image according to its distance. Therefore, size of an object has an influence on our depth estimates. Second cue related to the perspective is texture gradient. It happens when you look at a surface at a slant. The texture of the surface becomes denser as the distance increases. Another cue that we use in order to infer depth is called the atmospheric cue. It refers to the observation that objects get blurry and bluish as they move away from us. Moreover, we use patterns of light and shadows when perceiving depth. We consider things like objects casting shadows onto other objects or having shadows attached to their surfaces. The last cue that we use is height cue. Objects closer to the horizon seem farther away.

The most important cue here is the size cue. As humans, we have a rough estimate of the objects' size that we see in real world everyday. When we look at the world and observe 2D RGB images, our visual system estimates the 3D scene between endless number of geometrically possible 3D scenes, using our prior knowledge to choose the one that fits into the world as we know it. This is also the reason why we are fooled by the images that are similar to the ones in the **Figure 1.2.** Since there is no other cue that tells us otherwise, we assume the chair to have a usual size and accordingly estimate its depth closer to us. However, by looking at the relative sizes of the human and the chair in the right image, we understand that the chair is father away than we estimated since it is bigger than that we assumed.

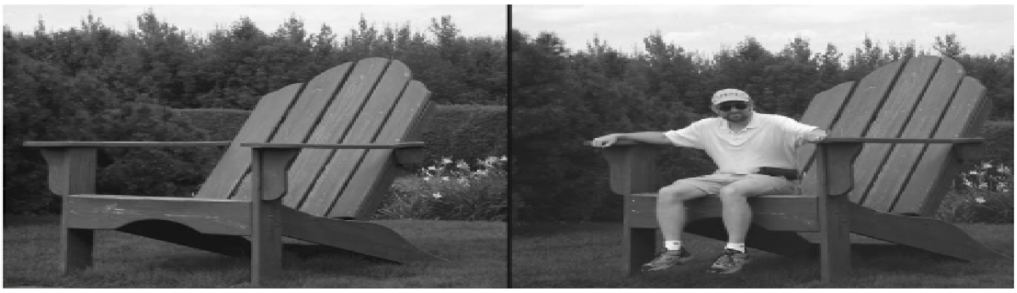

Figure 1.2 : An example of illusions that fools the human visual system. Image taken from [3].

All this information leads us to a very important conclusion: we as humans use learned prior knowledge and our visual system tends to work statistically [7]. This conclusion also directs the way the research in this area conducted. As can be seen in **Chapter 2,** statistical methods are heavily utilized to solve the SIDE problem.

1.1 Purpose of Thesis

The purpose of this work has three parts. Obviously, we want to develop a method to solve the SIDE problem in a better way compared to the existing methods (the meaning of the word "better" will be explained in detail later on). In the field of computer vision, estimating depth has always been an important problem. Estimating depth accurately can help us to do better object detection, semantic segmentation and improves scene understanding [8]. In addition, there are applications of depth estimation such as depth-aware image editing or rendering which can be used to enhance the virtual and augmented reality technologies, 3D modeling, obstacle avoidance, grasping, and robotics [9]. Problems that can be solved efficiently by estimating depth include estimation of detailed 3D structure of a scene [10] or obstacle avoidance with self driving robots [11]. Depth estimation is also utilized in full-self driving cars [12].

However, if our aim is to estimate the depth in a better way, why are we focusing on the "monocular" case? We know that the SIDE is a challenging problem. There are other alternatives that one can use to estimate the depth e.g. binocular depth estimation, structure from motion, or simply usage of depth sensors. Even though all of these alternatives can be used successfully, each has their own limitations. For binocular depth estimation, two cameras are needed and they need to be carefully calibrated [13]. Even all of this is given, it is still an open area of research since corresponding points in two stereo images need to be found. Moreover, it does not work on featureless regions [13] and has a limited range [11]. Another option is to incorporate motion to estimate the depth. This is referred as structure from motion (SfM) and requires motion. It is also not good on featureless regions in the image and normally assumes static scene, an assumption which does not always hold [14]. Lastly, different kinds of sensors exist to measure depth. However, these use high quality hardware which is very costly and power consuming, has a short range, produces sparse depth maps since the sensors do not work on all surfaces, or are sensitive to light conditions [13, 15]. As we can see, among these options SIDE is the most robust one with minimum requirements and limitations and that is the reason for us and for many other researchers to undertake the challenge of the SIDE problem.

The second point that makes our work important is the method that we use to tackle the SIDE. Following trends in the field and in similar fields, we have applied deep learning techniques which have achieved a lot of success recently. However, deep learning methods act like a black box tools and in order to understand and improve this tools, they have to be applied to different kinds of problems. We believe the fact that SIDE being a very challenging problem which can be done naturally by humans without difficulty shows that there is a room for improvement in terms of artificial intelligence development, and this makes the SIDE problem a suitable test bed for deep learning. We hope that our work will also contribute to the understanding and the success of deep learning techniques in depth estimation.

Lastly, artificial intelligence can help us to have a better understanding of human intelligence. For example, Moravec [16] expressed what is known as Moravec's paradox, which is the realization of the fact that high level tasks such as reasoning need less computational power compared to low level tasks such as bodily functions. Traditionally, it was believed to be the contrary, yet progress in the artificial intelligence allowed us to have a better understanding of the human intelligence. Similarly, it is possible to gain insights about human perception of depth if similar capabilities can be achieved by machines.

1.2 Contributions

This thesis offers an extensive literature review in **Chapter 2** which outlines the research categories regarding the SIDE problem. Works in the categories are summarized in a way that it highlights the logical progression. Common themes that are seen in multiple works, problem specific approaches and insights are emphasized.

Chapter 3 presents our work [17] and other works [18, 19] that investigate existing solutions in order to be able to shed light onto inner workings of depth estimating models. We have investigated whether models utilize global cues, how the training data set affects learning, to what extend the pictorial cues are utilized, and how models estimate depth.

Chapter 4 presents our work [20] in which we have for the first time formally formulated relative-SIDE problem as a ranking problem, proposed a new loss function and a new performance measure. We were able to achieve comparable performance with the state of the art.

Based on our literature review, analysis and insights on the subject, we share our conclusions and recommendations for future work in **Chapter 5.**

2. LITERATURE REVIEW

Early days In the early days of the field, the SIDE problem was not tackled directly. In the classical work of Hoeim et al. [7], the authors aim to automatically reconstruct a 3D scene from a given RGB image for virtual environment creation. Their approach makes the assumption that outdoor environments basically consist of sky, ground plane and vertical objects sticking out of ground. They use hand generated cues to classify superpixels in one of the three classes. Afterwards, using the three classes and the above-mentioned assumption, they automatically create the virtual environment by placing the objects on the ground plane vertically. Since the elements of the inferred scene are very simplified like a photo pop up from a child's book, there are some details missing. Nonetheless, the end results look pleasing to the eye. In this work, we see the first examples of two important recurring themes in the field:

- **Incorporating semantic segmentation** Semantic information is incredibly important for estimation of depth. It can help a computer vision system to exploit its prior knowledge for a given semantic class. For example, by looking at two pieces of blue patches from an image, we can estimate their depth by knowing one of them is sky and the other one is water. Even though semantic information is expected to be exploited by machine learning techniques implicitly, it also has been explicitly utilized to solve SIDE problem by many researchers in different ways [7, 21–25].

- **Separating indoor and outdoor** Although humans seamlessly estimate the depth of indoor and outdoor scenes without noticing any change between them, they are actually structured very differently. For instance, aforementioned assumptions made by Hoeim et al. [7] do not hold for indoor scenes. Even when researchers make no assumption about the structure of the environment, they still let their system work on a single type of environment most of the time. This is because of the inherent difference between indoor and outdoor environments which makes the statistical learning of a system that works on both type of environments a challenging problem. Nevertheless, some of the work in the field actually tackled the problem of SIDE in the wild [4, 26, 27]. Unless otherwise stated, all the works that are going to be mentioned are designed to work on either indoor or outdoor environments.

Another early work that uses SIDE to solve a given problem is the work of Michels et al. [11]. The task in this work is to navigate a high speed remote control car through obstacles in an uncontrolled outdoor environment. The designed framework consists of two parts: a vision part which fakes a 2D laser scanner and estimates the distance of nearest obstacle in each direction, and a reinforcement learning part that drives the car avoiding obstacles. The reason why depth is estimated from a single image in this case is that it gives a better range compared to usually preferred binocular vision. The vision system is trained using linear regression with hand crafted features in a supervised manner. The input image is divided into vertical strips. Each strip is labeled with the nearest obstacle's distance in log space. Hand crafted features are created for each stripe while preserving their spatial information. In order to be able to capture the global context, the system uses neighboring stripes' features as well as the features of the stripe that it is making its decision for. Additionally, we see different error metrics being used. The distance between estimated depth and the ground truth depth in log space is one of them. Also, relative depth error, where the mean is subtracted from estimated and ground truth depth values in log space, is used as an error metric. Moreover, synthetic data with different levels of realism is used to boost the success of the system. Since the task here is to avoid obstacles, the vision problem is formulated very differently (distance of the nearest obstacle in each direction) compared to our definition of SIDE (pixel by pixel dense depth estimation). Yet we see very important ideas that have been repeated by the later works in the field:

- **Working in log space** The challenge of estimating the depth of close objects and distant objects is not the same. While being a few centimeters off in our estimation of depth for an object that is meters away is acceptable, it is definitely a bigger mistake to be a few centimeters off if the object is only ten centimeters away. This is as much the case for humans as it is for computer vision systems. While we can be more precise in our estimations for smaller depths, we could only provide a rough depth range for bigger depths. For this reason, the errors are usually calculated in log space since the logarithm function maps the depth values in a way that error functions become more forgiving for mistakes in bigger depths. As can be seen from **Figure 2.1,** while the depth values between 0 and 3 meters are mapped into a approximately 2 unit range, depth values between 3 and 10 meters are mapped between approximately 1.3 unit range.

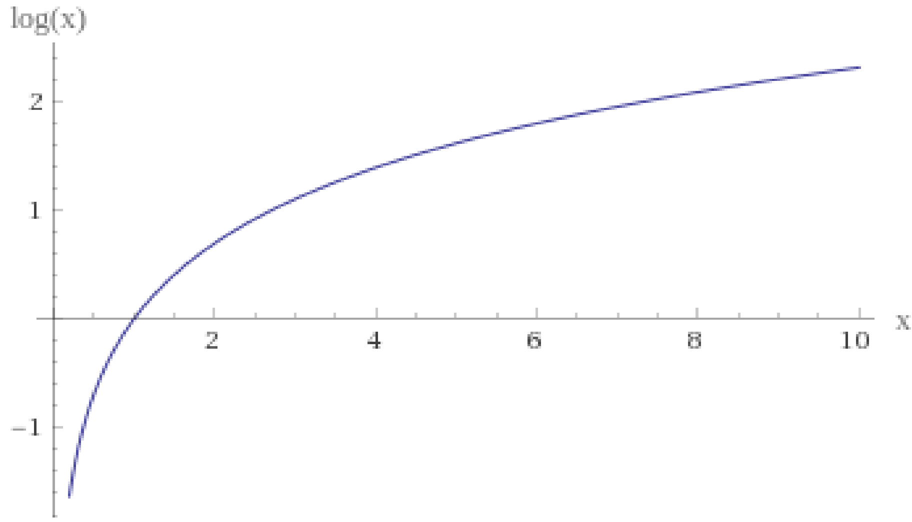

Figure 2.1 : Graph of logarithm function from 0 to 10.

- **Using spatial coordinates** Spatial coordinates can be an important cue to estimate depth. It can help to exploit the structure of the the scenes that system has seen before. When working on a data set consisting of outdoor images, the pixels in the upper rows have a high chance of being part of a sky and statistical learning systems can exploit this kind of relations easily. On the other hand, this exploitation can create bias in the system.

- **Incorporating global context** As stated earlier, the scale of the scene is ambiguous. However, this ambiguity can be ignored in real world scenarios since objects with known sizes can provide us enough cues to estimate the scale of the scene. These cues are called global cues that require us to look at a bigger part of the input image than just a local patch.

- **Using relative depth** The term "relative depth" refers to the ordering of the depth of pixels instead of absolute depth which refers to the metric depth values. Relative depth can be used to measure the performance of the system or can be used as an error function. This way, system would not be penalized for mistakes due to scale ambiguity.

- **Training with synthetic data** Machine learning systems need lots of data points in order to be able to learn the task at hand. For the SIDE task, data sets consist of RGB images and corresponding depth maps. Unfortunately, acquiring RGB images and corresponding depth maps is a costly job. Even though different data sets have been collected throughout last decades [28, 29], the need for labeled data is considered to be a problem in general in machine learning. One of the ways to overcome this problem is the usage of synthetic data. The system still learns with labeled data, however synthetic data can be created and labelled automatically with ease in great amounts. Moreover, similar to data augmentation, great diversity in the data can be achieved by changing the texture of the objects or the lighting of the scene while keeping the ground truth same, which will increase the robustness of the system. For all these reasons, usage of synthetic data is considered to be a solution. On the other hand, usage of synthetic data, in itself, creates a problem. Synthetic and real data are considered as different domains and the system needs to adapt to the real data after it is trained on synthetic data. Additionally, while diversity for a given scene can be achieved with ease, creating diverse sets of natural scenes synthetically is an incredibly time consuming job.

To best of our knowledge, the first work that tries to estimate a full metric depth map from a given single RGB image is the work of Saxena et al. [26]. They process an input image in small patches by applying hand crafted filters to extract image features. For each small patch, a single depth value is estimated. To be able to successfully determine the absolute depth, global cues are incorporated by applying the aforementioned filters in multiple scales, utilizing the neighboring patches' features and utilizing features from the same column. Features from the same image column are used based on the observation that most of the structures in the images are vertical. Additionally, to increase the understanding of the system for neighboring patches, histograms of the features are calculated for each patch and the system is fed with the difference between the histogram of a patch and its neighboring patches.

A Markov Random Field (MRF) model is trained in a supervised manner to estimate the metric depth from these features. Three different sets of parameters are learned from the training set for each row in the image. The reason for learning different parameters for different rows of the image is the observation that each row is statistically different from each other. First set of learned parameters is for estimating absolute depth for a given patch by looking at the features. Another set of parameters tries to estimate the uncertainty in the absolute depth estimation. The last set of parameters are related to the smoothing term in the model. Neighboring patches usually have a very similar depth when they are similar in RGB domain. Therefore a smoothing term is added to the model to bring neighboring patches' depth closer to each other. The effect of this term is controlled by the last set of parameters which determines the amount of the depth similarity of the patches. New themes encountered here are:

- **Working at multiple scales** The same object can look very differently when being looked at in different distances. While we do not have images of the same scene taken from different distances for the SIDE task, we could simulate a similar situation by changing the resolution of the image. By working in multiple scales, the system becomes more flexible since it can look at different cues at different scales to detect the depth of objects.

- **Incorporating human knowledge** Incorporating prior human knowledge in terms of hand designed filters, assumptions, and loss functions in MRF based models was a common practice. With the rise of deep learning techniques, this trend started to decline, yet we still see incorporated human knowledge in the form of network architecture design and loss functions.

- **Adding a smoothing term** A great example of added prior knowledge is smoothing term. We know that depth discontinuities only occur at the edge of objects. Depth maps mostly consist of smooth depth transitions where the neighboring pixels have a very similar depth. A smoothing term can be used in a loss function to achieve smoother depth maps. However, over smoothing of the actual depth discontinuities should be prevented.

In 2008, Saxena et al. published another work [10] that influenced the field with a very substantial assumption: scenes consist of small planar surfaces and the depth of all the pixels belonging to a surface can be calculated by the 3D location and orientation of the surface they belong to. This basically means that even the most complex 3D scenes can be expressed with the 3D location and the orientation of small surfaces. The validity of this assumption can be seen in graphics engines where many complex models can be created with simple triangular surfaces. They create these small surfaces by superpixelating the image in the RGB domain with the expectation of similar looking neighboring pixels belonging to the same surface.

We see lots of similarities with their previous work [26], the biggest difference being the use of superpixels. In this work, they again use the MRF model and train it in a supervised manner.

Image features are calculated via hand designed filters. To incorporate global information, the features of the neighboring superpixels are used. Similar to their previous work [26], the MRF model estimates different parameters for different rows of the image to model the relationship between image features and the 3D location and orientation of the superpixels. In their model, they also try to capture three more characteristics from the image:

- superpixels that are connected in 3D, since most of the neighboring superpixels should be connected except for the case of occlusion;
- superpixels that are on the same plane in 3D, since most of the superpixels are not only just connected but also parts of the same plane if no edges can be found between them;
- straight lines in RGB domain as they are most likely to be straight lines in 3D.

It is important to capture such characteristics as more constraints can be added for depth estimation based on these characteristics. Fractional (relative) depth error is used while applying these constraints. It is formulated as $\frac{(\hat{d}-d)}{d}$ where $\hat{d}$ is estimated depth and the d is the ground truth depth value.

They also extend their work by detecting objects and using prior knowledge to better estimate the depth of detected object such as detecting a human and expecting it to be connected to the ground or detecting two humans and using their sizes in pixels to better estimate their depth (the one with twice a size of other in pixels is most likely to be closer to the camera).

In this work, we see a very influential approach that allows the SIDE problem to be relaxed:

- **Superpixelating the input image** Based on the assumption that a 3D scene can be expressed with small planar surfaces, many scientist superpixelate the input image and estimate the depth of the superpixels [10, 21, 25, 27, 30–34]. This has the benefit of reducing the computational cost as the number of points for estimation is decreased by this assumption.

- **Coplanarity assumption** Superpixels that are similar in RGB domain are more likely to be on the same surface. This assumption is usually utilized to smooth the estimation of the model.

2.1 SIDE Landscape

Here we group the works based on their main consideration. Each work in the field approaches the SIDE problem in a different perspective, trying to improve different aspects of the existing solutions. While some of the works try to achieve better metric results in the data sets by simply applying new developments in the techniques or utilizing multi tasking, others try to overcome the need for labeled data, aim for better 3D structure, and so on. We group the works under 3 main categories as follows:

- **Eliminating the Need for Labeled Data** In this group, we present the works that solves the SIDE problem without annotated real world ground truth. Here we see two main approaches, usage of synthetic data where the costly ground truth data collection is replaced by automated creation of synthetic data and unsupervised learning where epipolar geometry is utilized to acquire supervision.

- **Increasing the Metric Performance** Here we summarize the works that mainly focus on increasing the metric performance. We organized works under two groups. First, we present works that tries to achieve better performance by architectural choices, loss functions, etc. Next, we present works that focuses on multitask learning.

- **Working in the Wild** Lastly, we summarize the works that focus on SIDE in the wild. Since the SIDE problem is reformulated in these works, we present the available data sets and ground truth annotations first, then summarize the works on SIDE in the wild.

2.1.1 Main consideration: eliminating the need for labeled data

In this research category, researchers try to eliminate the need for labeled data for the SIDE problem as labeled data is hard to acquire. Additionally, the need for labelled data prevents the life long learning opportunity, makes it hard to fine tune networks for unique situations. Two approaches have been developed by the researchers, namely usage of synthetic data and unsupervised learning without any labeled data.

2.1.1.1 Usage of synthetic data

One of the ways to overcome the need for labeled data is the usage of synthetic data. Even though the training is done in a supervised way with labeled data, synthetic data eliminates the costly real data collection since synthetic data can be created in an automatic way in great amounts and in great diversity.

Ren and Lee [35] try to solve the SIDE problem by learning more than one complementary task. Here they are inspired by humans as humans usually learn things jointly, utilizing more than one source of information. In particular, they try to estimate depth, surface normals and instance contours at the same time. They believe learning these tasks jointly would increase the performance of the system as the tasks share some common understanding for a given scene. However, it is very hard to find a data set that has labels for all of the tasks at hand. Therefore, they created their own data synthetically which allowed them to create a big enough data set with all the needed ground truths. They trained an estimator network, which is a CNN with three heads to estimate depth, surface normals and instance contours jointly.

One problem that the authors [35] face is the domain difference between the real and synthetic images. Since they do not assume labels for real data, they force the network to have similar low level feature maps for both real and synthetic images by applying adversarial training. Essentially, a discriminator network is trained for discriminating between synthetic and real world images by looking at low level feature maps produced by the estimator network. Afterwards, an adversarial loss is produced from the discriminator network and it is used to train the estimator network so that the gap between synthetic and real images in low level feature maps diminishes.

In this work, we see an example of an important theme:

- **Learning multi tasks** The reason many scientist tackled the SIDE problem by learning multiple tasks [21–25, 31, 34–37] is twofold. First, estimating depth from a given single RGB image requires a high level understanding of the scene such as detecting and recognizing objects and their relations in 3D. However, the loss from depth estimation alone may not be enough for the network model to discover those high level relations. Therefore, additional tasks such as surface normal estimation, semantic segmentation, intrinsic image decomposition along with their corresponding losses is utilized to train the networks. Moreover, we know that some of the aforementioned tasks shares a common understanding for the given scene; therefore, it is beneficial to learn them jointly to increase the robustness of the network.

