JYU DISSERTATIONS 332

Leevi Annala

Convolutional Neural Networks and Stochastic Modelling in Hyperspectral Data Analysis

UNIVERSITY OF JYVÄSKYLÄ
FACULTY OF INFORMATION TECHNOLOGY

JYU DISSERTATIONS 332

Leevi Annala

Convolutional Neural Networks and Stochastic Modelling in Hyperspectral Data Analysis

Esitetään Jyväskylän yliopiston informaatioteknologian tiedekunnan suostumuksella julkisesti tarkastettavaksi joulukuun 12. päivänä 2020 kello 10.

Academic dissertation to be publicly discussed, by permission of the Faculty of Information Technology of the University of Jyväskylä, on December 12, 2020 at 10 o'clock am.

JYVÄSKYLÄN YLIOPISTO
UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2020

Editors
Timo Männikkö
Faculty of Information Technology, University of Jyväskylä
Ville Korkiakangas
Open Science Centre, University of Jyväskylä

Permanent link to this publication: http://urn.fi/URN:ISBN:978-951-39-8453-3

ISBN 978-951-39-8453-3 (PDF)
URN:ISBN:978-951-39-8453-3
ISSN 2489-9003

ABSTRACT

Annala, Leevi
Convolutional neural networks and stochastic modelling in hyperspectral data analysis
Jyväskylä: University of Jyväskylä, 2020, 56 p. (+included articles)
(JYU Dissertations
ISSN 2489-9003; 332)
ISBN 978-951-39-8453-3 (PDF)

Hyperspectral imaging is relatively new and rapidly growing field of research. The datasets produced by hyperspectral imaging are large, and handling such data requires large computational resources. Therefore, there is a need for developing machine learning methods that can cope with the data, and methods to reduce the necessary amount of data gathering missions. For the latter, problem the author and his co-authors have developed stochastic modelling and generative adversarial neural networks for data augmentation. In machine learning, they have experimented with using convolutional neural network in conjunction with said stochastic model in order to retrieve useful information from hyperspectral data. Additionally, the author lists useful Python packages for hyperspectral data analysis.

Keywords: Hyperspectral imaging, Convolutional neural network, Stochastic modelling, Biophysical parameter retrieval, Data augmentation

TIIVISTELMÄ (ABSTRACT IN FINNISH)

Annala, Leevi
Konvoluutioneuroverkkojen ja stokastisen simuloinnin käyttö hyperspektrikuvien analysoinnissa
Jyväskylä: University of Jyväskylä, 2020, 56 s. (+artikkelit)
(JYU Dissertations
ISSN 2489-9003; 332)
ISBN 978-951-39-8453-3 (PDF)

Hyperspektrikuvantaminen on kasvava ala. Hyperspektrikuvantaminen on resurssien ja datan määrän suhteen vaativaa, ja siksi on tarpeen kehittää koneoppimismenetelmiä, jotka pystyvät käsittelemään dataa, ja menetelmiä tarvittavan datan keräämisen vähentämiseksi. Viimeksi mainittua ongelmaa varten kirjoittaja ja hänen kanssakirjoittajansa ovat kehittäneet stokastista mallintamista ja generatiivisia kilpailevia neuroverkkoja datan määrän kasvattamiseksi mitatun datan rinnalla. Koneoppimisessa he ovat käyttäneet konvoluutioneuroverkkoa mainitun stokastisen mallin kanssa saadakseen hyödyllistä tietoa hyperperspektridatasta. Lisäksi työssä etsittiin ja ja testattiin hyödyllisiä Python-paketteja hyperspektridatan analysointiin.

Avainsanat: Hyperspektrikuvantaminen, Konvoluutioneuroverkko, Stokastinen mallintaminen, biofysikaalisen parametrin palauttaminen, Datan lisääminen

Author Leevi Annala
Faculty of Information Technology
University of Jyväskylä
Finland

Supervisors Docent Ilkka Pölönen
Faculty of Information Technology
University of Jyväskylä
Finland

Docent Sami Äyrämö
Faculty of Information Technology
University of Jyväskylä
Finland

Professor Pekka Neittaanmäki
Faculty of Information Technology
University of Jyväskylä
Finland

Reviewers Professor Heikki Haario
Department of Computational Engineering
Lappeenranta University of Technology
Finland

Assistant professor Arto Klami
Department of Computer Science
Faculty of Science
University of Helsinki
Finland

Opponent Professor Keijo Ruotsalainen
Applied and Computational Mathematics
Faculty of Information Technology and Electrical Engineering
University of Oulu
Finland

ACKNOWLEDGEMENTS

I want to thank many people for helping me complete this work.

First, I want to thank my supervisors, Docents Ilkka Pölönen and Sami Äyrämö, and Professor Pekka Neittaanmäki. Thank you for your guidance and example. I have learned a lot from you and will continue to ask your guidance during the remainder of my academic career. Thank you for the reviewers, Professor Heikki Haario and Assistant professor Arto Klami, for reviewing the thesis and telling me how to improve this manuscript.

I wish to acknowledge the help, support and love I received from the current and former members of the Spectral Imaging Laboratory and other close colleagues. Without you I would have lost my mind sooner and my life would be boring. Special thanks to Anna-Maria Raita-Hakola for re-inventing behaviourism. Thank you for conditioning me with chocolate treats whenever I managed to do my job properly. Without those six pieces of chocolate you left for me during autumn break, the finishing of this work would have been much harder for me. Also Kimmo Riihiaho, Sampsa Kiiskinen and Pauliina Salmi helped me directly with this thesis by proofreading it and commenting on the draft. I am grateful for your help.

I should also acknowledge the work done by my co-authors. Science is a collaboration. Thank you for helping us publish our work and providing me with extremely interesting problems to work on. Thanks go also to the funders of the research: the Academy of Finland, Business Finland and the Faculty of Information Technology.

Lastly, I thank my family. Mom, Dad, Heikki, Mummu and Pappa, thank you for being part of my all-around great, loving and supporting family, and raising me up to be who I am today. Elias and Eetu, thank you for helping me maintain some resemblance of work-life balance and letting me be your hero. Maria, thank you for loving and supporting me through this process. Without you, this work would have been impossible.

With love,
Leevi

LIST OF ACRONYMS

CNN	Convolutional neural network
FPI	Fabry-Pérot interferometer
GAN	Generative adversarial neural network
ML	Machine learning
SM	Stochastic model
SLOP	Stochastic model for leaf optical properties
RGB	Red, green, blue
reLU	Rectified linear unit

LIST OF FIGURES

FIGURE 1 On the left: Example of a hyperspectral image. On the right: Spectrum from the hyperspectral image. 17

FIGURE 2 Network of states and transitions of light propagation in the Stochastic Model. 23

FIGURE 3 (a) Example spectra from skin stochastic model. (b) Example spectra from leaf stochastic model. 24

FIGURE 4 Top: An image and convolution kernels. Bottom: An image and convoluted images. 28

FIGURE 5 Example of a fully connected deep neural network with one input and one output, and two hidden layers with 3 and 5 nodes. 29

FIGURE 6 Example of (one dimensional) convolutional neural network that has nine inputs, convolution with one kernel of size two, pooling layer of size two and one output. Usually a fully connected part is added after enough convolution-pooling combinations. Note that the amount of connections is lower than if all layers would be fully connected. This means that in the machine learning training phase, there is less parameters to optimise. 34

CONTENTS

ABSTRACT
TIIVISTELMÄ (ABSTRACT IN FINNISH)
ACKNOWLEDGEMENTS
LIST OF ACRONYMS
LIST OF FIGURES
CONTENTS
LIST OF INCLUDED ARTICLES

1 INTRODUCTION 15
1.1 Background 15
1.2 Research environment 16
1.3 Objectives and Scope 18
1.4 Author's other works 19
1.5 Dissertation structure 19

2 THEORETICAL FOUNDATION 20
2.1 Hyperspectral imaging and hyperspectral data 20
2.2 Stochastic modelling of hyperspectral data 21
2.3 Machine learning problems in hyperspectral imaging 23
2.3.1 Hyperspectral classification problems 25
2.3.2 Biophysical parameter retrieval 25
2.3.3 Hyperspectral data augmentation 26
2.4 Convolutional neural network 26
2.4.1 Convolutional neural network – Intuition 27
2.4.2 Neural Network 27
2.4.3 Convolution 32
2.4.4 Convolutional neural network 32

3 RESEARCH RESULTS AND AUTHOR CONTRIBUTION 36
3.1 PI: Practical Approach for Hyperspectral Image Processing in Python 36
3.1.1 Results 36
3.1.2 Author contribution 36
3.2 PII: Tree Species Identification Using 3D Spectral Data and 3D Convolutional Neural Network 36
3.2.1 Results 36
3.2.2 Author contribution 37
3.3 PIII: Convolutional neural networks in skin cancer detection using spatial and spectral domain 37
3.3.1 Results 37
3.3.2 Author contribution 37

3.4 PIV: Chlorophyll Concentration Retrieval by Training Convolutional Neural Network for Stochastic Model of Leaf Optical Properties (SLOP) Inversion 37
3.4.1 Results 37
3.4.2 Author contribution 38
3.5 PV: Kubelka-Munk Model and Stochastic Model Comparison in Skin Physical Parameter Retrieval 38
3.5.1 Results 38
3.5.2 Author contribution 38
3.6 PVI: Comparison of Machine Learning Methods in Stochastic Skin Optical Model Inversion 38
3.6.1 Results 38
3.6.2 Author contribution 39
3.7 PVII: Generating Hyperspectral Skin Cancer Imagery using Generative Adversarial Neural Network 39
3.7.1 Results 39
3.7.2 Author contribution 39
3.8 Summary of the results 39

4 DISCUSSION 41
4.1 Limitations and further research 42

5 CONCLUSION 44

YHTEENVETO (SUMMARY IN FINNISH) 45

REFERENCES 46

INCLUDED ARTICLES

LIST OF INCLUDED ARTICLES

PI **Leevi Annala**, Matti A. Eskelinen, Jyri Hämäläinen, Aamos Riihinen and Ilkka Pölönen. Practical Approach for Hyperspectral Image Processing in Python. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XLII-3*, 2018.

PII Ilkka Pölönen, **Leevi Annala**, Samuli Rahkonen, Olli Nevalainen, Eija Honkavaara, Sakari Tuominen, Niko Viljanen and Teemu Hakala. Tree Species Identification Using 3D Spectral Data and 3D Convolutional Neural Network. *2018 9th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS), IEEE*, 2018.

PIII Ilkka Pölönen, Samuli Rahkonen, **Leevi Annala** and Noora Neittaanmäki. Convolutional neural networks in skin cancer detection using spatial and spectral domain. *Photonics in Dermatology and Plastic Surgery 2019, International Society for Optics and Photonics*, 2019.

PIV **Leevi Annala**, Eija Honkavaara, Sakari Tuominen and Ilkka Pölönen. Chlorophyll Concentration Retrieval by Training Convolutional Neural Network for Stochastic Model of Leaf Optical Properties (SLOP) Inversion. *Remote Sensing, 12(2):283*, 2020.

PV **Leevi Annala** and Ilkka Pölönen. Kubelka-Munk Model and Stochastic Model Comparison in Skin Physical Parameter Retrieval. *Computational Sciences and Artificial Intelligence in Industry – New digital technologies for solving future societal and economical challenges, Springer*, in press, 2020.

PVI **Leevi Annala**, Sami Äyrämö and Ilkka Pölönen. Comparison of Machine Learning Methods in Stochastic Skin Optical Model Inversion. *Applied Sciences, 10(20): 7097*, 2020.

PVII **Leevi Annala**, Noora Neittaanmäki, John Paoli, Oscar Zaar and Ilkka Pölönen. Generating Hyperspectral Skin Cancer Imagery using Generative Adversarial Neural Network. *2020 42st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), IEEE*, 2020.

1 INTRODUCTION

1.1 Background

It all started with a rainbow on a wall during the black death pandemic in 1660s. Isaac Newton had dimmed his living room and made a tiny hole in his curtains to experiment with light. Now we know that rainbow is produced by water droplets acting as a prism to refract the different light wavelengths of white light into their own beams. While prisms were known for centuries before him, Newtons work "Opticks" is the earliest known scientific accounting of light refraction in the prism. This was the beginning of the research field now known as spectroscopy. (Thomas, 1991)

Spectroscopy is the study of electromagnetic radiation, for which light is subcategory of, and its interaction with matter. When a photon, the light particle, hits the matter, it either reflects off of the matter, is absorbed into it, or transmits through it. In optical spectroscopy one measures the amount and wavelength of photons hitting the sensor. The measuring device takes into account the photons transmitted through the matter, the photons scattered from the surface of the matter and the photons emitted by the matter. Emission happens, when excited electron state is released, and the electron state is excited when the photon energy is absorbed into the matter. (Hof, 2003)

When we add spatial, i.e. photographic, information to the spectral information gathered in the field of spectroscopy, we enter the domain of spectral imaging. In order to do spectral imaging, we generally need a device with two parts:

1. We need a way to capture spatial data, like a camera, and
2. a way to control the wavelength of light detected by the camera.

In this research, it is achieved with a machine vision camera and Fabry-Perót interferometer (Saari et al., 2009, 2013). Such imager produces an image similar to normal digital photograph, but instead of channels for red, green and blue there is 20-200 channels for specific small wavelength ranges. A hyperspectral image is often referred to as datacube, because it is essentially a thick stack of monochromatic photographs. From each pixel of a hyperspectral datacube, one can plot a spectrum that is similar to one captured with a traditional spectrometric device. Example of hyperspectral image and a spectrum of a pixel can be seen in Figure 1.

Hyperspectral imaging has multitude of applications. For example, in the remote sensing field it has been used in forestry and agriculture applications (Adão et al., 2017), and in marine biology applications (Dumke et al., 2018). For example, Kaivosoja et al. (2013) studied the ways to use hyperspectral imaging to accurately apply fertiliser to the field. This includes using hyperspectral imaging in biomass and nitrogen content estimation. In (Zarco-Tejada et al., 2013), hyperspectral imaging was used to estimate carotenoid content from vineyard imagery. Näsi et al. (2015) took hyperspectral images with a drone and used the images to estimate the amount of bark beetle damage in Norway spruce. Dumke et al. (2018) used hyperspectral imaging underwater to identify taxonomical classes of deep-sea fauna.

In industry, hyperspectral imaging is used in various ways. For example, in the paper industry it is used to monitor the drying of the cellulose mass (Ma et al., 2020). In pharmaceutical industry, it is used in quality control (Hamilton and Lodder, 2002; Gendrin et al., 2007; Roggo et al., 2005). In is also used in plastic waste recycling (Serranti et al., 2011). In food industry, there is multitude of applications in quality control (Liu et al., 2017). Hyperspectral imaging is used, for example, in meat quality control (Elmasry et al., 2012), fish quality control (Cheng and Sun, 2014) and dairy quality control (Gowen et al., 2009).

Hyperspectral imaging is also finding use in the biomedical field. In the field, hyperspectral imaging is used, for example, in skin cancer research (Dicker et al., 2006; Zherdeva et al., 2016; Neittaanmäki-Perttu et al., 2013; Zheludev et al., 2015) and in diabetic foot ulceration monitoring (Nouvong et al., 2009; Yudovsky et al., 2010). More general wound monitoring applications are also available (Shah et al., 2003; Calin et al., 2015; Wahabzada et al., 2017).

1.2 Research environment

The research was carried out in the Spectral Imaging Laboratory of the Faculty of Information Technology in the University of Jyväskylä, headed by Docent Ilkka Pölönen. The author was mainly responsible of data analysis tasks, for which the computational clusters were provided by the Finnish Grid and Cloud Infrastructure.

The research described in this dissertation builds on the work of previous graduates of Spectral Imaging Laboratory. These graduates include the founder of the laboratory, Docent Ilkka Pölönen, Dr. Hannu-Heikki Puupponen and Dr. Matti A. Eskelinen. Pölönen published his dissertation by the name of "Discovering knowledge in various applications with a novel hyperspectral imager" in

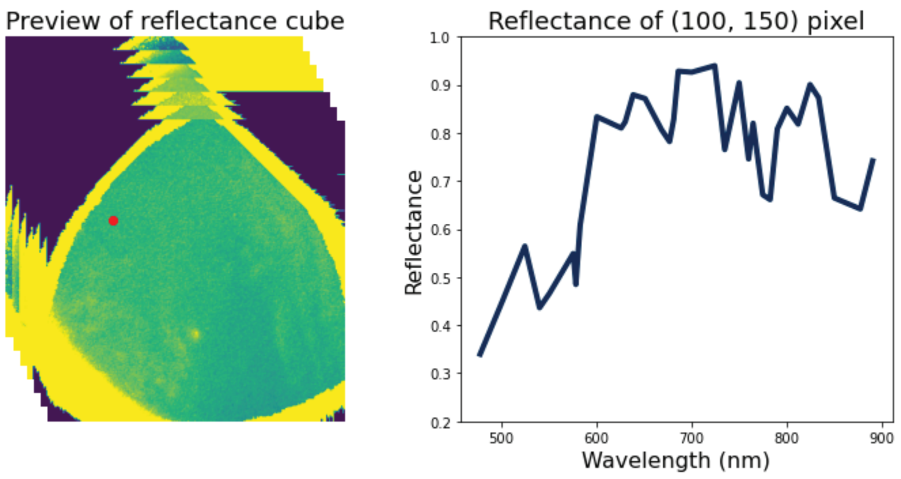


FIGURE 1 On the left: Example of a hyperspectral image, with example spectrum location marked with a red dot. On the right: Spectrum from the hyperspectral image. This Figure was produced by the author's contribution to the software used in (Trops et al., 2019).

2013. In his thesis, he describes the type of hyperspectral camera used in this research in addition to the preprocessing steps crucial to any analysis of data produced by such camera.

In 2014, Dr. Hannu-Heikki Puupponen wrote his disseration "Unmixing methods in novel applications of spectral imaging". His thesis serves as a reminder of the multitude of applications for hyperspectral imaging. Just in his list of included articles he lists applications in fields of crime scene investigation, explosive detection, environmental screening and skin cancer research.

Dr. Matti A. Eskelinen published his dissertation in 2019. His title was "Computational methods for hyperspectral imaging using Fabry-Perót interferometers and colour cameras". In his research, he has published open source tools for hyperspectral data processing in Python programming language. These tools are also used in the dissertation at hand.

The analysed empirical data originates from multiple sources. Data for PII and PIV was gathered on a drone mission over a research forest in Vesijako area in Pudasjoki (Nevalainen et al., 2017). The data was captured using hyperspectral camera with Fabry-Perót interferometer technology (Saari et al., 2009, 2013), that produces two dimensional spatial data instead of the one dimensional spatial data used in traditional pushbroom hyperspectral imagers.

Empirical data for articles PIII, PV, PVI and PVII came from three different hospitals:

– Department of Dermatology and Allergology of Helsinki University Hospital, Helsinki, Finland,
– Päijät-Häme Central Hospital, Lahti, Finland, and
– Department of Dermatology and Venereology of Sahlgrenska University Hospital, Göteborg, Sweden.

In each hospital, identical hyperspectral cameras were used. The camera was

Revenio Prototype 2016, which has spatial resolution of 1920×1200 pixels and spectral resolution of 120 wavelengths from 460 nm to approximately 842 nm.

1.3 Objectives and Scope

This dissertation provides an answer to the question: How can the hyperspectral images, deep learning algorithms and mathematical modelling be used in conjuction to increase the human knowledge? The context of the research are in fields of forestry and medicine. The author starts by finding the right tools for the work in PI. Then he uses convolutional neural network (CNN) in a direct way to solve hyperspectral classification problems in PII and PIII. In PIV the author uses a stochastic model to produce training data for CNN regression. The author uses CNN as an inversion method for the model. In PV and PVI this idea is expanded by extending the model used in forestry context in PIV to general stochastic model for layered media and using the model in producing skin spectra. The stochastic model and a Kubelka-Munk model are inverted in PV, and in PVI, the author proceeds to finding out which machine learning algorithms work well in inverting such mathematical models. PVII is a short excursion to finding out if the mathematical modelling part of articles PIV, PV and PVI is replaceable with another deep learning algorithm.

The research questions of these articles are the following:

- PI What are the best tools in Python programming language for hyperspectral image analysis, machine learning, and neural networks?
- PII How well does the CNN work in tree species classification?
- PIII How well does the CNN work in skin cancer classification?
- PIV
  1. How well can the Stochastic Model for Leaf Optical Properties (SLOP) be inverted in terms of chlorophyll content?
  2. How well can the inverted stochastic model be transfered to predict chlorophyll content in hyperspectral images?
- PV
  1. Can SLOP be transported to skin optical model?
  2. How well does it work compared to Kubelka-Munk model?
  3. How well do these two models work with CNN inversion?
- PVI Which machine learning algorithm (of the researched algorithms) is the best for stochastic model inversion?
- PVII Can hyperspectral data be easily modelled using generative adversarial neural network (GAN)?

Author's contributions are described in detail in chapter 3.

1.4 Author’s other works

During his PhD training, the author has contributed to the following works not listed as part of this thesis:

1. "Hexyl aminolevulinate, 5-aminolevulinic acid nanoemulsionand methyl aminolevulinate in photodynamic therapy ofnon-aggressive basal cell carcinomas: A non-sponsored,randomized, prospective and double-blinded trial" by Salmivuori et al. (2020)
2. "Hexylaminolevulinate and Aminolevulinic acid Nanoemulsion have Similar Tolerability, Initial Efficacy and Cosmetic Outcome as Methylaminolevulinate in Photodynamic Therapy of Basal Cell Carcinoma in a Prospective Randomized Double-blinded Trial" by Salmivuori et al. (2019)
3. "Miniature MOEMS hyperspectral imager with versatile analysis tools" by Trops et al. (2019)
4. "Minimal learning machine in anomaly detection from hyperspectral images" by Pölönen et al. (2020).

1.5 Dissertation structure

The dissertation is structured as follows. In Chapter 1, the author puts the dissertation in historical context, describes the environment the where work is done, presents the research questions and his other work. In Chapter 2, the foundations of the included articles are described and the necessary background information is given. Chapter 3 summarises the research results and presents the authors contributions to the articles. In Chapter 4, the author discusses the strengths and weaknesses of the work and in Chapter 5, the author provides concluding remarks on the work.

2 THEORETICAL FOUNDATION

2.1 Hyperspectral imaging and hyperspectral data

Hyperspectral imaging is combining spectroscopy and photogrammetry. Hyperspectral imagers can be broadly divided into four categories (Chang, 2007):

- line scanners,
- pushbroom scanners,
- whiskbroom scanners, and
- framing cameras.

The devices used during this research fall under the last category. The hyperspectral cameras consist of a machine vision camera and a device that controls the wavelength range of the electromagnetic radiation passed on to the camera, such as Fabry-Pérot interferometer (FPI) (Pérot and Fabry, 1899; Saari et al., 2009, 2013). The imaging procedure is to make small adjustments to the wavelength range and take a photograph after every adjustment. This is done quickly, and normal imaging time is less than one second for a full hypespectral image. The resulting hypespectral image has roughly the same spatial information as an RGB-photograph of the object, but it contains more information in the spectral dimension, upwards to 200 wavelength ranges compared to the 3 in the RGB-photograph. Example of a hyperspectral image can be seen in Figure 1.

Considering the amount of data contained in a hyperspectral image, it requires considerably more storage space than an RGB-photograph. While using hyperspectral imagers of the FPI type used in this research, one raw hyperspectral image consists of RGB-photographs for each wavelength setting. Therefore it takes up the disc space of 50-200 2 megapixel RGB-photographs. From the raw images, the radiance is calculated. Radiance is the amount of electromagnetic radiation leaving the imaged object towards the camera. Sometime reflectance, which describes the ratio of incoming and outgoing light, is more useful. Reflectance can be calculated for instance by calculating the ratio of radiance and *white reference*. Now, we have three to four sets of hyperspectral datacubes, depending on the amount of needed white references. This easily adds app to 2-4 gigabytes of data per imaging target. If we are doing data science with the entire hyperspectral images, for example classifying lesions in skin cancer research, there is no upper limit for the amount of data we would like to have. By the end of all this, we easily get datasets of hundreds of gigabytes. This creates benchmark for the used tools, as they need to be suited for data intensive science.

2.2 Stochastic modelling of hyperspectral data

Fortunately, hyperspectral data can be also modelled by physically based mathematical models. The goal is to calculate the spectral data by using known physical and biological properties of the imaged object and the physical principles behind light-matter interaction. These models are used to approximate the spectra.

For example, in skin reflectance modelling, some relevant properties would be the concentrations of different chromophores, such as melanin or hemoglobin, in the different skin layers, or the thicknesses of the layers. The physical interaction can be modeled for example by using by using Kubelka-Munk theory (Kubelka, 1931) as in (Angelopoulou, 2001; Jolivot et al., 2013), or Boltzmann photon transport theory (Ishimaru, 1978) or diffusion theory models (Van Gemert et al., 1987). These are called deterministic models by Baranoski et al. (2015). Other class of models are stochastic models which include randomness and are based on Monte Carlo modelling (Hammersley, 2013). Examples of different stochastic models can be found in (Shimada et al., 2001; Wang et al., 1995).

In the remote sensing field, the mathematical modelling task is slightly different compared to skin modelling, as the distance from the imaged object is longer. Therefore, in the forestry context, the modelling usually takes into account both canopy and individual leaf properties. The division into deterministic and stochastic models is less strict, as one can model canopy stochastically and leaf determinististically or other way around. Examples of models used in forestry can be found from (Jacquemoud and Baret, 1990; Jacquemoud et al., 2009; Jacquemoud and Ustin, 2008; Govaerts et al., 1996; Goel, 1988)

Stochastic modelling approach used in PIV, PV and PVI is based on Markov chain. The idea is to model the imaged object pixelwise as a network of states and their connections. The work is based on stochastic model for leaf optical properties by Maier et al. (1999) and Tucker and Garratt (1977). It is used without modifications in article PIV, and modified for skin optical properties by the author in PV and PVI by using skin chromophore parameters and scattering and absorption coefficients from Jacques (2013).