2.1.1.2 Unsupervised learning

Another way of truly overcoming the need for labeled data is unsupervised learning. Unsupervised learning for the SIDE problem uses stereo images or video recordings with small changes in camera positions between frames as two consequent frames can be considered as stereo images. All the unsupervised methods roughly work as follows: the system takes a pair of RGB images (lets call them I_{left} and I_{right}) which show the same scene from two slightly different perspectives. Then the depth is estimated for one of the images, for instance, let us assume that we estimate the depth of I_{left}, as D_{left}. Using the estimated depth and the camera motion between frames, other image in the pair can be deterministically warped to produce the first image using visual geometry as follows:

$$I_{right}(D_{left}) \sim I_{left}. \tag{2.1}$$

Afterwards, the network can be trained with the reconstruction error.

$$loss = f(I_{right}(D_{left}) - I_{left}). \tag{2.2}$$

Fundamentally, the system trains in a self supervised way where it tries to produce the input image itself as an output. Meanwhile, depth maps are produced by the system as a middle step and in order to minimize the reconstruction loss, the system learns how to predict more accurate depth maps.

This method makes some assumptions. First of all, as it assumes static scenes; objects' positions in 3D should be the same in I_{left} and I_{right}. Lack of occlusion/disocclusion is also assumed. Lastly, it assumes Lambertian surfaces, i.e. reflectance of the surfaces does not change with the point of view. Even though these assumptions may not hold all the time, researchers were able to successfully train networks in an unsupervised way.

To the best of our knowledge, the first work that tackles the SIDE problem in an unsupervised way is the work of Garg et al. [15]. They collect their own data set using a calibrated stereo gig, which allows them to know the camera motion between two frames. Since they do not specifically aim for the best possible results, as stated in their work, they apply our basic definition of unsupervised learning of the SIDE problem. Just like an auto encoder where the image is first encoded into a smaller space and then decoded into itself with a minimal loss, their end-to-end trained fully convolutional network encodes the input image into a depth map and then inversely warps the other of the pair using the depth map with a reconstruction minimisation loss. Note that the only part that is learned from data is prediction of the depth map. The warping can be calculated directly and in a differentiable manner using the depth map and the known camera motion.

Godard et al. [38] follow the same basic principles and improve them. They train their network with rectified stereo pairs with known camera baselines to estimate disparity maps. The main difference from the previous work of Garg et al. [15] is that the network produces left and right disparity maps ($Disp_{left}, Disp_{right}$) by looking at only one of the input images. Having left and right disparity maps allow us to reconstruct both of the images from each other,

$$I_{left}(Disp_{right}) \sim I_{right}, \quad I_{right}(Disp_{left}) \sim I_{left}, \tag{2.3}$$

and both of the reconstruction losses are used to train the network:

$$loss = f((I_{left}(Disp_{right}) - I_{right})) + f((I_{right}(Disp_{left}) - I_{left})) \tag{2.4}$$

where f is the loss function consists of weighted combination of SSIM and L1.

Additionally, they force their network to produce consistent disparity maps by penalizing the network for differences in disparity maps when they are projected onto each other:

$$loss = \frac{1}{N}\sum_{i,j} |Disp_{left}^{(i,j)} - Disp_{right}^{(j+Disp_{left}^{(i,j)})}|, \tag{2.5}$$

where (i, j) is the coordinate of a pixel. Note that index into the disparity map varies in one dimension since the images are rectified.

Another work that does unsupervised learning of SIDE problem is the work of Zhou et al. [39]. They train their network using a data set consisting of video recordings. Videos in the data set are acquired with a single camera, and subsequent frames are used to train the network as they have small camera position changes between them, hence they act like a stereo pair. Contrary to the previous work of Garg et al. [15] and Godard et al. [38], they do not use known camera positions. Instead they train an additional network to estimate the camera movement between frames. Therefore, they are able to use any video recording as an input to the training and eliminate the need for calibrated stereo image pairs. Simply, they estimate the depth D_t of the target image I_t with a network $DepthCNN$,

$$D_t = DepthCNN(I_t), \tag{2.6}$$

estimate the camera motions $M_{t->t-1}$, $M_{t->t+1}$ between neighboring frames I_{t-1}, I_{t+1} and the target frame I_t with a network $PoseCNN$,

$$M_{t->t-1}, M_{t->t+1} = PoseCNN(I_t, I_{t-1}, I_{t+1}), \tag{2.7}$$

and reconstruct the target image by warping the neighboring frames I_{t-1} and I_{t+1} using estimated camera motions $M_{t->t-1}$, $M_{t->t+1}$ and estimated depth D_t

$$W(I_{t-1}, M_{t->t-1}, D_t) =\sim I_t, \tag{2.8}$$

$$W(I_{t+1}, M_{t->t+1}, D_t) =\sim I_t, \tag{2.9}$$

where W refers to warping function.

Another improvement to previously mentioned works is that they model the model limitation. As we discussed earlier, there are some assumptions that need to hold for this method to work. The authors train an additional network to predict how much their model can explain each pixel. In order to prevent bad pixels (pixels for which the assumptions do not hold such as a pixel on a moving car) having a negative effect on the training process, they weight the loss from each pixel using the predicted belief for that pixel:

$$loss = ExplainabilityCNN(I_t) \otimes f\left(I_t, W(I_{t-1}, M_{t->t-1}, D_t), W(I_{t+1}, M_{t->t+1}, D_t)\right) \tag{2.10}$$

where $ExplainabilityCNN$ is the network that produces mask showing how much each pixel is explainable, f is the photometric loss function, and $\otimes$ represents pixel-wise multiplication.

Wang et al. [40] point out two important weakness of the previous approach of Zhou et al. [39] and improve upon them. The first one is the scale ambiguity of the scene. As we discussed earlier, a change in the scale in the depth map results in the same 2D image; therefore, the photometric reconstruction losses of the two depth maps that only differ in scale will be the same. Since we train our networks with the guidance of the photometric reconstruction loss function, different scales of a scene are equally likely from the perspective of the network. The only difference here is the regularization term in the loss function which is used for the smoothness of predicted depth map. Here the following problem occurs: as the scale of the scene decreases, the regularization loss also decreases. Therefore, the network learns to predict smaller and smaller scaled depth maps and eventually the training diverges. The authors successfully solve the problem by normalizing the output depth map before calculating the loss.

The second problem they detected is the separate estimation of depth and camera pose. They considered this as a problem since they believe these are related tasks. Additionally, with recent developments [41], camera pose estimation between frames is treated as an optimization process by using frames and the depth, without any learning required, and in a differentiable manner. They injected this module into the framework and removed the pose estimation CNN. As a result, the depth estimating network is now updated with the gradients of the pose estimation module and the number of parameters to be learnt is decreased.

Ranjan et al. [14], combine the idea of jointly solving related tasks and addressing the model limitation. Similar to Wang et al.'s work [40], they solve related tasks together, namely SIDE, camera motion estimation, optical flow, and motion segmentation (segmentation of the scene into static and moving parts). As they state, all of these tasks are related and solving them together benefits the system. Moreover, similar to the work of Zhou et al. [39], they model the limitations of their model by segmenting the input image into moving and static parts and employing different networks to estimate the depth of different parts of the image. What makes their work interesting is that all of these tasks are learned through unsupervised learning, in a framework that they called "Competitive Collaboration".

In the Competitive Collaboration framework [14], there are three players and a resource. Two of the players compete for the resources, hence the name competitive. The last player, called the moderator, distributes the resources to the players and to increase overall performance of the competitors, it is trained by competitors, hence the name collaboration. In our case, the players are networks and the resource is the training data. The first network, R, estimates the optical flow for the static parts of the scene using depth and camera motion while the other competitor network, F, estimates the optical flow for moving parts of the image. The moderator network distributes the resource by segmenting the input image into static and moving parts which will be used to train corresponding networks. Competitors and the moderator take their turn in a training cycle that is similar to the expectation maximization method [42]. For details of the training procedure, application of it to a different problem, and theoretical analysis, refer to the paper, as they are not directly related to the SIDE problem.

2.1.2 Main consideration: increasing the metric performance

In this section, we will examine the works that are mainly focused on getting better metric results. Naturally, some works closely follow previous works and just apply new extensions and developments in the given techniques. On the other hand, there are also works that incorporate human knowledge to boost the performance. Lastly, we will examine works that learn multiple tasks jointly to create synergy in order to get better results.

2.1.2.1 CNN based works

Here we list the works that use CNNs at the core of their frameworks. Note that this will not be an exhaustive list of works that use CNN, instead we are going to list works that use CNN while focusing on getting better results through architectural choices, loss functions, and so on.

In 2014, Eigen et al. [43] introduce CNNs to the SIDE problem and achieve relatively good performance, when compared to earlier methods. CNN based solutions were already achieving quite satisfactory results in different vision problems at that time. Eigen et al. utilize the experience gained so far on CNNs and combine it with problem specific knowledge to tackle the SIDE problem. They formulate the problem as a supervised regression problem and solve it with their framework, consisting of two networks, namely a *Coarse* and a *Fine* network, which are stacked on top of each other. Its operations can be summarized as follows:

$$Coarse(Input) = Depth_{coarse} \tag{2.11}$$

$$Fine(Input, Depth_{coarse}) = Depth_{fine} \tag{2.12}$$

The *Coarse* network consists of convolutional layers and fully connected layers at the end. Because of the fully connected layers, we could say that the network makes its decision by looking at the image as a whole. This allows it to utilize "global context" of the image and make a coarse estimation of the depth of a scene. However, fully connected layers come with a huge computational cost. In order to be able to keep the model reasonable in terms of memory, the output resolution is decreased.

The *Fine* network is a fully convolutional network and works by considering only the local parts of the image. It takes the original input image, and the estimation of the coarse network. In a sense, it refines the coarse estimation by working locally. To deal with the scale ambiguity, the authors define a scale invariant loss in the log space. Three reformulations of the same loss function can be seen below. y and y^* are the predicted depth map and the ground truth depth map, respectively. Sub indices indicate the pixels, and n is the total number of pixels in an image.

$$loss = \frac{1}{2n} \sum_{i=1}^{n} (\log y_i - \log y_i^* + \frac{1}{n} \sum_i (\log y_i^* - \log y_i))^2 \tag{2.13}$$

$$loss = \frac{1}{2n^2} \sum_{i,j} ((\log y_i - \log y_j) - (\log y_i^* - \log y_j^*))^2 \tag{2.14}$$

$$loss = \frac{1}{n} \sum_i (\log y_i - \log y_i^*)^2 - \frac{1}{n^2} \sum_{i,j} ((\log y_i - \log y_i^*)(\log y_j - \log y_j^*)) \tag{2.15}$$

Equation 2.13, subtracts the mean loss from the loss of each pixel in order to make up for mistakes due to scale ambiguity. We could have a different interpretation by reformulating it as **Equation 2.14,** which compares each pixel pair in the ground truth and the same pixel pair in the estimated depth map, in order to achieve the same distance between pixels in both the ground truth and the estimated depth maps. Similarly, the same loss function can be written as in **Equation 2.15,** which can be interpreted as the network being penalized for mistakes in different directions and being rewarded for mistakes in the same direction.

All three above formulations are equivalent, and actually a fourth reformulation where the function can be computed in linear time, is used to train the network:

$$loss = \frac{1}{n}\sum_{i}(\log y_i - \log y_i^*)^2 - \frac{\lambda}{n^2}(\sum_{i}(\log y_i - \log y_i^*))^2 \tag{2.16}$$

Here, λ is a hyperparameter which controls the scale invariance of the loss. $\lambda = 1$ refers to fully scale invariant, and $\lambda = 0$ refers to the normal L2 loss.

Following their previous work in [43] very closely, and building on top of it, Eigen et al. [24] devise a network architecture that can successfully estimate depth, surface normals, and semantic labels. Note that they do not optimize their system jointly for the said tasks, instead they merely show that a single neural network architecture can solve all the tasks. Although they experiment with shared layers between depth and surface normals, it does not improve their results.

As an improvement to their previous work, another scale is added to their multi-scale network architecture, which further refines the output using convolutional layers. Additionally, they incorporate an extra term to their loss as shown in **Equation 2.17,** which tries to match the gradient of the depth in the estimated depth map and the ground depth map, that results in a better local structure in the output depth maps:

$$loss = \frac{1}{n}\sum_{i} d_i^2 - \frac{\lambda}{n^2}\left(\sum_{i} d_i\right)^2 + \frac{1}{n}\sum\left[(\nabla_x d_i)^2 + (\nabla_y d_i)^2\right], \tag{2.17}$$

where $d_i = \log y_i - \log y_i^*$ and $\nabla_x d_i$ and $\nabla_y d_i$ are image gradients.

Both [43] and [24] were able to achieve very good results at that time. Nevertheless, there were still opportunities for improvement. First of all, most real world tasks require a real time vision pipeline, and Eigen et al.'s framework [43] is slow. Another of its weaknesses is having too many learnable parameters that leads to an increase in memory requirements, and the need for too many training points to train the network. Lastly, the resolution of the output is very low. Again, from the perspective of the application, higher resolution outputs are more desirable.

Laina et al. [6] aim to solve these problems in their work by applying recent developments in the CNN technology. Essentially, they follow the work of Eigen et al. [43] by training a CNN to regress per pixel depth value. Instead of having two networks, and fully connected layers, they train a fully convolutional residual network that is based on ResNet-50 [44], with up-convolutional blocks at the end. Their residual network has less parameters, runs in real time, and up-convolutional blocks increase the resolution of the output. Even though the lack of fully connected layers looks like a problem in terms of global context, the receptive field of the deep residual network covers the whole input image, providing the necessary global context.

Another improvement of [6] is on the loss function. After observing a heavy tailed distribution of depth values in the data sets, it uses a reverse Huber loss called BerHu, which acts like an L1 loss below threshold c and acts like an L2 loss above that threshold. The BerHu loss is given by

$$loss(y_i, \hat{y}_i) = \begin{cases} |y_i - \hat{y}_i| & |y_i - \hat{y}_i| \leq c \\ \frac{(y_i - \hat{y}_i)^2 + c^2}{2c} & |y_i - \hat{y}_i| > c \end{cases}, \tag{2.18}$$

where c is calculated over all the pixels of a batch of input images as $c = \frac{1}{5}\max_i(|y_i - \hat{y}_i|)$. This loss helps to put more emphasis on small residuals while still having the advantage of L2 loss for high residuals.

Cao et al. [8] formulate the SIDE problem as a classification problem. The main motivation for this is two fold. First of all, it is hard to regress to the exact depth value, even humans have a hard time estimating the exact depth. Instead, one can estimate the depth range with ease. Additionally, doing classification naturally produces a useful by-product that cannot be produced by doing regression without extra difficulty. The by-product is the confidence on the estimation that can be utilized to further enhance the estimation of the network by updating the network's estimation with the low confidence based on neighboring estimations with high confidence as a post processing.

Cao et al.'s framework [8] consists of two parts. First part is a ResNet based on a fully convolutional network that estimates the depth range for each pixel in the input RGB image. The depth ranges are acquired by uniformly discretizing the continuous depth values in the log space in the ground truth depth maps. This network is trained with a cross entropy loss weighted by an information gain matrix, so that depth ranges close to the ground truth depth range are also used to update the network's weight as follows:

$$loss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{D=1}^{B} H\left(D_i^{GT}, D\right)\log\left(P\left(D|z_i\right)\right) \tag{2.19}$$

where $H(p,q) = \exp\left[-\alpha(p-q)^2\right]$, and α is a constant. D refers to discrete depth labels while D_i^{GT} refers to ground truth depth label for i^{th} image, P refers to estimated probability for the depth label. Normally, the cross entropy loss tries to increase the probability of the correct class. Here, this loss also tries to increase the probability of the closer classes in depth domain. In the test time, center of the estimated depth range is assigned to a pixel's depth value.

Second part of Cao et al.'s framework is a fully connected conditional random fields (CRF) [45] that consists of unary potentials for each pixel and pairwise potentials for each pixel pair in the image. While the unary potential pushes the system to output correct labels for each pixel, the pairwise potential smooths the depth estimation by looking at pixels' positions and their appearance in RGB domain.

An important theme we see here is:

- **Classification and ordinality** A number of works decided to formulate the depth estimation problem in a way that does not require the system to estimate the exact depth value [1, 2, 4, 5, 8, 33]. While estimating the relative depth instead of the absolute depth is an alternative, estimating the depth range in a classification setting can be done with success. It is important to see that the classification formulation of the depth estimation is inherently different than the classic classification formulation since the classes in depth estimation problem represent depth ranges that have ordinal relations with one another. This has been utilized to increase the performance in [2, 8].

Fu et al. [2] determine two important drawbacks of the current approaches so far. Similar to [8], they advocate that the regression formulation of the depth estimation is hard to train with, thus results in poor solutions. Additionally, they claim that the networks that have been used consist of spatial pooling operations, which reduce the output resolution, followed by deconvolutions and skip connections to increase the resolution, which, as a whole, complicates the networks and adds computational cost.

To improve upon their observations, they formulate the problem as an ordinal regression problem. Despite the name having regression in it, their system actually consists of a network with multiple heads where each head of the network solves a pixel-wise binary classification problem in which the aim is to decide whether a pixel is closer or further away from a particular depth threshold. More specifically, they pick the number of depth thresholds $t_i \in t_0, t_1, .., t_{K-1}$ defined by:

$$t_i = e^{\log(\alpha) + \frac{\log(\beta/\alpha)*i}{K}}, \tag{2.20}$$

where $[\alpha, \beta]$ is the depth interval. The important thing to notice here is that it increases the distance between consecutive thresholds, which affects the loss in a way which working in the log space does in classic regression formulation.

The network has a head h_{t_i} for each of the t_is, and each head classifies all the pixels. The network is trained with the loss:

$$loss = -\frac{1}{N} \sum_{i=0}^{N-1} \sum_{k=0}^{K-1} [d_i > t_i] \log(1 - \hat{y}_i^k) + [d_i < t_i] \log(\hat{y}_i^k), \tag{2.21}$$

where N is the number of pixels, $[.]$ is the indicator function, d_i is the ground truth depth of the i$^{\text{th}}$ pixel, and the $\hat{y}_i^k$ is the prediction of the k$^{\text{th}}$ head for the i$^{\text{th}}$ pixel.

Inference stage also needs special care as we have multiple ordinal predictions for each pixel. The estimated depth of a pixel is calculated using the classification probabilities obtained by each head of the network for that pixel with the formula given by:

$$\begin{aligned} \hat{l}_{(w,h)} &= \sum_{k=0}^{K-1} [\hat{y}_i^k >= 0.5] \\ \hat{d}_i &= \frac{t_{\hat{l}_i} + t_{\hat{l}_i+1}}{2}. \end{aligned} \tag{2.22}$$

Following the trends of the time of the paper [2], they replace spatial pooling operations with dilated convolutions, which allow them to preserve resolution of the image and get rid of costly up sampling part of previous architectures. Their architecture consists of a generic feature extractor part followed by a scene understanding modular which, in parallel, applies dilated convolutions with different dilation rates, 1x1 convolutions to detect cross channel features, and a full image encoder to capture global information. All the feature maps created by scene understanding module is concatenated and fed to the heads of the network for final pixel-wise prediction.

2.1.2.2 Multitasking

Here, we list the works that solves SIDE problem jointly with similar task or tasks. In our case, similar means the closeness of tasks that are being solved jointly in the solution domain. As we discussed earlier, synergizing similar tasks helps to improve the results and increase the robustness of the acquired solutions. Note that this list does not contain all the works utilizing multitasking as some of the works are discussed under different sections to emphasize different aspects of the said works.

Liu et al. [21] combine SIDE with the semantic segmentation task. As explained earlier, the importance of the semantic information for depth estimation task is their main insight. They design a two staged system where semantic labels are predicted and used to apply additional constraints for the depth estimation task. Furthermore, the SIDE problem becomes easier to solve as simpler features may explain the depth of a point conditioned on the semantic class information.

MRFs are utilized to estimate the per pixel semantic classes by minimizing a unary label potential and a binary smoothing potential that is guided by similarity in the RGB domain. On top of semantic segmentation, they estimate the horizon, and semantically decompose the scene, which is similar to [7]. This decomposition is used to compute additional hand-crafted constraints on the depth of the pixels. To be able to fully exploit available semantic information, a separate MRF model for each semantic class is trained to estimate the depth in the second stage.

Ladicky et al. [23] observe that the older approaches do not incorporate perspective information. The look of an object in the images varies greatly depending on its depth and it is a problem for data driven approaches as they need samples for each object at each possible depth to be able to learn accurate representations. They claimed that this is an important problem for both semantic segmentation and depth estimation tasks.

To overcome this problem, they use the fact that the depth of a pixel in an image will change reversely proportionately to any scaling of the image. Explicitly, the expression:

$$H_d(I,i) = H_{d/\alpha}(\alpha * I, \alpha i) \tag{2.23}$$

holds for every pixel i, and scaling factor α, where d is an arbitrary depth, $H_d(I,i)$ is the probability of pixel i to be at depth d in image I, and $\alpha * I$ is the geometrically scaled image.

Using this information, they train L number of classifiers $H_{d,l}(I,i)$ that output the probability of a pixel i in image I being at an arbitrarily chosen depth d, and having the semantic class label l where L is the total number of semantic classes. The input image I can be scaled with different scaling factors α to determine the correct scaling that projects the pixel to the chosen depth d. Actual depth of the pixel can then be calculated using the obtained scaling factor. This reformulation of the problem helps the system by reducing the joint problem of estimating the depth and semantic class into simpler problem of detecting whether a pixel is of a particular depth and of a particular class. They claim that the learned features would be simpler since they only have to work for a specific depth as opposed to features that have to work across a depth range.

Wang et al. [25] try to unify SIDE and semantic segmentation in a framework by jointly solving both tasks instead of solving them sequentially. The latter is prone to error due to propagation from one task to another. Their framework consists of a complicated hand-crafted pipeline, which includes a CNN for pixel-wise depth and semantic label prediction, another CNN for a super-pixel based depth and semantic label prediction, and a hierarchical CRF that works on both pixel level and super pixel level estimations, and refines them by applying numerous hand crafted constraints similar to the constraints we have seen so far in other works. While the CNNs estimate depth and semantic labels in a global scale, hierarchical CRF refines the results locally. They were able to show that joint training of depth and semantic segmentation increases the performance of the both tasks individually.

Qi et al. [46] tackle the SIDE problem by combining it with surface normal estimation. Their network is called GeoNet, a geometric network, because they are utilizing geometric relations between depth and the surface normals to train the network. Their main observation is the inability of neural networks to learn these geometric relations from data directly, which they have shown by a poor performance of a CNN on the task of estimating surface normals from given depth maps.

The main assumption to allow applying geometric constraints is that the neighboring pixels lie in the same local tangent plane in 3D. This assumption allows us to fit a local tangent plane in a neighborhood of pixels using their depths, or estimating the depth of pixels in the neighborhood using surface normals assuming that they are on the same local tangent plane. This geometric relation between depth and surface normals is utilized to boost the performance of both tasks.

Qi et al.'s framework [46] consists of a generic network that makes the initial estimation for depth and the surface normals given the RGB image. Afterwards, these initial estimations are refined by "depth-to-normal" and "normal-to-depth" networks while both of the networks take the initial estimations as input.

The "Depth-to-normal" network takes the estimated initial depth values and calculates the 3D position of each pixel. The neighborhood for any pixel N_i is calculated based on the closeness in coordinates in 2D, and closeness in the depth value, where the closeness is controlled by hyperparameters β and γ as given by:

$$\begin{aligned} N_i = \{ (x_j, y_j, z_j) \mid \ & |u_i - u_j| < \beta , \\ & |v_i - v_j| < \beta, |z_i - z_j| < \gamma z_i \} , \end{aligned} \tag{2.24}$$

where (u, v) represents the coordinate of a pixel in image domain. Using the aforementioned assumption, a local tangent plane is fitted to each neighborhood, and then surface normals are calculated for each pixel. Note that these steps can be done deterministically, without the need for learning any parameters. Subsequently, a residual CNN module is applied to fuse the geometrically calculated normals and the initial normal estimation, and produces the refined end result for surface normals.

The "Normal-to-depth" network works in a similar fashion. It determines the local neighborhood by the distance in the 2D image coordinates and the surface normals in the initial estimation. For each pixel j in the neighborhood of pixel i, a depth estimation for pixel i is calculated based on the assumption that they are lying on the same local tangent plane. Subsequently, these assumptions are aggregated using a kernel regression with a linear kernel. Overall, the initial depth estimation is refined using the initial normal estimation and geometric constraints without introducing any learnable parameters to the framework.

Xu et al. [47] tackle the SIDE problem jointly with the semantic segmentation problem, by predicting and combining intermediate outputs consisting of complementary tasks. Their main idea is to utilize a pseudo multi-modal input for the estimation of each single task since we expect a network to perform better with multi-modal inputs compared to inputting just an RGB image.

To this end, they devise a two staged framework. In the first stage, a generic encoder E generates feature maps that are processed by separate decoders D_d, D_{sf}, D_c, D_{ss} to output pixel-wise depth $d_{intermediate}$, surface normals $sf_{intermediate}$, contour labels $c_{intermediate}$, and semantic labels $ss_{intermediate}$.

In the second stage, a multi-modal distillation module is devised to combine the intermediate outputs to predict depth estimation and semantic segmentation separately. First, intermediate outputs are turned into feature maps f_t where $t \in \{d, sf, c, ss\}$ by expanding the channel dimensions via applying convolutional layers. For the depth estimation task, an attention map a_d is produced using feature maps of intermediate depth output f_d, and feature maps of intermediate complementary tasks are gated with this attention map. Gated feature maps and feature maps of the intermediate depth output then are concatenated and passed to the final prediction module H_d that consists of convolutional layers to estimate the final depth map d_f as follows:

$$d_f = H_d\left(cat(f_d, a_d \otimes f_{sf}, a_d \otimes f_c, a_d \otimes f_{ss})\right), \tag{2.25}$$

where $\otimes$ represents pixel-wise multiplication. Similarly, feature maps of the intermediate semantic output used for attention mechanism and combined feature maps are passed to another module to predict final semantic labels.

A simple Euclidean loss for estimation of depth and surface normals, cross entropy loss for contour detection, and softmax loss for semantic segmentation are used for all intermediate and final outputs. The whole architecture is trained in an end-to-end fashion with linear combination of these losses.

Zhang et al. [48] also tackle the SIDE problem with semantic segmentation jointly. However, they bring a different perspective to the multi-task learning problem. Instead of having an architecture with shared layers up to a point and bifurcating from there on, they have devised a framework inspired by the human learning system where the framework alternates between the two complementary tasks back and forth. This allows them to utilize high level cross-task knowledge, compared to sharing layers where just common information is shared between tasks.

Zhang et al.'s architecture is an auto encoder with skip connections from the encoder part where the decoder part of the network outputs multi scale depth and semantic segmentation estimations. The decoder consists of four sequential main blocks $D^i \in D^1, D^2, D^3, D^4$, each of which has heads H_d^i, H_s^i to output depth and semantic segmentation predictions d^i, s^i in different scales. Each block of the decoder consists of two inner blocks D_d^i, D_s^i where the feature maps generated by the first inner block are used to output depth and the feature maps generated by the second inner block are used to estimate semantic labels. Furthermore, feature maps of previous depth and semantic segmentation outputs are gated with modules called *TAM* which applies the attention mechanism, and used as an input for the higher scales. **Equation 2.26** summarizes the process. Note that skip connections from the encoder part of the architecture are omitted for brevity.

$$\begin{aligned} feature_d^i &= D_d^i\left(TAM(feature_s^{i-1}, feature_d^{i-1})\right) \\ d^i &= H_d^i(feature_d^i) \\ feature_s^i &= D_s^i\left(TAM(feature_s^{i-1}, feature_d^{i})\right) \\ s^i &= H_s^i(feature_s^i) \end{aligned} \tag{2.26}$$

Essentially, their framework refines the depth and semantic segmentation outputs, going from smaller scales to bigger scales, alternating between depth estimation and semantic segmentation tasks utilizing the cross-task knowledge similar to [47].

Zhang et al. [49] combine SIDE, surface normal estimation and semantic segmentation tasks in their framework. They point out an important issue of multi-task approaches: ambiguity in the feature learning. Essentially, they claim that trying to learn features for multiple tasks may create ambiguity. Instead of letting a network learn the features that may be useful for all the tasks at hand, they propose to devise a method where they can explicitly look for similar features between tasks.

First of all, they investigate the existence of such patterns between tasks. To this end, they label pixel pairs as similar if their depth difference is smaller than a threshold or dissimilar if vice versa. The same process is applied to surface normal ground truths and semantic segmentation maps in which the similarity is decided based on the label equality. They have shown that 50% - 60% percent of pixel pairs that are labeled as similar in one task are similar in other tasks and same holds for dissimilar pairs too.

To exploit this observation, they devise a pattern-affinitive propagation method. Firstly, the RGB image is processed by a shared encoder, and task-specific decoders output initial estimates for each task. Using the last feature maps f_d, d_{sf}, f_{ss} generated by task specific decoders, affinity matrices M_d, M_{sf}, M_{ss} are calculated as

$$M_{t,ij} = e^{-s(f_{t,i}, f_{t,j})}, \tag{2.27}$$

where $s()$ is the similarity function such as an L1 distance, inner product, and so on. Additionally, each row of the affinity matrix is normalized. Basically, it expresses the similarity probability in the feature space for each pixel pair ij. Note that calculation of affinity matrices is done deterministically and does not introduce any learnable parameters. After the affinity matrices are calculated, they are combined by a linear weighting using a separate learnable weight set for each task, which is used to propagate affinities in the feature space. The final predictions are done using the affinity propagated features.

Chen et al. [50] tackle unsupervised SIDE and supervised semantic segmentation problems together. Unsupervised depth estimation is performed similar to the stereo approach of Godard et al. [38]. Similar to previous works, they propose architectural improvements to have a better multi-task learner as well as additional losses that utilize a cross-task knowledge to boost the performance of the tasks.

Their proposed architecture consists of an encoder E, which produces a scene representation z of the given RGB image I, and a decoder network D with skip connections from the encoder, which takes the scene representation z and the task identity t to generate cross-modal prediction $\hat{y}$. Task identity t is just a layer with all 1s for the depth estimation task and all 0s for the semantic segmentation task. Essentially, the same decoder learns to generate predictions conditioned on the task identity, allowing them to share weights and knowledge to a greater extent. Lastly, either softmax or pixel-wise average pooling is applied to cross-modal prediction to get semantic class probabilities or the depth map, respectively. **Equation 2.28** summarizes the framework:

$$\begin{aligned} z &= E(I), \\ \hat{y} &= D(cat(z,t)), \\ \hat{y}_{final} &= f(\hat{y}), \end{aligned} \tag{2.28}$$

where f is the softmax function if the task is semantic segmentation, and pixel-wise average pooling if the task is depth estimation.

Furthermore, two self supervised losses are defined to constrain the depth estimation using semantic knowledge. Similar to reconstruction of one stereo RGB image from another using the estimated depth information, estimated semantic segmentation maps are warped to reconstruct each other, and the L1 loss is applied to penalize inconsistencies between the reconstructed semantic segmentation maps. Additionally, a regularization loss that is guided by semantic class predictions is added since, intuitively, similar depth values within the same object are desired.

2.1.3 Main consideration: working in the wild

So far, the works that we summarized work on indoor and outdoor data sets separately. However, eventually we would like to train systems that can work on images coming from unstructured environments, i.e. systems that work in the wild. However, there are two important issues to consider for depth in the wild problem.

The first issue a system faces that aims to work in the wild is that the depth ranges may vary quite a lot which is a problem for the learning process. While the range of absolute depth values for an image of a table may be between 10 cm and a couple of meters, the image of a touristic place such as "Eiffel Tower" contains pixels that are hundred meters away, if they have a depth value at all (sky regions can be considered infinitely far away). This kind of diversity makes the problem of estimating an absolute depth quite hard.

Moreover, collecting data sets with absolute depth annotations for a diverse set of scenes that will allow a system to generalize unseen natural scenes is also very problematic. It requires a lot of effort in terms of time and money.

It is clear that a new approach with new data sets were needed and researchers have quickly come up with a new formulation to the SIDE problem, that we are going to refer as relative-SIDE, which allows the system to generalize better in the wild. To this end, new data sets and new methods to learn within this problem setting were introduced. We are going to summarize the data sets first, and discuss the works that are done so far.

Before we start discussing the data sets, we want to summarize the work of Zoran et al. [33], which we see as the spiritual predecessor of the works we are going to discuss later on.

Zoran et al. [33] tackle the SIDE problem in a supervised manner. They formulate the problem as a three-way classification problem where the network estimates the relation between a pixel pair as one of $d_i > d_j$, $d_j > d_i$, or $d_i = d_j$ where d_i and d_j are the depths of the pixel i and pixel j, respectively. Since it is infeasible to estimate the relations of all possible pixel pairs given an image, they have utilized the superpixel assumption and centers of the neighboring super pixels are compared with each other. In order to incorporate long distance relations, an input image is superpixelated at multiple scales. Once the relative relations are predicted, a global optimization procedure estimates the absolute depth values using predicted relative relations and data set statistics such as mean and variance of the depth values.

The advantage of this reformulation is twofold. First of all, the pixel-wise regression problem is reduced to a simple three-way classification which allows the usage of well studied classification methods. Additionally, it is more natural, and hopefully easier to learn to estimate the relative relations of pixels instead of estimating their absolute depth value.

2.1.3.1 Data sets

Chen et al. [4] make the first attempt towards depth estimation in the wild and bypass the aforementioned problems by collecting a data set with relative depth annotations. They crowdsourced the data set collection process using Amazon Mechanical Turk. Their data set, called Depth In the Wild (DIW), consists of randomly sampled internet images from diverse set of scenes. The ground truths consist of the relative relation between two pixels as can be seen in **Figure 2.2,** where green points are annotated as closer by the annotators.

Xian et al. [1] introduce another data set, called ReDWeb. The ReDWeb data set utilizes stereo images from the internet to reconstruct the corresponding depth maps

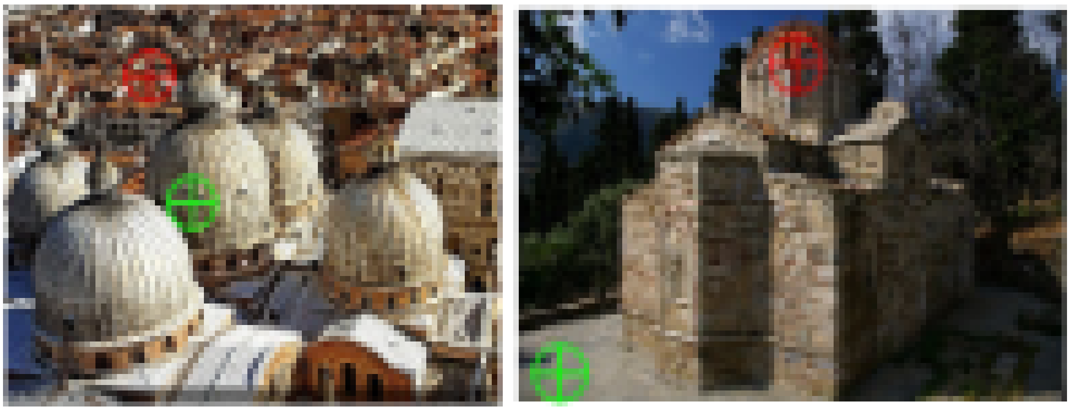

Figure 2.2 : An example input image from DIW data set. Green points are closer compared to red points. Taken from [4].

automatically. Some post processing is applied to the images to improve the quality of the acquired depth maps. This data set consist of dense relative depth annotations for each image in terms of pixel wise depth scores. The depth score s_i for pixel i and depth score s_j for pixel j are arbitrary numbers that satisfies the relations:

$$\begin{aligned} &s_i > s_j, \text{ if } d_i > d_j \\ &s_i < s_j, \text{ if } d_i < d_j \\ &s_i = s_j, \text{ otherwise.} \end{aligned} \tag{2.29}$$

Example input ground truth pairs from the data set can be seen in **Figure 2.3.**

Figure 2.3 : Examples from ReDWeb data set. Taken from [1].

Lastly, Chen et al. [5] introduce another data set called Youtube3D. They use structure from motion (SFM) techniques to automatically label Youtube videos with relative depth annotations. Their ground truths in the data set consist of relative relations of varying number of pixels as can be seen in **Figure 2.4.**

All the data sets that are designed to work in the wild have relative ground truths. Therefore in this research category, the SIDE problem is reformulated as

Figure 2.4 : Examples from Youtube3D data set. Red points are closer than blue points. Taken from [5].

a relative-SIDE problem, where the aim is to estimate pairwise relations of pixels instead of their absolute depths. Throughout the text, we are going to use the term "relative-SIDE problem" and the term "depth in the wild problem" interchangeably.

2.1.3.2 Learning from relative depth annotations

The main motivation of the aforementioned three works is to present an automatic way to collect and annotate data for depth estimation in the wild problem. Therefore in these papers, the main focus is the data set. Therefore they use very similar frameworks to tackle the problem.

In general, their framework consists of an auto encoder style network which takes the input RGB image and regresses a score map that satisfies the ground truth relative depth annotation in the data set. The most important part of the framework is the loss function, which uses the ground truth relative depth relations to train the network.

The loss that is used in the work of Chen et al. [4] for relative depth estimation is given by

$$loss = \begin{cases} \log\left(1+\exp\left(-\hat{s}_i+\hat{s}_j\right)\right), & \pi_{ij}=+1 \\ \log\left(1+\exp\left(\hat{s}_i-\hat{s}_j\right)\right), & \pi_{ij}=-1 \\ \left(\hat{s}_i-\hat{s}_j\right)^2, & \pi_{ij}=0 \end{cases}, \tag{2.30}$$

where $\hat{s}_i$ is the depth score prediction of the network for the pixel i, and the $\hat{s}_j$ is the depth score prediction for the pixel j. π_{ij} shows the relative depth relation between the two pixels where $\pi_{ij}=+1$ means pixel i is closer, $\pi_{ij}=-1$ means pixel j is closer and $\pi_{ij}=0$ means two pixels are at the same depth.

What this loss does is very simple. For $\pi_{ij}=0$, it penalizes the network for outputting different values. For $\pi_{ij}=+1$ and $\pi_{ij}=-1$, the loss function pushes network to increase the distance between the estimated depth for the given pixels in the correct direction depending on the relative depth relation [1].

[1]Note that the actual value that network outputs is irrelevant, only the difference between predicted depth scores affects the loss.

Xian et al. [1] use the exact same loss with very small modifications. In the previous work, Chen et al. [4] experiment with the only available data set, DIW, which has ground truths of only one pixel pair per image to apply this loss. But in this work, ReDWeb data set is introduced that has dense relative depth ground truths in the form of depth score maps. They randomly sample 3K pairs of pixels per image, calculate the loss, and use the top 75% of the pairs with the highest loss. They claim that discarding the low 25% focuses the learning on the hard pairs. Additionally, there are few pixels that are on the same depth. The proposed loss balances the unequal - equal relations ratio.

Chen et al. [5] also use the same loss. However, the Youtube3D data set does not have equality relations in the ground truths, so they disregard the corresponding term.

2.1.4 Other considerably related works

In this section, we list some of the works that we encountered along the way that do not fit any category mentioned earlier. Some of them are not even tackling the SIDE problem as we defined. Nonetheless, they are considerably interesting work that we want to share without going into too many details.

Non-parametric Karsch et al. [22] devise a depth sampling method to estimate the depth without the need for learning any parameter, i.e. non-parametric depth sampling. Their method can be applied to single images as well as videos. For the video case, they use additional steps incorporating temporal relations of frames to increase the quality of the outcome.

The main insight of their method is that similarity in features derived from RGB images means similar depth values. To utilize this insight for depth estimation, they collect a data set of RGB images and their corresponding depth maps and devised a three staged method for estimating a depth of a query image. First stage is to find k candidate images for the query image. Candidate images are found searching the data set based on their similarity in RGB domain. In the second stage, warping functions that warps candidate images to query image are determined using SIFT flow in RGB domain. In the last stage, depth maps of the candidate images are warped using the warping functions calculated in the second stage, and an optimization procedure is utilized to merge all the depth maps to get the final prediction.

They were able to achieve relatively good results at that time, without learning any parameters. The important part of their work is they were able to show the validity of their insight which is essentially how statistical methods work as well.

Sensor Fusion Liao et al. [51] look at the SIDE problem from a robotics perspective. Even though researchers are getting better and better results metrically, the scale ambiguity is still a problem, and the success of the performance that is measured with the available metrics is not necessarily a good indication of success, particularly when we deploy estimated depths for a robotics task.

To increase the performance and the applicability of the acquired solutions to the robotics tasks, additional sensory information is incorporated as an input to their system. To this end, they experiment with a 2D laser range finder, which is cheap and already available in lots of robots.

As an input, the network of Liao et al. takes an RGB image and a reference depth map which is created by extending the 2D laser scan values vertically to the ground plane. By doing that, a 2D planar depth observation can be turned into a dense 2D reference depth map by projecting it along the gravity direction. Reference depth map can be concatenated to the input image and together they are fed to the network. The network then does a pixel-wise classification to estimate a residual depth map, showing the difference between the reference depth map and the ground truth depth map. Moreover, they utilize a regression loss combined with the classification loss, which allows them to make use of ordinal relations between classes similar to the idea of [2].

The main contribution of this work is the usage of a laser scan, which helps alleviate the scale ambiguity problem, resulting in better solutions. To the best of our knowledge, they were the first to do "depth completion", a new branch of research where the aim is to utilize sparse ground truth depth information alongside the RGB image as input, mimicking additional sensory information.

Mal and Karaman [13] also work on depth completion. Their framework is very similar to previous works, with the single contribution of an additional input they choose to use. Instead of using 2D laser scan information, they mimic structured light based sensors and use the depth information of randomly sampled points across the image domain as an additional input. This can also be viewed as super-resolution for sensors, which is demonstrated as an application in their work.