The stochastic model used in PV and PVI is described in Figure 2. It consists of states, and their connecting probabilities. Each layer in the figure represents a layer in the skin (in PV and PVI context). The probabilities are calculated in each layer as follows:

$$P_{\text{direct reflection}} = \begin{cases} 0.02 & \text{if the current state is illumination,} \\ 1 & \text{if the current state is direct reflected,} \\ 0 & \text{otherwise} \end{cases}$$

$$P_{\text{absorption}}(\lambda) = \begin{cases} 1 & \text{if the current state is absorbed,} \\ \frac{a(\lambda)}{a(\lambda)+s(\lambda)} \cdot (1 - e^{-(a(\lambda)+s(\lambda))\cdot L}) & \text{otherwise,} \end{cases}$$

$$P_{\text{scattering}}(\lambda) = \begin{cases} 0 & \text{if the current state is absorbed,} \\ \frac{s(\lambda)}{a(\lambda)+s(\lambda)} \cdot (1 - e^{-(a(\lambda)+s(\lambda))\cdot L}) & \text{otherwise,} \end{cases}$$

$$P_{\text{up/down}}(\lambda) = \begin{cases} 1 - P_{\text{absorption}} - P_{\text{scattering}} & \text{if the current state is up or down,} \\ \frac{1-P_{\text{absorption}}-P_{\text{scattering}}}{2} & \text{if the current state is scattered,} \\ 0 & \text{if the current state is absorbed,} \\ 0.98 & \text{if the current state is illumination,} \end{cases}$$

where

- $P_{\text{direct reflection}}$ represents the probability of photon moving to state direct reflected,
- $P_{\text{absorption}}$ represents the probability of photon moving to state absorbed (note that the photon can move to this state only from the same layer),
- $P_{\text{scattered}}$ represents the probability of photon moving to state scattered (note that the photon can move to this state only from the same layer),
- $P_{\text{up/down}}$ represents the probability of photon moving from one layer to another,
- λ is wavelength in nanometers,
- $a(\lambda) = \sum_n a_i(\lambda)c_i$ is the absorption coefficient,
- $s(\lambda) = s(500\,\text{nm}) \cdot f_{\text{Ray}} \left(\frac{\lambda}{500\,\text{nm}}\right)^{-4} + (1 - f_{\text{Ray}}) \left(\frac{\lambda}{500\,\text{nm}}\right)^{-b_{\text{Mie}}}$ is the reduced scattering coefficient,
- L is the length of the light path, which is assumed to be the thickness of the layer,
- a_i are the chromophore (i.e. part of a molecule responsible for its color) absorption coefficients,
- c_i are the chromophore concentrations,
- $s(500\,\text{nm})$ the measured reduced scattering coefficient at 500 nm,
- f_{Ray} fraction of the Rayleigh scattering, and
- b_{Mie} the Mie scattering power.

These parameters and their definitions are for skin modelling, and we used slightly different definitions in article PIV, where the stochastic model for leaf optical properties (SLOP) is used.

The chromophores used in the skin studies PV and PVI were melanin, deoxygenated and oxygenated hemoglobin and water, and additional parameters

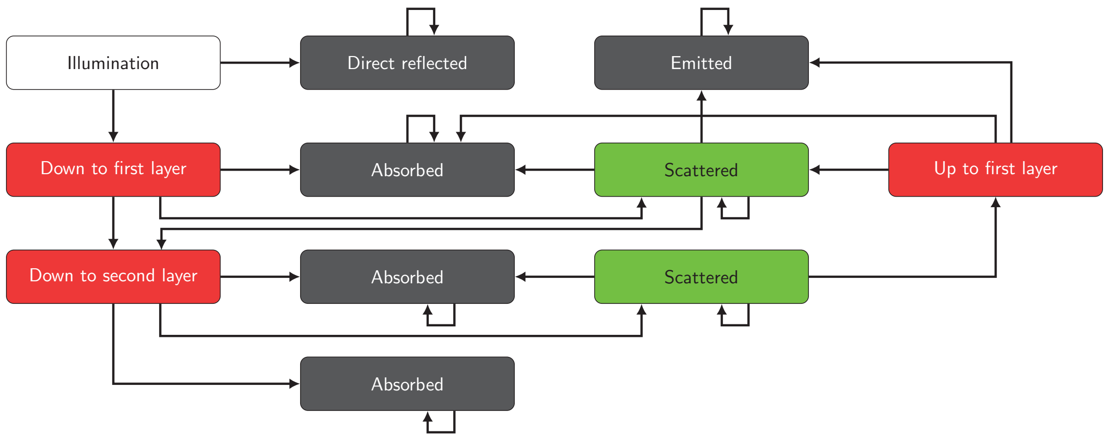


FIGURE 2 Network of states and transitions of light propagation in the Stochastic Model. Adapted from PVI.

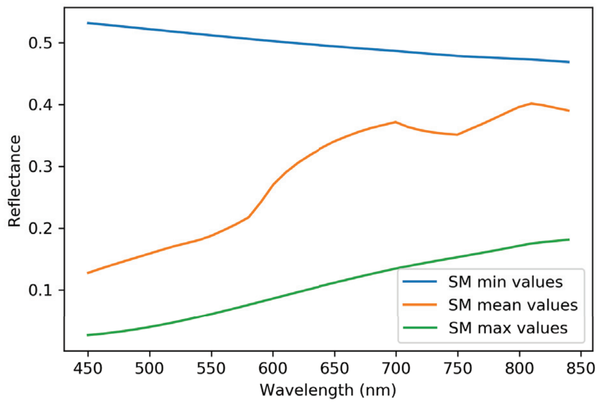
were scattering and absorption parameters and skin layer thicknesses. For the forestry study in PIV the chromophores were chlorophyll *a* and *b*, *β*-carotene, lutein, violaxanthin, neoxanthin and water. Additionally, scattering coefficient and probability of direct reflection were varied and chloroplast diameter and concentration and leaf layer thicknesses were handled as constants. Example spectra produced by the stochastic models is in Figure 3.

This stochastic model can be used in a direct way to produce spectra based on the input parameters. It can also be inverted by various methods, for example using machine learning. After inversion, the machine learning methods takes an input of a spectrum, and uses that input to produce information of the stochastic model input parameters. For example in PIV the inverted model is used to estimate chlorophyll concentrations from hyperspectral drone imagery. This is called physical parameter retrieval and is described in more detail in section 2.3.2.

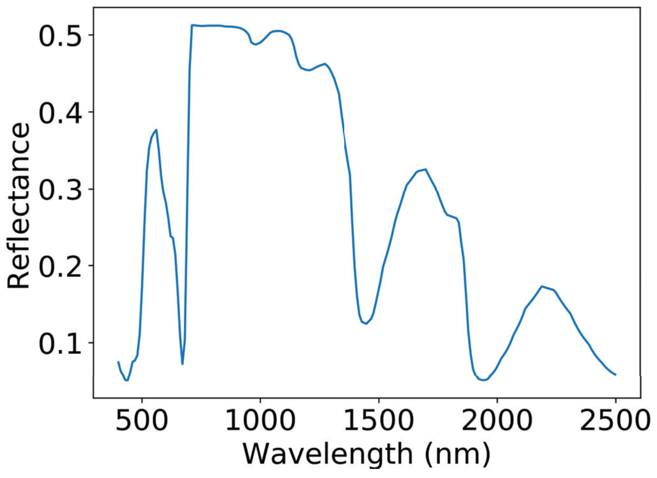
2.3 Machine learning problems in hyperspectral imaging

Machine learning problems can be divided in to four categories; supervised and unsupervised machine learning, semi-supervised learning, and reinforcement learning (Kubat, 2017). In hyperspectral image analysis the machine learning tasks can be divided into categories of classification, detection, spectral unmixing and parameter estimation (Gewali et al., 2019). Of these the scope of this work is mainly on the classification and parameter estimation problems, for which mainly supervised machine learning methods are used.

The problems can be also classified by field. This classification includes remote sensing, biomedical applications, food and agriculture, and other applications (Signoroni et al., 2019). Using this classification, the work at hand is in the fields of remote sensing and biomedical applications. Signoroni et al. (2019) also adds data enhancement to the list of tasks by Gewali et al. (2019). In this section,



(a)



(b)

FIGURE 3 (a) Example spectra from skin stochastic model. Figure from PVI. (b) Example spectra from leaf stochastic model. Figure from PIV.

the reader will be provided sufficient knowledge for machine learning tasks for understanding the articles the author has written.

2.3.1 Hyperspectral classification problems

Classification is conceptually one of the easiest to understand among machine learning tasks. The idea is to take data and classify it according to pre-determined labels. This is called training of a machine learnig algorithm. Mathematically the idea is to minimize a loss function, which with well defined loss function means that the the desired metric, usually accuracy, of the resulting trained algorithm is the best possible with the used input data.

In hyperspectral imaging, the classification can be done either by classifying the image pixel by pixel or by using a spatial-spectral approach (Gewali et al., 2019; Signoroni et al., 2019). In pixel by pixel approach, only the spectral domain affects the classification, and in spatial-spectral approach, the pixel's surroundings with spatial features are taken into account. The author's articles PII and PIII are about hyperspectral data classification. In both spatial-spectral approach is utilized.

2.3.2 Biophysical parameter retrieval

Biophysical parameter retrieval is a subclass of hyperspectral regression problems. In biophysical parameter retrieval, the machine learning algorithm is trained similarly to classification algorithm. Only the loss metric is different and the label is a value that represent some parameter of the imaged object. The goal is to estimate the value of the parameter, such as leaf chlorophyll content or skin melanin content.

According to Verrelst et al. (2018), the parameter estimation strategies can be divided into four categories, which are in order of complexity:

1. Parametric regression methods,
2. Nonparametric regression methods,
3. Physically based model inversion methods, and
4. Hybrid regression methods.

Of these, the first category generally means spectral indices, such as NDVI (Pontailler et al., 2003; Stenberg et al., 2004) or TCARI/OSAVI (Haboudane et al., 2002) and similar methods that are calculated directly from the hyperspectral data. The second category is generally machine learning with direct measurements of the parameters as labels, as used in (Lazaridis et al., 2011; Feilhauer et al., 2017). Third category means mathematically modelling the result, like in with PROSAIL for tree canopies (Jacquemoud et al., 2009). Fourth category means combining some of the previous categories in one model, such as using previously defined stochastic model in producing training data for a machine learning algorithm and using the trained algorithm for the estimation. This is the paradigm used in articles PIV, PV and PVI. Other examples of similar methodology can be found in (Rivera-Caicedo et al., 2017; Malenovskỳ et al., 2013)

2.3.3 Hyperspectral data augmentation

Hyperspectral data augmentation is another interesting problem class (Nalepa et al., 2019), where the amount of usable data is increased by modifying the existing data or manufacturing synthetic data. The desire to use data augmentation rises from the data gathering being a laborious process and the data size being in orders of gigabytes per captured hyperspectral image. The augmentation process is used to spend minimal amount of time in data gathering and to minimize the amount of original data needed.

The traditional ways of reusing the data are normal transformations of the used data, such as flipping an image upside down or mirroring it. One can 8-fold the data with the eight symmetry axels of rectangle shaped image data. There are also more interesting new ways to go about it, like generative adversarial neural network (Goodfellow et al., 2014) used in article PVII.

The idea of the generative adversarial neural network is to have two competing neural networks. The generator network's objective is to fool discriminator network to believe the data is from the training dataset, which has been introduced to the discriminator beforehand, but not to the generator. Mathematically the objective is achieved by using the true/false labels produced by discriminator to change the discriminator and generator losses to the opposite directions. When trained, the generator can be used for producing data similar to the training data in the dataset.

2.4 Convolutional neural network

The main machine learning algorithm used is this work is the convolutional neural network (CNN). It was invented in its current form by Yann LeCun, who in his article "Backpropagation applied to handwritten zip code recognition" (1989), thought of combining previously known hand tuned convolutional neural network by Fukushima and Miyake (1982) and backprobagation algorithm studied previously by Linnainmaa (1970), Werbos (1974) and Rumelhart et al. (1986). The resulting network correctly updated its convolution kernels by itself–without human intervention.

This section is structured as follows: First we take a look at the intuitional explination why CNN works well in image analysis tasks. Then we introduce the neural network (NN), also known as multi-layer perceptron (MLP), and go through the backpropagation process with an example. After that we look at the mathematical definition of convolution and lastly we introduce the CNN with an example.

2.4.1 Convolutional neural network – Intuition

The usefulness of convolutional neural network (CNN) in image and signal analysis is easy to understand by looking at convolution in a small example. Let us consider the image and convolution kernels in Figure 4a. If we want to look for features described in the convolution kernels, we can calculate the convolutions and see the results in Figure 4b. The resulting *feature maps* are brightest where the searched feature was most likely to be found, which is the intuition behind CNN. By reducing the image to the searched features, the machine learning tasks become much simpler, because the features act as a link between the individual pixels and the label. As already mentioned, in the CNN these feature kernels, officially known as convolution kernels, are automatically formed through the training process.

The intuition holds true. If we look at state-of-the-art object recognition algorithms (Xie et al., 2020; Touvron et al., 2020), they have been based on convolutional neural network since AlexNet by Krizhevsky et al. (2012). CNN has also been successfully used in various non-image applications, for example in signal analysis tasks (Jafarzadeh et al., 2019), natural language processing (Collobert and Weston, 2008) and time series forecasting (Borovykh et al., 2017). It was also used in Google's AlphaGo Go program in evaluating the best moves in the game (Silver et al., 2016).

2.4.2 Neural Network

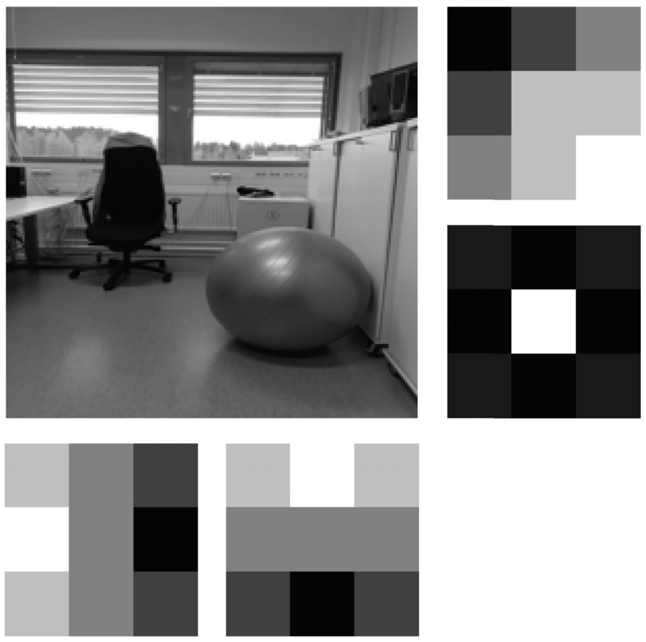
Mathematically, a neural network is a network of nodes and their connections. The connections have weights, which are initially random values. Then the goal is to adjust the weights in the training phase so that the loss function is minimised. The loss function (error function) is always non-negative, and zero, when the true and predicted labels of the training dataset are the same. The usual way for minimizing the loss function is through gradient descent with backpropagation as introduced by Amari (1993).

Let us discuss an example of a neural network. Let us assume the network has the same architecture as in the Figure 5. The first step of training a neural network is *forward pass*. The input is forwarded to the first hidden layer where the input of a node is the sum of the input connections multiplied by the output value in the connected input node

$$h_1^{in} = i^{out} \cdot w_{i1}, \tag{1}$$

$$g_1^{in} = \sum_{k=1}^{3} h_k^{out} \cdot w_{k1}. \tag{2}$$

If the nodes have bias, as introduced by Blum (1989); Lisboa and Perantonis (1991) in order to solve the XOR-problem with neural networks, a constant bias is added


(a) Image at top left to be convoluted with the convolution kernels. Top right seeks left facing corners from the image, bottom right is sharpening kernel, bottom left seeks vertical edges and bottom middle horisontal.

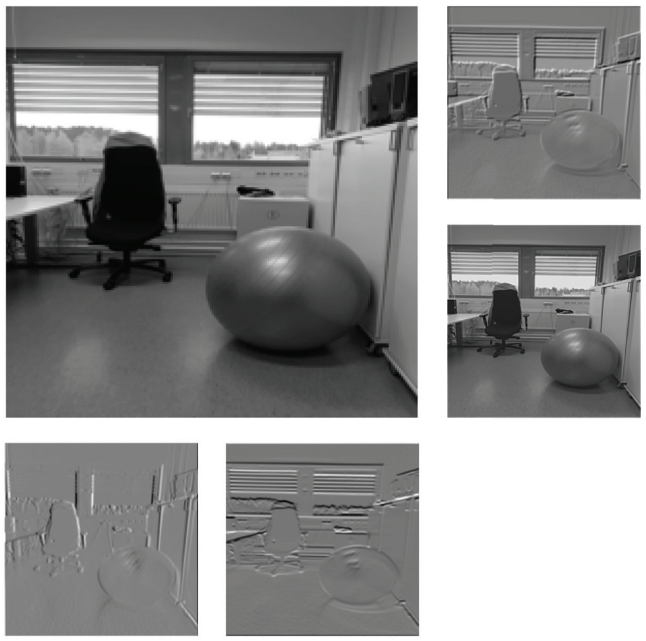

(b) Image at top left and its convolutions with kernels from Figure 4a.

FIGURE 4 Top: An image and convolution kernels. Bottom: An image and convoluted images.

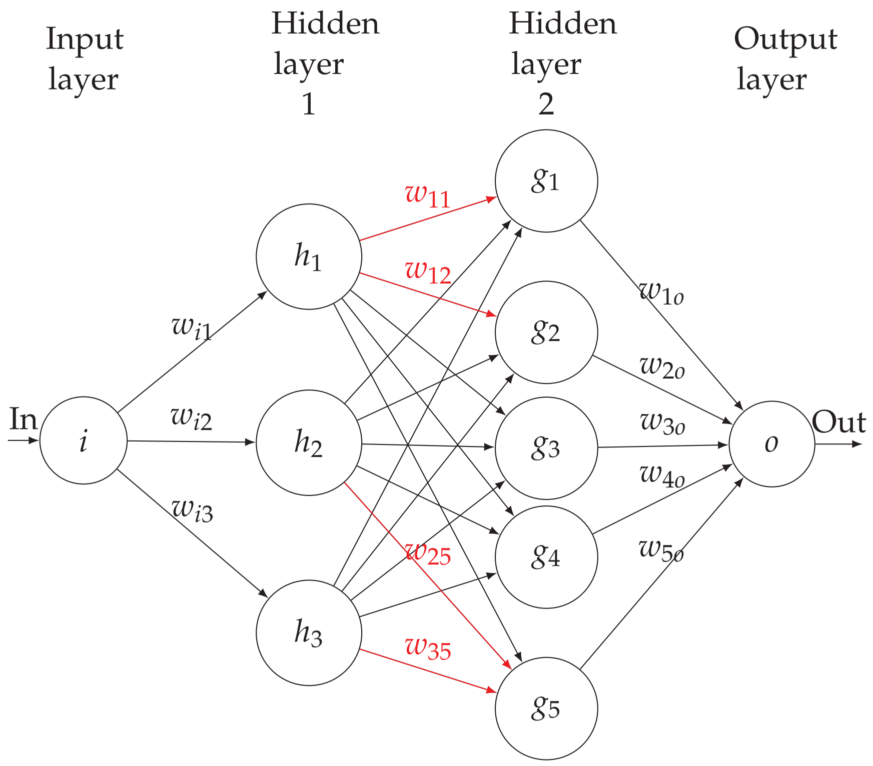


FIGURE 5 Example of a fully connected deep neural network with one input and one output, and two hidden layers with 3 and 5 nodes.

to the Equations 1 and 2

$$h_1^{in} = i^{out} \cdot w_{i1} + b_{h_1}, \tag{3}$$

$$g_1^{in} = \sum_{k=1}^{3} h_k^{out} \cdot w_{k1} + b_{g_1}. \tag{4}$$

The output of the node is the input that goes through activation function σ, such as rectified linear unit (reLU) (Nair and Hinton, 2010), sigmoid (Kilian and Siegelmann, 1993), or softmax (Bridle, 1990)

$$h_1^{out} = \sigma(h_1^{in}) \tag{5}$$

Finally, we arrive at the output layer, for which the input is calculated similarly to Equation 2 and output similarly to Equation 5

$$o^{out} = \sigma o^{in} = \sigma \left(\sum_{k=1}^{5} g_k^{out} \cdot w_{ko} \right). \tag{6}$$

Now, we can compare the training data to the output data by calculating the loss function. Let us use *mean square error* in this example. Mean square error in the case of one output is just square error (Eq. 7). In the case there are multiple outputs it is the mean of the squared errors (Eq. 8):

$$L = (o^{out} - o^{exp})^2, \tag{7}$$

$$L = \frac{1}{n} \sum_{k=1}^{n} (o_k^{out} - o_k^{exp})^2. \tag{8}$$

In order to update the weights, we need to know each node contributes to the error propagation in the loss function. This is called backpropagation. According to the law of error propagation (Birge, 1939), the contribution is the partial derivative of loss function by the weight. For weight w it is $\frac{\partial L}{\partial w}$. The actual update is in Eq. 9, where l is the learning rate, which determines how fast the weights are allowed to change in gradient descent algorithm

$$w^{new} = w^{old} - l \left(\frac{\partial L}{\partial w} \right). \tag{9}$$

In the usual case when there are more than one data point in the training data, the update in Eq. 9 is the average of the batch of partial derivatives (Eq. 10)

$$w^{new} = w^{old} - l \left(\frac{1}{n} \sum_{k=1}^{k=n} \frac{\partial L_k}{\partial w} \right). \tag{10}$$

Now, let us calculate the partial derivative for w_{1o}. First we need to represent L as a function of w_{1o}:

$$\begin{aligned} L &= (o^{out} - o^{exp})^2 \\ &= (\sigma(o^{in}) - o^{exp})^2 \\ &= (\sigma \left(\sum_{k=1}^{5} g_k^{out} \cdot w_{ko} \right) - o^{exp})^2. \end{aligned}$$

Let us denote $f(x) = (\sigma(x) - o^{exp})^2$, and the L simplifies as

$$\begin{aligned} L &= f\left(\sum_{k=1}^{5} g_k^{out} \cdot w_{ko}\right) \\ &= f\left(g_1^{out} \cdot w_{1o} + \sum_{k=2}^{5} g_k^{out} \cdot w_{ko}\right) \\ &= f(g(w_{1o})), \end{aligned}$$

where $g(x) = g_1^{out}x + \sum_{k=2}^{5} g_k^{out} \cdot w_{ko}$. Now the partial derivative of this is easy to calculate through the chain rule:

$$\frac{\partial L}{\partial w_{1o}} = f'(g(w_{1o})) \cdot g'(w_{1o}).$$

Now, the derivative of g is easy to calculate:

$$g'(w_{1o}) = g_1^{out},$$

as the weights and node outputs in the sum are not functions of w_{1o}. The derivative of f on the other hand must be calculated again using chain rule: let us say $h = (x + o^{exp})^2$. Now f can be written as

$$\begin{aligned} f(x) &= h(\sigma(x)), \text{ and} \\ f'(x) &= h'(\sigma(x))\sigma'(x). \end{aligned}$$

Now we can put it all together:

$$\begin{aligned} \frac{\partial L}{\partial w_{1o}} &= f'(g(w_{1o})) \cdot g'(w_{1o}) \\ &= h'(\sigma(g(w_{1o}))) \cdot \sigma'(g(w_{1o})) \cdot g'(w_{1o}) \\ &= h'(\sigma(o^{in})) \cdot \sigma'(o^{in}) \cdot g_1^{out} \\ &= 2 \cdot (o^{out} - o^{exp}) \cdot \sigma'(o^{in}) \cdot g_1^{out}. \end{aligned}$$

We did not set the activation function σ for this example, but they are usually simple to differentiate. For example with logistic activation (Elliott, 1993) the function and derivative are

$$\begin{aligned} \sigma(x) &= \frac{1}{1 + e^{-x}}, \text{ and} \\ \sigma'(x) &= \frac{e^x}{(1 + e^x)^2} = \sigma(x)\sigma(-x). \end{aligned}$$

Then the partial derivative would come to

$$\frac{\partial L}{\partial w_{1o}} = 2 \cdot (o^{out} - o^{exp}) \cdot \sigma(o^{in}) \cdot \sigma(-o^{in}) \cdot g_1^{out}.$$

Similar calculation would be done for each of the weights and biases, and the update would be calculated according to Eq. 10. The current neural network implementations use automatic differentation for backpropagation (Baydin et al., 2017), which means that for any computational graph, such as neural network, the backpropagation can be automatically computed.

2.4.3 Convolution

This section is based on "Deep Learning Book" Chapter 9.1 by Goodfellow et al. (2016). Convolution is defined as follows:

$$(f * g)(x) = \int f(t)g(t-x)dt. \tag{11}$$

However, in computer science, the computation is discrete, which leads to discrete convolution, described as

$$(f * g)(x) = \sum_{t=-\infty}^{\infty} f(t)g(t-x). \tag{12}$$

As we can see, in the Equation 12 the integral of the Equation 11 is replaced with summation. Now this can be extended to multiple dimensions if needed:

$$(f * g)(x,y) = \sum_{t=-\infty}^{\infty} \sum_{u=-\infty}^{\infty} f(t,u)g(t-x,u-y) \tag{13}$$

In Equations 11 – 13 the function f is the convoluted function, and function g is called *kernel*. Now, as the input data in machine learning application is always both finite and discrete, functions f and g can be represented as finite vector in one dimension, matrix in two dimensions or tensor in higher dimensions. Now, equation 12 can be written as

$$(f * g)(x) = \sum_{t \in I} f(t)g(t-x) = \sum_{u \in K} f(x+u)g(u), \tag{14}$$

where I denotes the input data, and K the kernel. This final form is very close to the convolution used in deep learning libraries such as Tensorflow (Abadi et al., 2016).