3D Reconstruction Izadinia et al. [52] design a system that infers the 3D CAD model of a given scene, inspired by the 1963 Ph.D. thesis of Lawrence Roberts [53]. Using a database of known objects, Roberts's system was able to infer the 3D scene from a given single RGB image, even with back-facing objects or occlusions. Even though a lot of progress has been made in computer vision since then, it is still a very challenging problem to estimate 3D model of a complex, natural scene.

To tackle the problem, Izadiana et al. [52] develop a pipeline that processes the input RGB image, and estimates the 3D CAD model of the scene using an object database containing CAD models of known objects. Their pipeline consists of roughly two parallel processes.

In one process, an off the shelf object detector is used to detect objects in the given scene. Using the CAD models of the detected objects, possible alignments of them are rendered and compared to the input RGB image to infer their alignment in the scene.

In the second process, pixels are semantically labeled as floor, wall or ceiling in order to be able to estimate the layout of the scene. Essentially, a box is fitted with the assumption of indoor scenes.

Having the room layout and objects with their alignments, the 3D scene is reconstructed. As a last step, the reconstructed scene is used as an initial estimate, and it is further refined through optimization by rendering and comparing it to the input image.

Even though there are obvious fail cases, such as object detector missing the object, object database not containing an entry for the detected object, or room layout not being box shaped, they were able to show that inferring qualitatively pleasing 3D scenes from a natural, complex single image is possible.

3. INVESTIGATION OF PERCEPTUAL PRINCIPLES EXPLOITED BY SIDE METHODS

In last years, lots of efforts have been put into making neural networks capable to estimate the depth from a single image, as we have discussed in **Chapter 2.** As part of this work, we analyze the work of Laina et al. [6] in order to see how these neural networks perform beyond the metrics and to examine its qualitative performance on the given data set[1]. We investigate the networks capacity to generalize outside its data set and the biases caused by the data set; try to understand how it makes its decisions, what type of cues it uses, what types of images cause problems for the neural network. In general, it is crucial to understand how a network works for number of reasons as explained by Selvaraju et al. [54]. Firstly, determining the failure modes in the network leads to more guided and therefore efficient research paths [55,56]. Understanding how the network works builds trust between the user and the system. Additionally, where the network model performs better than humans can be used for learning purposes, namely, machine teaching [57]. Lastly, this type of investigations are important to scientifically build an understanding of the principles of operation, and the perceptual problem itself. It can help us to have a better understanding of how biological systems solve the perceptual problem.

To have a better understanding of how the depth-estimating networks work, we collect a small data set of interesting images that we think would expose the behavior of the network, create adversarial examples using fast gradient sign method [58] to see what kind of changes it would create on the RGB image, and where these changes will occur, and apply our variant on Grad-CAM [54] to understand where the network looks in the RGB image when determining the depth of a pixel. Specifically, we want to gauge whether the network could do global inference using the whole input image as this is one of the important aspects of the SIDE problem. Additionally, we want to see whether the network utilizes cues that are being used by the human visual system.

[1] Work of Laina et al. [6] was state of the art at the time of analysis.

3.1 Background on Investigation Techniques

As deep learning evolves, the trade-off between interpretability and accuracy [54] becomes more clear. For deep learning to continue to evolve and take part in everyday life, we need not only better performing models, but also models that we can understand how they work. Therefore researchers have come up with different ways of investigating how a neural network works.

One relatively recent pieces of work towards making convolutional neural networks interpretable is Grad-CAM [54], developed for creating visual explanations for classifiers. For a given input, a given convolutional layer, a given feature map A^k_{ij} where sub indices show the spatial coordinates and k represents the k$^{\text{th}}$ feature map at that layer, and a given class c and score y^c, the sensitivity of the class probability output by the network with respect to each feature map in the convolutional layer while processing the input image is calculated (this is the gradient of the feature with respect to the output class).

$$\alpha^c_k = \frac{1}{Z}\sum_i\sum_j \frac{\partial y^c}{\partial A^k_{ij}} \tag{3.1}$$

where α^c_k is a scalar value that can be interpreted as the importance of k$^{\text{th}}$ feature map for the class c. It is then used in weighted sum of activation maps which results in a rough approximation of the contribution of each pixel to the decision to choose that class.

$$L^c_{\text{Grad-CAM}} = \text{ReLU}\left(\sum_k \alpha^c_k A^k\right) \tag{3.2}$$

It is also shown that the results of Grad-CAM is very similar with the earlier work using occlusion maps to probe visual predictors which is presented in [59]. In our work, we adapt Grad-CAM for the case of regression in order to detect the parts of the RGB image that contribute to the depth of a pixel in an increasing or decreasing way.

In 2013, Szegedy et al. [60] are able to fool a trained CNN to predict wrong classes by changing the input image slightly and creating so called adversarial examples, in a way that it would not be noticeable to a human eye. This is presumably because the neural networks architecture is such that it only interpolates well in a probabilistic way according to the data it has seen and because of the vast number of parameters that create a system can be highly sensitive to the input in unpredictable ways. Subsequently, Goodfellow et al. [58] show that adversarial examples can be created very easily in one step of gradient-sign descent as follows:

$$x_{adversarial} = x_{original} + \varepsilon \operatorname{sign}\left(\nabla_{x_{original}} J(\theta, x_{original}, y)\right) \tag{3.3}$$

where $x_{adversarial}$ is the created adversarial example and $x_{original}$ is the original input image, θ is the parameters of the model, J is the loss function used to train the network, and y is the target. We adapt the work of Goodfellow et al. [58] to study regression (over depth) rather than classification to create adversarial examples that we subsequently examine to see what kind of changes are needed to change the depth of a pixel and where these changes occur, whether these changes will still be invisible to the human eye, so that we can get a feel for how the network makes use of global vs. local information in an image in general.

3.2 Approach

3.2.1 Model

For our investigative purposes, we implement and train the network of Laina et al. [6]. We train the network using the BerHu loss function with a learning rate of 0.015, and use cosine annealing as a learning rate scheduler. As a regularizer, we use weight decay with the rate of 5e-5, and the data augmentations described by Eigen et al. [43] excluding rotations. The network is trained on the NYU-Depth V2 data-set [28] with the batch size of 32. We use every tenth image in the raw data set in order to optimize the training time. We adopt the official train/test split. The input images are projected onto the depth images by using the script provided with the data set. We are able to get very close results [2] on the test set compared to what is shared by [6] as can be seen in **Table 3.1.**

3.2.2 Collecting the data set

In order to discover the biases caused by the data set on which the neural network is trained and understand what the network has learned, we collect a small number of

[2] During training, as the loss decreases, the denominator of the BerHu loss function, given in **Equation 2.18**, gets smaller and may cause numerical issues depending on the randomly sampled batch, since the denominator is calculated over a batch of inputs. Our training diverged because of this particular numerical issue, however we didn't continue training solving this problem due to time constraints and as we believe our results are sufficiently well enough for the purposes of the investigation.

Table 3.1 : Comparison of results on NYU depth V2 data set. For the rel, RMSEs and $\log_{10}$ lower is better. For the accuracies, higher is better.

NYU depth V2	rel	RMSE	RMSE(log)	log 10	$\boldsymbol{\delta}_1$	$\boldsymbol{\delta}_2$	$\boldsymbol{\delta}_1$
Eigan et al. [43]	0.215	0.907	0.285	-	0.611	0.887	0.971
Laina et al. [6]	0.127	0.573	0.195	0.055	0.811	0.953	0.988
Our implementation of [6]	0.148	0.659	0.210	0.063	0.796	0.950	0.986

images and sort them into categories. These images are considerably different than the images that the network has seen during its training. **Figure 3.1** shows some examples.

The categories in detail are as follows:

(a) Top-down In this category, we have images that are taken from an angle perpendicular to the ground, which is a different angle compared to the training images. It has images of lands and images of objects that are sitting on a surface. See **Figure 3.1**-(a) for examples.

(b) Corners This category contains images of empty rooms. These images are actually quite like the images from the data set, but they do not have any objects in them.

(c) Unattached objects These images contains planes hanging in the sky. See Fig. 1 for examples. We were curious to see how the network would perform on these images since they are unlike any image in the data-set, and it is clearly a possibility that the network can learn to rely on ground-plane related relationships.

(d) Illusions Images in this category contains illusions of different kinds where the objects size or its shape (being convex or concave) are ambiguous. We wanted to see if the network would do the same mistakes that are made by humans or not.

(e) Water surfaces In this category, we have images of ocean. Since the waves can be used to detect the texture gradient, we wanted to see if the network could use this cue and come up with a good estimate.

(f) Strong shadows These images contains strongly evident shadows of the objects. We wanted to test the network's ability to identify shadows and use them as a cue, or alternatively be fooled by them.

(g) Simple colors & gradients Here we have images that consist of only one color or contains simple color gradients. We wanted to see the response of the network to these images. For human beings, the gradients can create sense of depth. However,

(a) Top-down

(b) Corners

(c) Unattached objects

(d) Real illusions

(e) Water surfaces

(f) Strong shadows

(g) Simple colors and gradients

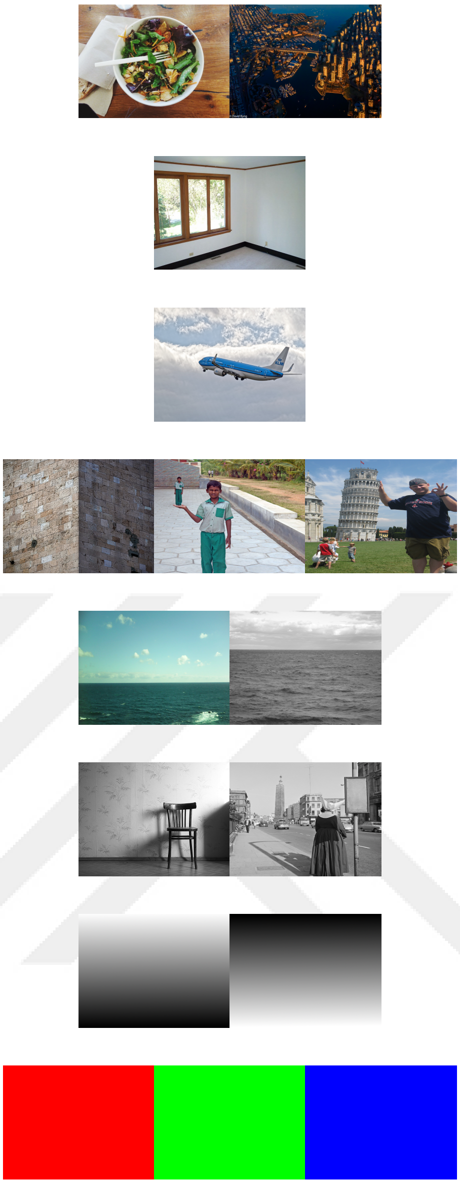

Figure 3.1 : Probe image categories.

in general acquiring depth from such an image is a poorly specified problem and the behaviour of the network may offer clues as to what it is doing.

3.2.3 Adversarial examples

We want to generate adversarial examples to see where the changes occur in the input image. Our intuition is that pixels that are affected by the adversarial attack could give us an insight about how the network works. In order to do that, we use the fast gradient sign method described in the work of Goodfellow et al. [58].

$$x_{adversarial} = x_{original} + \varepsilon \, \text{sign} \left(\nabla_{x_{original}} J(\theta, x_{original}, d_{target}) \right) \tag{3.4}$$

where xs are the RGB images, J is the loss function and d_{target} is the target depth map. For our purposes, we devise probes by modifying loss function J, target depth map d_{target} and so on. The probes are:

AdvDepth Our first probe uses the fast gradient sign method on the input RGB image to try to change the predicted depth of one pixel of the output depth map to a specified target depth. The loss function (mean squared error in the predicted depth of the target pixel) that we use for this probe ignores all the other pixels in the output depth map.

FixDepth This probe uses the fast gradient sign method on the input RGB image to make the same adversarial attack as in AdvDepth but also to make a minimal change in the output depth map apart from the target pixel. It adds an extra term to the loss function to penalize the change in the prediction for the pixels other than the targeted pixel that we are trying to change the depth of. This probe allows us to see how coupled depth values are locally in the output.

Non-local This probe applies a mask into the gradients over the RGB image. By using this probe, we could set out a probe to change the depth of a pixel while allowing it only to change some specified part of the input RGB image. This will allow us to get an idea for how non-local information can be propagated through the network.

We apply these probes in order to see what kind of changes occurs in the input RGB image. For the classification task, the adversarial attack is stopped either when the prediction of the network becomes wrong or it predicts the wrong class with a specified certainty. In our case, either we do only one step of the adversarial attack, just to see which pixels are affected by it or we do the adversarial attack until we reach the target depth value within an epsilon of 0.1 in meters or until we reach the maximum number of iterations of 100. Note also that we check the results of the adversarial attack by casting the image onto the original 8-bit integer to make sure that the changes remain.

3.2.4 Grad-CAM

In the work of Selvaraju et al. [54], the weights for each feature in the activation matrix of a layer are calculated as the average gradient received for the feature and multiplied into the activation map for that feature and summed. Subsequently, ReLU is applied in order to show only the parts of the image that have a positive effect on the specified class. For our case, we followed the method in the same way, except the ReLU part:

$$\begin{aligned} \alpha_k^{target} &= \frac{1}{Z}\sum_i\sum_j \frac{\partial(d_{prediction} \odot m)}{\partial A_{ij}^k} \\ L_{\text{Grad-CAM}}^{target} &= \sum_k \alpha_k^{target} A^k \end{aligned} \tag{3.5}$$

where $d_{prediction}$ is the predicted depth map for the given input image and the m is the mask that contains all 0s except for the targeted pixel, for which it contains 1. Instead of applying ReLU and showing only the parts that have a positive effect on the targeted pixel's depth (parts in the input image that can make the pixel further away with activation in the corresponding layer), we discarded the ReLU, and showed the parts that also have a negative effect on the target depth (i.e., parts in the input image that make the pixel closer with activation in the corresponding layer).

3.3 Experimental Results

3.3.1 Qualitative results

This section presents some of the images from our qualitative data set that exhibit interesting results. We scaled the depth images in order to increase the contrast and make the depth differences between points more evident. Therefore in this section, the same shade of blue in different depth maps does not necessarily imply the same depth value for those pixels.

In **Figure 3.2,** we feed the network with a top-down view image, and an unattached plane, which is also horizontally and vertically flipped. As it can be observed in the results, the bottom side of the depth maps tend to be closer and objects tend to be connected to an inferred ground plane. Since the NYU data set consists of indoor images, this is the common theme in the training set and clearly the network is biased towards making this decision. An open question is whether this bias is necessary given the network architecture or only contingent on the data available. Indeed, the behaviour of the trained neural network is relatively unpredictable in areas where it has not been trained on.

Even when the scene lacks detail, turning the input image upside down before feeding to the trained neural network results in poor estimations as can be seen in **Figure 3.3.**

Figure 3.2 : Top-down images results. **Left**: original images. **Right**: output of our version of Laina et al. [6] neural network trained on the NYU data set.

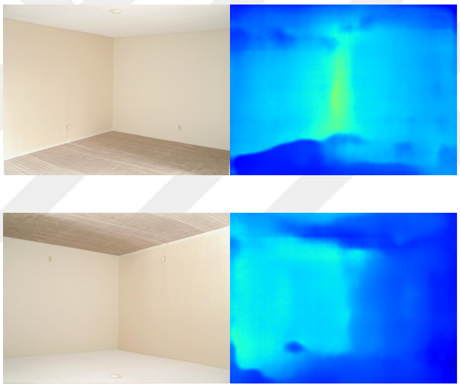

Figure 3.3 : Turning an image with a textured floor upside down. **Left**: original images. **Right**: output of the trained network.

Our hypothesis is that the textured ceiling could put the network into a situation that is too far from the training data set.

For proportion illusions, the network does show a behaviour which is somewhat consistent with human perception. In the case of Pisa tower (see **Figure 3.4** first row), it successfully estimates that its distant. Our hypothesis is that in this case the ground plane is supporting this inference, another is that the tower texture helps make it appear distant. Indeed, this is not a full illusion as the illusion is not generally a global or a stable one.

In the case of the boys (see **Figure 3.4** second row), the trained neural network gets fooled. There are only a few images in the training set with human beings; therefore the

Figure 3.4 : Proportion Illusions. Left: original images. Right: output of trained network.

network is not able to use the known size of this object as a cue even if it has learned to utilize this cue with other objects. Indeed, with humans, the illusion is stronger. Even though the network is familiar with the kind of object in question, in this case, the ground plane does not offer any clues about the distance of the object because it is contingent with the foreground object and not the horizon.

From **Figure 3.5,** it can be seen that network hallucinates things for plain colors, and it does not exhibit any consistent behaviour for color gradients. This could be due to the fact that the network has not seen any examples in this category and it shows that network is not particularly generalising rather essentially learning statistical patterns on low level features. Furthermore, possibly because the homogeneous scene parts that it has seen are always placed withing a setting where there are multiple scene components or objects, the network model hallucinates the structured world it expects to see. Similar behaviour can be observed in the human visual system as explained in the carpentered world hypothesis [61].

In the image in the **Figure 3.6,** we see a textbook example of the sensitivity of human perception to the developmental environment. Even though the estimation of the network is not well in general, it can be seen that the network makes the same kind of mistake as a human being and thinks that the shape in the first image is concave and the shape in the second image is convex. We believe that this is an extreme version of a bias induced by the data-set where the network seemingly learned how to utilize the shading and lighting due to above-lighting and the preponderance of upward-facing horizontal surfaces.

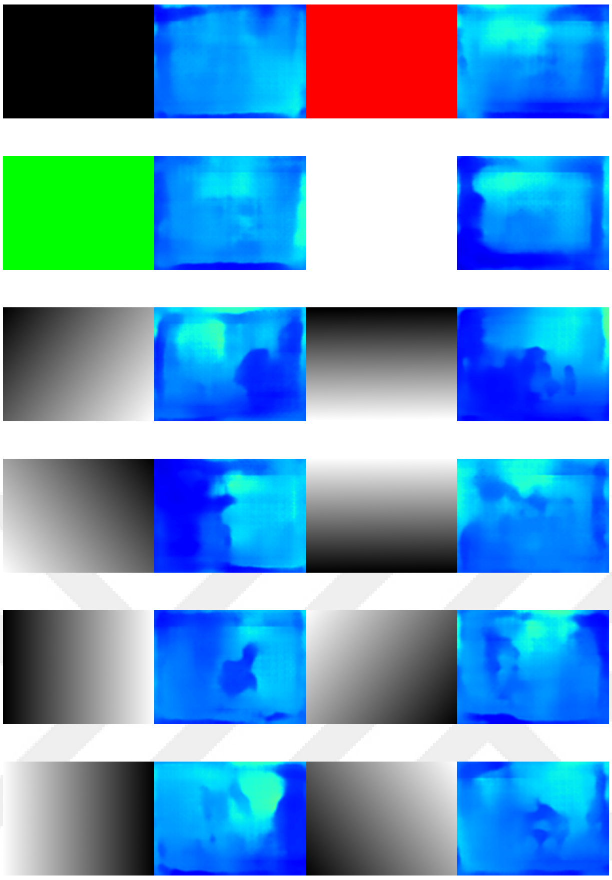

Figure 3.5 : Network's response to simple colors and gradients. **Left**: original images. **Right**: output of trained network.

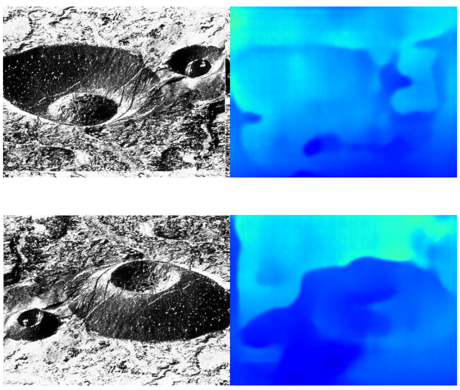

Figure 3.6 : Results of running the trained neural network on a common inverted image illusion. **Left**: original images. **Right**: output of trained network.

Figure 3.7 : Results of running the trained neural network on a water surface. Left: original images. Right: output of trained network.

Figure 3.8 : Results of running the trained neural network on images with strong shadows. **Left**: original images. **Right**: output of trained network.

The network never saw such an example as in **Figure 3.7** throughout its training. However if it was able to learn texture gradient as a general cue, we believe that it should have been able to make a relatively good estimation of the direction of depth drop-off since the waves in the ocean create a texture gradient, becoming smaller and denser as the distance increases. As the result indicates though, we believe that the network has not learned to make use of texture gradients.

Shadows in the images can definitely fool the network as it can be observed in **Figure 3.8.** Again in the data set, there are not many examples that clearly show shadows of the objects, since the images are taken from the inside with lighting conditions hiding the shadows. However, we can see that sometimes the network is able to ignore them, particularly when support relationships are clear and shadows are attached to objects.

3.3.2 Adversarial examples

We used different probes to investigate how the network responds to adversarial examples. There is a consistent color-map scale across all depth map images shown in this section to enable comparisons.

Figure 3.9 : The image that is used to demonstrate the adversarial probe. **Left**: Original RGB image. **Middle**: Output of the trained network. **Right**: Ground truth depth map.

AdvDepth AdvDepth probe targets a single pixel with adversarial attack and aims to change its depth. By using this probe with multiple steps of adversarial attacks, we create adversarial images. **Figure 3.9** contains the original input image, the original prediction of the network, and the ground truth.

Figure 3.10 contains the adversarially created RGB image, the networks depth prediction over the adversarial image, and the differences between the original RGB image and the adversarial image. The pixels surrounding the target pixel in the depth map are also influenced by the adversarial attack. Clearly we replicate the original adversarial attack [58, 60] on depth from a single image network. Also, clearly the change in the RGB image that affects the prediction of depth in a pixel also affects the neighbouring pixels (and even more distant pixels).

When we allow AdvDepth to make only one step of change in the input image (of **Figure 3.10**), as it can be observed in **Figure 3.11,** even when we do not allow the gradient-sign descent algorithm to explore the whole space of images, the target pixel depth is directly sensitive to pixels in the input image over a relatively wide area, roughly $\frac{1}{4}^{th}$ to $\frac{1}{3}^{rd}$ of the image, even in the AdvDepth case.

FixDepth Using this probe, we attempt to constrain the adversarial attack to minimize changes to depth of pixels apart from the target pixel. While making the target pixel further away, we force network to keep other pixels depth value the same by adding a term to the loss function. As we increase the weight of the term in the loss function fixing the distances of other pixels, beyond a certain weight (about 100), small changes are observed in the result, and eventually numerical issues arise; therefore we show the results of using a depth-fixing loss term with weights 1 and 100. The results are shown in **Figure 3.12.**

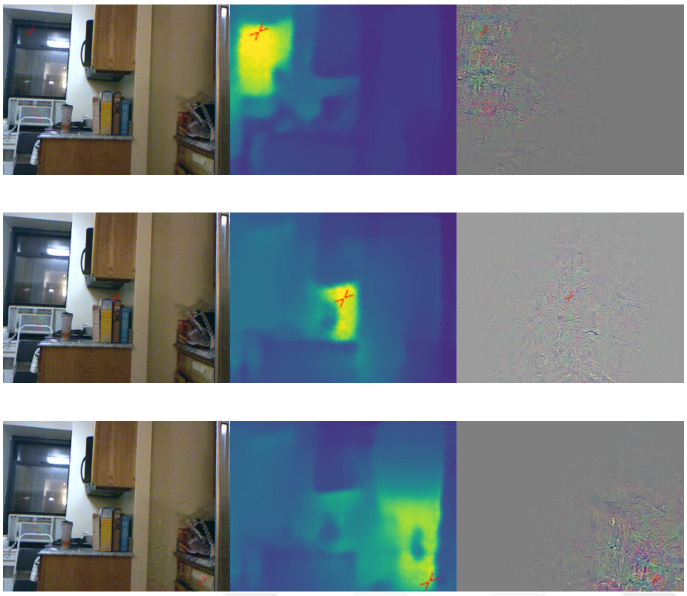

Figure 3.10 : The results of the AdvDepth adversarial probe. **Left**: RGB images created using the multi-step adversarial method; target pixels contain a red cross. **Middle**: Depth map after running these images through the neural network. **Right**: Difference map between the original RGB image and the adversarially created image. **Top**: target pixel top left. **Middle**: target pixel centre. **Bottom**: target pixel bottom right.

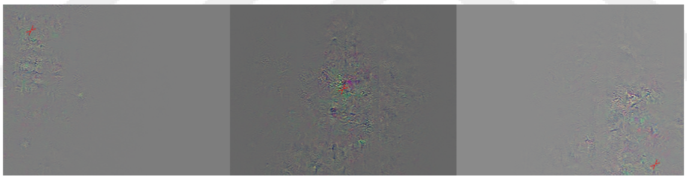

Figure 3.11 : The results of the AdvDepth adversarial probe with one-step gradient-sign descent. Only the difference images are shown: that is the difference map between the original RGB image and the adversarially created image. **Left**: target pixel top left. **Middle**: target pixel centre. **Right**: target pixel bottom right.

As a result of this attack, pixels surrounding the target pixel become more distant, though less extremely. We might surmise as a result that the non-locality in the network is somewhat fundamental since it seems difficult to avoid even with a focused adversarial attack. Even when we assert 100 times more weight to the probe FixDepth, the depth estimation of the immediately surrounding pixels get affected severely and further depth structures can also be affected.

Non-local Lastly, we allow only an 80 pixel by 80 pixel area in the RGB image to be changed by the adversarial attack. The results that can be seen in **Figure 3.13** show that the depth estimation of a pixel can be affected by non-neighbouring pixels,

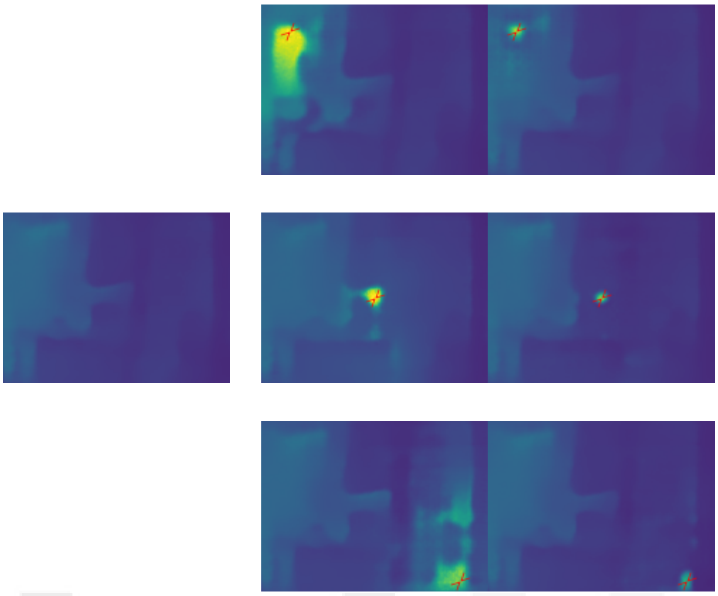

Figure 3.12 : The results of the FixDepth adversarial probe. **Left**: The depth map before running the adversarial attack. **Middle**: Depth maps after running the adversarial attack with a depth-fixing loss term weight of 1. The target pixel for each row is indicated with a red cross. **Right**: With a depth-fixing loss term of 100. Beyond 100, only small changes are observed in the output.

which shows that global cues are utilized by the network to some extent. However, for distant pixels, there are noticeable changes (glitches) in the RGB images which means that bigger changes are needed to change the output. Additionally, smaller changes in depth are achievable using distant masks; indeed for very long-distance relationships where the target pixel and the attacked pixels are placed diagonally (see **Figure 3.14**), the depth can not be changed in any noticeable way, and the pixels in the RGB image that are changed are the ones at the edge of the mask closest to the target pixel. This provides evidence for the assertion that the network is mostly relying on local information. This might be a result of how convolutional neural networks operate as it is shown that effective receptive fields of CNNs tend to be too small [62].

We also note that the masked area has a stronger relationship when it is offset vertically but not horizontally from the target pixel. We claim that this is because the network is highly dependent on the structure of the scene coming from the orientation of the camera (e.g. see the results in the previous section for unattached objects, for example).

3.3.3 Grad-CAM

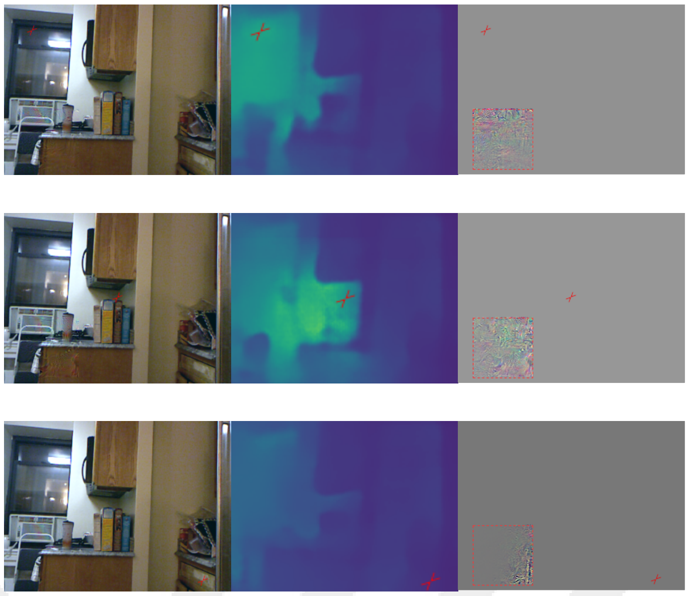

Figure 3.13 : The results of the Non-local adversarial probe. Left: RGB images created using the multi-step adversarial method using an 80x80 mask in the bottom left corner on gradients; target pixels contain a red cross and attacked pixels are surrounded with dashed line. **Middle**: Depth map after running these images through the neural network. **Right**: Difference map between the original RGB image and the adversarially created image. **Top**: target pixel top left. **Middle**: target pixel middle. **Bottom**: target pixel bottom right.

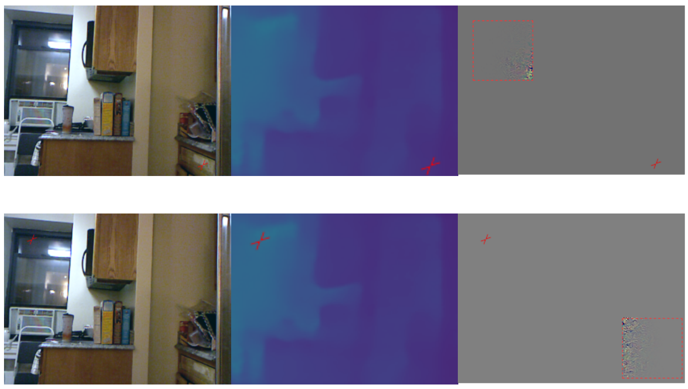

Figure 3.14 : The results of the Nonlocal adversarial probe for very long-distance relations. **Left**: RGB images created using the multi-step adversarial method using an 80x80 mask on gradients; target pixels contain a red cross and attacked pixels are surrounded with dashed line. **Middle**: Depth map after running these images through the neural network. **Right**: Difference map between the original RGB image and the adversarially created image. **Top**: target pixel bottom right, mask top left corner. **Bottom**: target pixel top left, mask bottom right corner.

Figure 3.15 : Grad-CAM variant for depth regression applied to our trained network. **Left**: target pixel top left. **Middle**: target pixel middle. **Right**: target pixel bottom right. Green parts are the parts contributing to the target pixels depth and blue parts are decreasing the depth of the target pixel. The targeted pixel is marked with a red cross in the images.

We apply our variant on Grad-CAM, as described in Section 3.2.4 to the last convolutional layer of the encoder part of the network, as it captures the best semantic level information and preserves spatial information [54]. The results can be seen in **Figure 3.15.**

The extent to which the color coding here is appropriate (green corresponding to activations that are responsible for depth, and blue corresponding to activations that are responsible for closeness) is the extent to which we can interpret activation as cues. If we consider that lack of an activation may be a cue, then the depth-closeness coding becomes less appropriate. However, this can reveal something about how the network is coding depth at the semantic convolutional layers.

Clearly, when it comes to semantic information held at the last layer of the convolutional neural network, there are some areas that are responsible for increased depth of all pixels (top right of the images in **Figure 3.15,** for example, which are coded green in all images). We may surmise that there is some cue in that flat area that is statistically associated with distance. Moreover, areas around the target pixel are more likely to contribute to closeness or no net change. We may conclude that a variety of cues are available at the final layer and their net effect is not strong in one direction or another. However, it is not possible to make more meaningful deductions looking at the results. Overall, the results of Grad-CAM experiments are inconclusive.

3.4 Other Investigative Works

Concurrently with our work [17], Dijk and Croon [18] investigate the networks of Godard et al. [38], Zhou et al. [39], Kuznietsov et al. [63], and Wang et al. [40] in an autonomous driving scenario. They make their analysis using probe images designed by hand to reveal high level behaviours of the networks, instead of examining feature map visualizations which can only give insights about low level understanding of the networks.

First, they focus on revealing how objects' positions in the image and their apparent sizes affect the depth estimation. To this end, they crop images of cars and insert them into other images in vertically different positions keeping the apparent size same, same position with different apparent sizes, and vertically different positions with an appropriate size change. Networks' depth estimation for the inserted car is reduced to a scalar by averaging the estimated depth on a flat surface of front or rear end of the car, depending on whether the inserted car faces the camera or not. They are able to show that the networks mostly use vertical position cue when estimating the depth, disregarding the apparent size cue. This finding is similar to ours: connection to the estimated ground plane heavily influences the estimated depth. Moreover, this behaviour is shown to be consistent in all the investigated networks which have different architectures and training methods.

They also experiment on Godard et al.'s network [38] to see how different camera setups affect the estimated depth of objects. Cropping images with different vertical offsets, they mimic the pitch change of the camera and they show that the pitch changes affect the estimated depth values, which further strengthen the claim of vertical position being the most influential cue.

As another experiment, they investigate whether the inserted object's properties such as color, shape, and so on, are important. They show that the network essentially tries to detect a connection point to the ground to estimate the depth of objects, in fact when such a connection is not detected, the network ignores the object altogether. Furthermore, even when the object's inside is emptied, the network still fills in the interior in the estimated depth, showing that edges are guiding the depth estimation process. While grayscale objects or falsely colored objects do not degrade the performance too much, objects that are colored completely to a single color degrades the performance showing that the important thing for the network is to be able to distinguish parts of the objects, not their actual color or texture.

Hu et al. [19] also attempt to find an answer to the questions of how neural nets estimate the depth from a single image; which cues do they use? Instead of visualizing intermediate feature maps, they also work on the input image. Unlike our work [17] and Dijk and Croon's work [18], their goal is to see which parts of the input image has a bigger impact on the estimated depth. They formulate this as an optimization problem where a network learns to mask out irrelevant pixels from the input image that results in a minimal change in the estimated depth map from the entire image.

To clarify, first they train a network N for the SIDE task on the NYU and KITTI data sets with common architectures [3] seen in the literature such as ResNet50 in Laina et al. [6]. Afterwards, while the weights of the network N is fixed, a mask predicting network G is trained to optimize the following problem:

$$\min_{G} \; l_{\text{dif}} \left(N(I), \, N(I \otimes G(I)) \right) + \lambda \frac{1}{n} \|G(I)\|_1, \tag{3.6}$$

where $\|G(I)\|_1$ pushes the network to mask out more pixels (otherwise a trivial solution to the optimization problem would be just outputting all 1s), λ controls its weight, n is the number of pixels, and l_{dif} is the function calculating the difference between estimated depth maps that consists of a depth loss, a depth gradient loss, and a depth normal loss. Here, $\otimes$ refers to an elementwise multiplication.

When the predicted masks are examined, they are able to show that the edges are utilized by the networks to estimate the depth. However, not all image edges are present in the masks, which claimed to be the indication of the network's ability to distinguish informative edges. Another important finding is that the interiors of some of the objects look important to the network. This looks especially true for smaller objects. This finding may seem to be contradicting with the results of Dijk and Croon [18] at first. However, the latter work only experimented with cars, and it is possible that the networks that are examined may have overfitted to cars. One last important finding is the importance of the vanishing points for outdoor images. Network models seem to be paying great attention to those areas of images. They conjecture that this might very well be the result of errors in those pixels being larger due to their depth, however, they may also be more important for understanding the geometry of the scene.

[3] All experimented architectures result in similar findings showing that it is not dependent on any particular architecture, therefore we are not going to dive into details of them.

3.5 Discussion

Our work as well as other investigative works show that models learn to exploit regularities in their training set. Simple cues such as ground plane connection, vanishing points, and object edges are utilized to estimate depth. However it is questionable how well these cues generalize outside of their training domain. While models seemingly use some of the pictorial cues that human visual system uses such as shading and lighting, there are also cues such as texture gradient that models do not utilize. We believe it is important to systematically investigate how well pictorial cues are utilized by the models and how the learning can be structured in a way that models learn these type of generalizable cues. In addition to that, investigations show that data sets cause biases and affect the learned cues. In order to be able to have a model that can work in the wild, it is crucial to have diverse data sets. In that sense, these investigative works justify in the wild data sets that we discuss in **Section 2.1.3.**

4. SINGLE IMAGE RELATIVE DEPTH ESTIMATION AS A RANKING PROBLEM

Based on our literature review and analysis of existing solutions, it seems that limits are being reached in the SIDE problem as it is, since we observe only marginal improvements in the metrics over limited data sets such as the NYU or KITTI in recent works. However, it is not straightforward to deploy those systems in real world scenarios where images are coming from diverse natural scenes since their performances are unpredictable in such cases. Therefore, we believe that it is important to design systems that have predictable performances in the wild, that is why we focus our attention to solve the SIDE problem in the wild. Particularly, we believe that the most promising research direction is the relative depth estimation, where networks learn to predict ordinal relations of pixels. It allows us to collect diverse data sets with ease and alleviates the scale ambiguity problem.

In this chapter, we discuss our work [20] on the relative-SIDE problem. Our main insight is that the previous works use a simple ranking loss which can be replaced with a better loss as suggested in learning to rank literature. In our work, we examine the usage of one such option. Specifically, we formulate the relative depth estimation as a ranking problem, and apply a theoretically better ranking loss compared to the losses that are used so far. Rest of this chapter is organized as follows: **Section 4.1** gives a short summary on the literature about learning to rank, **Section 4.2** describes our proposed approach to the relative-SIDE problem, and **Section 4.4** presents our results.

4.1 Learning to Rank

The ranking literature is mainly focused on the information retrieval problem. In information retrieval, the aim is to find a permutation[1] of a set of documents $D = \{D_1, .., D_K\}$ in which the documents are ranked in the descending order of their relevance to a given query Q. To achieve this, one can consider each possible

[1]Permutation and ranking will be used interchangeably throughout the chapter and both refers an ordered set of documents.

permutations of the D and learn a ranking function to choose the correct ranking for the given Q_l. However, this formulation quickly becomes intractable as the number of documents to rank increases since the number of possible permutations to consider increases exponentially. Instead, one can learn a ranking function to assign a scalar score value $\hat{S}^{Q_l}_{D_k}$ to each document D_k for a given query Q_l, which shows the relevance of each document so that correct ranking can be achieved by sorting the set of scores $\hat{S}^{Q_l} = \{\hat{S}^{Q_l}_{D_1}, .., \hat{S}^{Q_l}_{D_k}\}$ in descending order.

Methods that tackle the information retrieval problem as we describe above, mainly falls into three categories in terms of the number of documents considered in their loss functions, namely pointwise, pairwise and listwise methods. Pointwise methods assume that the ground truth score information $S^{Q_l}_{D_k}$ is available for each document-query pair $\{D_k, Q_l\}$ and tries to estimate the ground truth value. However, pointwise methods have two major drawbacks. First of all, the assumption of existence of ground truth scores is too strong as often times only the ground truth permutation is available. Although one can map the permutation to scores, there is not a systematic approach for finding the best mapping given the problem. Additionally, estimating the ground truth score values is a strictly harder problem compared to estimating only the correct permutation as an infinite number of scores can result in the same permutation. Instead of finding a set of scores that result in the correct permutation, pointwise methods try to estimate the one given as a ground truth, even though both are equally correct.

As one can expect, pairwise methods consider pairs of documents in their loss functions. A ground truth permutation Π^{Q_l} provides us with all pairwise ordinal relations $R^{Q_l}_{D_i,D_j}$. Instead of trying to estimate a specific score for each document-query pair, pairwise methods try to estimate scores $\hat{S}^{Q_l}_{D_i}$ and $\hat{S}^{Q_l}_{D_j}$ so that $\hat{S}^{Q_l}_{D_i} > \hat{S}^{Q_l}_{D_j}$ if document D_i is ranked higher than document D_j, $\hat{S}^{Q_l}_{D_i} < \hat{S}^{Q_l}_{D_j}$ if document D_j is ranked higher than document D_i, and $\hat{S}^{Q_l}_{D_i} = \hat{S}^{Q_l}_{D_j}$ if documents D_i and D_j are ranked the same for a given query Q_l. Note that the actual value of the score is irrelevant as long as it follows the ordinal relation. A loss function that can be used in pairwise methods is given by

$$\text{loss}(\hat{S}^{Q_l}_{D_i}, \hat{S}^{Q_l}_{D_j}, R^{Q_l}_{D_i,D_j}) = \begin{cases} \log\left(1+\exp\left(-\hat{S}^{Q_l}_{D_i} + \hat{S}^{Q_l}_{D_j}\right)\right), & R^{Q_l}_{D_i,D_j} = +1 \\ \log\left(1+\exp\left(\hat{S}^{Q_l}_{D_i} - \hat{S}^{Q_l}_{D_j}\right)\right), & R^{Q_l}_{D_i,D_j} = -1 \\ \left(\hat{S}^{Q_l}_{D_i} - \hat{S}^{Q_l}_{D_j}\right)^2, & R^{Q_l}_{D_i,D_j} = 0 \end{cases}, \quad (4.1)$$

where $R^{Q_l}_{D_i,D_j}$ is $+1$ when the document D_i is ranked higher than the document D_j in the ground truth permutation Π^{Q_l}, $R^{Q_l}_{D_i,D_j}$ is -1 when the document D_j is ranked higher than the document D_i in the ground truth permutation Π^{Q_l}, and $R^{Q_l}_{D_i,D_j}$ is 0 when the document D_i and the document D_j are ranked same in the ground truth permutation Π^{Q_l}. This loss function is the one that has been used in relative-SIDE problem so far [1,4,5].