2.4.4 Convolutional neural network

CNN differs from normal deep neural network by the forward pass method. While in dense neural network every node in a layer affects the input of the next layer's nodes, in CNN the nodes far from each other have no cross-effect on the next layer.

Let us take a look at an one dimensional neural network described in Figure 6. It consist of input layer, convolutional layer, pooling layer and output layer. Let us look at them one by one. First the convolution layer calculates the convolution for the input layer. Let us assume the input is

$$[x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9]$$

and the original convolution kernel is randomly chosen to be

$$[w_1, w_2].$$

Now, the output of convolutional layer, before the activation function is

$$\left[\sum_{i=1}^{2} w_i \cdot x_i, \sum_{i=1}^{2} w_i \cdot x_{i+1}, \sum_{i=1}^{2} w_i \cdot x_{i+2}, \sum_{i=1}^{2} w_i \cdot x_{i+3}, \right.$$
$$\left. \sum_{i=1}^{2} w_i \cdot x_{i+4}, \sum_{i=1}^{2} w_i \cdot x_{i+5}, \sum_{i=1}^{2} w_i \cdot x_{i+6}, \sum_{i=1}^{2} w_i \cdot x_{i+7}\right].$$

Let us assume, for simplicity, that our activation function is linear, so the output after activation is the same as input for activation. Next we have pooling layer, which is intended to drop some of the data in a controlled way in order to further reduce the dimensionality of the network (Passricha and Aggarwal, 2020). For example, here we use max pooling with kernel size two and stride two, which means we take the first two inputs to the layer and choose the maximum of the two, and then take the next two values and take the maximum of the two and so forth. The output of the layer is

$$\left[\max\left(\sum_{i=1}^{2} w_i \cdot x_i, \sum_{i=1}^{2} w_i \cdot x_{i+1}\right), \max\left(\sum_{i=1}^{2} w_i \cdot x_{i+2}, \sum_{i=1}^{2} w_i \cdot x_{i+3}\right), \right.$$
$$\left. \max\left(\sum_{i=1}^{2} w_i \cdot x_{i+4}, \sum_{i=1}^{2} w_i \cdot x_{i+5}\right), \max\left(\sum_{i=1}^{2} w_i \cdot x_{i+6}, \sum_{i=1}^{2} w_i \cdot x_{i+7}\right)\right]$$

The output layer is just a normal dense layer, so the input (and output after linear activation) is

$$w_3 \cdot \max\left(\sum_{i=1}^{2} w_i \cdot x_i, \sum_{i=1}^{2} w_i \cdot x_{i+1}\right) + w_4 \cdot \max\left(\sum_{i=1}^{2} w_i \cdot x_{i+2}, \sum_{i=1}^{2} w_i \cdot x_{i+3}\right) +$$
$$w_5 \cdot \max\left(\sum_{i=1}^{2} w_i \cdot x_{i+4}, \sum_{i=1}^{2} w_i \cdot x_{i+5}\right) + w_6 \cdot \max\left(\sum_{i=1}^{2} w_i \cdot x_{i+6}, \sum_{i=1}^{2} w_i \cdot x_{i+7}\right)$$

Now the backpropagation is similar to the non-convolutional neural network, but due to weight sharing in the convolutional layer, the arithmetic is slightly more difficult. Let us consider the contribution of weight w_1 to the loss L:

$$\frac{\partial L}{\partial w_1}.$$

By chain rule this can be expressed as

$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial y_{out}} \frac{\partial y_{out}}{\partial w_1},$$

where loss function L is defined as

$$L = (y_{out} - y_{exp})^2.$$

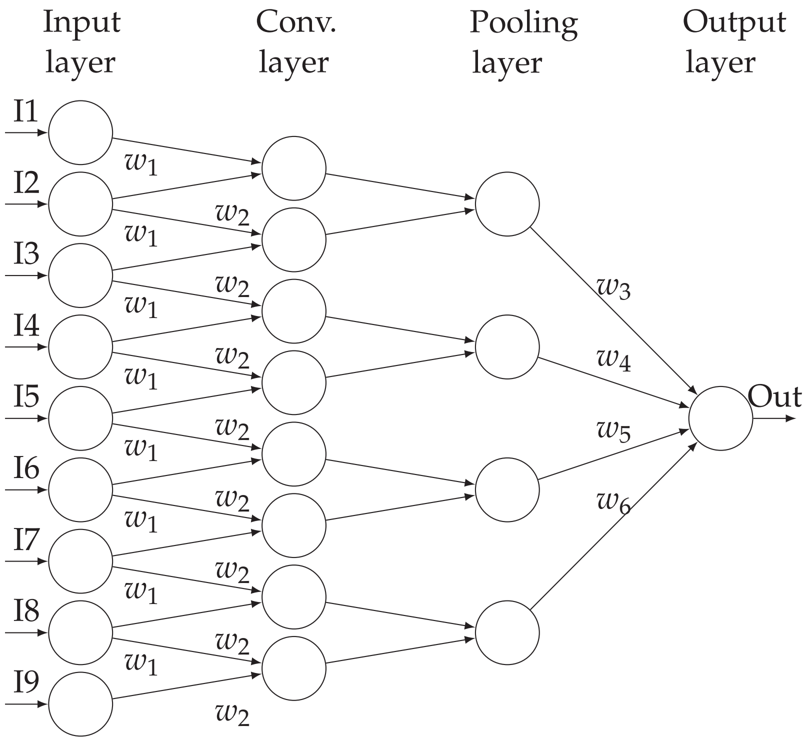


FIGURE 6 Example of (one dimensional) convolutional neural network that has nine inputs, convolution with one kernel of size two, pooling layer of size two and one output. Usually a fully connected part is added after enough convolution-pooling combinations. Note that the amount of connections is lower than if all layers would be fully connected. This means that in the machine learning training phase, there is less parameters to optimise.

Now, y_{out} can be expressed as a function of previous layer:

$$y_{out} = \sum_{i=1}^{4} w_{i+2} p_i(w_1),$$

where p_i is output of pooling layer, which are functions of w_1. Now we can use the chain rule again:

$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial y_{out}} \frac{\partial y_{out}}{\partial w_1} = \frac{\partial L}{\partial y_{out}} \sum_{i=1}^{4} \frac{\partial y_{out}}{\partial p_i} \frac{\partial p_i}{\partial w_1}.$$

Again, p_i can be further expanded:

$$p_i = \max\left(c_j(w_1), c_{j+1}(w_1)\right),$$

where $j = (i-1) \cdot 2 + 1$ and c_j is the output of the convolution layer. We use the chain rule and get

$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial y_{out}} \sum_{i=1}^{4} \frac{\partial y_{out}}{\partial p_i} \frac{\partial p_i}{\partial w_1} = \frac{\partial L}{\partial y_{out}} \sum_{i=1}^{4} \frac{\partial y_{out}}{\partial p_i} \sum_{j=(i-1)\cdot 2+1}^{(i-1)\cdot 2+2} \frac{\partial p_i}{\partial c_j} \frac{\partial c_j}{\partial w_1}.$$

Now, when we remember that $c_j = w_1 x_j + w_2 x_{j+1}$, where x_j are inputs to the network, we can calculate all the partial derivatives in the chain:

$$\begin{aligned}
\frac{\partial L}{\partial y_{out}} &= 2 \cdot (y_{out} - y_{exp}) \\
\frac{\partial y_{out}}{\partial p_i} &= w_{i+2} \\
\frac{\partial p_i}{\partial c_{(i-1)\cdot 2+1}} &= \begin{cases} 1, & \text{if } c_{(i-1)\cdot 2+1} \geq c_{(i-1)\cdot 2+2} \\ 0, & \text{otherwise} \end{cases} \\
\frac{\partial p_i}{\partial c_{(i-1)\cdot 2+2}} &= \begin{cases} 1, & \text{if } c_{(i-1)\cdot 2+1} < c_{(i-1)\cdot 2+2} \\ 0, & \text{otherwise} \end{cases} \\
\frac{\partial c_j}{\partial w_1} &= x_j.
\end{aligned}$$

Now, the only difference a different activation function would make is that one would have to calculate more partial derivatives. The remaining steps are calculating the backpropagated errors for the other weights as well, and calculating the update to the weights with Equation 10.

The useful CNN's would usually include more convolutional and pooling layers, a few dense layers at the end, and some dropout layers to battle overfitting (Srivastava et al., 2014). Depending on application, the convolution layers can be one, two or three dimensional. Building even higher dimensional networks are entirely possible, but the scientific community has not yet found use for such networks. The purpose of this example was to familiarise the reader with basic principles of training a convolutional neural network.

3 RESEARCH RESULTS AND AUTHOR CONTRIBUTION

3.1 PI: Practical Approach for Hyperspectral Image Processing in Python

3.1.1 Results

This article identified the tools that were used throughout the project: scikit-learn (Pedregosa et al., 2011), Tensorflow and keras (Abadi et al., 2016) and xarray (Hoyer and Hamman, 2017). The article also showed that the Python programming language ecosystem is sufficient for our line of work.

3.1.2 Author contribution

In this paper the author did most of the research on the Python packages in collaboration with Matti Eskelinen and Jyri Hämäläinen, wrote the draft and final versions of the article and wrote 2/3 of the original software introduced (MaskAccessor and VisualisorAccessor). The author also translated one algorithm from Matlab to Python (3D histogram (Eskelinen, 2017), part of VisualisorAccessor) and was a contributor in the spectral indices library, which was mainly done by Matti Eskelinen and Aamos Riihinen.

3.2 PII: Tree Species Identification Using 3D Spectral Data and 3D Convolutional Neural Network

3.2.1 Results

This article establishes the three dimensional convolutional neural network (3DCNN) as an accurate and robust tool for tree classification in hyperspectral forest imagery. It also concludes that another benefit of using the 3DCNN in classification is the automation of time consuming feature extraction and selection.

3.2.2 Author contribution

In this article, the author was a part of the team (with Ilkka Pölönen and Samuli Rahkonen) that was resposible of producing the neural networks that was used to obtain the results. The author also proofread the original draft and provided comments to increase the quality of the manuscript.

3.3 PIII: Convolutional neural networks in skin cancer detection using spatial and spectral domain

3.3.1 Results

The results of this article were promising, even though the authors had very small dataset (n=61) to work with. The results show that when the convolutional neural networks with different amount of dimensions (1, 2 or 3) can be used together to improve the classification. The positive predictive value of the classifiers matched the clinical accuracy of specialized medical professionals, even though the sensitivities were poor. The results show promise that is still, at the time of writing, to be verified with a larger dataset.

3.3.2 Author contribution

In this article, the author was a part of the team (with Ilkka Pölönen and Samuli Rahkonen) that was resposible of producing the neural networks that was used to obtain the results. The author also proofread the original draft and provided comments to increase the quality of the manuscript.

3.4 PIV: Chlorophyll Concentration Retrieval by Training Convolutional Neural Network for Stochastic Model of Leaf Optical Properties (SLOP) Inversion

3.4.1 Results

This article found the one dimensional convolutional neural network a good way to invert the stochastic model for leaf optical properties (SLOP) by Maier et al. (1999). The correlation coefficients between predicted and SLOP input values were consistently high. With empirical data, the CNN chlorophyll concentration predictors correlated well with a more traditional way of estimating the chlorophyll concentration from hyperspectral image. Also the chlorophyll *a/b* of the predictions were within the expected range.

The results indicate that there is a way to build a remote tree health monitor based on hyperpsectral imaging, convolutional neural networks, and stochastic modelling.

3.4.2 Author contribution

In this article, the author was the main author. The author chose the studied models and algorithms, designed the experiments, wrote the original manuscript draft, edited the article based on co-author and review comments, and wrote the final version of the article.

3.5 PV: Kubelka-Munk Model and Stochastic Model Comparison in Skin Physical Parameter Retrieval

3.5.1 Results

This article establishes the skin stochastic model for light scattering in layered media, developed by the author extending the SLOP model from PIV, as a worthy object for further research. The leaf to skin modification from SLOP and other sources is successful, as the spectrum is similar to spectra provided by Kubelka-Munk model and measured skin spectra. It can also be successfully inverted using convolutional neural network.

3.5.2 Author contribution

In this article the author the main author. The author chose the studied models and algorithms, designed the experiments, wrote the original manuscript draft, edited the article based on co-author and review comments, and wrote the final version of the article.

3.6 PVI: Comparison of Machine Learning Methods in Stochastic Skin Optical Model Inversion

3.6.1 Results

This article compared the convolutional neural network to many other algorithms in inversion of stochastic model developed in PV. The CNN was the most accurate based on all used metrics.

3.6.2 Author contribution

The subject of this study was suggested to the author by his supervisor and co-author Ilkka Pölönen. Sami Äyrämö suggested some of the used algorithms and metrics they were evaluated with. Other than that the research design and article preparation was conducted by the author.

3.7 PVII: Generating Hyperspectral Skin Cancer Imagery using Generative Adversarial Neural Network

3.7.1 Results

This article found that using generative adversarial neural network (GAN) is a prospective way to produce/augment hyperspectral data for machine learning purposes. The results show that with very little effort GAN can produce something similar to hyperspectral data. With more effort in the designing of the GAN and defining the goals of the generator and discriminator, and with more specialized data, the produced data could become more useful in the machine learning applications. This remains an open research subject for the future.

3.7.2 Author contribution

This article was the authors idea. The data gathering and labeling were done by Noora Neittaanmäki, John Paoli, and Oscar Zaar, and the author used their data to conduct his experiments. The research design was the authors task as well as the writing of the article and editing based on co-author and review comments. The data gathering process was not originally for this article, and Ilkka Pölönen was responsible for combining the article idea with suitable data.

3.8 Summary of the results

The results are in line with the research questions. In PI the tools used in the other articles were selected and evaluated. In PII and PIII the convolutional neural network was succesfully used in hyperspectral classification problems. In PIV the SLOP model was found out to be of the type that inverts well with convolutional neural network. It is also established that the predictions from measured hyperspectral data correlate with the ground truth.

PV discovered that the SLOP model can be transferred to skin model by adapting the transfer probabilities and using skin pigments. The resulting model compares well with the Kubelka-Munk model and measured skin spectra. It is also found that the resulting stochastic model is invertible for the most important parameters of the model. PVI compared the convolutional neural network to other machine learning methods and discovered it to be the best tool for inverting the stochastic model.

PVII found that there is great hope for future research in using GAN to produce hyperspectral data for different purposes.

4 DISCUSSION

One of the main themes in this thesis is data augmentation. The data gathering process for hyperspectral imaging physical parameter retrieval is a resource intensive process. In addition to the hyperspectral data one needs a sufficient dataset of corresponding measurements of the desired parameters. This often requires highly trained work force that can be divided into three categories:

1. Sample gatherers: in skin cancer research these are the nurses and clinical doctors who take the samples and hyperspectral images. In forestry they are drone/satellite operators and forest care takers.
2. Laboratory workers: pathologists in skin cancer research and biologist in forestry.
3. Machine learning specialists: people working on the data.

Group one does the field work and produces spectra, group two does the laboratory work and produces measurements. Group three the builds the model to estimate one from another. Of course these groups can be intermixed, but due to the extreme specialisation needed for the tasks, this is rare. Data augmentation reduces the work required from the first two categories if the a data similar enough can be produced.

The stochastic model for light scattering in complex media (PIV, PV and PVI) seems to perform fairly well in the physical parameter retrieval tasks. The model described in PIII, which had already been verified in terms of accuracy in leaf spectrum production by Maier et al. (1999), showed consistency also in the canopy level when the results of the chlorophyll concentration estimation were compared to TCARI/OSAVI (Haboudane et al., 2002), a spectral index that is known to correlate well with chlorophyll concentration. The results are similar to previous research by Croft et al. (2015); Atzberger et al. (2003), although they are directly comparable to the research scheme used in PIV.

In PV and PVI, similar index comparison was not available since most of the known spectral indices are developed for remote sensing tasks and vegetation in mind. However, visual comparison with measured hyperspectral data suggested that the inverted stochastic model is viable. The lesions were distinguishable from the feature maps and the values seem to correlate as expected. At least the melanin concentration was higher where the skin was darker, as is expected. In PV the stochastic model was compared to more traditional Kubelka-Munk model (Jolivot et al., 2013). The comparison showed no reason to doubt the accuracy and usefulness of the stochastic model, even though they should be further verified. Also the generative adversarial neural network showed promise in the data augmentation task (PVII).

Another contribution of this research is further establishing CNN in hyperspectral classification and regression tasks. In article PIII we showed that even with relatively low amount of data we can match the clinical diagnosis on a difficult task of skin cancer classification with CNN. In PII we showed that the three dimensional CNN is a useful tool in the classification task. We achieved greater accuracy compared to the previous research on the same dataset by Nevalainen et al. (2017), with less preprocessing necessary. In PIII we used CNN to achieve similar accuracy to clinical examination by a doctor (Heal et al., 2008).

In PIV and PV we used one dimensional CNN in stochastic model inversion and the accuracy was good. The correlation results were slightly inferior to results by Vyas et al. (2013, 2015). It may be due to the difference in used mathematical models, as they used Kubleka-Munk modelling (Doi and Tominaga, 2003; Jolivot et al., 2013) in their research. The one and three dimensional CNN's are still less known compared to normal two dimensional CNN. In PVI we compared the one dimensional CNN to other machine learning algorithms in stochastic model inversion. It outshone the other algorithms by every metric. In PVII we used three dimensional GAN, which is not yet much researched, and found it useful. The results in the spectral domain of the generated hyperspectral images were as encouraging as in the spatial domain.

Finally, the research done in PI listed the necessary tools for researcher in hyperspectral imaging field in Python. The field is still working mainly on Matlab, and while it is not inherently wrong, free open source tools are better for the accessibility of the research. In PI we listed some good data processing tools in Python, such as NumPy (Oliphant, 2006), rasterio (Gillies et al., 2013), xarray (Hoyer and Hamman, 2017) and GDAL (GDAL Development Team, 2018), as well as tools for data visualisation (Jupyter notebook Kluyver et al. (2016), matplotlib (Hunter, 2007), HoloViews (Rudiger et al., 2020)) and machine learning tools (tensorflow (Abadi et al., 2016) and scikit-learn (Pedregosa et al., 2011)). With these tools, one with some prior experience in hyperspectral imaging and programming can start to transfer towards more open scientific programming.

4.1 Limitations and further research

The research of article PI was not made in a systematic manner. This means we likely missed some useful tools. However, the packages found in the research have served us well. Some packages developed in this research have found very little use, but they served as a useful programming practice.

The article PIII suffered from lack of data, as did PVII. While the results of PVII showed promise, the results were not yet good enough for data augmentation purposes. The main pitfall of the article was that due to the lack of data the different skin lesion labels were mixed in the training data set, resulting in generated hyperspectral images that failed to represent any of them. The encouraging part of the results should be verified with a larger dataset specified on a single lesion type. The network design should also be revisited. The results in PIII are also expected to be better with larger dataset.

The stochastic model developed in PIV – PVI is one dimensional, therefore it does not provide additional help when using the spatial information of the hyperspectral images. It does not take into account the shape or structure of the leaves, canopy or lesions, and therefore the inverted model can not be used in assessing those parameters. Especially in the skin cancer research the shape of the lesion is important in determining the stage of the cancer (Abuzaghleh et al., 2014).

In further research, the stochastic model could be extended to three dimensions by combining it with a different model for the leaf, canopy or lesion structure. For canopy level forestry research, this could be achieved by combining the stochastic model with a radiative transfer simulator, such as Librat by Disney et al. (2000). One could also try to evaluate how the stochastic model parameters fluctuate with the structure.

The articles PIV – PVI also lacked verification datasets. In further research these leaf and skin parameters should be measured in order to verify the stochastic model inversion with the CNN.

5 CONCLUSION

In this thesis the author has provided the reader with fundamental understanding of convolutional neural network, hyperspectral imaging, and relating machine learning problems. The research shows practical implications in the fields of forestry and skin cancer research. The author has provided evidence of convolutional neural network being useful in the inversion of mathematical models of hyperspectral data, and that the inversion is accurate. The author also showed that the stochastic model he designed by adjusting an existing model to another field did not fail rudimentary tests for accuracy and usefulness. The author also experimented with generative adversarial neural networks and listed useful Python tools for hyperspectral image analysis.

YHTEENVETO (SUMMARY IN FINNISH)

Väitöskirjatyössä tutkittiin konvoluutioneuroverkkoa ja stokastista simulointia hyperspektrikuvien analysoinnissa. Stokastinen malli toimii hyvin. Aiemmassa tutkimuksessa julkaistu lehden optisten ominaisuuksien stokastinen malli (SLOP), jota käytettiin artikkelissa PIII oli yhtenevä verrattuna spektri-indeksiin, joka korreloi vahvasti klorofyllin määrän kanssa. Artikkelissa PV malli sovitettiin ihosyöpätutkimukseen eri lähteistä löydettyjen ihon parametrien avulla. Näin saatua mallia verrattiin yleisesti käytössä olevaan Kubelka-Munk mallin tuottamaan spektriin ja ihon spektriin. Molemmista malleista tehtiin myös käänteismallit konvoluutioneuroverkon avulla ja artikkelissa PVI verrattiin eri koneoppimismenetelmiä käänteismallien tuottamisessa. Näiden tutkimusten perusteella on syytä olettaa, että stokastista mallintamista kannattaa jatkossakin tutkia, ja jatkotutkimuksessa painopisteenä voisi olla mallin toimivuuden varmistaminen mitatulla datalla ja mallin laajentaminen kolmeulotteiseksi. Myös generatiivinen kilpaileva neuroverkko näytti lupaavalta hyperspektridatan lisäämisessä.

Työn toinen tuotos oli konvoluutioneuroverkon käytön vakiinnuttaminen hyperspektrikuvantamisen luokittelu- ja regressio-ongelmissa. Artikkelissa PIII näytettiin, että käytettäessä konvoluutioneuroverkkoa verrattain vähäisellä datan määrällä päästään ihotautilääkärin kliinisen diagnoosin kanssa likimain samaan tulokseen ihosyöpäkasvaimien luokittelussa. Artikkelissa PII näytettiin, että kolmiulotteinen konvoluutioneurkko on hyödyllinen hyperspektrikuvien luokittelussa. Saavutettu tarkkuus oli suurempi aiempaan samalla datalla tehtyyn tutkimukseen verrattuna, eikä dataa tarvinnut esikäsitellä yhtä paljon. Artikkeleissa PIV ja PV käytettiin yksiulotteista konvoluutioneuroverkkoa stokastisen mallin käänteismallin ratkaisemiseen. Näin saadun käänteismallin tarkkuus oli hyvä. Artikkelissa PVI verrattiin yksiulotteista konvoluutioneuroverkkoa muihin koneoppimismenetelmiin stokastisen mallin käänteismallin ratkaisemisessa. Konvoluutioneuroverkko oli kaikkien käytettyjen mittareiden mukaan paras. Artikkelissa PVII käytettiin kolmiulotteista generatiivista kilpailevaa verkkoa, jota ei olla vielä paljoa tutkittu verrattuna kaksiulotteiseen malliin. Kaksiulotteinen malli on itsessäänkin varsin tuore keksintö. Generatiivisen kilpailevan neuroverkon data näytti lupaavalta niin spatiaalisessa kuin spektrinsuuntaisessa ulottuvuudessa, joten kolmiulotteisen verkon käyttäminen näyttää perustellulta.

Lisäksi artikkelissa PI listattiin hyperspektrikuvantamisen Python-työkaluja. Alan tutkijat käyttävät edelleen yleisesti suljettua ja maksullista Matlab-ohjelmointikieltä, mikä ei sinänsä ole väärin, mutta avoimen lähdekoodin ja ilmaisten työkalujen käyttö lisää tutkimuksen saavutettavuutta. Työssä listattuja hyperspektridatan prosessointiin käytettäviä työkaluja ovat NumPy, rasterio, xarray ja GDAL. Datan visualisointiin käytettyjä työkaluja olivat Jupyter notebook, matplotlib ja HoloViews ja koneoppimiskirjastoja Tensorflow and scikit-learn.

REFERENCES

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. et al. 2016. Tensorflow: A system for large-scale machine learning. In 12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16), 265–283.

Abuzaghleh, O., Barkana, B. D. & Faezipour, M. 2014. Automated skin lesion analysis based on color and shape geometry feature set for melanoma early detection and prevention. In IEEE Long Island Systems, Applications and Technology (LISAT) Conference 2014. IEEE, 1–6.

Adão, T., Hruška, J., Pádua, L., Bessa, J., Peres, E., Morais, R. & Sousa, J. J. 2017. Hyperspectral imaging: A review on UAV-based sensors, data processing and applications for agriculture and forestry. Remote Sensing 9 (11), 1110.

Amari, S.-i. 1993. Backpropagation and stochastic gradient descent method. Neurocomputing 5 (4-5), 185–196.

Angelopoulou, E. 2001. Understanding the color of human skin. In Human Vision and Electronic Imaging VI, Vol. 4299. International Society for Optics and Photonics, 243–251. doi:10.1117/12.429495.

Atzberger, C., Jarmer, T., Schlerf, M., Kötz, B. & Werner, W. 2003. Retrieval of wheat bio-physical attributes from hyperspectral data and SAILH+ PROSPECT radiative transfer model. In Proceedings of the 3rd EARSeL Workshop on Imaging Spectroscopy. Citeseer, 473–482.