While the pairwise approach alleviates drawbacks of the pointwise approach, Cao et al. [64] argue that the assumption of i.i.d. document pairs is detrimental to the learning algorithms, as it ignores the complete ranking of documents. Instead, they develop ListNet, which falls into what we refer to as listwise approach that uses a complete ranking of documents in the loss function. To the best of our knowledge, they are the first group to do so.

To be able to calculate a loss between the predicted permutation and the ground truth permutation, they express the estimated and ground truth scores as probability distributions, which allows them to utilize any metric that measures the distance between two probability distributions. To this end, they define a top one probability $P^{Q_l}(D_i)$ for each document as follows:

$$P(D_i|S^{Q_l}) = \frac{\phi\left(S^{Q_l}_{D_i}\right)}{\sum_{k=1}^{K}\phi\left(S^{Q_l}_{D_k}\right)}, \tag{4.2}$$

where $\phi()$ is an increasing and strictly positive function. It is easy to see that top one probability forms a probability distribution over a set of documents, and $P(D_i|S^{Q_l}) > P(D_j|S^{Q_l})$ for documents D_i and D_j with $S^{Q_l}_{D_i} > S^{Q_l}_{D_j}$. Using top one probability, one can try to minimize any loss that measures the distance between two probability distributions, such as cross entropy that is defined by

$$\text{loss}(S^{Q_l},\hat{S}^{Q_l}) = -\sum_{i=1}^{K} P(D_i|S^{Q_l})\log(P(D_i|\hat{S}^{Q_l})). \tag{4.3}$$

Note that one can use the exp function as $\phi()$ in **Equation 4.2**, which can be implemented in neural network models easily by applying a softmax layer to the output.

With the listwise loss function given in **Equation 4.3,** the work by [64] is able to outperform pairwise methods, which empirically strengthens the motivation for the listwise approach. Additionally, it is empirically shown that the listwise loss is correlated with a commonly used ranking measure normalized discounted cumulative gain (NDCG) [65]. It was important to show that since ranking measures are not directly optimized as they are not differentiable. On the other hand, the need for ground truth scores is still a drawback.

Qin et al. [66] propose another listwise loss, called RankCosine. They argue that a good loss function for ranking should be insensitive to the number of documents for a given query. That is, the loss should have an upper bound in order to prevent a learning bias to some queries with more documents or with bigger losses. Additionally, it should be able to put more importance on the documents that are ranked higher since naturally we care more about mistakes that are done in the higher rankings compared to mistakes done in lower rankings.

To satisfy those properties, a listwise loss that uses cosine similarity measure on estimated and ground truth score vectors is devised, as follows:

$$\text{loss}(S^{Q_l}, \hat{S}^{Q_l}) = \frac{1}{2}\left(1 - \frac{{S^{Q_l}}^{\mathrm{T}}\hat{S}^{Q_l}}{\|S^{Q_l}\|\|\hat{S}^{Q_l}\|}\right). \tag{4.4}$$

It can be seen easily that the RankCosine loss is insensitive to the number documents for the given query, and its upper bound is 1, which prevents the learning from being overly biased towards queries with more documents or with bigger losses. Additionally, it follows the ground truth scores, if available. Otherwise one can map the ground truth ranking to scores in an appropriate way, e.g. e^i for the document ranked i^{th}, to put more emphasis on items that are ranked higher.

Xia et al. [67] analyze the existing solutions RankCosine and ListNet theoretically in terms of their consistency, soundness, mathematical properties such as continuity, differentiability, and convexity, and computational efficiency. In their analysis, consistency means whether optimizing our listwise loss will result in optimal ranking function in terms of minimizing a 0-1 loss that mirrors the task where it takes 0 for correct permutation and 1 otherwise. A sound loss means that it can assign larger penalties for worse solutions. Based on their analysis, they propose a new loss that is theoretically and empirically superior to previous losses.

The proposed loss is called ListMLE, which minimizes the negative log likelihood of the ground truth ranking, given the estimated scores. The ListMLE loss is defined by

$$\operatorname{loss}(\Pi^{Q_l}, \hat{S}^{Q_l}) = -\log\left(\prod_{k=1}^{K} \frac{\exp\left(\hat{S}^{Q_l}_{D_{\Pi(k)}}\right)}{\sum_{j=k}^{K} \exp\left(\hat{S}^{Q_l}_{D_{\Pi(j)}}\right)}\right), \tag{4.5}$$

where $\Pi(i)$ is the index of the i^{th} item in the ground truth ranking. It defines a Plackett-Luce model probability distribution over permutations given the scores. A more intuitive explanation is that, similar to the drawing items without replacement, probability of a permutation can be expressed as multiplication of probabilities of choosing a particular item amongst remaining items, which is available in our case in the form of top one probability calculated from scores for the remainder of items. It is similar to ListNet approach defined in **Equation 4.3** in terms of usage of top one probability. However in ListMLE, top one probability is calculated for the remainder of items and instead of trying to match the ground truth probability distribution as in ListNet, ListMLE approach aims to estimate scores that maximizes the probability of ground truth permutation.

It is shown that the ListMLE loss is consistent, sound, continuous, differentiable, convex, and can be calculated in linear time. Additionally, it does not need the ground truth scores for items. The experiments in [67] also show that the ListMLE loss empirically performs better compared to losses in previous works we have discussed so far.

Chen et al. [68] analyze the ListMLE loss in terms of its relation to the commonly used ranking measures NDCG and mean average precision (MAP) [69]. They are able to prove that the ListMLE loss is the upper bound of the ranking measure losses $(1-NDCG)$ and $(1-MAP)$, meaning that minimizing this loss will maximize the ranking measures. Furthermore, their theoretical analysis results in a weighting for the ListMLE loss as follows

$$\operatorname{loss}(\Pi^{Q_l}, S^{Q_l}, \hat{S}^{Q_l}) = \sum_{k=1}^{K-1} \underbrace{\mathrm{G}(S^{Q_l}_{D_{\Pi^{Q_l}(k)}})\mathrm{D}(k)}_{\textit{weighting}} \overbrace{\left(-\hat{S}^{Q_l}_{D_{\Pi^{Q_l}(k)}} + \ln\left(\sum_{j=k}^{K} \exp \hat{S}^{Q_l}_{D_{\Pi^{Q_l}(j)}}\right)\right)}^{\textit{ListMLE}} \tag{4.6}$$

where $G()$ is an increasing function named as the gain function, and $D()$ is a decreasing function named as the discount function. Gain is the function of score, as the score increases the gain increases. On the other hand discount is the function of rank, as the rank increases in number the discount decreases. It is stated that common choices for $G()$ and $D()$ are $G(z) = 2^z - 1; D(z) = \frac{1}{\log_2(1+z)}$. The weigthing in this loss provides a tighter bound for ranking measures, which theoretically and empirically improved the performance of the ListMLE loss.

4.2 Proposed Method: Relative-SIDE as a Ranking Problem

In this section, for the first time, we formulate the relative-SIDE problem as a ranking problem in a formal way. We have a set of images $\mathbf{I} = \{I_1, .., I_L\}$. For each image I_l we have, a set of pixels $\mathbf{P^{I_l}} = \{P_1^{I_l}, .., P_N^{I_l}\}$ where $N = I_{lH} \times I_{lW}$, I_{lH} and I_{lW} are the height and width of the image I_l respectively[2], and a ground truth ranking $\Pi^{I_l}(i)$ which returns the index of the i^{th} item in the ranking. $\Pi^{I_l}(i)$ can be used to obtain the ground truth ordering as $(P^{I_l}_{\Pi^{I_l}(1)}, .., P^{I_l}_{\Pi^{I_l}(N)})$. The aim is to find a ranking function $f(I_l; \theta) = \mathbf{\hat{S}^{I_l}}$, where $\mathbf{\hat{S}^{I_l}} = \{\hat{S}_{P_1^{I_l}}, .., \hat{S}_{P_N^{I_l}}\}$, are the scores assigned to each pixel, and θ are the parameters of the ranking function f. The optimization problem in this case is defined by

$$\begin{aligned} \theta^* &= \operatorname*{argmin}_{\theta} \sum_{l=1}^{L} \phi(f(I_l; \theta), \Pi^{I_l}) \\ \phi(\mathbf{\hat{S}^{I_l}}, \Pi^{I_l}) &= \begin{cases} 0, & \text{if } \mathrm{sort}(\mathbf{\hat{S}^{I_l}}) = (\hat{S}_{P^{I_l}_{\Pi^{I_l}(1)}}, .., \hat{S}_{P^{I_l}_{\Pi^{I_l}(N)}}) \\ 1, & \text{otherwise} \end{cases} \end{aligned} \tag{4.7}$$

where ϕ is a 0-1 loss. and the $\mathrm{sort}(\cdot)$ is a sorting operator that sorts the given set in a descending order.

In practice, we have a subset of $\mathbf{I}$, divided into training, validation and test sets $\mathbf{I^{trn}}$, $\mathbf{I^{val}}$, and $\mathbf{I^{tst}}$, respectively. The ranking function f is modelled as a neural network and its parameters θ are chosen between set of parameters Θ that is learned through a stochastic gradient descent (SGD) optimization algorithm on a training set $\mathbf{I^{trn}}$ using a differentiable loss function ϕ^{loss}. Optimal parameters θ^* for the network is chosen based on the validation set $\mathbf{I^{val}}$ statistics, and its performance $\mathbf{PM} = \{PM_1, .., PM_M\}$ on the test set $\mathbf{I^{tst}}$ is measured by performance measures $\Phi^{PM} = \{\phi_1^{PM}, .., \phi_M^{PM}\}$ as an approximation of its performance on a set of unseen images. The whole process is

[2] In practice, N is the same for all images as input resolution to the model is fixed.

summarized by

$$
\begin{aligned}
\Theta &:= \mathrm{SGD}(\ \mathbf{I^{trn}},\ \phi^{loss},\ f,\ \Pi^{I^{trn}})\\
\theta^{*} &= \underset{\theta\ \in\ \Theta}{\mathrm{argmin}}\ \frac{1}{L_{val}}\sum_{l=1}^{L_{val}} \phi^{loss}(\ f(I_l^{val};\theta),\ \Pi^{I_l^{val}}) \\
PM_m &= \frac{1}{L_{tst}}\sum_{l=1}^{L_{tst}} \phi_m^{PM}(\ f(I_l^{tst};\theta),\ \Pi^{I_l^{tst}})
\end{aligned} \tag{4.8}
$$

To demonstrate the effectiveness of our approach, we adapt the network model of Xian et al. [1] as a baseline which is the state of the art work on RedWeb data set. It is an encoder-decoder network with an ImageNet pretrained ResNet-50 [44] backbone. The last pooling layer, the fully connected layer and the softmax layer are removed from the ResNet-50, and it is used as a fully convolutional encoder that produces multi scale feature maps which are the outputs of each residual block. The decoder part fuses information coming from the main stream and from the encoder via skip connections and increases the resolution of the output by applying upsampling operations. **Figure 4.1** illustrates the network architecture. For simplicity, we are going to refer the model as **EncDecResNet**.

Figure 4.2 depicts the inner structures of the original blocks. ResNet-50 blocks are directly taken from the original work and initialized with pretrained weights. Following the original work, we use ten times smaller learning rates for ResNet-50 blocks to be able to make better use of pretrained weights.

We introduce the ListMLE loss to the SIDE framework, in order to train our network. Following our formal formulation of the relative-SIDE problem as a ranking problem in **Equation 4.8,** we define the ListMLE loss for the relative-SIDE problem with the aforementioned notation, as follows:

$$
\phi_d(\Pi^{I_l},\hat{\mathbf{S}}^{\mathbf{I_l}},\mathbf{S^{I_l}}) = \sum_{n=1}^{N-1} W(\mathbf{S^{I_l}},n)\left(-\hat{S}^{I_l}_{P_{\Pi^{I_l}(n)}} + \ln\left(\sum_{j=n}^{N}\exp \hat{S}^{I_l}_{P_{\Pi^{Q_l}(j)}}\right)\right) \tag{4.9}
$$

$$
W(\mathbf{S^{I_l}},n) = \mathbb{I}(\mathbf{S^{I_l}})\left(\mathrm{G}(S^{I_l}_{P_{\Pi^{I_l}(n)}})\mathrm{D}(n)\right) + 1 - \mathbb{I}(\mathbf{S^{I_l}}) \tag{4.10}
$$

$$
\mathbb{I}(\mathbf{S^{I_l}}) = \begin{cases} 1, & \textit{if } \mathbf{S^{I_l}} \textit{ is available} \\ 0, & \textit{otherwise} \end{cases} \tag{4.11}
$$

$$
\mathrm{G}(s) = 2^s - 1 \tag{4.12}
$$

$$
\mathrm{D}(n) = \frac{1}{\log(n+1)} \tag{4.13}
$$

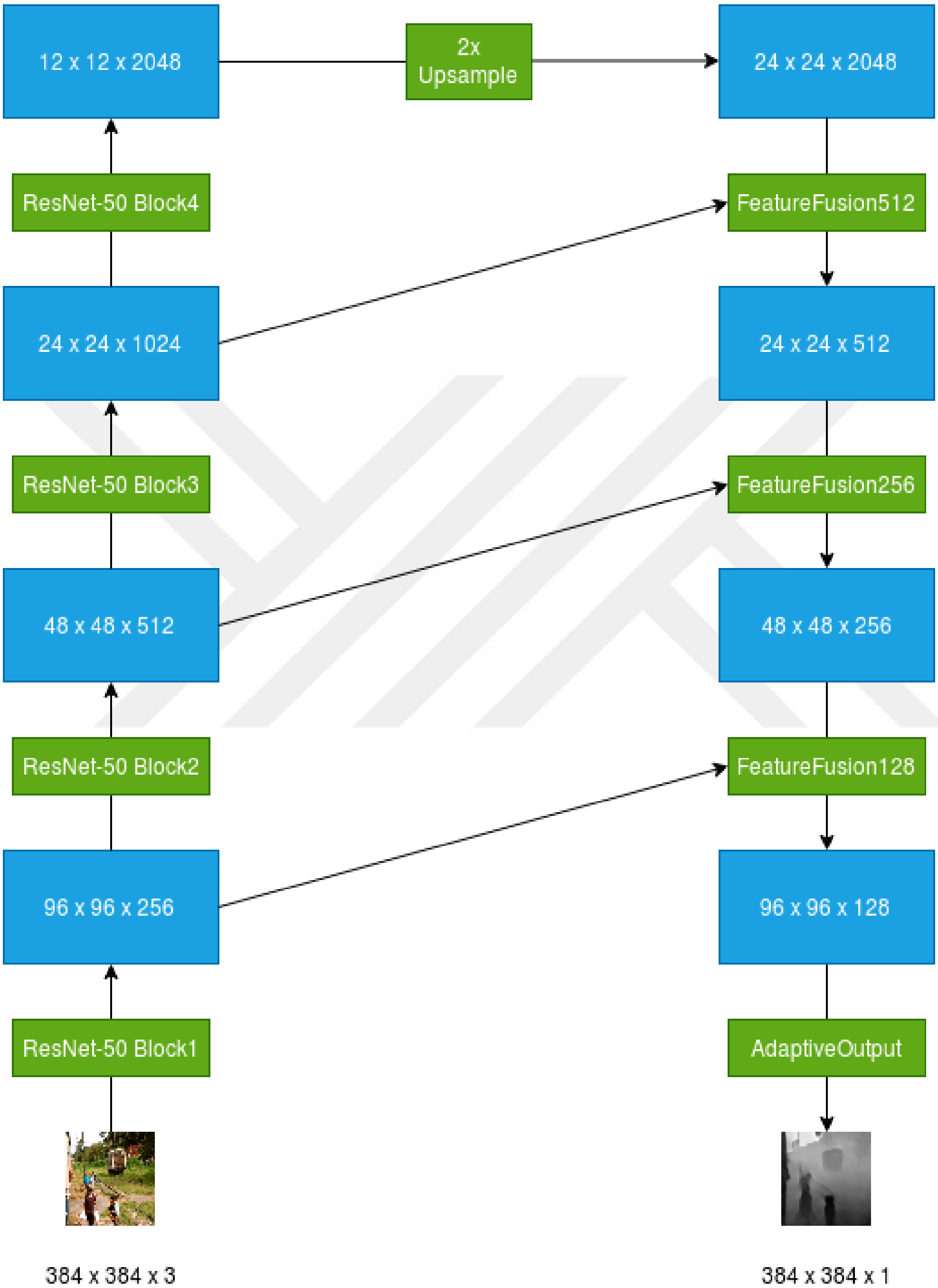

Figure 4.1 : Illustration of the network architecture. Blue blocks represent the feature maps, green blocks represent layers.

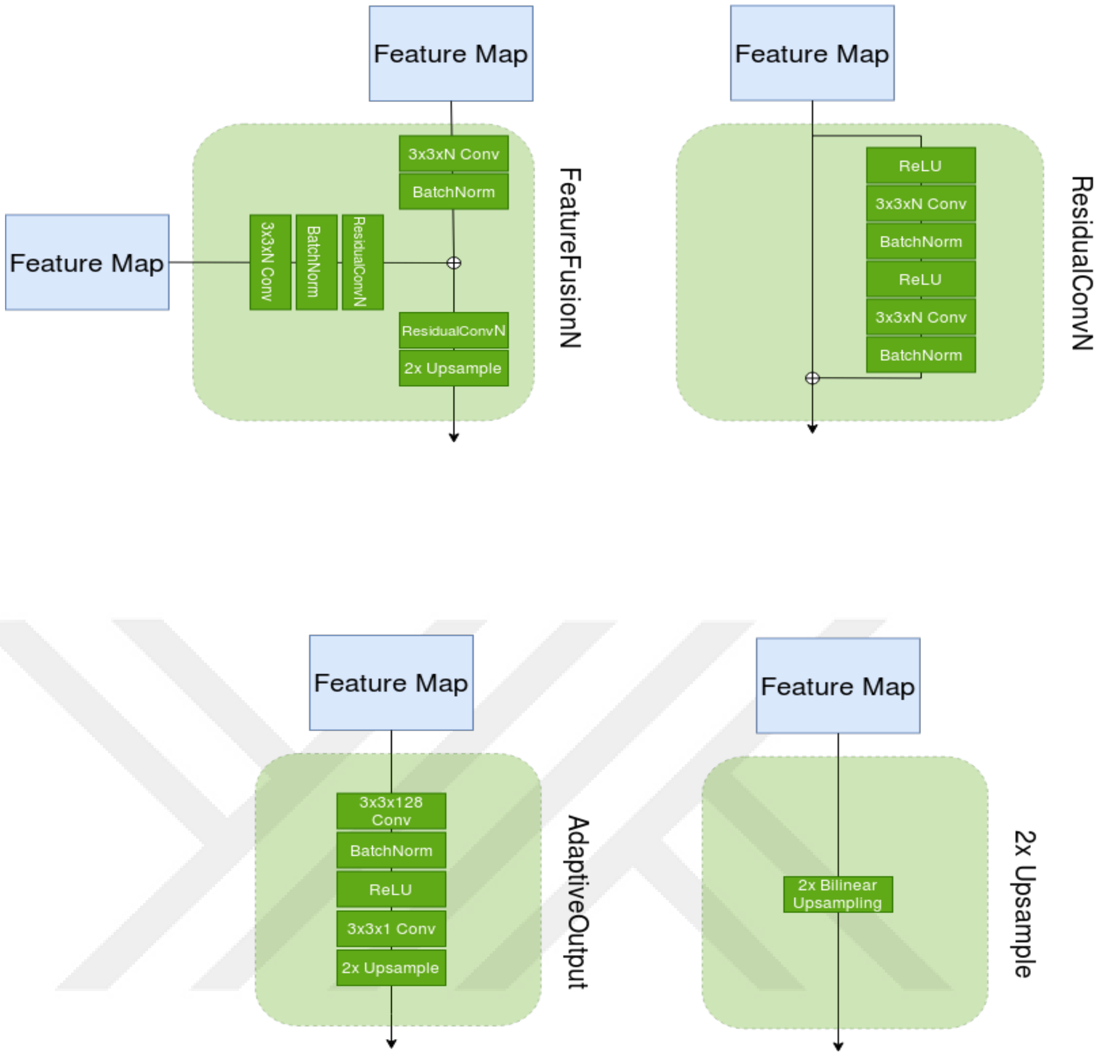

Figure 4.2 : Illustrations of the green blocks from EncDecResNet in Figure 4.1.

Depending on the availability of the ground truth scores $\mathbf{S}^{\mathbf{I}_1}$, our loss formulated in **Equation 4.9,** switches between a vanilla ListMLE loss given in **Equation 4.5** or a weighted ListMLE (W-ListMLE) loss given in **Equation 4.6.** Switching mechanism is controlled by the weighting function in **Equation 4.10**, which applies the weighting as defined in **Equation 4.12** and **Equation 4.13** if the ground truth scores $\mathbf{S}^{\mathbf{I}_1}$ are available.