Baranoski, G. V., Chen, T. F. & Krishnaswamy, A. 2015. Multilayer Modeling of Skin Color and Translucency. In Computational Biophysics of the Skin. Taylor & Francis.

Baydin, A. G., Pearlmutter, B. A., Radul, A. A. & Siskind, J. M. 2017. Automatic differentiation in machine learning: a survey. The Journal of Machine Learning Research 18 (1), 5595–5637.

Birge, R. T. 1939. The propagation of errors. American Journal of Physics 7 (6), 351–357.

Blum, E. 1989. Approximation of Boolean functions by sigmoidal networks: Part I: XOR and other two-variable functions. Neural computation 1 (4), 532–540.

Borovykh, A., Bohte, S. & Oosterlee, C. W. 2017. Conditional time series forecasting with convolutional neural networks. arXiv preprint arXiv:1703.04691.

Bridle, J. S. 1990. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In Neurocomputing. Springer, 227–236.

Calin, M.-A., Coman, T., Parasca, S. V., Bercaru, N., Savastru, R. S. & Manea, D. 2015. Hyperspectral imaging-based wound analysis using mixture-tuned matched filtering classification method. Journal of Biomedical Optics 20 (4), 046004. doi:10.1117/1.JBO.20.4.046004.

Chang, C.-I. 2007. Hyperspectral Data Exploitation: Theory and Applications. John Wiley & Sons.

Cheng, J.-H. & Sun, D.-W. 2014. Hyperspectral imaging as an effective tool for quality analysis and control of fish and other seafoods: Current research and potential applications. Trends in Food Science & Technology 37 (2), 78–91. doi: 10.1016/j.tifs.2014.03.006.

Collobert, R. & Weston, J. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In Proceedings of the 25th International Conference on Machine Learning, 160–167.

Croft, H., Chen, J., Zhang, Y., Simic, A., Noland, T., Nesbitt, N. & Arabian, J. 2015. Evaluating leaf chlorophyll content prediction from multispectral remote sensing data within a physically-based modelling framework. ISPRS Journal of Photogrammetry and Remote Sensing 102, 85–95.

Dicker, D. T., Lerner, J., Belle, P. V., DuPont Guerry, t., Herlyn, M., Elder, D. E. & El-Deiry, W. S. 2006. Differentiation of normal skin and melanoma using high resolution hyperspectral imaging. Cancer Biology & Therapy 5 (8), 1033–1038. doi:10.4161/cbt.5.8.3261.

Disney, M., Lewis, P. & North, P. 2000. Monte Carlo ray tracing in optical canopy reflectance modelling. Remote Sensing Reviews 18 (2-4), 163–196.

Doi, M. & Tominaga, S. 2003. Spectral estimation of human skin color using the Kubelka-Munk theory. In Color Imaging VIII: Processing, Hardcopy, and Applications, Vol. 5008. International Society for Optics and Photonics, 221–228.

Dumke, I., Purser, A., Marcon, Y., Nornes, S. M., Johnsen, G., Ludvigsen, M. & Søreide, F. 2018. Underwater hyperspectral imaging as an in situ taxonomic tool for deep-sea megafauna. Scientific reports 8 (1), 1–11.

Elliott, D. L. 1993. A Better Activation Function for Artificial Neural Networks.

Elmasry, G., Barbin, D. F., Sun, D.-W. & Allen, P. 2012. Meat Quality Evaluation by Hyperspectral Imaging Technique: An Overview. Critical Reviews in Food Science and Nutrition 52 (8), 689–711. doi:10.1080/10408398.2010.507908.

Eskelinen, M. A. 2017. SOFTWARE FRAMEWORK FOR HYPERSPECTRAL DATA EXPLORATION AND PROCESSING IN MATLAB. ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XLII-3/W3, 47–50. doi:10.5194/isprs-archives-XLII-3-W3-47-2017.

Eskelinen, M. 2019. Computational methods for hyperspectral imaging using Fabry–Perot interferometers and colour cameras. JYU dissertations.

Feilhauer, H., Somers, B. & van der Linden, S. 2017. Optical trait indicators for remote sensing of plant species composition: Predictive power and seasonal variability. Ecological Indicators 73, 825–833.

Fukushima, K. & Miyake, S. 1982. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In Competition and Cooperation in Neural Nets. Springer, 267–285.

Gendrin, C., Roggo, Y. & Collet, C. 2007. Content uniformity of pharmaceutical solid dosage forms by near infrared hyperspectral imaging: A feasibility study. Talanta 73 (4), 733–741. doi:10.1016/j.talanta.2007.04.054.

Gewali, U. B., Monteiro, S. T. & Saber, E. 2019. Machine learning based hyperspectral image analysis: A survey. arXiv:1802.08701.

Gillies, S. et al. 2013. Rasterio: Geospatial Raster I/O for Python Programmers. Mapbox.

Goel, N. S. 1988. Models of vegetation canopy reflectance and their use in estimation of biophysical parameters from reflectance data. Remote Sensing Reviews 4 (1), 1–212.

Goodfellow, I., Bengio, Y. & Courville, A. 2016. Deep Learning. MIT Press.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. & Bengio, Y. 2014. Generative adversarial nets. In Advances in Neural Information Processing Systems, 2672–2680.

Govaerts, Y. M., Jacquemoud, S., Verstraete, M. M. & Ustin, S. L. 1996. Three-dimensional radiation transfer modeling in a dicotyledon leaf. Applied Optics 35 (33), 6585–6598.

Gowen, A. A., Burger, J., O'Callaghan, D. & O'Donnell, C. 2009. Potential applications of hyperspectral imaging for quality control in dairy foods. In 1st International Workshop on Computer Image Analysis in Agriculture, Potsdam, Germany.

Haboudane, D., Miller, J. R., Tremblay, N., Zarco-Tejada, P. J. & Dextraze, L. 2002. Integrated narrow-band vegetation indices for prediction of crop chlorophyll content for application to precision agriculture. Remote sensing of environment 81 (2-3), 416–426.

Hamilton, S. J. & Lodder, R. A. 2002. Hyperspectral imaging technology for pharmaceutical analysis. In Biomedical Nanotechnology Architectures and Applications, Vol. 4626. International Society for Optics and Photonics, 136–147.

Hammersley, J. 2013. Monte Carlo Methods. Springer Science & Business Media.

Heal, C. F., Raasch, B. A., Buettner, P. & Weedon, D. 2008. Accuracy of clinical diagnosis of skin lesions. British Journal of Dermatology 159 (3), 661–668.

Hof, M. 2003. Basics of optical spectroscopy. Handbook of Spectroscopy 1, 39–47.

Hoyer, S. & Hamman, J. 2017. Xarray: N-D labeled arrays and datasets in Python. Journal of Open Research Software 5 (1). doi:10.5334/jors.148.

Hunter, J. D. 2007. Matplotlib: A 2D graphics environment. Computing In Science & Engineering 9 (3), 90–95. doi:10.1109/MCSE.2007.55.

Ishimaru, A. 1978. Wave Propagation and Scattering in Random Media, Vol. 2. Academic press New York.

Jacquemoud, S. & Ustin, L. 2008. Modeling leaf optical properties. Photobiological Sciences Online.

Jacquemoud, S., Verhoef, W., Baret, F., Bacour, C., Zarco-Tejada, P. J., Asner, G. P., François, C. & Ustin, S. L. 2009. PROSPECT+ SAIL models: A review of use for vegetation characterization. Remote sensing of environment 113, S56–S66.

Jacquemoud, S. & Baret, F. 1990. PROSPECT: A model of leaf optical properties spectra. Remote sensing of environment 34 (2), 75–91.

Jacques, S. L. 2013. Optical properties of biological tissues: A review. Physics in Medicine & Biology 58 (11), R37.

Jafarzadeh, M., Hussey, D. C. & Tadesse, Y. 2019. Deep learning approach to control of prosthetic hands with electromyography signals. In 2019 IEEE International Symposium on Measurement and Control in Robotics (ISMCR). IEEE, A1–4.

Jolivot, R., Benezeth, Y. & Marzani, F. 2013. Skin parameter map retrieval from a dedicated multispectral imaging system applied to dermatology/cosmetology. Journal of Biomedical Imaging 2013, 26. doi:10.1155/2013/978289.

Kaivosoja, J., Pesonen, L., Kleemola, J., Pölönen, I., Salo, H., Honkavaara, E., Saari, H., Mäkynen, J. & Rajala, A. 2013. A case study of a precision fertilizer application task generation for wheat based on classified hyperspectral data from UAV combined with farm history data. In Remote Sensing for Agriculture, Ecosystems, and Hydrology XV, Vol. 8887. International Society for Optics and Photonics, 88870H.

Kilian, J. & Siegelmann, H. T. 1993. On the power of sigmoid neural networks. In Proceedings of the Sixth Annual Conference on Computational Learning Theory, 137–143.

Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B., Bussonnier, M., Frederic, J., Kelley, K., Hamrick, J., Grout, J., Corlay, S., Ivanov, P., Avila, D., Abdalla,

S. & Willing, C. 2016. Jupyter Notebooks – a publishing format for reproducible computational workflows. Positioning and Power in Academic Publishing: Players, Agents and Agendas, 87–90.

Krizhevsky, A., Sutskever, I. & Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In Advances in Neural Information Processing Systems, 1097–1105.

Kubat, M. 2017. An Introduction to Machine Learning. Springer.

Kubelka, P. 1931. Ein Beitrag zur Optik der Farbanstriche (Contribution to the optic of paint). Zeitschrift fur technische Physik 12, 593–601.

Lazaridis, D. C., Verbesselt, J. & Robinson, A. P. 2011. Penalized regression techniques for prediction: A case study for predicting tree mortality using remotely sensed vegetation indices. Canadian Journal of Forest Research 41 (1), 24–34.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. & Jackel, L. D. 1989. Backpropagation applied to handwritten zip code recognition. Neural computation 1 (4), 541–551.

Linnainmaa, S. 1970. The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors. Master's Thesis (in Finnish), Univ. Helsinki, 6–7.

Lisboa, P. & Perantonis, S. 1991. Complete solution of the local minima in the XOR problem. Network: Computation in Neural Systems 2 (1), 119–124.

Liu, Y., Pu, H. & Sun, D.-W. 2017. Hyperspectral imaging technique for evaluating food quality and safety during various processes: A review of recent applications. Trends in Food Science & Technology 69, 25–35. doi:10.1016/j.tifs.2017.08.013.

Ma, T., Inagaki, T. & Tsuchikawa, S. 2020. Rapidly visualizing the dynamic state of free, weakly, and strongly hydrogen-bonded water with lignocellulosic material during drying by near-infrared hyperspectral imaging. Cellulose 27 (9), 4857–4869.

Maier, S. W., Lüdeker, W. & Günther, K. P. 1999. SLOP: A Revised Version of the Stochastic Model for Leaf Optical Properties. Remote Sensing of Environment 68 (3), 273–280. doi:10.1016/S0034-4257(98)00118-7.

Malenovskỳ, Z., Homolová, L., Zurita-Milla, R., Lukeš, P., Kaplan, V., Hanuš, J., Gastellu-Etchegorry, J.-P. & Schaepman, M. E. 2013. Retrieval of spruce leaf chlorophyll content from airborne image data using continuum removal and radiative transfer. Remote Sensing of Environment 131, 85–102.

Nair, V. & Hinton, G. E. 2010. Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th International Conference on Machine Learning (ICML-10), 807–814.

Nalepa, J., Myller, M. & Kawulok, M. 2019. Hyperspectral data augmentation. arXiv preprint arXiv:1903.05580.

Neittaanmäki-Perttu, N., Grönroos, M., Tani, T., Pölönen, I., Ranki, A., Saksela, O. & Snellman, E. 2013. Detecting field cancerization using a hyperspectral imaging system. Lasers in Surgery and Medicine 45 (7), 410–417. doi:10.1002/lsm.22160.

Nevalainen, O., Honkavaara, E., Tuominen, S., Viljanen, N., Hakala, T., Yu, X., Hyyppä, J., Saari, H., Pölönen, I., Imai, N. N. et al. 2017. Individual Tree Detection and Classification with UAV-Based Photogrammetric Point Clouds and Hyperspectral Imaging. Remote Sensing 9 (3), 185.

Newton, I. 1704. Opticks: Or, A Treatise of the Reflections, Refractions, Inflections &of Light. Royal Society.

Nouvong, A., Hoogwerf, B., Mohler, E., Davis, B., Tajaddini, A. & Medenilla, E. 2009. Evaluation of Diabetic Foot Ulcer Healing With Hyperspectral Imaging of Oxyhemoglobin and Deoxyhemoglobin. Diabetes Care 32 (11), 2056–2061. doi:10.2337/dc08-2246.

Näsi, R., Honkavaara, E., Lyytikäinen-Saarenmaa, P., Blomqvist, M., Litkey, P., Hakala, T., Viljanen, N., Kantola, T., Tanhuanpää, T. & Holopainen, M. 2015. Using UAV-based photogrammetry and hyperspectral imaging for mapping bark beetle damage at tree-level. Remote Sensing 7 (11), 15467–15493.

Oliphant, T. 2006. A Guide to NumPy. USA: Trelgol Publishing.

Passricha, V. & Aggarwal, R. K. 2020. A comparative analysis of pooling strategies for convolutional neural network based Hindi ASR. Journal of Ambient Intelligence and Humanized Computing 11 (2), 675–691.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. & Duchesnay, E. 2011. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research 12, 2825–2830. doi:10.5555/1953048.2078195.

Pontailler, J.-Y., Hymus, G. J. & Drake, B. G. 2003. Estimation of leaf area index using ground-based remote sensed NDVI measurements: Validation and comparison with two indirect techniques. Canadian Journal of Remote Sensing 29 (3), 381–387.

Puupponen, H.-H. 2014. Unmixing methods in novel applications of spectral imaging. Jyväskylä studies in computing 211.

Pölönen, I., Riihiaho, K., Hakola, A.-M. & Annala, L. 2020. Minimal learning machine in anomaly detection from hyperspectral images. The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 43, 467–472.

Pölönen, I. 2013. Discovering Knowledge in Various Applications with a Novel Hyperspectral Imager. University of Jyväskylä.

Pérot, A. & Fabry, C. 1899. On the application of interference phenomena to the solution of various problems of spectroscopy and metrology. The Astrophysical Journal 9, 87.

Roggo, Y., Edmond, A., Chalus, P. & Ulmschneider, M. 2005. Infrared hyperspectral imaging for qualitative analysis of pharmaceutical solid forms. Analytica Chimica Acta 535 (1), 79–87. doi:10.1016/j.aca.2004.12.037.

Rudiger, P., Stevens, J.-L., Bednar, J. A., Nijholt, B., Andrew, B, C., Randelhoff, A., Mease, J., Tenner, V., maxalbert, Kaiser, M., ea42gh, Samuels, J., stonebig, LB, F., Tolmie, A., Stephan, D., Lowe, S., Bampton, J., henriqueribeiro, Lustig, I., Signell, J., Bois, J., Talirz, L., Barth, L., Liquet, M., Rachum, R., Langer, Y., arabidopsis & kbowen 2020. Holoviz/Holoviews: Version 1.13.3. Zenodo. doi: 10.5281/zenodo.3904606.

Rumelhart, D. E., Hinton, G. E. & Williams, R. J. 1986. Learning representations by back-propagating errors. nature 323 (6088), 533–536.

Saari, H., Aallos, V.-V., Akujärvi, A., Antila, T., Holmlund, C., Kantojärvi, U., Mäkynen, J. & Ollila, J. 2009. Novel miniaturized hyperspectral sensor for UAV and space applications. In Sensors, Systems, and next-Generation Satellites XIII, Vol. 7474. International Society for Optics and Photonics, 74741M.

Saari, H., Pölönen, I., Salo, H., Honkavaara, E., Hakala, T., Holmlund, C., Mäkynen, J., Mannila, R., Antila, T. & Akujärvi, A. 2013. Miniaturized hyperspectral imager calibration and UAV flight campaigns. In Sensors, Systems, and next-Generation Satellites Xvii, Vol. 8889. International Society for Optics and Photonics, 88891O.

Salmivuori, M., Grönroos, M., Tani, T., Pölönen, I., Räsänen, J., Annala, L., Snellman, E. & Neittaanmäki, N. 2019. Hexylaminolevulinate and Aminolevulinic acid Nanoemulsion have Similar Tolerability, Initial Efficacy and Cosmetic Outcome as Methylaminolevulinate in Photodynamic Therapy of Basal Cell Carcinoma in a Prospective Randomized Double-blinded Trial. Journal of Investigative Dermatology 139, S234. doi:10.1016/j.jid.2019.07.119.

Salmivuori, M., Grönroos, M., Tani, T., Pölönen, I., Räsänen, J., Annala, L., Snellman, E. & Neittaanmäki, N. 2020. Hexyl aminolevulinate, 5-aminolevulinic acid nanoemulsion, and methyl aminolevulinate in photodynamic therapy of non-aggressive basal cell carcinomas: A non-sponsored, randomized, prospective and double-blinded trial. Journal of the European Academy of Dermatology and Venereology.

Serranti, S., Gargiulo, A. & Bonifazi, G. 2011. Characterization of post-consumer polyolefin wastes by hyperspectral imaging for quality control in recycling processes. Waste Management 31 (11), 2217–2227. doi:10.1016/j.wasman.2011.06.007.

Shah, S., Bachrach, N., Spear, S., Letbetter, D., Stone, R., Dhir, R., Prichard, J., Brown, H. & LaFramboise, W. 2003. Cutaneous Wound Analysis Using Hyperspectral Imaging. BioTechniques 34 (2), 408–413. doi:10.2144/03342pf01.

Shimada, M., Yamada, Y., Itoh, M. & Yatagai, T. 2001. Melanin and blood concentration in a human skin model studied by multiple regression analysis: Assessment by Monte Carlo simulation. Physics in Medicine & Biology 46 (9), 2397.

Signoroni, A., Savardi, M., Baronio, A. & Benini, S. 2019. Deep learning meets hyperspectral image analysis: A multidisciplinary review. Journal of Imaging 5 (5). doi:10.3390/jimaging5050052.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M. et al. 2016. Mastering the game of Go with deep neural networks and tree search. nature 529 (7587), 484–489.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research 15, 1929–1958.

Stenberg, P., Rautiainen, M., Manninen, T., Voipio, P. & Smolander, H. 2004. Reduced simple ratio better than NDVI for estimating LAI in Finnish pine and spruce stands. Silva Fennica 38. doi:10.14214/sf.431.

Thomas, N. C. 1991. The early history of spectroscopy. Journal of chemical education 68 (8), 631.

Touvron, H., Vedaldi, A., Douze, M. & Jégou, H. 2020. Fixing the train-test resolution discrepancy: FixEfficientNet. arXiv preprint arXiv:2003.08237.

Trops, R., Hakola, A.-M., Jääskeläinen, S., Näsilä, A., Annala, L., Eskelinen, M. A., Saari, H., Pölönen, I. & Rissanen, A. 2019. Miniature MOEMS hyperspectral imager with versatile analysis tools. In MOEMS and Miniaturized Systems XVIII, Vol. 10931. International Society for Optics and Photonics, 109310W.

Tucker, C. J. & Garratt, M. W. 1977. Leaf optical system modeled as a stochastic process. Applied Optics 16 (3), 635–642. doi:10.1364/AO.16.000635.

Van Gemert, M. J., Welch, A. J., Star, W. M., Motamedi, M. & Cheong, W.-F. 1987. Tissue optics for a slab geometry in the diffusion approximation. Lasers in medical Science 2 (4), 295–302.

Verrelst, J., Malenovskỳ, Z., Van der Tol, C., Camps-Valls, G., Gastellu-Etchegorry, J.-P., Lewis, P., North, P. & Moreno, J. 2018. Quantifying vegetation biophysical variables from imaging spectroscopy data: A review on retrieval methods. Surveys in Geophysics, 1–41.

Vyas, S., Banerjee, A. & Burlina, P. 2013. Estimating physiological skin parameters from hyperspectral signatures. Journal of Biomedical Optics 18 (5), 057008. doi: 10.1117/1.JBO.18.5.057008.

Vyas, S., Meyerle, J. & Burlina, P. 2015. Non-invasive estimation of skin thickness from hyperspectral imaging and validation using echography. Computers in biology and medicine 57, 173–181. doi:10.1016/j.compbiomed.2014.12.010.

Wahabzada, M., Besser, M., Khosravani, M., Kuska, M. T., Kersting, K., Mahlein, A.-K. & Stürmer, E. 2017. Monitoring wound healing in a 3D wound model by hyperspectral imaging and efficient clustering. PLOS ONE 12 (12), e0186425. doi:10.1371/journal.pone.0186425.

Wang, L., Jacques, S. L. & Zheng, L. 1995. MCML—Monte Carlo modeling of light transport in multi-layered tissues. Computer Methods and Programs in Biomedicine 47 (2), 131–146. doi:10.1016/0169-2607(95)01640-F.

Werbos, P. 1974. Beyond regression:" new tools for prediction and analysis in the behavioral sciences. Ph. D. dissertation, Harvard University.

Xie, Q., Luong, M.-T., Hovy, E. & Le, Q. V. 2020. Self-training with noisy student improves imagenet classification. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 10687–10698.

Yudovsky, D., Nouvong, A. & Pilon, L. 2010. Hyperspectral Imaging in Diabetic Foot Wound Care. Journal of Diabetes Science and Technology 4 (5), 1099–1113. doi:10.1177/193229681000400508.

Zheludev, V., Pölönen, I., Neittaanmäki-Perttu, N., Averbuch, A., Neittaanmäki, P., Grönroos, M. & Saari, H. 2015. Delineation of malignant skin tumors by hyperspectral imaging using diffusion maps dimensionality reduction". Biomedical Signal Processing and Control 16 (Supplement C), 48–60. doi:10.1016/j.bspc.2014.10.010.

Zherdeva, L. A., Bratchenko, I. A., Myakinin, O. O., Moryatov, A. A., Kozlov, S. V. & Zakharov, V. P. 2016. In vivo hyperspectral imaging and differentiation of skin cancer. In Optics in Health Care and Biomedical Optics VII, Vol. 10024. International Society for Optics and Photonics, 100244G.

GDAL Development Team 2018. GDAL - Geospatial Data Abstraction Library, Version 2.2.3. Open Source Geospatial Foundation.

Rivera-Caicedo, J. P., Verrelst, J., Muñoz-Marí, J., Camps-Valls, G. & Moreno, J. 2017. Hyperspectral dimensionality reduction for biophysical variable statistical retrieval. ISPRS journal of photogrammetry and remote sensing 132, 88–101.

Zarco-Tejada, P. J., Guillén-Climent, M. L., Hernández-Clemente, R., Catalina, A., González, M. & Martín, P. 2013. Estimating leaf carotenoid content in vineyards using high resolution hyperspectral imagery acquired from an unmanned aerial vehicle (UAV). Agricultural and forest meteorology 171, 281–294.

ORIGINAL PAPERS

PI

PRACTICAL APPROACH FOR HYPERSPECTRAL IMAGE PROCESSING IN PYTHON

by

Leevi Annala, Matti A. Eskelinen, Jyri Hämäläinen, Aamos Riihinen and Ilkka Pölönen 2018

ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XLII-3

https://doi.org/10.5194/isprs-archives-XLII-3-45-2018

PRACTICAL APPROACH FOR HYPERSPECTRAL IMAGE PROCESSING IN PYTHON

Leevi Annala*, Matti A. Eskelinen, Jyri Hämäläinen, Aamos Riihinen, Ilkka Pölönen

Faculty of Information Technology, University of Jyvaskyla - leevi.a.annala@student.jyu.fi

Commission III, WG III/4

KEY WORDS: Python, Data analysis, Hyperspectral imaging, Image processing, Machine learning, Open source

ABSTRACT:

Python is a very popular programming language among data scientists around the world. Python can also be used in hyperspectral data analysis. There are some toolboxes designed for spectral imaging, such as Spectral Python and HyperSpy, but there is a need for analysis pipeline, which is easy to use and agile for different solutions. We propose a Python pipeline which is built on packages xarray, Holoviews and scikit-learn. We have developed some of own tools, MaskAccessor, VisualisorAccessor and a spectral index library. They also fulfill our goal of easy and agile data processing. In this paper we will present our processing pipeline and demonstrate it in practice.

1. INTRODUCTION AND MOTIVATION

Python is a go-to programming language of many scientists and it could also be good programming language for hyperspectral data analysis. It has advantage of being actively developed, free, open source programming language. In addition, since it looks like pseudocode, it is easy to learn and write. There are Python tools and packages for all kinds of users, and especially for scientists. There are specialized open source tools for hyperspectral data analysis like Spectral Python (Boggs, n.d.) and HyperSpy (de la Peña et al., 2017), but the scope of potential usage may be too narrow and the structure of such an specialized tool can be too strict for some purposes, for example for transferring data to machine learning algorithm and developing tools that work together with them.

In this paper, we utilize some general open source tools for different aspects of hyperspectral data analysis and determine if they are useful for analysing and visualising hyperspectral images. We also introduce some new tools and packages, which are our own work. We aim at providing the reader with a modular set of tools that can be used in many contexes. These tools are reusable elements, which work fine on their own and can be used for building more complex tools. The packages and tools will be evaluated using following questions: How easy is it to use? How agile is it? What can we do with it?

2. DIFFERENT ASPECTS OF HYPERSPECTRAL DATA ANALYSIS

In this section we will go through different aspects of hyperspectral data analysis and an example of how the selected tools can be used in these subjects. The example is divided into smaller examples and what has been done on previously is assumed to hold on to the new example. We go through the example in figures and in text, and the source code is included in the figures. The example problem is that we have a hyperspectral image of a forest and a dataset of two tree species, birch and pine, in that forest, and we want to use machine learning to differentiate one from the other. First we of course need to import all of the packages, like in figure 1.

```
import xarray as xr
import numpy as np
import pandas as pd
import holoviews as hv
from sklearn import svm
import sklearn
from sklearn.model_selection import GridSearchCV
import visacc
import maskacc

hv.notebook_extension('matplotlib')
```

Figure 1. Importing all necessary packages and declaring that Holoviews should use Matplotlib backend.