4.3 Performance Measures

To demonstrate the performance of the methods, the only measure that is used in previous works is the weighted human disagreement rate (WHDR). Formulation of the WHDR metric is given by

$$\phi_{WHDR}(\hat{S}^{I_l}, \Pi^{I_l}) = \frac{\sum_{i,j\in\Omega} \mathbb{1}\left(\left(\hat{S}_{P^{I_l}_{\Pi(i)}} > \hat{S}_{P^{I_l}_{\Pi(j)}} \wedge i > j\right) \vee \left(\hat{S}_{P^{I_l}_{\Pi(i)}} < \hat{S}_{P^{I_l}_{\Pi(j)}} \wedge i < j\right)\right)}{\#\Omega} \tag{4.14}$$

where $\Omega = \binom{S}{2}$, $S = \{1, 2, .., N\}$ and N is the number of pixels. Essentially, it shows the percentage of incorrectly ordered pixel pairs. Since the DIW data set has only one pair of ground truth ordinal relations between two pixels, and all the previous works have tested their models on the DIW data set regardless of the training data set, the usage of WHDR has been the only option as a performance measure.

We propose to use an additional performance measure in the relative SIDE problem: the mean average precision (MAP). Since new data sets such as RedWeb have dense ground truth ordinal relation annotations, we believe that commonly used MAP is a good candidate for an additional performance measure as it puts more emphasis on top ranked pixels. This is a plausible property since we naturally care more about top ranked pixels and it is similar to the idea of working in the log space. First, we give the definition of MAP and then we give a motivating example for proposed performance measure.

Definition Since average precision (AP) is defined for binary labeled items, a common practice is to label items up to the i^{th} rank as positive. Let $\Pi_i^{I_l}$ represent such a ground truth ranking, then the MAP is calculated as follows:

$$\begin{aligned} \phi_{MAP} &= \frac{1}{N-1} \sum_{n=1}^{N-1} \text{AP} \\ \text{AP} &= \sum_{i=1}^{N} \text{Precision}(i)(\text{Recall}(i) - \text{Recall}(i-1)). \end{aligned} \tag{4.15}$$

Motivating Example Let's assume we have a set of documents $D = \{a, b, c, d\}$ and their ground truth permutation for a query as (a, b, c, d). We have two ranking functions f_1 and f_2 which rank the set of documents D as $\pi_1 = (a, b, d, c)$ and $\pi_2 = (b, a, c, d)$, respectively. There are 6 pairwise relations for a 4-document set, and both functions have WHDR of $1/6$ as both of them get one of the pairwise relation wrong. On the other hand, as **Table 4.1** shows f_1 that ranks top-ranked documents correctly has a better MAP score compared to f_2 that ranks top-ranked documents wrong.

Table 4.1 : A motivating example for MAP measure. + signs represent relevant documents and - signs represent irrelevant documents.

relevant documents	f_1				f_2			
a	a b d c				b a c d			
	+ - - -				- + - -			
	i	Precision	Recall	AP@1	i	Precision	Recall	AP@1
	1	1	1	1	1	0	0	1/2
	2	1/2	1		2	1/2	1	
	3	1/3	1		3	1/3	1	
	4	1/4	1		4	1/4	1	
a b	a b d c				b a c d			
	+ + - -				+ + - -			
	i	Precision	Recall	AP@2	i	Precision	Recall	AP@2
	1	1	1/2	1	1	1	1/2	1
	2	1	1		2	1	1	
	3	2/3	1		3	2/3	1	
	4	1/2	1		4	1/2	1	
a b c	a b d c				b a c d			
	+ + - +				+ + + -			
	i	Precision	Recall	AP@3	i	Precision	Recall	AP@3
	1	1	1/3	35/36	1	1	1/3	30/36
	2	1	2/3		2	1	2/3	
	3	2/3	2/3		3	1	1	
	4	3/4	1		4	3/4	1	
MAP	0.9907				0.7778			

Table 4.2 : Result of the original work [1] and its replicated implementation results.

	Training data set	Testing data set	Network	Loss	Implementation	WHDR
Replication	RedWeb	DIW	EncDecResNet	Pairwise	Ours	16.28%
					Chen et al. [5]	16.31%
Original					Xian et al. [1]	14.33%

4.4 Experiments and Results

In order to demonstrate the effectiveness of our proposed loss, we compare our results to those of Xian et al. [1]. However, as the authors did not share their implementation, we had to implement and train their model. We train EncDecResNet with pairwise loss defined in **Equation 2.30** on the RedWeb data set [1], and test on the DIW data set [4] to replicate their results shared in their paper. **Table 4.2** shows the WHDR of our replication, Chen et al.'s [5] replication, and the original results reported in their paper. While our replication is worse than the original results shared by the authors, we were able to replicate slightly better compared to [5].

Next, we train the EncDecResNet model on the RedWeb data set with our proposed ListMLE loss defined in **Equation 4.9**. Since the ground truth scores are available in the RedWeb data set, our loss can take the form of W-ListMLE or ListMLE. We

Table 4.3 : Comparison of proposed listwise losses vs pairwise loss. Lower is better.

Training data set	Testing data set	Network	Loss	WHDR
RedWeb	DIW	EncDecResNet	Pairwise	16.28%
			W-ListMLE	16.33%
			ListMLE	16.71%

Table 4.4 : Results of training on the ReDWeB data set and testing on the YouTube3D data set. Lower is better for WHDR and higher is better for MAP.

Training data set	Testing data set	Network	Loss	WHDR	MAP
RedWeb	YouTube3D	EncDecResNet	Pairwise	**27.19%**	70.36%
			W-ListMLE	27.60%	**70.67%**

experiment with both versions to verify whether weighting increases the performance. We report the WHDR measure on the DIW test split. To optimize the training time, we randomly sample 500 pixels per image and construct the ground truth ranking amongst them. **Table 4.3** shows the comparison of our proposed loss against the pairwise loss. We show that weighting increases the performance marginally in ListMLE loss, and get comparable results with W-ListMLE loss.

In order to enable a better comparison, we introduce the MAP as a performance measure to the SIDE domain. We choose to rank pixels from closest to furthest so MAP measure reflects the usage of logarithm in absolute depth estimation. Since the DIW test split has only an annotation of one pixel pair per image, we test our models on the YouTube3D data set [5]. **Table 4.4** shows the results of training the EncDecResNet with the pairwise loss and the proposed W-ListMLE loss on the RedWeb dataset, and testing on the YouTube3D data set with reporting of the WHDR and MAP measures. The results are comparable on both measures, while the pairwise loss performs slightly better on the WHDR metric, the W-ListMLE loss slightly outperforms the pairwise loss on the MAP metric.

Figure 4.3 shows sample qualitative results from the proposed W-ListMLE loss method from the DIW test split. It can be seen that the network learns to predict overall structure of the scene well for very diverse set of images.

4.5 Discussion

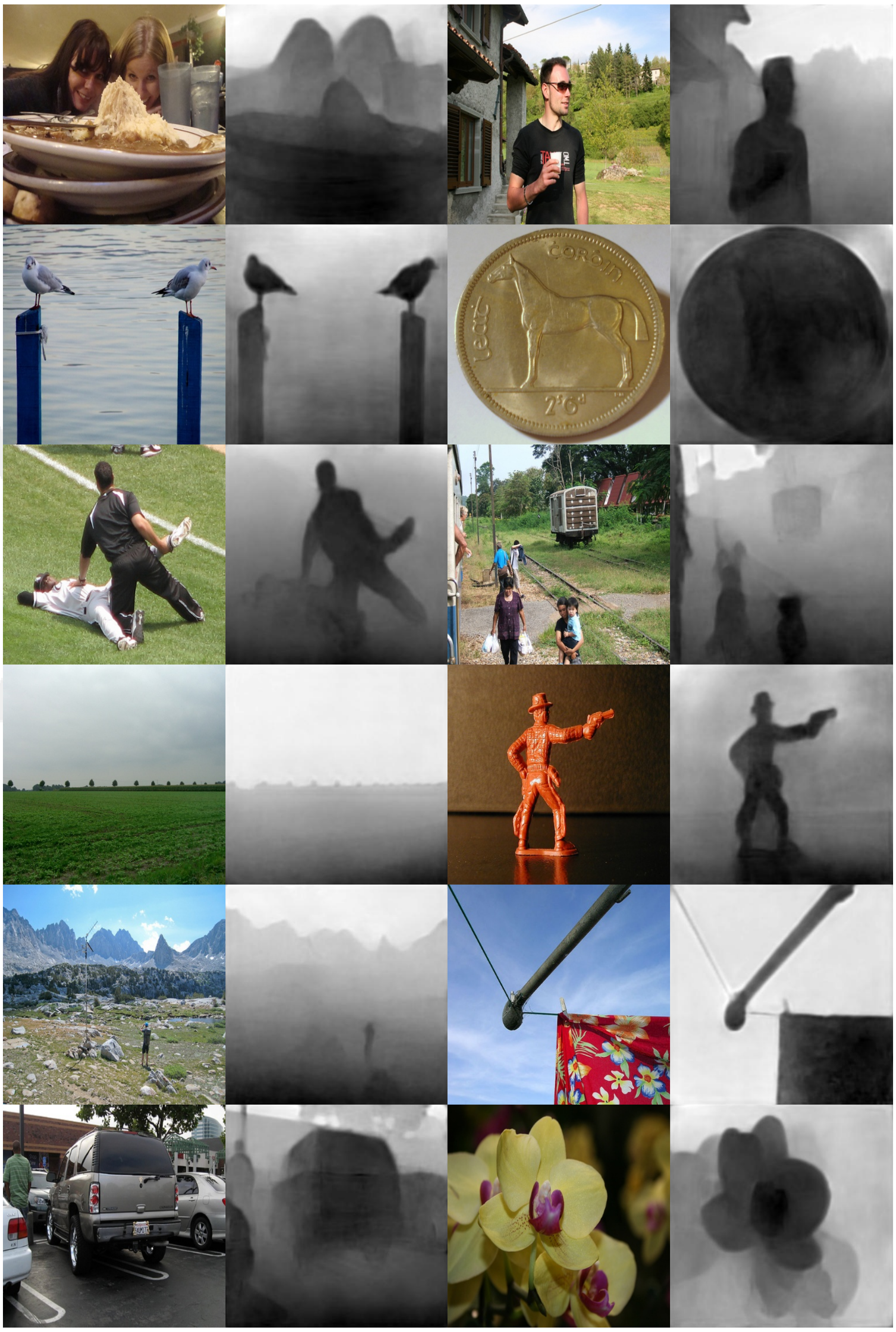

Figure 4.3 : Depth map estimation results of the proposed W-ListMLE approach.

Figure 4.4 : Some of the failure cases. Images are from DIW test split.

Our results display that proposed method with ListMLE loss show similar performances with state of the art methods. We believe our model learns to rank closer pixels better, indicated by the better performance on the MAP measure. The MAP measure penalizes highly those mistakes that are done on higher rankings while the WHDR measure does not discriminate pixels based on their ranking. Moreover, we achieve comparable performances by using 500 pixels per image compared to 3000 pixel pairs per image.

Figure 4.4 shows some of the failure cases. Featureless surfaces (e.g. see the background of bottom row images), and flat surfaces with distinct features (e.g. see painted wall in the top-right image) pose a challenge. In the first case, the model needs to utilize global cues for accurate prediction. In the latter case, it needs to ignore the depth inducing painting texture as depth features. Additionally, image in the top-left exhibits an interesting behaviour. One can argue that it is failure case since in the training data set sky pixels and cloud pixels are considered to be at the same depth. On the other hand, it can be argued that model learns to generalize therefore can recognize clouds as objects that are closer compared to sky.

Figure 4.5 shows results obtained from models trained with pairwise loss, ListMLE loss, and W-ListMLE loss. While model trained with pairwise loss produces relatively sharper results, it tends to miss some objects such as person in the fourth row and small bush in the bottom row. Models trained with ListMLE and W-ListMLE produces similar estimations.

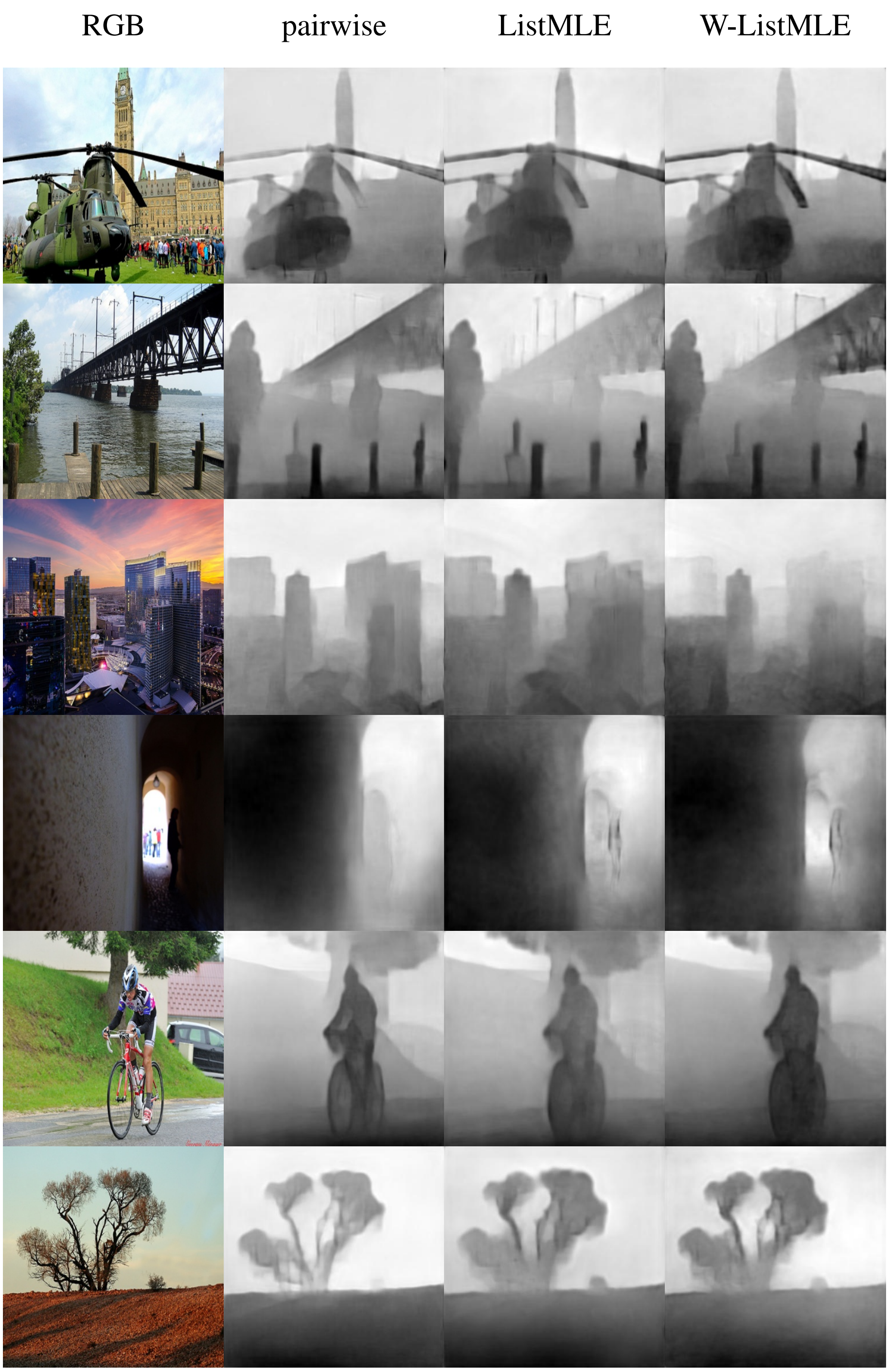

Figure 4.5 : Qualitative results of models trained with different losses.

5. CONCLUSIONS AND RECOMMENDATIONS

In this thesis, we investigate the SIDE problem by providing an extensive literature review, analyzing existing solutions to find the perceptual and computational principles by which they work, and proposing novel contributions by introducing a global pixel ordering sensitive loss to one of the sub-problems of the SIDE, namely the relative-SIDE problem.

Our analysis [17], as well as other investigative works [18, 19], show that the SIDE models trained on limited data sets such as the NYU or the KITTI, are not able to learn generalizable pictorial cues for the SIDE, even though these cues are present in the data sets. We can say that the models tend to "subtly overfit" to the given data by exploiting simple regularities observed in the data sets. [18] shows a nice example of this by empirically proving that the SIDE models search for a ground connection in order to estimate the depth of objects and ignore the object all together when it lacks that connection. While these models will perform successfully in the test set and on unseen images coming from a similar domain, can we really say that they have "learned" to predict depth? Even though the metric performances on the given data sets have increased and continue to increase, we believe that the analyzed models lack the capability of learning generalized cues, hence are incapable of generalization.

In our work [20], we focus on a promising direction of the SIDE for working in the wild, i.e. the relative-SIDE. Here, the problem is to predict ordinal relations of pixels instead of absolute depth in meters. This relaxation allows diverse data sets with appropriate ground truth annotations to be collected and successfully used to train models that work in the wild. We contributed to the existing solutions by formally formulating the problem as a ranking problem, and applying a novel listwise ranking loss, the weighted ListMLE, used in the context of the relative-SIDE problem for the first time.

Furthermore, we introduce the MAP as a performance measure since it captures the importance of correct rankings in higher ranks. Our proposed solution shows comparable performance with the state of the art, even slightly better in terms of the proposed metric MAP.

To sum up, contributions of the thesis are outlined as follows:

- We offer a detailed literature review, summarizing main research categories and common approaches.
- We present our work on the analysis of depth estimating networks, which is the very first work that tries to shed light on the inner workings of such networks.
- We give the formal formulation of the relative-SIDE problem as a ranking problem, apply a listwise loss and achieve comparable results with state of the art.
- We introduce a new metric, namely MAP, which puts more importance to high-ranked pixels. Our proposed method outperforms state of the art methods on this metric.

5.1 Future Work

Here we list some of the research directions and open questions, as well as specific future work suggestions for the SIDE problem.

Ranking losses Losses we have discussed for the relative-SIDE problem are called surrogate losses, as we are not directly optimizing ranking measures such as MAP and NDCG due to their not being differentiable. In recent years, relaxations of ranking metrics that make them differentiable have been devised, allowing them to be used in the optimization process directly [70–72]. These new losses should be investigated in the context of the relative-SIDE problem. Furthermore, [73, 74] proposed new approaches for ranking based on the optimal transport. Their potential success in high level tasks such as the relative-SIDE can be examined.

Analysis of relative-SIDE solutions So far, models that estimate the absolute depth are investigated. It has been shown that these methods tend to focus on different parts of images depending on the data set they are trained on, suggesting that they have learned different features. Models that are trained to work in the wild by estimating relative depth should be investigated.

Absolute depth from relative depth Similar to the idea of using additional sparse sensory information in depth completion in [13, 51], one can use off-the-shelf relative depth estimator models as they are designed to work in the wild, and estimate absolute depth with additional relative depth information alongside RGB image. The success of this approach in terms of metric performance, the need for labeled data, and the ability to adapt to cross domain scenarios can be investigated.

Scalability of unsupervised methods The main premise of unsupervised methods is the ability to learn without ground truth annotations of depth, allowing models to learn from virtually unlimited amount of data. Yet, does unsupervised methods perform better as the data increases? Which unsupervised learning methods benefit more from the increase in the training data?

3D structure Using camera parameters, the depth map of a scene can be projected onto 3D. So far, researchers only qualitatively analyzed 3D projected depth maps. It has been claimed that the methods may exhibit qualitatively different performances even though they have very similar quantitative performances in the metrics calculated from depth maps [9]. A systematic way to measure the quantitative performance on 3D projections of depth maps needs to be devised.

Embodied AI We expect the SIDE models to learn pictorial cues that are used by the human visual system to estimate depth, yet our models are trained very differently when compared to how humans learn. While humans learn and build representations of the world actively through interactions with the world, models that we train learn passively from a data set of static examples. It is possible that agents that learn through interactions may build better representations of the world for the tasks they are optimized to solve. Similar to [75], agents that learn to predict depth through interacting with the world can be an interesting future research direction.

REFERENCES

[1] **Xian, K.**, **Shen, C.**, **Cao, Z.**, **Lu, H.**, **Xiao, Y.**, **Li, R. and Luo, Z.** (2018). Monocular relative depth perception with web stereo data supervision, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.311–320.

[2] **Fu, H.**, **Gong, M.**, **Wang, C.**, **Batmanghelich, K. and Tao, D.** (2018). Deep ordinal regression network for monocular depth estimation, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.2002–2011.

[3] **Foley, H. and Matlin, M.** (2015). *Sensation and Perception*, Psychology Press, fifth edition edition, google-Books-ID: jLBmCgAAQBAJ.

[4] **Chen, W.**, **Fu, Z.**, **Yang, D. and Deng, J.** (2016). Single-image depth perception in the wild, *Advances in Neural Information Processing Systems*, pp.730–738.