2.1 Handling hyperspectral data

For handling hyperspectral data, we recommend the xarray[1] package (Hoyer and Hamman, 2017). It provides multidimensional arrays and datasets with metadata. It is an actively developed open source project by the pydata team. The basic usage of xarray is relatively easy and for more advanced users it offers plenty of options for handling the data. Xarray’s basic idea is to have netCDF (Rew et al., 1997) compatible multidimensional array object in Python. NetCDF stands for network Common Data Form and the basic idea is that the netCDF file describes itself to the reader. Xarray is also easily extendable, which means that one can add new properties as they are needed.

Xarray supports reading spectral image formats like ENVI or TIFF, and other formats. For reading it uses Rasterio (Gillies et al., 2013–), which in turn uses GDAL (GDAL Development Team, 2018). Rasterio is a python toolbox developed solely to read and write geospatial data, and it does it well. GDAL (Geospatial Data Abstraction Library) is a lower level C++ library that translates geospatial raster and vector data.

When xarray has read dataset from file (see figure 2), it is either DataArray or Dataset. There are differences between the two, but

*Corresponding author

[1] Xarray can be installed with pip (`pip install xarray`) or conda (`conda install xarray`) Python package managers.

from now on we will assume that the data is in DaraArray format. DataArray has following properties (see figure 3):

- data, N-dimensional NumPy (Oliphant, 2006) or Dask (Dask Development Team, 2016) array,
- coords, dictionary of coordinate arrays, one array for each dimension of the data,
- dims, names of the dimensions,
- attrs, dictionary keeping track of other metadata,
- name, the name of the DataArray,

which follow the netCDF specification. These properties help in extracting data from the DataArray, since the user can use either index based lookups or label based lookups. For example, if we only had NumPy[2] array, we would only know the dimensions by index, but with DataArray we have names like latitude, longitude and wavelength[3]. Then we can extract data from DataArray like in figure 4 by telling it that we want to see data where latitude is between 39° N and 40° N, longitude is between 116° E and 117° E, and wavelength is between 400 nm and 700 nm.

```
cube = xr.open_dataarray(
        'C:/Users/lealanna/DATAA/vvkk2.nc'
        )
wavelength = [507.60, 509.50, 514.50, 520.80,
              529.00, 537.40, 545.80, 554.40,
              562.70, 574.20, 583.60, 590.40,
              598.80, 605.70, 617.50, 630.70,
              644.20, 657.20, 670.10, 677.80,
              691.10, 698.40, 705.30, 711.10,
              717.90, 731.30, 738.50, 751.50,
              763.70, 778.50, 794.00, 806.30,
              819.70, 833.70, 845.80, 859.10,
              872.80, 885.60]
cube.coords['wavelength'] = ('band', wavelength)
cube = cube.swap_dims({'band': 'wavelength'})
cube.values[cube.values<0]=np.nan
```

Figure 2. Here we read the cube, attach wavelength data to it and remove non-physical negative values.

```
print(cube)

<xarray.DataArray (wavelength: 38, y: 4120, x: 3930)>
array([[[ nan,  nan, ...,  nan,  nan],
        [ nan,  nan, ...,  nan,  nan],
        ...,
        [ nan,  nan, ...,  nan,  nan],
        [ nan,  nan, ...,  nan,  nan]],

        ...,
        [ nan,  nan, ...,  nan,  nan],
        [ nan,  nan, ...,  nan,  nan]],
        ...,
        [ nan,  nan, ...,  nan,  nan],
        [ nan,  nan, ...,  nan,  nan]]],
        dtype=float32)
Coordinates:
    * longitude   (longitude) float64 6.804e+06 ...
    * latitude    (latitude) float64 3.983e+05  ...
    band         (wavelength) int32 1 2 3 4 5 6 7 ...
    * wavelength  (wavelength) float64 507.6 509.5...
Attributes:
    res:          [ 1.  -1.]
    is_tiled:     1
    transform:    [1.00000000e-01  0.00000000e+00 ...
    ncols:        3930
    rows:         4120
    xllcorner:    398296
    yllcorner:    6804299
    cellsize:     0.1
```

Figure 3. Simple print-command to see what the cube holds inside.

```
cube.sel(latitude=slice(39,40),
         longitude=slice(116,117),
         wavelength=slice(400,700))
```

Figure 4. Here we use xarray's sel-method to extract the data we want.

There are also other useful functionalities of xarray DataArray. For example two or more arrays can be attached to each other with easy one line command, where the user only has to align the arrays by common dimension. Generally speaking, xarray handles dimensions well and altering and extracting data using them is generally quite easy. Xarray also handles missing data well and there is possibility to use Dask arrays to parallel compute.

Xarray fullfills our criteria of being easy to use and agile. It has a lot of functionality, enough to keep basic and advanced users satisfied most of the time.

2.2 Visualisation

For visualizing the xarray data, one excellent solution is Holoviews[4] (Stevens et al., n.d.). Holoviews is a visualization library that uses Bokeh (Bokeh Development Team, 2014), Matplotlib (Hunter, 2007) or Plotly (Plotly Technologies Inc., 2015) for showing images. All figures in this paper are produced with Holoviews using Bokeh or Matplotlib visualisation backends.

Basic idea of Holoviews is that visualizing of data should be easy and simple. If user wants to see anything, it should not take many lines of code. In our opinion, Holoviews succeeds in that goal. As we move on, one will see that all images in this paper are produced with less than four lines of code. One basic example of producing Holoviews image is to look at one band of a hyperspectral image like in figure 5.

Now that we have figured out how to visualise a single channel of an image, the next logical step is to want to visualise the entire multidimensional dataset. This is also easy. Holoviews supports multidimensional datasets very well and there are data backends that support multiple different data formats including xarray. As we can see in figure 6, more complex visualisation is easy to make. In the example we make a Holoviews dataset out of xarray DataArray, and tell Holoviews to make a series of images out of the dataset.

One of the properties of Holoviews is that one can make interactive figures using the Bokeh backend with no extra effort. By having Bokeh backend selected user can right away use interactive tools like zooming the image either by scrolling or drawing boxes on the image. A little more work is required for using hover, tapping or selection tools, which all can be programmed to do what the user wants them to do. An example of usage of tapping and selection tools are using them to select data for further analysis or activating other visualisation with them.

[2]NumPy is in practice the Python standard array library.

[3]Note, that the user can freely name the dimensions. The user is not stuck with these names.

[4]Holoviews can be installed with pip (`pip install holoviews`) or conda (`conda install holoviews`) Python package managers.

```
cube = xarray.DataArray(...)
hv.Image(cube.sel(wavelength=800,
                  method='nearest'))
```

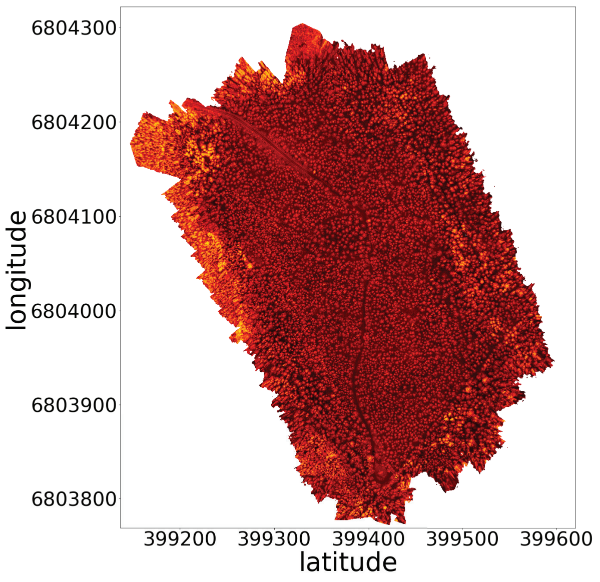


Figure 5. Here we produce a very simple Holoviews visualisation by telling Holoviews Image to use the xarray data we provide it. This is an image of a Finnish forest.

A good platform for using Holoviews is Jupyter Notebook (Kluyver et al., 2016). Jupyter Notebook is a web application where user can code in Python and output images and write narratives between code blocks. The user has to activate Holoviews by importing it and declaring the visualisation backend like in figure 1. When one is visualising figures in Jupyter Notebook, it is possible to fine tune figures, by using *output cell magic* and Holoviews *opts*. We use output cell magic and opts in figure 6, where lines starting with % or %% are the cell magic lines. In the example of the figure we tune the size of the font and the size of the image. This fine tuning is absolutely necessary if one needs to produce figures for a publication or needs good looking images for any reason. Matplotlib backend is better suited for publication quality figures.

Holoviews is purely a visualisation library. The user can make data move between two images in the same visualisation, but the developers have not build a way to get this data for further use and the only way of getting a data output is by coding it. However, once coded, these background processes are relatively easy to attach to an image. Holoviews is an open source project and it is developed by the ioam team. Holoviews is easy to use and it can be bended to do many things. It makes beautiful images, and in all is an excellent choise for visualisation.

2.3 Masking and visualizing xarray

Using xarray and Holoviews together is made easy by Holoviews developers. Xarray is one of the available backends for Holoviews. That means, one can easily produce an image from xarray using Holoviews. There is still some difficulties involved, and to address those difficulties, we use xarrays extendability. Making an extension to xarray in figure 7 is done by making a Python class and declaring it as dataset or DataArray accessor.

```
%%opts Image [fontsize={{'title':15, \
                        'xlabel':15, \
                        'ylabel':15, \
                        'ticks':15}},\
          fig_size=350]

ds = hv.Dataset(cube,
                vdims=['Value'])
ds.to(hv.Image,
      kdims=['x', 'y'],
      dynamic=True)
```

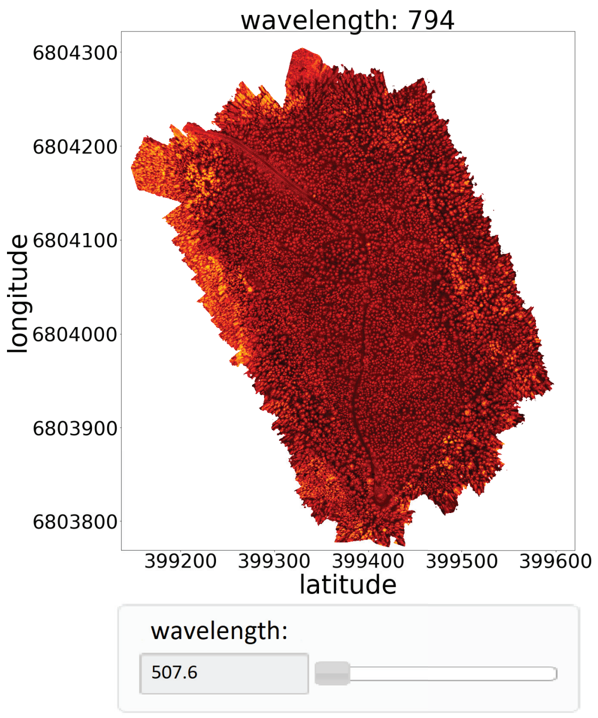


Figure 6. Here we make a more complicated Holoviews visualisation by using Holoviews dataset. From using Dataset, we get a slider that goes through the wavelength bands.

These extensions are relatively easy to make and can extend xarray's functionalities to anything one might want it to do, within reasonable limits.

```
@xr.register_DataArray_accessor('cat')
class CatAccessor(object):
    def __init__(self, xarray_obj):
        self._obj = xarray_obj
        self.cat = 'a cat'
```

Figure 7. Extending xarray with a simple Accessor. Here we declare that CatAccessor is a DataArray accessor and define it.

We have developed two DataArray accessors, MaskAccessor and VisualisorAccessor[5]. The reason for developing both of these tools is that we want to use more complicted background interactivity tracking with Holoviews and get the data out of the visualisation.

MaskAccessor is a general masking tool for xarray, and the main function of it is to help collect data points to further analysis, such as machine learning or modelling. It provides an interface

[5]These tools are available at our groups github page `http://github.com/silmae`

for selecting pixels in n-dimensional datasets. In figure 8 we see that the accessor is initiated when one imports the xarray and the accessor. After that every DataArray has the property, and the user can use the accessor by calling it by name.

```
import xarray as xr
import maskacc

cube = xr.DataArray(...)
cube.M.dims
```

Figure 8. When the accessor is imported, every xarray DataArray created after that has the accessor attribute.

The mask dimensions are set at the initialisation to be the first two dimensions of the DataArray, but there is the reset method that is used to change the dimensions, as we see on figure 9. One can also initialise the mask here or just assign a new mask afterwards. The MaskAccessor class checks that the shape of the mask is correct.

```
import numpy as np
cube.M.reset(dims=['a', 'b'],
             matrix=[[0,1,0,1],
                     [1,0,1,0],
                     [0,1,0,1]])

# OR
cube.M.reset(dims=['a', 'b'])
cube.M.mask = np.array([[0,1,0,1],
                        [1,0,1,0],
                        [0,1,0,1]])
```

Figure 9. Different ways of assigning a specific matrix as the mask.

On figure 10 one can see four different selection methods to set mask on individual points.

```
# Select
cube.M.select([0,0])
cube.M.select([(0,2),(1,1)])

# Unselect
cube.M.unselect([(0,2),(1,1)])

# All to ones
cube.M.selected_ones()

# All to zeros
cube.M.selected_zeros()
```

Figure 10. Different selection methods for MaskAccessor.

Finally, on figure 11 there is three different methods to get the mask or masked data.

```
# Get the mask as xarray.DataArray
cube.M.mask_as_xarray()

# Get the masked points as xarray.DataArray
cube.M.where_masked()

# Get the masked points as a list
cube.M.to_list()
```

Figure 11. Methods for getting data out of MaskAccessor and underlyind DataArray.

VisualisorAccessor is a hyperspectral imaging specific visualising tool for xarray and MaskAccessor. It is designed to make basic visualizations of xarray DataArray and MaskAccessor mask with easy one-line commands. For example the image in figure 6 can now be produced with the one line code of figure 12. It is also easy to add visualisations like this to the VisualisorAccessor.

```
cube.visualize.basic(sliders = ['wavelength'])
```

Figure 12. The visualisation on figure 6 can be done with one line code with VisualisorAccessor.

We have implemented three chooser functions, which access the mask and select or unselect pixels. They are called Point Chooser, Box Chooser and Spectre Chooser. Spectre Chooser and Box Chooser use Bokeh’s box drawing tools for selecting which pixels are chosen and Point Chooser uses tap tool. Example uses of the Choosers is on figure 13, and screenshots of the Choosers are on figures 14 (Point Chooser), 15 (Box Chooser) and 16 (Spectre Chooser).

```
layout_box = cube.visualize.box_chooser()
layout_point = cube.visualize.point_chooser()
layout_spectre = cube.visualize.spectre_chooser()
```

Figure 13. VisualisorAccessor has three different chooser tools.

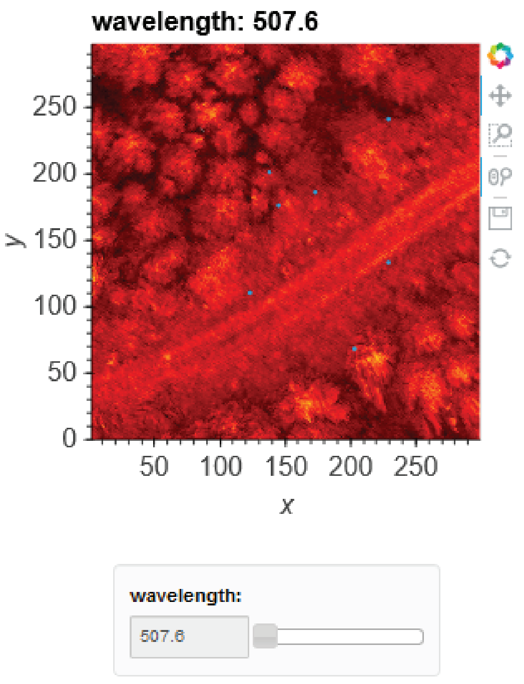


Figure 14. Screenshot of the point chooser.

Finally there is a histogram method (figure 17), that calculates histograms for each bands and shows those histograms side by side. This is translated from hsicube (Eskelinen, 2017) MATLAB package to Python.

2.4 Machine learning

Machine learning can be handled using scikit-learn[6] package (Pedregosa et al., 2011). The main idea of scikit-learn is to make

[6]Scikit-learn can be installed with pip (`pip install sklearn`) or conda (`conda install sklearn`) Python package managers.

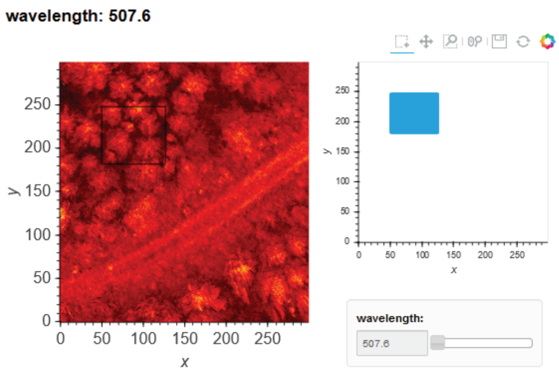


Figure 15. Screenshot of the box chooser.

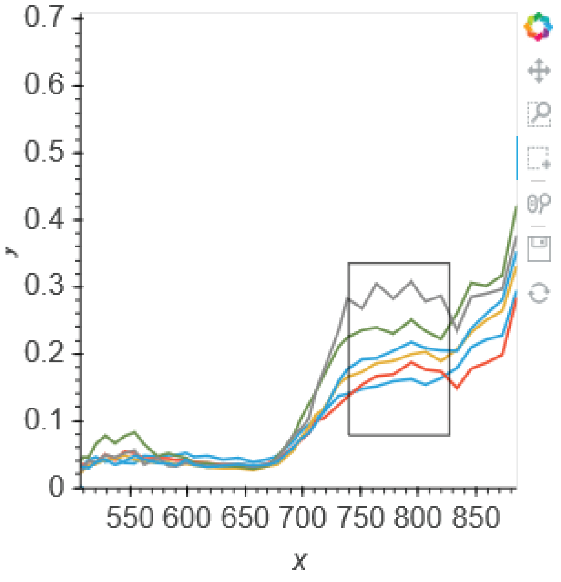


Figure 16. Screenshot of the spectre chooser.

simple and efficient tools for data analysis. Part of the simplicity is documentation, and with scikit-learn it is done well. There is flowchart for finding a suitable estimator, and every estimator is documented so well, that one can easily learn to use them decently.

Another thing we want to point out is the variety of implemented algorithms. Every well established machine learning algorithm can be found. Still, there are no duplicates, and the user does not have to worry about competing implementations, and the API is consistent through the algorithms.

Other useful properties are flows, parallel computing, fine tuning. The user can relatively easily make a workflow, that preprosesses data, does cross-validation on desired estimators with desired parameters and returns the estimator, that seems to produce the best result. The basic forms of the estimators are simple, but there are multiple parameters that one can use to fine tune the estimator.

The usage is simple since the algorithms are well documented and their API is simple, yet agile. Scikit-learn is also free and open source, and it is developed by scikit-learn team. It fullfills

```
result = cube.visualize.histogram(band_dim = 'band')
hist_image = result[0]
hist_counts = result[1]
bin_edges = result[2]
hist_image
```

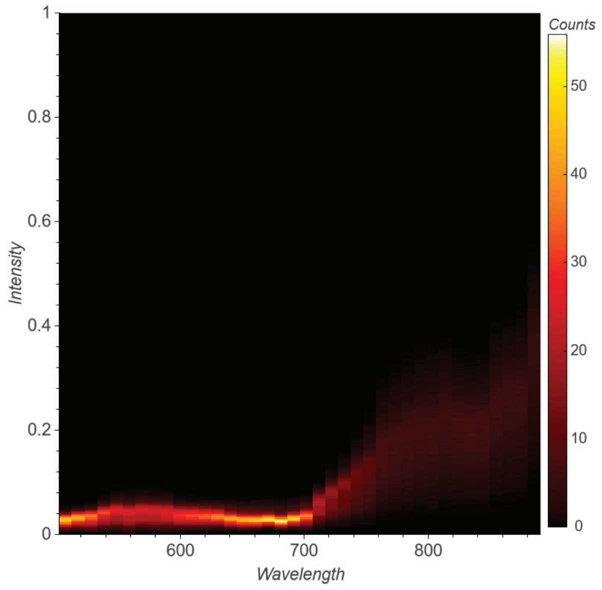


Figure 17. Visualisation of the histogram. The histogram tool returns an image of the histogram, the values of the histogram and the bin edges.

our criteria of being easy to use. It is agile in a way that user can make own flows through the algorithms and the user can fine-tune the algorithms as much as is needed.

Now we can use machine learning on our example problem. First, in figure 18 we use pandas[7] (McKinney, 2010) for reading the tree dataset and plot it over one of the bands in our cube.

Now we can use the tree coordinates to train a machine learning algorithm to recognise birch from pine. We take 30 * 30 box around every tree and calculate histogram of the box. These histograms are used to train the algorithm. We also have to make nan-values zero for this. In figure 19 we use VisualisorAccessor to make the histograms and goal vector and prepare them for machine learning.

In figure 20 we train the machine learning algorithm. For this example we are using support vector machine algorithm. We also do cross-validiation with GridSearchCV. Both of these functions are functions from scikit-learn. Then we print out the results, and that tells us the best accuracy score[8] and the best parameters.

In figure 21 use the predictor to predict the species of a 30x30 histogram, that is made from a hyperspectral image of a tree. From the result we could then interpret wheter the estimator estimates the histogram as a pine or a birch.

[7]Pandas is in practice the Python standard for tabular and Excel type data.
[8]Note, that the score should not be taken too seriously, since this is a toy example, and birch and pine are really easy to recognise from eatch other.

```
trees_panda = pd.read_csv('trees_panda_kuva_2',
                          index_col=0)
trees_panda = trees_panda.sort_values("I"). \
drop_duplicates(["I", "P"],
keep="first")
points = hv.Points(trees_panda,
kdims=['I','P'])
hv.Image(cube[20]) * points
```

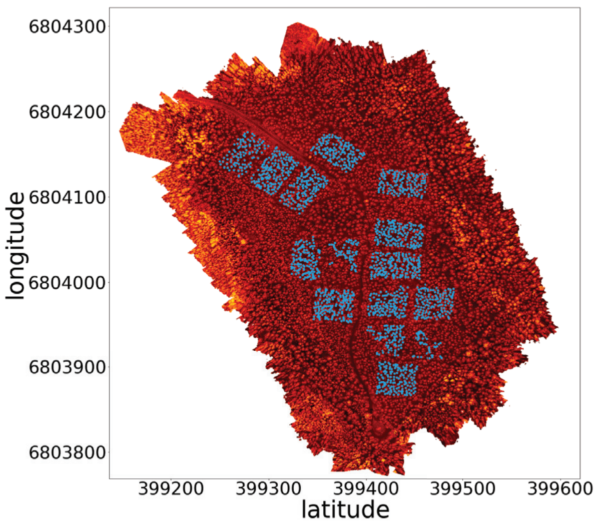


Figure 18. Visualisation of the trees over the image. In this block we read tree data as pandas DataFrame and visualise the trees on the top of one band of the cube.

```
cube.values = np.nan_to_num(cube.values)
X_list = []
size = 30
cellsize = float(cube.attrs['cellsize'])
add = cellsize * size / 2
bin_edges = np.arange(0, 1, 1/20)
i = 0
for puu in trees_panda.values:
print(i)
i = i+1
x_coord = puu[1]
#print(x_coord)
y_coord = puu[2]
#print(y_coord)
cropped = cube.sel(y=slice(y_coord + add,
                          y_coord - add),
                 x=slice(x_coord-add,
                          x_coord+add))
_, hist, _ = cropped.visualize.\
            histogram(
          bin_edges=bin_edges,
                      flag='linear',
                      show_plot=False,
                      band_dim="wavelength"
          )
hist_flat = np.array(hist).flatten()
X_list.append(hist_flat)
X = np.array(X_list)
y = np.array(trees_panda['pl'])
```

Figure 19. Calculating histograms and preparing data for machine learning algorithm.

2.5 Other aspects

Other notable libraries for hyperspectral data analysis are already mentioned Bokeh for advanced visualisation, scikit-image (van der Walt et al., 2014) for image data analysis and Tensor-Flow (Abadi et al., 2015) and Keras[9] (Chollet et al., 2015) for deep learning.

```
svc = svm.SVC()
parameters = {'kernel':['linear',
                        'poly',
                        'rbf',
                        'sigmoid'],
            'C':[10**i for i in range(-5,4)]}

clf = GridSearchCV(svc, parameters, n_jobs=20)
clf.fit(X,y)
clf_best = clf.best_estimator_
print(clf.best_score_)
print(clf.best_estimator_)
```

```
0.965723612622
SVC(C=0.1, cache_size=200, class_weight=None,
coef0=0.0, decision_function_shape='ovr',
degree=3, gamma='auto', kernel='poly',
max_iter=-1, probability=False,
random_state=None, shrinking=True,
tol=0.001, verbose=False)
```

Figure 20. Here we train the machine learning algorithm and print out the result. caption=Results of the training. Here we see that best estimator predicts correctly 96.6% of the time.

```
tree_1_pred = clf_best.predict(X_new.reshape(1, -1))
```

Figure 21. Here we use the estimator.

Bokeh is a package that has been on the rise in 2017. Bokeh makes interactive Python visualisations, using JavaScript. It is a backend of Holoviews, and if one wants to understand Holoviews deeply, this is one place to look at. Bokeh visualisations are generally quite beautiful, but it comes with expence of computational complexity and increased memory usage.

Scikit-image is a sister package of scikit-learn. Scikit-image is focused on computer vision and image processing. The same advantages as with scikit-learn apply here. The API is consistent and simple and the wide variety of algorithms is well curated.

TensorFlow and Keras are deep learning libraries. TensorFlow is considered to be the state of the art at this field, but the syntax is difficult and learning curve extremely steep. Keras uses TensorFlow as a backend, and offers simpler syntax. If one is a beginner on deep learning, Keras is a library to more easily get started, but as one is becoming more advanced user, TensorFlow's flexibility and increased tuning possibilities start becoming more attractive.

3. CONCLUSIONS AND FURTHER WORK

We have gathered and further developed an agile and easy to use pipeline for hyperspectral data analysis in Python. The tools we have investigated have wide range of advantages such as simple API:s, variety of different implementations, back ends and tools and extendibility.

In addition to that, Python programming language has large user base and active developer community, which guarantees that Python keeps up with needs of scientists. The packages mentioned in this paper are all actively developed and thoroughly tested.

[9]These tools can also be installed with pip or conda.

Also, especially xarray and Holoviews are good Python tools for hyperspectral data processing and visualisation. These tools seem like they are made for this use, but they still provide the generality of non-specialised tools. Compared to HyperSpy end Spectral Python our solution is much more modular and open to extending with new blocks.

Finally we would like to suggest, that in the context of using Python in hyperspectral data analysis, there is need for developing a graphical user interface that uses these tools and finding out best practises for utilising deep learning algorithms. We are starting to develop the graphical user interface in this summer.

Deep learning algorithms are becoming more and more attractive when there is more and more computational power available. The algorithms are computationally intense, but when they are used correctly they provide strikingly good results. These algorithms can be applied on many of the problems on the field of hyperspectral data analysis, such as object recognition, classification or for example analysing the health of a crop.

On this specific toolset there is work to do with parallelisation, since the datasets are huge and paralellisation would make the computations faster.

We have also started to develop a Python library for spectral indices, and are quite far in it already. The leading principle of our implementation is to make a simple implementation of every index on website indexdatabase.de, and wrap the implementation lightly with features that help in the usage. The point was to make the indices easily computable, so that the user could easily use a loop to go through the indices.

One thing we need to define was the API for selecting bands. For many of the indices, they are not defined for exact wavelengths like 745nm, but rather for red light and user needs to define this as he/she wishes. This is for now done by declaring the defaults in form of a Python dictionary. The other thing to consider is how is a band selected. Is it selected only if there is a clear match, the indice wants wavelength 500nm and our data has exactly that or is there room for approximation? If the used data is in format of xarray DataArray or Dataset, then it is possible to use the xarrays nearest neighbor -selection like in figure 5, otherwise one needs to implement their own selector. Once the index library is initialised like in figure 22, one can loop through the indices and find all the indices that can be computed on the dataset like in figure 23. Then the user can plot all possible indices with Holoviews like in figure 24.

```
from pyspindl import Indices, selectors
defaults =  {’NIR’: 815.7,
             ’GREEN’: 544.2,
             ’RED’: 595.3}
defaults.update(
        {k: defaults[’RED’] for k in [’Red’, ’R’]}
        )
defaults[’G’] = defaults[’GREEN’]
#Without defaults, we can not calculate some indices.
indices = Indices(selectors.from_xarray(
                    ’wavelength’,
                    method=’nearest’,
                    tolerance=8.0
                    ),
                  defaults=defaults)
```

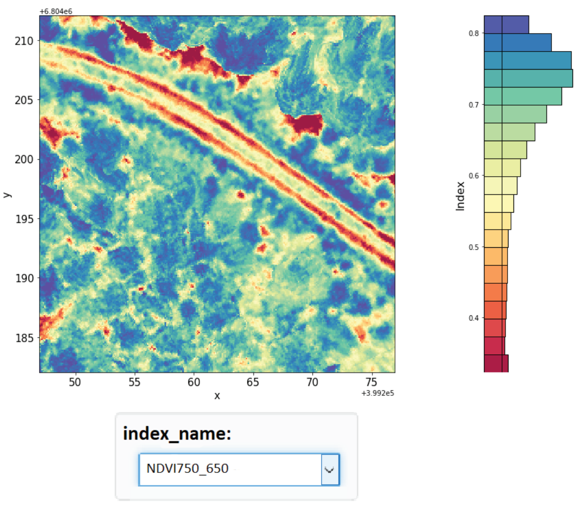
Figure 22. The initialisation process of spectral indices library. This is still work in progress.

Other thing we we are considering in developing this package is bands. How are they defined? There is big difference between a camera that has the same response on a interval around the middle value and camera that has more gaussian response. These differences should somehow be accounted for with software. The response function could be used in selection, and inbetween values coud be interpolated from two or more bands based on their responses. The response function is definately important in presicion applications and this problem needs to be solved.

```
matches = dict()
for iname, ifunc in indices.items():
    try:
        matches[iname] = ifunc(cube_cropped)
        # The following is necessary to
        # remove indices that result
        # only in +inf, −inf and NaN
        if not np.any(np.isfinite(matches[iname])):
            matches.pop(iname)
            continue
        # We have now built a dictionary
        # of index names and corresponding data.
        matches[iname].coords[’index_name’] = iname
        # We also want to clean up
        # unnecessary coordinates, if any remaim
        for coordinate in [’band’,
                           ’fwhm’,
                           ’wavelength’]:
            if coordinate in matches[iname].coords:
                matches[iname] = matches[iname]. \
                                 drop(coordinate)
    except (KeyError, TypeError, NameError):
        pass
print(str(len(matches))+’␣matching␣indices␣found.’)
```

Figure 23. We loop through the indices, and take those that are sensible.

```
%%output size = 250
%%opts Image [invert_yaxis=True] (cmap=’Spectral’)
dataset = hv.Dataset(prettyfield,
                     kdims=[’index_name’, ’x’, ’y’],
                     vdims=’Index’)
dataset.to(hv.Image,
           kdims=[’x’, ’y’],
           dynamic=True).hist()
```



index_name: NDVI750_650

Figure 24. All indices in a dropdown menu. Dropdown menu comes from the use of Holoviews Dataset.

REFERENCES

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y. and Zheng, X., 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Official website: `https://www.tensorflow.org/`.

Boggs, T., n.d. Spectral python. Source code available at `https://github.com/spectralpython`.

Bokeh Development Team, 2014. Bokeh: Python library for interactive visualization. Official website: `http://www.bokeh.pydata.org`.

Chollet, F. et al., 2015. Keras. Source code available at `https://github.com/keras-team/keras`.

Dask Development Team, 2016. Dask: Library for dynamic task scheduling. Official website: `http://dask.pydata.org`.

de la Peña, F. et al., 2017. hyperspy/hyperspy: Hyperspy 1.3.

Eskelinen, M. A., 2017. Software framework for hyperspectral data exploration and processing in matlab. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XLII-3/W3, pp. 47–50. Source code available at `https://github.com/silmae/hsicube`.

GDAL Development Team, 2018. Gdal - geospatial data abstraction library, version 2.2.3. Official website: `http://www.gdal.org`.

Gillies, S. et al., 2013–. Rasterio: geospatial raster i/o for Python programmers. Source code available at `https://github.com/mapbox/rasterio`.

Hoyer, S. and Hamman, J., 2017. xarray: N-D labeled arrays and datasets in Python. *Journal of Open Research Software*. Source code available at `https://github.com/pydata/xarray`.

Hunter, J. D., 2007. Matplotlib: A 2d graphics environment. *Computing In Science & Engineering* 9(3), pp. 90–95. Source code is available at `https://github.com/matplotlib/matplotlib`.

Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B., Bussonnier, M., Frederic, J., Kelley, K., Hamrick, J., Grout, J., Corlay, S., Ivanov, P., Avila, D., Abdalla, S. and Willing, C., 2016. Jupyter notebooks – a publishing format for reproducible computational workflows. pp. 87 – 90.

McKinney, W., 2010. Data structures for statistical computing in python. In: S. van der Walt and J. Millman (eds), *Proceedings of the 9th Python in Science Conference*, pp. 51 – 56.

Oliphant, T., 2006. A guide to numpy. Source code available at `https://github.com/numpy/numpy`.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12, pp. 2825–2830.

Plotly Technologies Inc., 2015. Collaborative data science. Official website: `https://plot.ly`.

Rew, R. K., Davis, G. P., Emmerson, S. and Davies, H., 1997. NetCDF User’s Guide for C, An Interface for Data Access, Version 3.

Stevens, J.-L., Rudiger, P. and Bednar, J. A., n.d. Holoviews. Source code available at `https://github.com/ioam/holoviews`.

van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., Gouillart, E., Yu, T. and the scikit-image contributors, 2014. scikit-image: image processing in Python. *PeerJ* 2, pp. e453. Source code available at `https://github.com/scikit-image/scikit-image`.

PII

TREE SPECIES IDENTIFICATION USING 3D SPECTRAL DATA AND 3D CONVOLUTIONAL NEURAL NETWORK

by

Ilkka Pölönen, **Leevi Annala**, Samuli Rahkonen, Olli Nevalainen, Eija Honkavaara, Sakari Tuominen, Niko Viljanen and Teemu Hakala 2018

2018 9th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS), IEEE

https://doi.org/10.1109/WHISPERS.2018.8747253

TREE SPECIES IDENTIFICATION USING 3D SPECTRAL DATA AND 3D CONVOLUTIONAL NEURAL NETWORK

Ilkka Pölönen[1], Leevi Annala[1], Samuli Rahkonen[1], Olli Nevalainen[2], Eija Honkavaara[2], Sakari Tuominen[3], Niko Viljanen[2], Teemu Hakala[2]

[1]Faculty of Information Technology, University of Jyväskylä, P.O. Box 35, FI-40014 Jyväskylä, Finland
[2]Finnish Geospatial Research Insititute, National Land Survey of Finland, Geodeetinrinne 2, 02430 Masala, Finland
[3]Natural Resources Institute Finland, PL 2 00791 Helsinki, Finland

ABSTRACT

In this study we apply 3D convolutional neural network (CNN) for tree species identification. Study includes the three most common Finnish tree species. Study uses a relatively large high-resolution spectral data set, which contains also a digital surface model for the trees. Data has been gathered using an unmanned aerial vehicle, a framing hyperspectral imager and a regular RGB camera. Achieved classification results are promising by with overall accuracy of 96.2 % for the classification of the validation data set.

Index Terms— Tree species, spectral imaging, 3D, convolutional neural network, UAV

1. INTRODUCTION

This study is continuum for [1], where the individual tree detection and classification pipeline for the hyperspectral and point cloud data is clearly described. We are interested to see if deep learning methods could improve or simplify the data processing chain for identifying the species of individual trees.

There exists plenty of research concerning tree species identification, but it is mainly concentrated on large scale remote sensing, which uses forest stand and plot level data. For example in Scandinavia combination of airborne laser scanning and aerial images is used in forest inventory [2]. There are less studies and applications for the tree species identification from unmanned aerial vehicles (UAV) using hyperspectral sensors. If hyperspectral data has been used for tree species identification, the platform for data gathering has been manned aircraft or satellite.

As in [1], these remote sensing studies use quite traditional feature extraction and selection methods before classification. Deep learning methods have dramatically improved performance of pattern recognition [3]. Especially deep convolutional neural networks (CNN) have provided breakthroughs in image, video and audio processing. If we consider hyperspectral data, it seems that they should handle hyperspectral data combined with 3D data as well. There is currently increasing number of research, which applies CNN's and 3D CNN's to hyperspectral imager [4, 5].

In this paper we first test performance of 3D CNN for tree species classification. Neural networks has the nature of being a black box, that doesn't reveal how it has reasoned its results. However, while doing classification we can calculate saliency maps, which will give us hints on which parts of the input data are relevant for the CNN [6].

This paper has the following structure. First, in Section 2 we describe the used data set, its acquisition and preprocessing. Then the structure and functionality of the used 3D CNN is described. In Section 3, the results are presented and Section 4 includes the conclusion.

2. MATERIALS AND METHODS

2.1. Data gathering and preprocessing

The research data is the same as reported in [1]. The collected remote data was captured in Vesijako research forest area in the municipality of Padasjoki in southern Finland (approximately 61^{o}24'N and 25^{o}02'E). Area has been used for forestry research by Natural Resources Institute of Finland. The area contains experimental plots with different research setups. All the trees with the diameter of at least 50 mm at the breast-height were measured and estimated with various metrics, such as the tree species, diameter, height and volume. Locations of these trees were collected with GPS.

In total, 4142 trees were selected for further study. The data set contained three most common species of Finnish forests: scots pine (*Pinus sylvestris*, 2821 samples), norway spruce (*Picea abies*, 742 samples) and silver birch (*Betula bendula*, 579 samples). These selected trees were compared to aerial orthoimage mosaics to ensure that the GPS coordi-

This research has been co-financed by Finnish Funding Agency for Innovation Tekes (grants 2208/31/2013 and 1711/31/2016)

nates were in the centres of the treetops.

The used remote sensing data was a combination of two data modalities captured by the UAV remote sensing system, which belongs to Finnish Geospatial Research Institute. System consist of a Tarot 960 hexacopter and a Pixhawk autopilot. System is capable of carrying 3 kg payload at maximum. Average flying time of the system is 30 minutes. As a payload, we had a tunable Fabry-Pérot inteferometer based spectral imager (FPI) and an ordinary RGB camera, the Samsung NX1000 (RGB). Flying height from ground level varied between 83-94 meters.

The FPI imager captures raw data, which is processed to radiance based on the radiometric laboratory calibration [7]. The geometric imaging model was then determinated. The model includes both the interior and exterior orientations of the images. The digital surface model is calculated by dense image matching. Because of the slight variations between bands in the FPI camera, we had to apply registration of the spectral bands of FPI images. To make data cubes and further mosaics radiometrically homogenous, the radiometric imaging model has to be determined [8, 9]. The hyperspectral image mosaic is calculated after the radiometric model is applied to each cube. The detailed radiometric and geometric processing of the data set is explained in [1]. Finally, the spectral mosaics with 33 bands and digital surface model (DSM) both with 10 cm GSD are created.

For the tree species identification, 4×4 meter windows surrounding each treetop were extracted. The windows contained both DSMs as rasters and spectral cubes. For each treetop, the extracted DSMs were scaled by the minimum value of the whole DSM. The DSM and spectral cube for each treetop were concatenated in spectral axis to unified data cubes $(41 \times 41 \times 34)$. In Finland there exist laser scanned nation wide ground surface elevation model, which is freely available. Thus, canopy surface model could have been calculated, but it isn't actually needed, because we are only using height of the treetops.

Figure 1 illustrates average treetops for each species. We can see that there are slight differences between the shapes. Pine's treetops are quite symmetric. Spruce's treetops are more of ellipses and aligned on north-west to south-east axis. Birch is more irregular, but its leaves and branches are towards south where the Sun shines.

Figure 2 represents how spectral distribution diverges to different wavelengths for each tree species. The line in the figure represents the average spectra for each treetop. Quite obvious differences can be found between birches and Nordic coniferous trees. Birches have stronger reflection in green and infrared regions. Birches have steeper spectrum at red edge area.

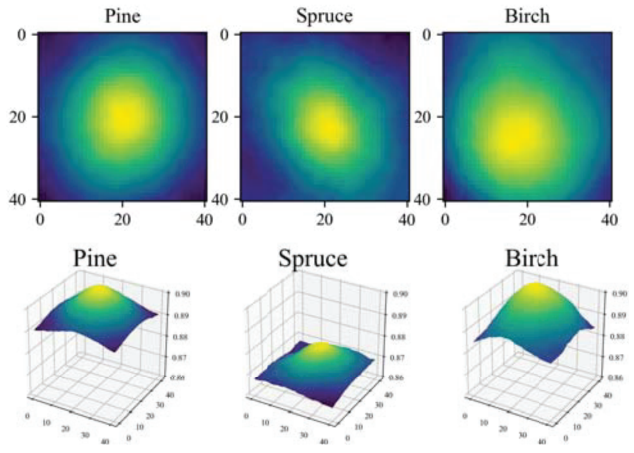


Fig. 1. Avarage shape of treetop for each tree species.

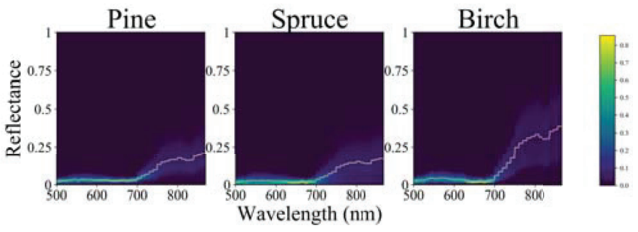


Fig. 2. Histogram spectra of each tree species. Black line is average spectrum.

2.2. Convolutional Neural Network

Originally CNN's were presented by LeCun and Bengio [10]. The idea was to tackle feature extraction and selection problem in fully connected feed-forward networks. The network uses a convolution matrices. Traditional neural network layers are usually based on consecutive dense (fully connected) neurons. In convolutional neural networks, there exists at least one convolution operation in the network. We applied quite simple structure to our CNN, using four types of layers: 3D convolutional, pooling, dropout and fully connected layers. Our network's structure is presented in Table 1.

In general, convolutional layers have trainable filters, which use convolution operations to extract features. In our implementation, the convolution layer uses activation called rectified linear unit (ReLU). ReLU has advantages of being efficient with non-linear relations and having less vanishing gradient problems during the network optimisation compared to other popular activation functions [11]. Pooling layers, which usually follow convolutional layers, are non-linear downsampling functions, which reduce dimensions of input data. Dropout layer is a regularization method for reducing overfitting in the neural network by introducing noise to the network. Flatten layer translates data to one dimensional stack.

Layer	Kernel / pool size or Activation	Output Shape	Parameters
Conv3D	(3,3,1) ReLU	(39, 39, 33, 64)	640
Conv3D	(3,3,3) ReLU	(37, 37, 31, 64)	110656
MaxPooling3D	(2,2,1)	(18, 18, 31, 64)	0
Conv3D	(3,3,3) ReLU	(16, 16, 29, 128)	221312
MaxPooling3D	(2,2,3)	(8, 8, 9, 128)	0
Conv3D	(3,3,3) ReLU	(6, 6, 7, 256)	884992
MaxPooling3D	(2,2,3)	(3, 3, 2, 256)	0
Flatten		(4608)	0
Dense	ReLU	(128)	589952
Dropout (0.25)		(128)	0
Dense	SoftMax	(3)	387
Total params:	1,807,939		
Trainable params:	1,807,939		
Non-trainable params:	0		

Table 1. Structure of our experimental CNN.

A dense layer is a fully connected layer, which consists of parallel neurons which are connected to all previous layer's outputs. Weights of the connections and activation functions determine which features are correlating with different tree species. The last dense layer is activated with *softmax* function, whose output is the final classification.

If the amount of data is limited, meaning that the number of training samples is low, then there is option to apply data augmentation. Basically this means that we will generate new training data from existing ones. In this study we fivefold our training data by using simple rotation and flipping operations. Selected training data was flipped both horizontally and vertically. Data was also rotated 90 degrees to left and right.

In machine learning structures like neural networks are so called "black box" solutions. We don't have clear vision how data is classified. It is reasonable to ask, is the classification based on real feature of wanted object or something secondary such as ground type in tree species recognition. Luckily there are methods to see where network is putting weight in classified data. It is possible to calculate gradient over layers from output to input. This way to get actually image, where areas with higher values contributes most to classification result. These maps are called *saliency maps*.

Stochastic gradient decent was used to tune weights between layers. We used categorical cross entropy as a loss function, which basically calculates cross entropy between categories probability distributions. Primary metric for model evaluation was accuracy

$$acc = \frac{TP + TN}{TP + FP + FN + TN},$$

where TP is true positive, TN is true negative, FP is false positive and FN is false negative classification result.

CNN's were trained by using IBM PowerAI platform which includes two Tesla V100-SXM2 16 GB GPU units. Tensorflow was used as a computational backend [12]. All machine learning phase coding was done using Python 3.6 and Keras library [13]. Saliency maps were calculated using Keras-vis library [14].

3. RESULTS

Altogether 3311 trees were randomly selected for the training of the 3D CNN. After data augmentation there was 16555 samples. Training was performed with batch size 128 and with 100 epocs. Training took two and half hours (approx. 88 seconds/epoch). Results were validated with 831 samples, which weren't included in training set.

Figure 3 shows that accuracy of trained model is relatively high. It seems that we can with quite large confidence identify tree species from each other. Overall accuracy for classification of validation set was 96.2%, which is higher with earlier results achieved in [1]. Producer accuracies were for each tree species were 96.2% (Pine), 86.6 % (Spruce) and 98.2 % (Birch). Respectively users accuracies were 96.3 %, 83.8 % and 95.7 %.

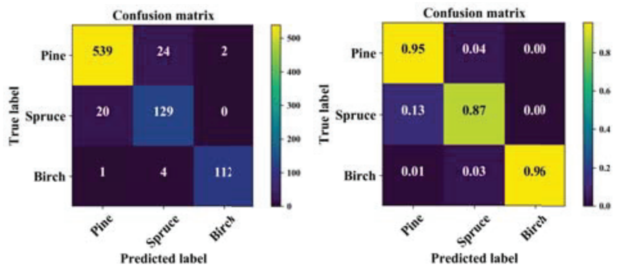


Fig. 3. Confusion matrices show good separation between tree species.

Figure 4 is presenting average saliency maps in spatial domain over all input bands of validation data. It seems that most of the important features are handling data surrounding tree top. This is shown more clear in the figure 5, where figure's 4 maps are rendered over validation sets average 3D treetops. Thus, we can be quite confident that, at least in spatial domain, tree top's shape is relevant feature in classification.

In spectral domain most characterising features seems to be located between wavelengths from 600 to 720 nm. Figure 6 presents average salience in each spectral band. It can seen that there is differences between tree species. For example birch has lower saliency in 560 nm and higher in 700 nm than coniferous trees.

If we consider individual trees, it seems that classifying is working quite efficiently. In figure 7 there is one tree of each species from the validation set. It can be seen that for example pine in this case doesn't have very clear treetop, but classifier is able to find one and saliency map seems to confirm the result.

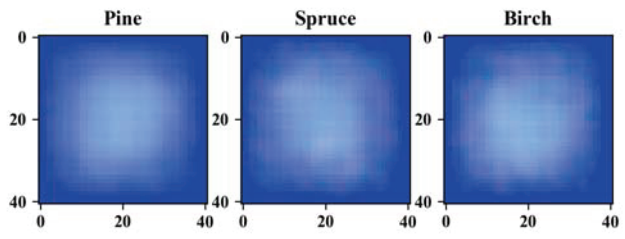


Fig. 4. Average saliency maps in spatial domain over all input bands of validation data for each tree species. Brighter pixel indicates that band is probably more meaningful in classification.

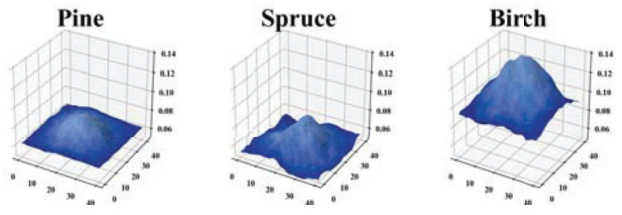


Fig. 5. Here figure's 4 saliency maps are rendered over validation set's average 3D treetops. It can be seen that maps surround quite well treetops.

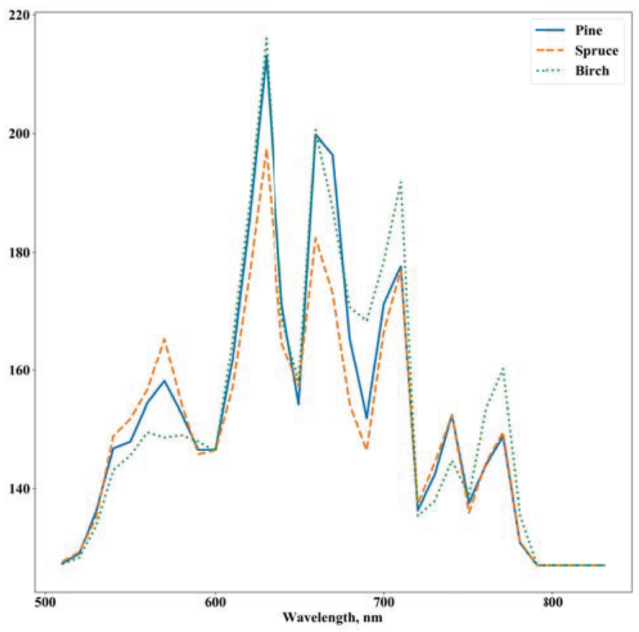


Fig. 6. Average saliencies of spectral domain for each tree species. Higher value indicates that band is probably more meaningful in classification.

4. CONCLUSIONS

In this paper we demonstrate how 3D hyperspectral data can be analysed using 3D convolutional neural networks. As a concluded result we can see that even with quite simple 3D CNN, it is possible to create network, which has good capability to classify single trees based on their shape and spectral features.

In classical machine learning one of the most time consuming thing for data analysist has been feature extraction and selection. In case of convolutional neural network this phase is now automated. After preprocessing there is quite limited amount of things to do, if you want to utilize trained network. Network training itself is time consuming, but before hand trained network can deliver results almost in realtime. In our case training took two and half hours.

Compared to earlier work [1] we actually used all captured test areas. In original paper one area was left behind, because of the poor quality of image block. Based on that, our results seems to show that trained 3D CNN is actually more robust as a classifier than methods used in previous study.

It is obvious that more studies is needed. Used network structure is one of the most simple ones. With more sophisticated structures it might be possible to improve learning results. One of the tested things in the future is, how general trained model actually is. If we have another data set, can we have similar classification results? We used quite limited amount of data augmentation. Even tough overfitting wasn't observed based loss and accuracy curves during the training, it would be useful to do more cross-validation within data.