[5] **Chen, W.**, **Qian, S. and Deng, J.** (2019). Learning single-image depth from videos using quality assessment networks, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.5604–5613.

[6] **Laina, I.**, **Rupprecht, C.**, **Belagiannis, V.**, **Tombari, F. and Navab, N.** (2016). Deeper depth prediction with fully convolutional residual networks, *2016 Fourth international conference on 3D vision (3DV)*, IEEE, pp.239–248.

[7] **Hoiem, D.**, **Efros, A.A. and Hebert, M.** (2005). Automatic photo pop-up, *ACM transactions on graphics (TOG)*, volume 24, ACM, pp.577–584.

[8] **Cao, Y.**, **Wu, Z. and Shen, C.** (2017). Estimating depth from monocular images as classification using deep fully convolutional residual networks, *IEEE Transactions on Circuits and Systems for Video Technology*, *28*(11), 3174–3182.

[9] **Li, J.**, **Klein, R. and Yao, A.** (2017). A two-streamed network for estimating fine-scaled depth maps from single rgb images, *Proceedings of the IEEE International Conference on Computer Vision*, pp.3372–3380.

[10] **Saxena, A.**, **Sun, M. and Ng, A.Y.** (2008). Make3d: Learning 3d scene structure from a single still image, *IEEE transactions on pattern analysis and machine intelligence*, *31*(5), 824–840.

[11] **Michels, J.**, **Saxena, A. and Ng, A.Y.** (2005). High speed obstacle avoidance using monocular vision and reinforcement learning, *Proceedings of the 22nd international conference on Machine learning*, ACM, pp.593–600.

[12] **Karpathy, A.**, (2020), AI for Full-Self Driving.

[13] **Mal, F. and Karaman, S.** (2018). Sparse-to-dense: Depth prediction from sparse depth samples and a single image, *2018 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, pp.1–8.

[14] **Ranjan, A.**, **Jampani, V.**, **Balles, L.**, **Kim, K.**, **Sun, D.**, **Wulff, J. and Black, M.J.** (2018). Competitive Collaboration: Joint Unsupervised Learning of Depth, Camera Motion, Optical Flow and Motion Segmentation, *arXiv preprint arXiv:1805.09806*.

[15] **Garg, R.**, **BG, V.K.**, **Carneiro, G. and Reid, I.** (2016). Unsupervised cnn for single view depth estimation: Geometry to the rescue, *European Conference on Computer Vision*, Springer, pp.740–756.

[16] **Moravec, H.** (1988). *Mind children: The future of robot and human intelligence*, Harvard University Press.

[17] **Mertan, A. and Duff, D.J.**, (2019), Analysis of Neural Networks for Depth Estimation.

[18] **Dijk, T.v. and Croon, G.d.** (2019). How Do Neural Networks See Depth in Single Images?, *Proceedings of the IEEE International Conference on Computer Vision*, pp.2183–2191.

[19] **Hu, J.**, **Zhang, Y. and Okatani, T.** (2019). Visualization of convolutional neural networks for monocular depth estimation, *Proceedings of the IEEE International Conference on Computer Vision*, pp.3869–3878.

[20] **Mertan, A.**, **Duff, D.J. and Unal, G.**, (2020). Relative Depth Estimation as a Ranking Problem, unpublished.

[21] **Liu, B.**, **Gould, S. and Koller, D.** (2010). Single image depth estimation from predicted semantic labels, *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE, pp.1253–1260.

[22] **Karsch, K.**, **Liu, C. and Kang, S.B.** (2014). Depth transfer: Depth extraction from video using non-parametric sampling, *IEEE transactions on pattern analysis and machine intelligence*, *36*(11), 2144–2158.

[23] **Ladicky, L.**, **Shi, J. and Pollefeys, M.** (2014). Pulling things out of perspective, *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp.89–96.

[24] **Eigen, D. and Fergus, R.** (2015). Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture, *Proceedings of the IEEE international conference on computer vision*, pp.2650–2658.

[25] **Wang, P.**, **Shen, X.**, **Lin, Z.**, **Cohen, S.**, **Price, B. and Yuille, A.L.** (2015). Towards unified depth and semantic prediction from a single image, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.2800–2809.

[26] **Saxena, A.**, **Chung, S.H. and Ng, A.Y.** (2006). Learning depth from single monocular images, *Advances in neural information processing systems*, pp.1161–1168.

[27] **Liu, M.**, **Salzmann, M. and He, X.** (2014). Discrete-continuous depth estimation from a single image, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.716–723.

[28] **Silberman, N.**, **Hoiem, D.**, **Kohli, P. and Fergus, R.** (2012). Indoor segmentation and support inference from rgbd images, *European Conference on Computer Vision*, Springer, pp.746–760.

[29] **Geiger, A.**, **Lenz, P.**, **Stiller, C. and Urtasun, R.** (2013). Vision meets robotics: The KITTI dataset, *The International Journal of Robotics Research*, *32*(11), 1231–1237.

[30] **Liu, X.**, **Zhao, Y. and Zhu, S.C.** (2014). Single-view 3d scene parsing by attributed grammar, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.684–691.

[31] **Li, B.**, **Shen, C.**, **Dai, Y.**, **Van Den Hengel, A. and He, M.** (2015). Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs, *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp.1119–1127.

[32] **Liu, F.**, **Shen, C. and Lin, G.** (2015). Deep convolutional neural fields for depth estimation from a single image, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.5162–5170.

[33] **Zoran, D.**, **Isola, P.**, **Krishnan, D. and Freeman, W.T.** (2015). Learning ordinal relationships for mid-level vision, *Proceedings of the IEEE International Conference on Computer Vision*, pp.388–396.

[34] **Yan, H.**, **Yu, X.**, **Zhang, Y.**, **Zhanga, S.**, **Zhao, X. and Zhang, L.** (2017). Single Image Depth Estimation with Normal Guided Scale Invariant Deep Convolutional Fields, *IEEE Transactions on Circuits and Systems for Video Technology*.

[35] **Ren, Z. and Jae Lee, Y.** (2018). Cross-domain self-supervised multi-task feature learning using synthetic imagery, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.762–771.

[36] **Hoiem, D.**, **Efros, A.A. and Hebert, M.** (2008). Closing the loop in scene interpretation, *2008 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, pp.1–8.

[37] **Kim, S.**, **Park, K.**, **Sohn, K. and Lin, S.** (2016). Unified depth prediction and intrinsic image decomposition from a single image via joint convolutional neural fields, *European conference on computer vision*, Springer, pp.143–159.

[38] **Godard, C.**, **Mac Aodha, O. and Brostow, G.J.** (2017). Unsupervised monocular depth estimation with left-right consistency, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.270–279.

[39] **Zhou, T.**, **Brown, M.**, **Snavely, N. and Lowe, D.G.** (2017). Unsupervised learning of depth and ego-motion from video, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.1851–1858.

[40] **Wang, C.**, **Miguel Buenaposada, J.**, **Zhu, R. and Lucey, S.** (2018). Learning depth from monocular videos using direct methods, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.2022–2030.

[41] **Engel, J.**, **Schöps, T. and Cremers, D.** (2014). LSD-SLAM: Large-scale direct monocular SLAM, *European conference on computer vision*, Springer, pp.834–849.

[42] **Greff, K.**, **Van Steenkiste, S. and Schmidhuber, J.** (2017). Neural expectation maximization, *Advances in Neural Information Processing Systems*, pp.6691–6701.

[43] **Eigen, D.**, **Puhrsch, C. and Fergus, R.** (2014). Depth map prediction from a single image using a multi-scale deep network, *Advances in neural information processing systems*, pp.2366–2374.

[44] **He, K.**, **Zhang, X.**, **Ren, S. and Sun, J.** (2016). Deep residual learning for image recognition, *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp.770–778.

[45] **Lafferty, J.**, **McCallum, A. and Pereira, F.C.** (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

[46] **Qi, X.**, **Liao, R.**, **Liu, Z.**, **Urtasun, R. and Jia, J.** (2018). Geonet: Geometric neural network for joint depth and surface normal estimation, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.283–291.

[47] **Xu, D.**, **Ouyang, W.**, **Wang, X. and Sebe, N.** (2018). Pad-net: Multi-tasks guided prediction-and-distillation network for simultaneous depth estimation and scene parsing, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.675–684.

[48] **Zhang, Z.**, **Cui, Z.**, **Xu, C.**, **Jie, Z.**, **Li, X. and Yang, J.** (2018). Joint task-recursive learning for semantic segmentation and depth estimation, *Proceedings of the European Conference on Computer Vision (ECCV)*, pp.235–251.

[49] **Zhang, Z.**, **Cui, Z.**, **Xu, C.**, **Yan, Y.**, **Sebe, N. and Yang, J.** (2019). Pattern-Affinitive Propagation across Depth, Surface Normal and Semantic Segmentation, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.4106–4115.

[50] **Chen, P.Y.**, **Liu, A.H.**, **Liu, Y.C. and Wang, Y.C.F.** (2019). Towards Scene Understanding: Unsupervised Monocular Depth Estimation With Semantic-Aware Representation, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.2624–2632.

[51] **Liao, Y.**, **Huang, L.**, **Wang, Y.**, **Kodagoda, S.**, **Yu, Y. and Liu, Y.** (2017). Parse geometry from a line: Monocular depth estimation with partial laser observation, *2017 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, pp.5059–5066.

[52] **Izadinia, H.**, **Shan, Q. and Seitz, S.M.** (2017). Im2cad, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.5134–5143.

[53] **Roberts, L.G.** (1963). Machine perception of three-dimensional solids, *Ph.D. thesis*, Massachusetts Institute of Technology.

[54] **Selvaraju, R.R.**, **Cogswell, M.**, **Das, A.**, **Vedantam, R.**, **Parikh, D. and Batra, D.** (2017). Grad-cam: Visual explanations from deep networks via gradient-based localization, *Proceedings of the IEEE international conference on computer vision*, pp.618–626.

[55] **Agrawal, A.**, **Batra, D. and Parikh, D.** (2016). Analyzing the behavior of visual question answering models, *arXiv preprint arXiv:1606.07356*.

[56] **Hoiem, D.**, **Chodpathumwan, Y. and Dai, Q.** (2012). Diagnosing error in object detectors, *European conference on computer vision*, Springer, pp.340–353.

[57] **Johns, E.**, **Mac Aodha, O. and Brostow, G.J.** (2015). Becoming the expert-interactive multi-class machine teaching, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.2616–2624.

[58] **Goodfellow, I.J.**, **Shlens, J. and Szegedy, C.** (2014). Explaining and harnessing adversarial examples, *arXiv preprint arXiv:1412.6572*.

[59] **Zeiler, M.D. and Fergus, R.** (2014). Visualizing and understanding convolutional networks, *European conference on computer vision*, Springer, pp.818–833.

[60] **Szegedy, C.**, **Zaremba, W.**, **Sutskever, I.**, **Bruna, J.**, **Erhan, D.**, **Goodfellow, I. and Fergus, R.** (2013). Intriguing properties of neural networks, *arXiv preprint arXiv:1312.6199*.

[61] **Segall, M.H.**, **Campbell, D.T. and Herskovits, M.J.** (1963). Cultural differences in the perception of geometric illusions, *Science*, *139*(3556), 769–771, iSBN: 0036-8075 Publisher: American Association for the Advancement of Science.

[62] **Luo, W.**, **Li, Y.**, **Urtasun, R. and Zemel, R.** (2016). Understanding the effective receptive field in deep convolutional neural networks, *Advances in neural information processing systems*, pp.4898–4906.

[63] **Kuznietsov, Y.**, **Stuckler, J. and Leibe, B.** (2017). Semi-supervised deep learning for monocular depth map prediction, *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp.6647–6655.

[64] **Cao, Z.**, **Qin, T.**, **Liu, T.Y.**, **Tsai, M.F. and Li, H.** (2007). Learning to rank: from pairwise approach to listwise approach, *Proceedings of the 24th international conference on Machine learning*, ACM, pp.129–136.

[65] **Järvelin, K. and Kekäläinen, J.** (2002). Cumulated gain-based evaluation of IR techniques, *ACM Transactions on Information Systems (TOIS)*, *20*(4), 422–446, iSBN: 1046-8188 Publisher: ACM New York, NY, USA.

[66] **Qin, T.**, **Zhang, X.D.**, **Tsai, M.F.**, **Wang, D.S.**, **Liu, T.Y. and Li, H.** (2008). Query-level loss functions for information retrieval, *Information Processing & Management*, *44*(2), 838–855, iSBN: 0306-4573 Publisher: Elsevier.

[67] **Xia, F.**, **Liu, T.Y.**, **Wang, J.**, **Zhang, W. and Li, H.** (2008). Listwise approach to learning to rank: theory and algorithm, *Proceedings of the 25th international conference on Machine learning*, ACM, pp.1192–1199.

[68] **Chen, W.**, **Liu, T.Y.**, **Lan, Y.**, **Ma, Z.M. and Li, H.** (2009). Ranking measures and loss functions in learning to rank, *Advances in Neural Information Processing Systems*, pp.315–323.

[69] **Baeza-Yates, R. and Ribeiro-Neto, B.** (1999). *Modern information retrieval*, volume463, ACM press New York.

[70] **Cakir, F.**, **He, K.**, **Xia, X.**, **Kulis, B. and Sclaroff, S.** (2019). Deep Metric Learning to Rank, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.1861–1870.

[71] **Revaud, J.**, **Almazan, J.**, **de Rezende, R.S. and de Souza, C.R.** (2019). Learning with Average Precision: Training Image Retrieval with a Listwise Loss, *arXiv preprint arXiv:1906.07589*.

[72] **Grover, A.**, **Wang, E.**, **Zweig, A. and Ermon, S.** (2019). Stochastic optimization of sorting networks via continuous relaxations, *arXiv preprint arXiv:1903.08850*.

[73] **Cuturi, M.**, **Teboul, O. and Vert, J.P.** (2019). Differentiable Sorting using Optimal Transport: The Sinkhorn CDF and Quantile Operator, *arXiv preprint arXiv:1905.11885*.

[74] **Mena, G.**, **Belanger, D.**, **Muñoz, G. and Snoek, J.** (2017). Sinkhorn Networks: Using Optimal Transport Techniques to Learn Permutations, *NIPS Workshop in Optimal Transport and Machine Learning*, volume 3.

[75] **Pinto, L.**, **Gandhi, D.**, **Han, Y.**, **Park, Y.L. and Gupta, A.** (2016). The curious robot: Learning visual representations via physical interactions, *European Conference on Computer Vision*, Springer, pp.3–18.

[76] **Firman, M.** (2016). RGBD datasets: Past, present and future, *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp.19–31.

[77] **Uhrig, J.**, **Schneider, N.**, **Schneider, L.**, **Franke, U.**, **Brox, T. and Geiger, A.** (2017). Sparsity invariant cnns, *2017 International Conference on 3D Vision (3DV)*, IEEE, pp.11–20.

[78] **Song, S.**, **Lichtenberg, S.P. and Xiao, J.** (2015). Sun rgb-d: A rgb-d scene understanding benchmark suite, *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp.567–576.

[79] **McCormac, J.**, **Handa, A.**, **Leutenegger, S. and Davison, A.J.** (2016). SceneNet RGB-D: 5M photorealistic images of synthetic indoor trajectories with ground truth, *arXiv preprint arXiv:1612.05079*.

APPENDICES

APPENDIX A : Common Data Sets
APPENDIX B : Performance Measures
APPENDIX C : 3D Vision Basics

APPENDIX A

Here we summarize some of the data sets that are frequently used in the literature. For a better review of available data sets, please refer to [76].

NYU [28] It consists of indoor images, and their corresponding depth images and semantic segmentation maps. It contains 1449 densely labelled images from 464 indoor scenes such as bathrooms, kitchens, libraries, living rooms, etc. Additionally, it has a raw version consists of over 400k images and corresponding depth maps coming from video recording of same indoor environments.

KITTI [77] It consists of over 93k images with their corresponding depth maps, collected by driving a car in a mid-size city, in rural areas and in highways. It contains outdoor images with visible cars and pedestrians.

Sun RGB-D [78] It contains 10,335 indoor images with corresponding depth maps. Additionally, it contains ground truth annotations for scene category, 2D and 3D bounding boxes for object detection, 2D semantic segmentation, object orientation, and room layout estimation.

SceneNet RGB-D [79] It contains synthetically created 5M indoor images with their depth maps. Room conditions, lighting and textures are randomly generated.

APPENDIX B

SIDE The metrics that are used for quantitative measurements can be seen below. d_{gt} shows the depth of ground truth pixel and d_p shows the depth of the pixel in predicted depth map. T is the total number of pixel in all of the images that are evaluated for the loss function.

- Absolute relative error (rel):

$$\frac{1}{T}\sum_{p}\frac{|d_{gt}-d_p|}{d_{gt}}T$$

- Root mean squared error (RMSE):

$$\sqrt{\frac{1}{T}\sum_{p}(d_{gt}-d_p)^2}$$

- Root mean squared error in log space (RMSE log):

$$\sqrt{\frac{1}{T}\sum_{p}(\log d_{gt}-\log d_p)^2}$$

- Absolute log error:

$$\sqrt{\frac{1}{T}\sum_{p}|\log d_{gt}-\log d_p|}$$

- Accuracy with thresholding (δ_i): percentage of d_p s.t.

$$max(\frac{d_{gt}}{d_p},\frac{d_p}{d_{gt}})=\delta_i<1.25^i, i\in\{1,2,3\}$$

Ranking For n documents with ground truth scores $\mathbf{S}$, NDCG of a ranking π is defined as follows:

$$NDCG(\mathbf{S},\pi)=\frac{1}{N_{max}}\sum_{r=1}^{n}G\left(S_{\pi(r)}\right)D(r)$$

where G is an increasing function, D is a decreasing function, $\pi(r)$ returns the index of the r^{th} item's index, S_i is the score of the item i, and $N_{max}=\max_{\pi}\sum_{r=1}^{n}G\left(S_{\pi(r)}\right)D(r)$.

Average precision is defined for binary labeled documents as follows:

$$AP=\sum_{i=1}^{n}\text{Precision}(i)(\text{Recall}(i)-\text{Recall}(i-1))$$

where $Precision(i)=\frac{tp}{tp+fp}$, $Recall(i)=\frac{tp}{tp+fn}$ up until i^{th} document in the ranking and $Precision(0)=Recall(0)=0$

APPENDIX C

We summarize basic 3D vision properties that are used in unsupervised depth estimation. In unsupervised depth estimation, we have two images of the same scene taken from different angles. With estimated depth, one image is warped into the other image and a reconstruction loss is used to train the models.

Using rectified images with known camera baseline In this case, pixels only move horizontally in the image space. Depth d can be turned into disparity $\hat{d}$ using camera baseline b and focal length f as $\hat{d} = \frac{bf}{d}$. Image warping can be performed by simply mapping pixels from one image into other as follows:

$$w(I^l(x,y),\hat{d}) = I^r(x,y+\hat{d})$$

where w is the warping function which warps pixel $I^l(x,y)$ into $I^r(x,y+\hat{d})$ and $\hat{d}$ is the disparity of the pixel.

Using camera motion and intrinsics In this case, image reconstruction can be done in 3 steps. First, pixels of the image with estimated depth are projected into camera coordinates $\mathbf{X} = [X,Y,Z]^T$ using image coordinates (x,y) depth d, and camera intrinsics (c_x, c_y, f).

$$X = \frac{d}{f}(x - c_x)$$
$$Y = \frac{d}{f}(y - c_y)$$
$$Z = d$$

Next, camera coordinates are transformed according to camera motion.

$$\mathbf{X}' = R_x R_y R_z \mathbf{X} + t$$

Lastly, new camera coordinates $\mathbf{X}'$ are turned into new image coordinates to produce other image.

$$x' = \frac{f}{Z'}X' + c_x$$
$$y' = \frac{f}{Z'}Y' + c_y$$

CURRICULUM VITAE

Name Surname: Alican Mertan

Place and Date of Birth: Samsun, Turkey - 12.02.1994

E-Mail: mertana@itu.edu.tr

EDUCATION:

- **B.Sc.:** 2018, Istanbul Technical University, Computer Engineering
- **M.Sc.:** 2020, Istanbul Technical University, Computer Engineering

PROFESSIONAL EXPERIENCE AND REWARDS:

- 2018-2019 Beneficiary of TUBITAK scholarship on project no: 116E167

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

- **Mertan, A., Duff, D.J., and Unal, G.** (2020) ”Relative Depth Estimation as a Ranking Problem”, 28th Signal Processing and Communications Applications Conference (SIU).
- **Mertan, A. and Duff, D.J.** (2019) ”Analysis of Neural Networks for Depth Estimation”, 27th Signal Processing and Communications Applications Conference (SIU), Poster.

OTHER PUBLICATIONS, PRESENTATIONS AND PATENTS:

- **Kamar, D., Akyol, G., Mertan, A., and Inceoglu, A.** (2020) ”Comparative Analysis of Reinforcement Learning Algorithms on TORCS Environment”, 28th Signal Processing and Communications Applications Conference (SIU).