One potential research question is that how many bands and what GSD is needed, if we want to gain similar results. Our next steps include more augmented data to training such as scaling, adding noise, chancing lightness and adding more rotation to see if we could detect trees also with lower resolution.

The used data set has more parameters for single trees (height, estimated volume, etc..) and there is also 300 fixed radius (9 m) sample plots, which have been used for area based forest inventory. In near future we will also test how well 3D CNN approach is able to estimate these parameters.

Our consortium has ongoing research project where our aim is to produce real time processing for the DSM and hyperspectral mosaics. This combined with pre-trained CNN classifier, could be significant tool to provide forest tree identification and parameter estimation without wasting time on massive preprocessing.

5. REFERENCES

[1] Olli Nevalainen, Eija Honkavaara, Sakari Tuominen, Niko Viljanen, Teemu Hakala, Xiaowei Yu, Juha Hyyppä, Heikki Saari, Ilkka Pölönen, Nilton N Imai, et al., "Individual tree detection and classification with uav-based photogrammetric point clouds and hy-

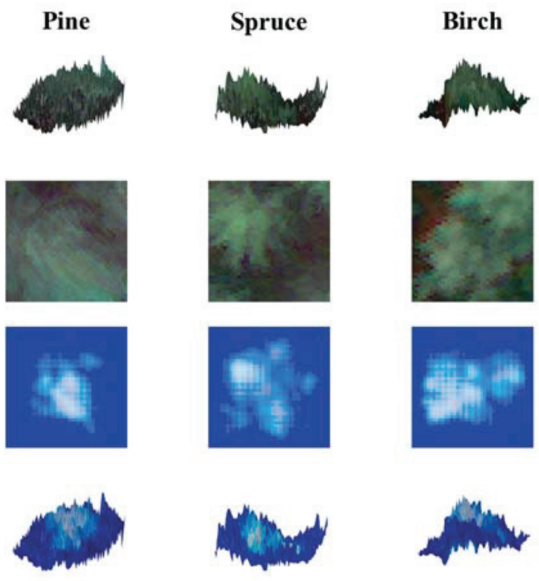


Fig. 7. Comparison individual trees. CNN is capable to detect trees surprisingly well.

perspectral imaging," *Remote Sensing*, vol. 9, no. 3, pp. 185, 2017.

[2] Erik Næsset, "Predicting forest stand characteristics with airborne scanning laser using a practical two-stage procedure and field data," *Remote sensing of environment*, vol. 80, no. 1, pp. 88–99, 2002.

[3] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[4] Sungbin Choi, "Plant identification with deep convolutional neural network: Snumedinfo at lifeclef plant identification task 2015.," in *CLEF (Working Notes)*, 2015.

[5] Luiz G Hafemann, Luiz S Oliveira, and Paulo Cavalin, "Forest species recognition using deep convolutional neural networks," in *Pattern Recognition (ICPR), 2014 22nd International Conference on*. IEEE, 2014, pp. 1103–1107.

[6] Mariusz Bojarski, Philip Yeres, Anna Choromanska, Krzysztof Choromanski, Bernhard Firner, Lawrence Jackel, and Urs Muller, "Explaining how a deep neural network trained with end-to-end learning steers a car," *arXiv preprint arXiv:1704.07911*, 2017.

[7] Heikki Saari, Ilkka Pölönen, Heikki Salo, Eija Honkavaara, Teemu Hakala, Christer Holmlund, Jussi Mäkynen, Rami Mannila, Tapani Antila, and Altti Akujärvi, "Miniaturized hyperspectral imager calibration and uav flight campaigns," 2013, vol. 8889, pp. 88891O–88891O–12.

[8] L. Markelin, E. Honkavaara, R. Näsi, K. Nurminen, and T. Hakala, "Geometric processing workflow for vertical and oblique hyperspectral frame images collected using uav," *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XL-3, pp. 205–210, 2014.

[9] Eija Honkavaara, Heikki Saari, Jere Kaivosoja, Ilkka Pölönen, Teemu Hakala, Paula Litkey, Jussi Mäkynen, and Liisa Pesonen, "Processing and assessment of spectrometric, stereoscopic imagery collected using a lightweight uav spectral camera for precision agriculture," *Remote Sensing*, vol. 5, no. 10, pp. 5006–5039, 2013.

[10] Yann LeCun and Yoshua Bengio, "Convolutional networks for images, speech, and time-series," 1995.

[11] Xavier Glorot, Antoine Bordes, and Yoshua Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011, pp. 315–323.

[12] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, Software available from tensorflow.org.

[13] François Chollet et al., "Keras," 2015.

[14] Raghavendra Kotikalapudi and contributors, "keras-vis," https://github.com/raghakot/keras-vis, 2017.

PIII

CONVOLUTIONAL NEURAL NETWORKS IN SKIN CANCER DETECTION USING SPATIAL AND SPECTRAL DOMAIN

by

Ilkka Pölönen, Samuli Rahkonen, **Leevi Annala** and Noora Neittaanmäki 2019

Photonics in Dermatology and Plastic Surgery 2019, International Society for Optics and Photonics

https://doi.org/10.1117/12.2509871

Convolutional neural networks in skin cancer detection using spatial and spectral domain

Ilkka Pölönen[a], Samuli Rahkonen[a], Leevi Annala[a], and Noora Neittaanmäki[b]

[a]Faculty of Information Technology, University of Jyväskylä, Mattilanniemi 2, Jyväskylä, Finland
[b]Departments of Pathology and Dermatology, Institutes of Biomedicine and Clinical Sciences, Sahlgrenska Academy, University of Gothenburg, Gothenburg, Sweden

ABSTRACT

Skin cancers are world wide deathly health problem, where significant life and cost savings could be achieved if detection of cancer can be done in early phase. Hypespectral imaging is prominent tool for non-invasive screening. In this study we compare how use of both spectral and spatial domain increase classification performance of convolutional neural networks. We compare five different neural network architectures for real patient data. Our models gain same or slightly better positive predictive value as clinicians. Towards more general and reliable model more data is needed and collection of training data should be systematic.

Keywords: Hyperspectral imaging, convolutional neural network, skin cancer, melanoma

1. INTRODUCTION

Skin cancers are constantly increasing problem world wide. Traditionally this has been concern of people whose skin is relatively lightly coloured and annual portion of sunlight is high. Because of increased traveling and ageing of the population, melanoma is increasing problem also in the Nordic countries. For example in Sweden,[1] 50 % of all the annual skin cancer related costs are caused by melanomas.

There is a need for tools, which are able to detect early stage skin cancers and delineate them properly from healthy tissue. With proper detection it is possible to reduce amount of re-surgeries, when part of the malignant tissue has been left to the patient in original tumor removal. This is highlighted by the fact that overall positive predictive value of clinical melanoma diagnosis is 33 %.[2] In non-specialised clinics this is even lower. For every melanoma removal there will be 9 to 30 non-melanoma lesions removed depending on how specialised clinic is.[3] Thus, early detection will lower the treatment costs and will ensure higher survival rate.

Hypersepctral imaging is method where hundreds narrow wavebands of light are imaged simultaneously. This method will provide almost continuous spectrum for each pixel of the image as figure 1 is showing. Hyperspectral imaging is non-invasive imaging modality, because it is using only visible and near infra-red illumination to capture images. Previously we have used it in delineation of tumor border and distinguish in-situ melanoma from malignant melanoma.[4,5]

If you look at closely two spectra in the figure 1 , it is quite easy to see that in clear cases melanoma and healthy skin have characteristic spectra. Unfortunately this is not so in all the cases. In figure 2 we have spectral distributions of malignant melanoma, lentigo-maligna, dysplastic nevus and benign nevus. We can see that these distributions are overlapping. This means that if the melanoma is hard to recognise in clinical study, it will be hard distinguish using just spectral information. Thus, it seems natural that we also utilize spatial domain in the classification task.

Further author information: (Send correspondence to Ilkka Pölönen)
Ilkka Pölönen: E-mail: ilkka.polonen@jyu.fi, Telephone: +358 400 248 140
Samuli Rahkonen: E-mail: samuli.rahkonen@jyu.fi
Leevi Annala: E-mail: leevi.a.annala@jyu.fi
Noora Neittaanmäki: E-mail: noora.neittaanmaki@fimnet.fi

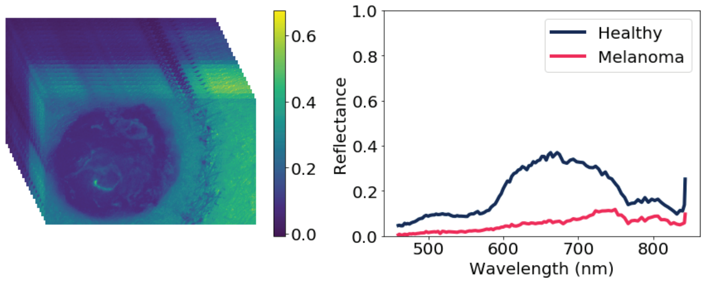


Figure 1. Hyperspectral image consist of intensity images, which are narrow wavebands from visible and near-infra red region of the electromagnetic spectrum. Each pixel in the image is spectrum through spectral domain.

Convolutional neural networks have shown great success in different kind of pattern recognition tasks.[6–8] They have also been recently used in classifying melanomas and other skin cancers from dermatoscope and regular color images.[9] In these cases, results are given for the whole images. Because of such binary classification we don't actually have an opportunity to determinate lesion's borders from analysis or what kind of other irregularities there are in the tumor.

There are multiple strategies to utilize convolutional neural networks. We are introducing efficient strategy, which contains utilization of both spectral and spatial domain. With hyperspectral data containing wavebands from visible to infra-red region, we are able to gather more information from each pixel than using regular imaging systems.[4,5] Using sliding window method over captured spectral image, we will have spectral and spatial domain for further analysis. In this study we describe some incremental steps, which are taking us closer to automatic skin cancer detection, identification and delineation.

2. MATERIAL AND METHODS

In this study we have small data set (n=61) of hyperspectral images covering narrow wavebands from 450 to 850 nm. Data set consist of several lesions, which were imaged and diagnosed by histopathology. Lesions consist of malignant melanomas, melanoma in-situs, dysplastic nevi and bening nevi. All patients have volunteered to participate in the study. The study protocol has followed the Declaration of Helsinki and it was approved by the local Ethics committee. Patient were recruited and imaged by the Department of Dermatology and Allergology of Helsinki University Hospital, Helsinki, Finland and by the Päijät-Häme Central Hospital, Lahti, Finland, between June 2016 and October 2017.

All hyperspectral images were collected with two identical hyperspectral imagers (Revenio Prototype 2016). Spectral separation of the imager is based on Fabry-Pérot interferometer (FPI). Use of FPI enables fast scanning in the spectral domain. The imager works on wavebands from 450 nm to 850 nm. The imager captures 120 wavebands within few seconds. Full width of each waveband's half maximum (FWHM) vary from 5 to 15 nm. Variation in FWMH comes as a function of wavelength. Another source of the variation comes from which multiple of FPI's is used. Imaging system contains a broadband halogen light source, which produces diffuse illumination to the imaged region of the interest (ROI). At the imager there is covering tube, which blocks illumination from other sources. Image acquisition is done with color cmos machine vision camera, which is integrated to the imager. The used machine vision camera is capable to take images in 1920×1200 pixel resolution. This corresponds approximately to 15 μm/pixel spatial resolution.

The spectral imager produces a raw data cube, which is calibrated to the radiance by following method of Saari et. al (2013).[10] There was some indeterminated fluctuation at the end of recorded spectra. Thus, twenty last wavebands were left outside from further analysis. For each data cube there was captured white reference target. This was used to convert imaged radiance to reflectance $R = I/I_0$, where I is imaged region of interest and I_0 is data cube from white reference. To improve quality of the data in spectral domain and reduce memory consumption in further processing, the data is downsampled. This was done by averaging nearest pixels of every fifth pixel. Also, only every second waveband was used in further analysis. By these operations data cubes size reduced to $384 \times 240 \times 50$ pixels.

Training of the classifier needs labelled data. For each image there were annotated areas, which indicated either healthy skin, lesion or used marker. From each image's annotated areas 1000 data sets (or less if annotated area contained less than 1000 pixels) were selected for training purposes. These data sets contained annotated pixel and its 10×10 neighborhood. Figure 2 shows distribution of spectra of melanoma, lentigo maligna, dysplastic nevus and benign nevus. As we can see that there are overlapping in the distributions and some of the spectra has deviation. To reduce these effects and some problems from the vignetting and lightning irregularities, each imaged spectrum was subtracted by its average in spectral domain.

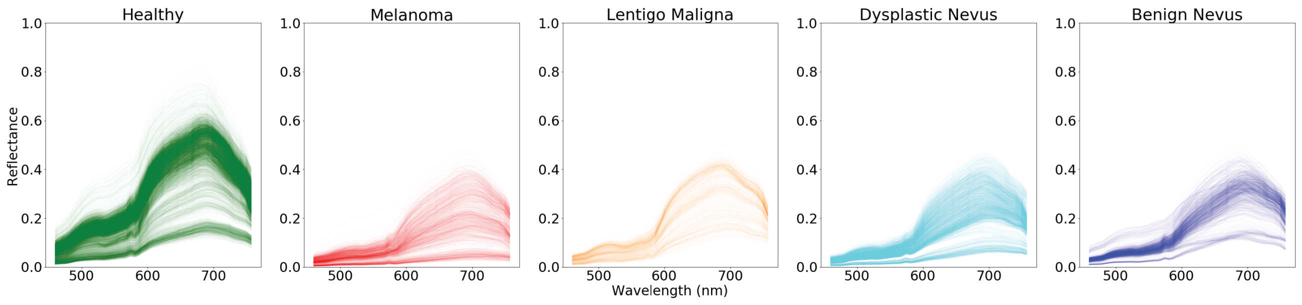


Figure 2. The distributions of spectra of melanoma, lentigo maligna, dysplastic nevus and benign nevus.

During recent years, deep neural networks have made new records in pattern recognition.[6] Our aim was to use both spatial and spectral domain simultaneously. Convolutional neural networks have been used to classify melanomas and other skin cancers from dermatoscope and regular color images.[9] Spectral data cube has three-dimensional nature, thus, standard 2D convolutional neural network might not be enough to utilize spectral data.

In deep learning and especially with convolutional neural networks classification task has two parts. In the feature learning we are calculating features using convolution operations with different weights. By tuning weights during back-propagation we will eventually achieve optimized feature space for our classification task. The actual classification model is just a deep regular multi-layer perceptron network. This structure is illustrated in the figure 3.

In this study we tested three different kind of feature learning structures - 1D, 2D and 3D convolutions. We will also have basically two different types of inputs. Single spectra and small window surrounding this spectra. As figure 3 shows, a 1D convolution input takes a single spectra. For 2D and 3D convolution input will be a subset's of spectral cube. Difference between 2D and 3D convolution is that 2D case doesn't operate over spectral domain while 3D does.

Used deep neural networks consist from two parts. First part executes feature learning by using the convolutional operator by the Conv layers. The Maxpooling layers reduces data's dimensionality. Machine learning part and actual classification is done with deep feed-forward neural network, which consist of six Dense layers. Convolutional and dense layers use rectified linear unit activation (ReLU) function. The single Dropout layer is added to avoid overfitting of the model. Last dense layer does final classification using Softmax activation function. Parameters of each layer are shown in the figure 3.

By variating the described architecture we tested five different kind of networks - 1D, 2D and 3D convolutional neural networks and two combinations where feature learning was executed by using 3D+1D convolutions and 3D+2D+1D convolutions.

The actual training data was sampled randomly from annotated points, so that there were 10000 data points from each class. An annotation was based on its histopathological results. Whole lesions were marked the same way. Annotation was done by a non-expert.

For annotated points, data augmentations was utilized so, that each training cube was mirrored and flipped horizontally and vertically. These operations fourfold the number of the inputs in the training phase. Training

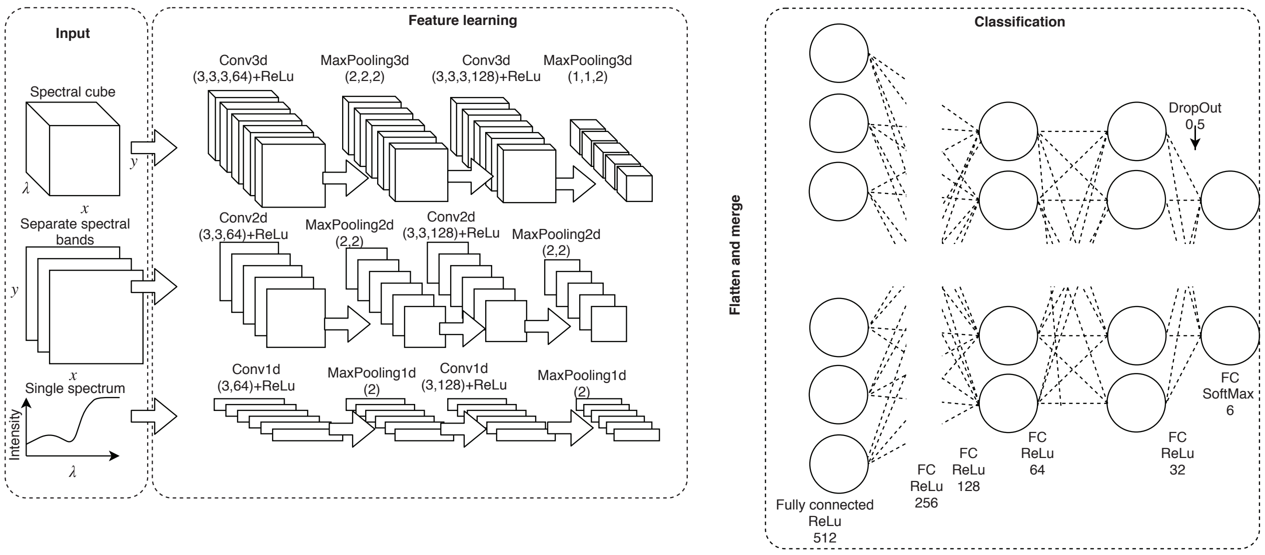


Figure 3. Schematic structure of used convolutional neural networks. Best results were gained using all inputs and all three different convolutional feature learning parts simultaneously.

set consisted of approximately 240 000 data points. For the optimization we used Adam, which is a first-order gradient-based optimization method of stochastic objective functions. The used hyperparameters for the optimization was the learning rate of 0.001, $\beta_1 = 0.9$ and $\beta_2 = 0.999$, while the learning rate decay over each update stayed at 0. The used cost function was categorical cross-entropy.

Our implementation used Keras with Tensorflow backend and Python 3.6 . All calculations were executed using IBM PowerAI platform, which includes two Nvidia Tesla V100-SXM2 16 GB GPU units.

There were only 61 imaged lesions (15 malignant melanoma, 6 lentigo maligna, 26 dysplastic nevus and 14 benign nevus). Thus, leave-one-out cross-validation was used. In this procedure classifier is trained 61 times for each image separately. This will guarantee that training set does not include data points from the image which is currently under classification.

3. RESULTS

Our ground truth consist of the results of histopathology. This meant that whole lesion was labelled based on most dangerous diagnosis. Because our approach gives us pixel wise information we ended situations where one lesion had several differently classified pixels. This actually might be quite realistic situation. Malignant lesions can have non-malignant parts. Thus, final classification for each lesion was made based on most dangerous pixel, which was found from lesion. If there was even a single pixel, which was classified as melanoma, whole lesion was classified to melanoma. In melanoma detection with this approach we will gain relatively high sensitivity, but low specificity which is seen in table 1 and in figure 8. And as opposite for benign nevus will have high specificity and low sensitivity.

Table 1. Sensitivity, specificity and positive predictive value of different classifiers for the melanoma classification

	CNN 1D	CNN 2D	CNN 3D	CNN 3D+1D	CNN 3D+2D+1D
Sensitivity	1	1	0.93	0.93	0.93
Specificity	0.15	0.12	0.14	0.14	0.21
Positive predictive value	0.34	0.35	0.32	0.32	0.34

When we are looking at sensitivities of different classifiers, we can see, that all 15 melanoma cases actually were classified correctly using only 1D and 2D convolution networks. 14 of 15 melanoma cases were classified correctly when 3D convolution was utilised. These metrics are actually misleading. If we look at actual classification results as shown in figures 4, 5, 6 and 7, we can see that actually classification results based on single spectra are often noisy. Figure 4 was confirmed to be dysplastic nevus in histopathology. All single source convolutional neural networks fail to classify it correctly. On lesion boundaries there is quite typical error, where trained model for some reason mis-classify lesion to melanoma. Here best result is achieved using multiple inputs and three different kind on CNN's.

In general level it seems that spatial features give more reliable looking results. When we combine those with spectral domain, results get better, because specificity increases. If we take closer look one false positive case in the figure 7 we can see that majority of the pixels in the lesion is actually classified correctly. This is promising result because with more training cases we might have chance to train better models.

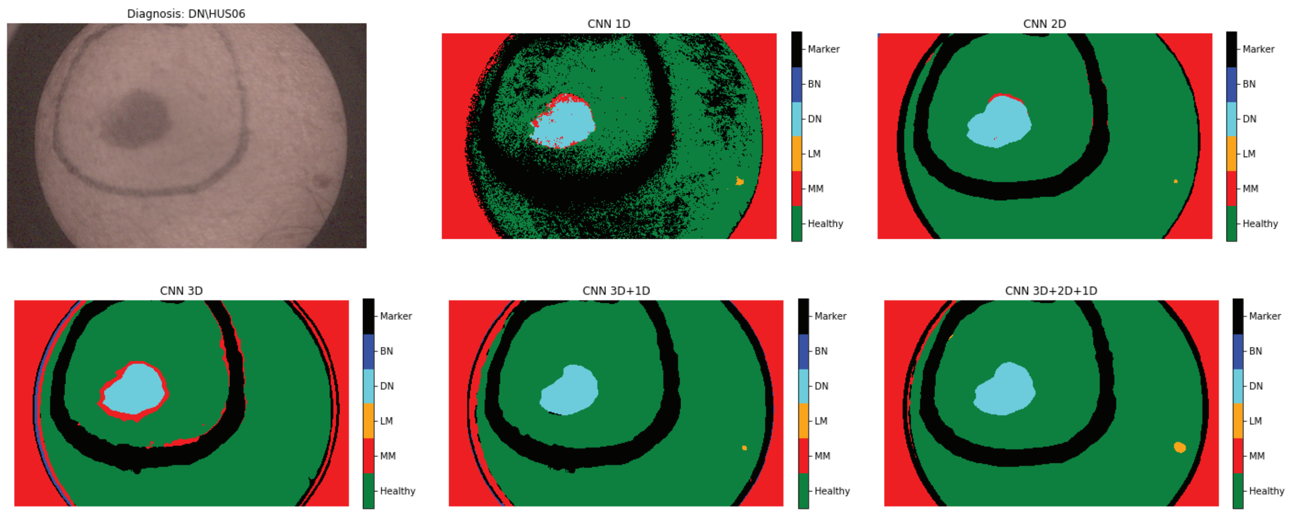


Figure 4. Classification results of five different classifiers for dysplastic nevus.

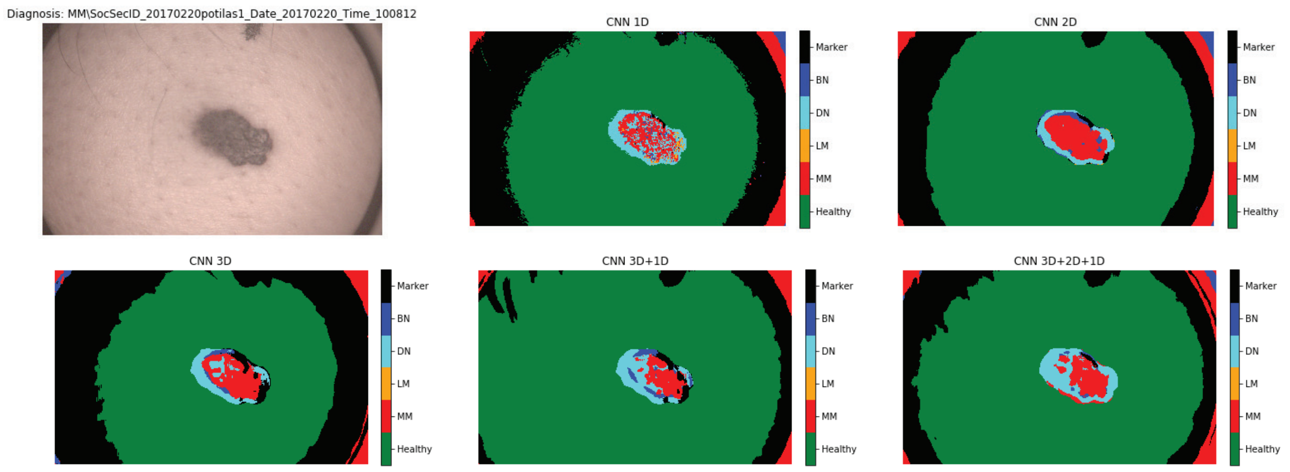

Diagnosis: MM\SocSecID_20170220potilas1_Date_20170220_Time_100812
CNN 1D
CNN 2D
CNN 3D
CNN 3D+1D
CNN 3D+2D+1D
Marker
BN
DN
LM
MM
Healthy

Figure 5. Classification results of five different classifiers for malignant melanoma.

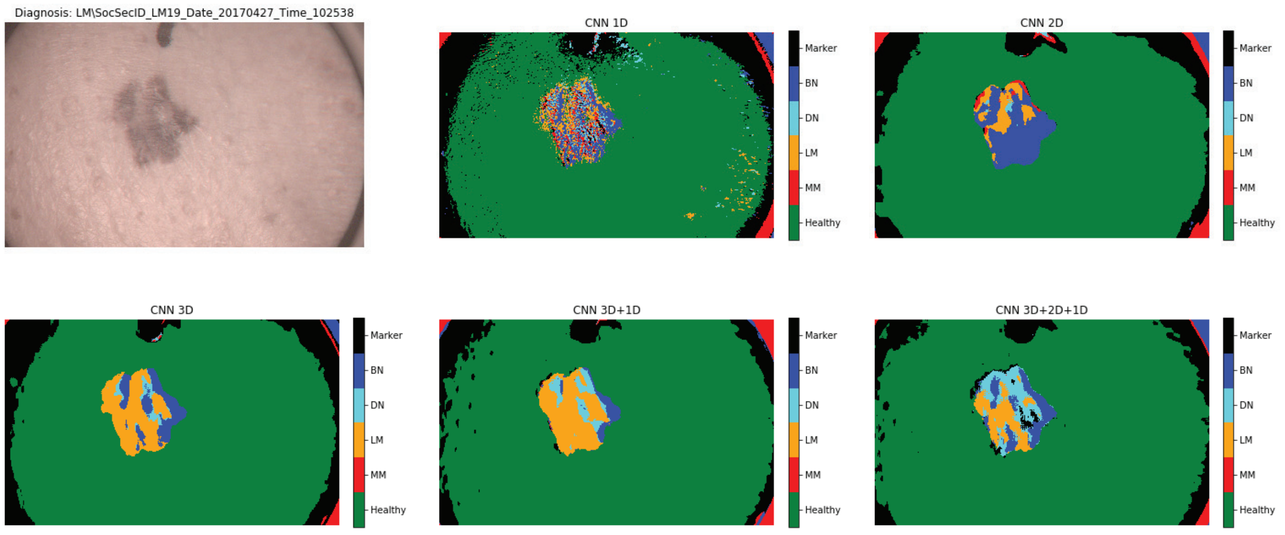


Figure 6. Classification results of five different classifiers for lentigo maligna.

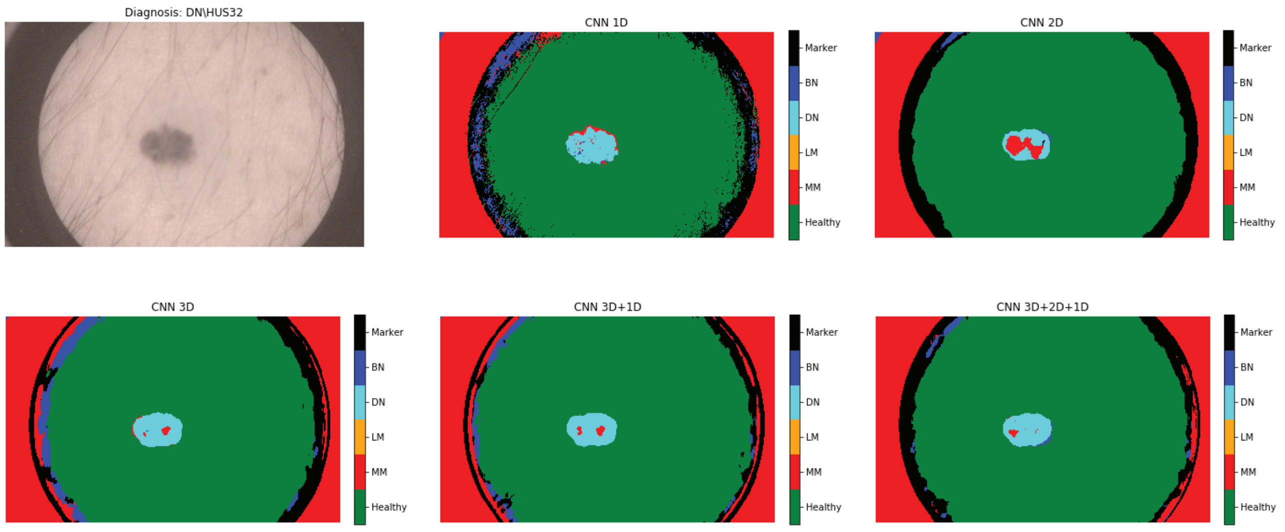


Figure 7. Classification results of five different classifiers for dysplatic nevus. Here all classifiers give false positive as a result. Even though majority of pixels are classified correctly, the end result will be false positive for whole lesion.

4. DISCUSSION

Shown results are promising. With all classifiers we achieved same positive prediction value (PPV) as clinicians. It is shown that utilisation of the spectral and spatial domain increases classification performance.

There is work to be done to gain higher specificity and PPV. We could play around with detection probabilities provided by the softmax layer and take some threshold probabilities, which would be concerned during classification (for example only classification results over 90% confident would be recognised). Or we could calculate which class has majority of pixels on lesion area. Unfortunately both of these approaches would actually decrease the sensitivity and the number of false negatives would rise.

Our study's first limitation comes from the small data set. Even thought we had over hundred million pixels at our disposal, we eventually had only 61 different lesions. This is a quite limited data set and more data is needed to develop and calculate a more robust and accurate neural network model. This would mean that we will need multi center studies, where patient data is gathered in several countries simultaneously. For example Finnish population is too small to produce enough patients to train enough general models.

Another limitation is that the ground truth labeling is based on histopathological diagnosis of whole lesion. There is a great possibility that a lesion can include several classes. Thus, our ground truth contains bias and this bias is also transferred to our training data. What we actually should do is that we should have several biopsied training points from each lesion so that we could use those spots in our training data. This would decrease bias in the training data, but it would also lead to reduced training data size.

Process of validating results and gathering training data should be similarly iterative as training of the neural network itself. When a hyperspectral imager and a classification model is used in a clinical study, we should take biopsies based on results. The spatial locations of these biopsies should be saved and the model should be updated using histopathological results of these studies afterwards.

The approach to use spectral and spatial domains seems feasible. Our next ideas are to add more features to the data. By modifying the illumination source we can take photogrammetric stereo images. From these images it is possible to calculate surface normals, a digital elevation model and skin's albedo as a function of wavelength. Each of these can be used as new features in cancer classification and delineation.

5. CONCLUSION

We have shown that use of spectral and spatial domain will increase classification performance of convolutional neural network. Our results show that with a relative small data set we are able to get same or slightly better positive prediction values as clinicians. This information was achieved by using a novel hyperspectral imager prototype in a clinical setup and train five different neural network models based on histopathological diagnoses. Because of the climate change proportion direct of sun radiation seems to grow, thus non-invasive automatic skin cancer detection and delineation systems will be needed even more in the future. These results are incremental steps towards this goal.

REFERENCES

[1] Eriksson, T. and Tinghög, G., "Societal cost of skin cancer in sweden in 2011," *Acta dermato-venereologica* **95**(3), 347–348 (2015).

[2] Heal, C. F., Raasch, B. A., Buettner, P., and Weedon, D., "Accuracy of clinical diagnosis of skin lesions," *British Journal of Dermatology* **159**(3), 661–668 (2008).

[3] Argenziano, G., Cerroni, L., Zalaudek, I., Staibano, S., Hofmann-Wellenhof, R., Arpaia, N., Bakos, R. M., Balme, B., Bandic, J., Bandelloni, R., et al., "Accuracy in melanoma detection: a 10-year multicenter survey," *Journal of the American Academy of Dermatology* **67**(1), 54–59 (2012).

[4] Neittaanmäki-Perttu, N., Grönroos, M., Jeskanen, L., Pölönen, I., Ranki, A., Saksela, O., and Snellman, E., "Delineating margins of lentigo maligna using a hyperspectral imaging system," *Acta dermato-venereologica* **95**(5), 549–552 (2015).

[5] Neittaanmäki, N., Salmivuori, M., Pölönen, I., Jeskanen, L., Ranki, A., Saksela, O., Snellman, E., and Gronroos, M., "Hyperspectral imaging in detecting dermal invasion in lentigo maligna melanoma," *Br J Dermatol* (2016).

[6] Krizhevsky, A., Sutskever, I., and Hinton, G. E., "Imagenet classification with deep convolutional neural networks," in [*Advances in neural information processing systems*], 1097–1105 (2012).

[7] Lawrence, S., Giles, C. L., Tsoi, A. C., and Back, A. D., "Face recognition: A convolutional neural-network approach," *IEEE transactions on neural networks* **8**(1), 98–113 (1997).

[8] Kim, Y., "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882* (2014).

[9] Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M., and Thrun, S., "Dermatologist-level classification of skin cancer with deep neural networks," *Nature* **542**(7639), 115 (2017).

[10] Saari, H., Pölönen, I., Salo, H., Honkavaara, E., Hakala, T., Holmlund, C., Mäkynen, J., Mannila, R., Antila, T., and Akujärvi, A., "Miniaturized hyperspectral imager calibration and uav flight campaigns," in [*Sensors, Systems, and Next-Generation Satellites XVII*], **8889**, 88891O, International Society for Optics and Photonics (2013).

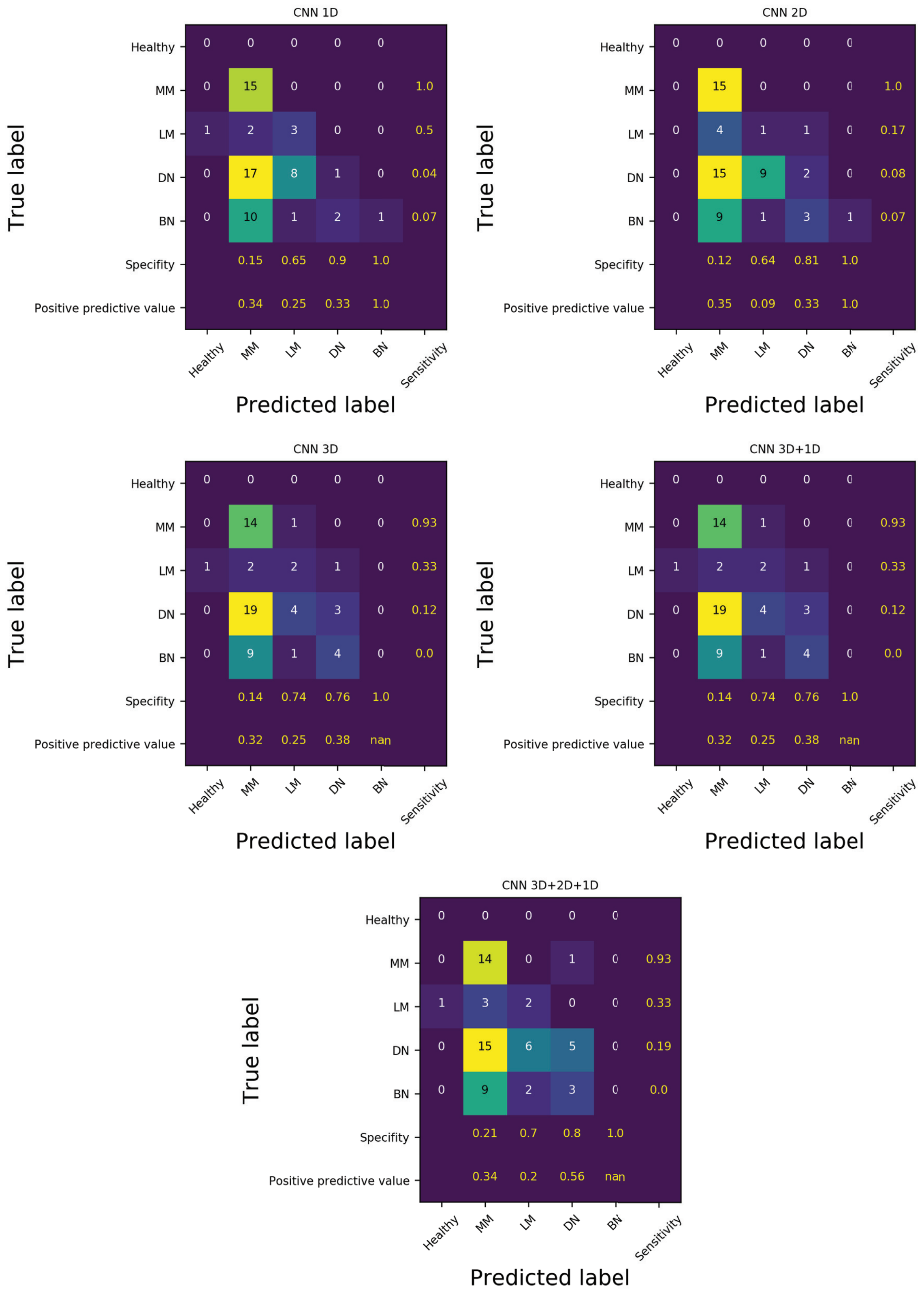


Figure 8. Confusion matrices for the different convolutional neural network models.

PIV

CHLOROPHYLL CONCENTRATION RETRIEVAL BY TRAINING CONVOLUTIONAL NEURAL NETWORK FOR STOCHASTIC MODEL OF LEAF OPTICAL PROPERTIES (SLOP) INVERSION

by

Leevi Annala, Eija Honkavaara, Sakari Tuominen and Ilkka Pölönen 2020

Remote Sensing, 12(2):283

https://doi.org/10.3390/rs12020283

remote sensing

Article

Chlorophyll Concentration Retrieval by Training Convolutional Neural Network for Stochastic Model of Leaf Optical Properties (SLOP) Inversion

Leevi Annala [1,*], Eija Honkavaara [2], Sakari Tuominen [3] and Ilkka Pölönen [1]

[1] Faculty of Information Technology, University of Jyväskylä, FI-40014 Jyväskylä, Finland; ilkka.polonen@jyu.fi
[2] Finnish Geospatial Research Institute, National Land Survey of Finland, FI-02430 Masala, Finland; eija.honkavaara@nls.fi
[3] Natural Resources Institute Finland, FI-00790 Helsinki, Finland; sakari.tuominen@luke.fi
[*] Correspondence: leevi.a.annala@jyu.fi; Tel.: +358-45-890-3185

Received: 24 October 2019; Accepted: 4 January 2020; Published: 15 January 2020

Abstract: Miniaturized hyperspectral imaging techniques have developed rapidly in recent years and have become widely available for different applications. Combining calibrated hyperspectral imagery with inverse physically based reflectance models is an interesting approach for estimating chlorophyll concentrations that are good indicators of vegetation health. The objective of this study was to develop a novel approach for retrieving chlorophyll *a* and *b* values from remotely sensed data by inverting the stochastic model of leaf optical properties using a one-dimensional convolutional neural network. The inversion results and retrieved values are validated in two ways: A classical machine learning validation dataset and calculating chlorophyll maps from empirical remotely sensed hyperspectral data and comparing them to $\frac{\text{TCARI}}{\text{OSAVI}}$, an index that has strong negative correlation with chlorophyll concentration. With the validation dataset, coefficients of determination (R^2) of 0.97 were obtained for chlorophyll *a* and 0.95 for chlorophyll *b*. The chlorophyll maps correlate with the $\frac{\text{TCARI}}{\text{OSAVI}}$ map. The correlation coefficient (R) is -0.87 for chlorophyll *a* and -0.68 for chlorophyll *b* in selected plots. These results indicate that the approach is highly promising approach for estimating vegetation chlorophyll content.

Keywords: optical properties; convolutional neural network; deep learning; chlorophyll; stochastic modeling; physical parameter retrieval; forestry

1. Introduction

Forests play an important role in the atmospheric carbon cycle. Growing forests serve as carbon sinks as well as a providers of various ecosystem services [1]. The condition of forests, however, is endangered by, among other factors, degradation by anthropogenic pressures on land use and forest resources. In addition, biotic factors such as insect damage and fungal diseases, along with abiotic risks due to climate factors (such as storm or drought), form a constant threat to the general health and carbon sequestration capability of forests [2,3].

In monitoring the health of forests, a key factor is their photosynthetic capacity. There exists several options for monitoring this capacity. One is to quantify the amount of chlorophyll *a* and *b*, which have been shown to have strong correlation with tree health [2]. Hyperspectral imaging technologies have developed greatly in recent years and are today available for practical use and are highly interesting for forest health monitoring [4].

Physical parameter retrieval from hyperspectral images is a difficult task that concerns estimating the values of different biophysical parameters from spectral data [5]. According to Verrelst et al., the methods for biophysical parameter retrieval can be divided into four categories [6,7]: parametric and non-parametric regression methods, physically based methods, and hybrid methods. Parametric regression methods directly take the spectral data and make estimations based on them, such as spectral indices. Non-parametric regression generally means using machine learning methods directly on the measured data. Physically based methods are based on physical cause–effect relationships of the interaction of light and matter. Hybrid methods are combinations of the previous three methods, in which a machine learning algorithm and a physically based model are typically used. A physically based model is inverted by teaching the machine learning algorithm using the outputs of the model as inputs and the inputs of the model as outputs. Once it has been taught, the machine learning algorithm can then receive inputs in the form of spectra and provide results in the form of a desired parameter value. In this article we develop a methodology that belongs to the hybrid method category.

The hyperspectral datasets used in remote sensing applications can be based on imaging or non-imaging spectrometers [4]. The convolutional neural networks (CNNs) have been shown to be powerful in regression and classification tasks with both data types [8–12]. In such networks, the dense layers that utilize matrix multiplications are replaced with convolutional layers [13].

Chlorophyll retrieval from hyperspectral images of a forest by model inversion has been studied previously [14,15]. Croft et al., for example, found promising results for inverting a PROSPECT models [16] using lookup tables [14]. However, this approach inverts the function only point-wise, when it would be useful to have a more precise approximation of the true inverse function. In another example, Atzberger et al. [15] used artificial neural networks to invert the PROSPECT+SAIL model [17]. However, neural network research has progressed significantly since these attempts, and particularly, CNNs have become popular in recent years.

Machine learning and artificial neural networks have been previously used in different physical parameter retrieval applications. For example, the density and depth of snow cover have been retrieved utilizing microwave image satellite data and a model for brightness temperature, which was inverted using an artificial neural network. The results were verified with on-site measurements with good accuracy [18]. Notarnicola et al. compared two methods, neural network and Bayesian method, for soil moisture retrieval. In their study, training data were simulated and validation data were on-site measurements, and the neural network outperformed the Bayesian approach [19]. Trombetti et al. used a radiative transfer model inverted by an artificial neural network in retrieving the water content of a canopy. The results correlated well with the amount of rain in the research area [20]. In another study, a generalized radial basis function neural network was used to retrieve optically active parameters of seawater from hyperspectral images [21]. This study concluded that the neural network outperformed other algorithms that are typically used in the field.

Although there is a significant amount of research on using neural networks in physical parameter retrieval, there appears to be a lack of research on using CNNs in such retrieval. The objective of this study is to develop a model for non-invasive prediction of chlorophyll values. This is achieved by solving the inverse function of the stochastic model of leaf optical properties (SLOP) [22,23] with respect to chlorophyll a and b values by utilizing a one-dimensional convolutional neural network (1DCNN). Our proposed model utilizes the SLOP for CNN training, and then uses the trained network to predict the chlorophyll values in the hyperspectral image of a forest. We selected SLOP for inversion because it is mathematically appealing and it appears to have the potential to be more accurate when compared to plate or Kubelka–Munk models [24] while the simulations are still fast enough.

2. Materials and Methods

2.1. Stochastic Model of Leaf Optical Properties

SLOP is a stochastic model of leaf optical properties based on Markov chains. It was first introduced by Tucker and Garratt in 1977 [23] and improved by Maier et al. in 1999 [22]. It takes leaf properties (see Table 1) and wavelength as an input and calculates leaf reflectance, transmittance, and absorbance. The basic idea of the stochastic modeling approach is that the leaf is modeled as a network of different states and their connections. For each connection, there is a corresponding probability for transition between states. The possible transitions in a leaf are described in Figure 1. The black boxes are end states where the only possible transition is to itself. In each layer of the leaf (see Figure 1) the photon goes to the next layer (or out), goes to the previous layer (or out) due to scattering, or it is absorbed into the layer. Each of these can also happen after scattering and therefore each layer contains four basic states: down, up, absorbed, and scattered. In addition, there are two white illumination states—for input—and four end states on the outside of the leaf. For each of these events probabilities are calculated based on the layer structure and pigment concentrations with the help of Beer's law. For each transition, the probability is calculated as follows:

1. For each up (similarly down) state, the up (down), scattering and absorption probabilities are

$$P_{\text{absorption}}(\lambda) = \frac{a(\lambda)}{a(\lambda)+s} \cdot (1 - e^{-(a(\lambda)+s)\cdot L}), \tag{1}$$

$$P_{\text{scattering}}(\lambda) = \frac{s}{a(\lambda)+s} \cdot (1 - e^{-(a(\lambda)+s)\cdot L}), \tag{2}$$

$$P_{\text{up (down)}}(\lambda) = 1 - P_{\text{absorption}} - P_{\text{scattering}}. \tag{3}$$

2. For each scattered state the scattering and absorption probabilities are same as for the up and down states on the same layer. The up and down probabilities are

$$P_{\text{down}}(\lambda) = P_{\text{up}}(\lambda) = \frac{1 - P_{\text{absorption}} - P_{\text{scattering}}}{2}. \tag{4}$$

3. The probability of direct reflection is given as a parameter and the probability of entering the first layer is $1 - P_{\text{direct reflection}}$.
4. The probability of going from absorbed state, reflected state, or emitted state to itself is 1.
5. All other transition probabilities are 0.

In the previous equations,

- $a(\lambda)$ is the absorption coefficient,

$$a(\lambda) = \frac{\pi}{4} \cdot (1 + \frac{2e^{-\rho(\lambda)}}{\rho(\lambda)} + \frac{2(e^{-\rho(\lambda)} - 1)}{\rho(\lambda)^2}) \cdot d^2_{\text{chloroplast}} \cdot c_{\text{chloroplast}} + a_{H_2O}(\lambda) \cdot w_{H_2O}, \tag{5}$$

where

$$\rho(\lambda) = \frac{6}{\pi} \cdot \frac{1}{d^2_{\text{chloroplast}} \cdot c_{\text{chloroplast}}} \sum_{\text{pigments}} a_i \cdot c_i, \tag{6}$$

- s is the scattering coefficient,
- L is the length of the light path, which is assumed to be the same as the thickness of the layer,
- a_i are the absorption coefficients of the pigments (chlorophyll a, chlorophyll b, beta-carotene, lutein, violaxanthin, neoxanthin) [22,25–29] and
- c_i are their concentrations,
- a_{H_2O} and w_{H_2O} are the absorption coefficient [30–32] and volume concentration (cm^3/cm^3) of liquid water, and
- $d_{\text{chloroplast}}$ is the diameter of chloroplast (cm) and $c_{\text{chloroplast}}$ is its concentration (1/cm^3).

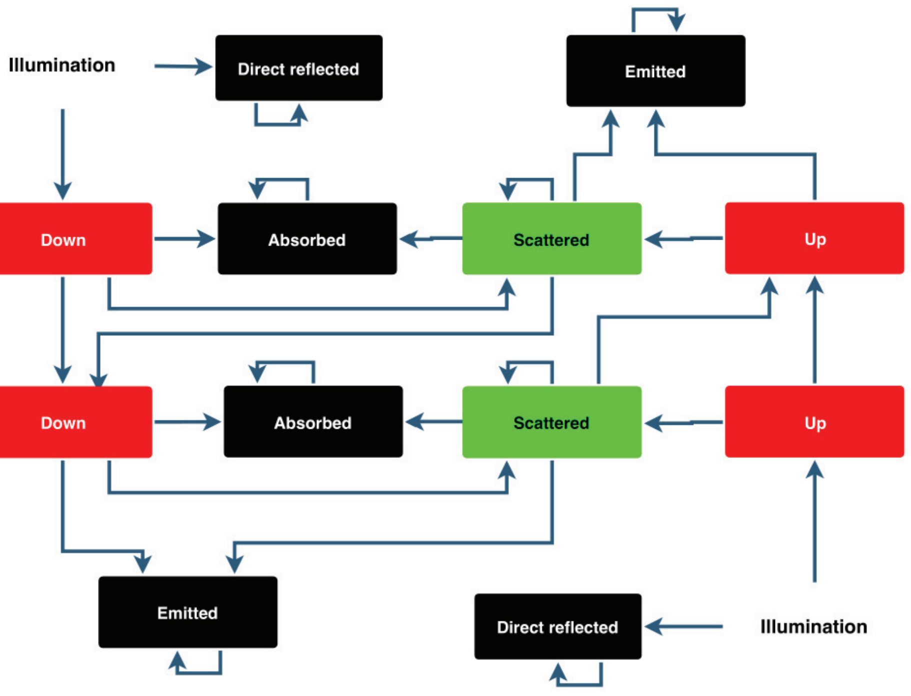


Figure 1. Diagram representation of stochastic model of leaf optical properties (SLOP). A tree leaf is assumed to have two major layers: a palisade layer and a spongy layer. In both layers, a photon can go straight through, be absorbed or scatter until it is absorbed or it moves to the previous or next layer. Adapted from [22].

Table 1 shows the inputs and constants SLOP uses to calculate the previously introduced probabilities.

Table 1. Constants and variables used in making training, testing, and validation data with SLOP. The training, testing, and validation data consist of 500,000 spectra made with SLOP. Each spectrum is produced by taking a random value from each interval in the table and calculating SLOP for each specified wavelength. The spectra are divided into training, testing, and validation data randomly, with constant sizes.

		Leaf Layer	
		Palisade	**Spongy**
Variables	Chlorophyll *a* concentration (mg/cm^3)	[1, 10]	[0, 4]
	Chlorophyll *b* concentration (mg/cm^3)	[0.5, 5.5]	[0, 3]
	β-carotene concentration (mg/cm^3)	[0, 1]	[0, 0.5]
	Lutein concentration (mg/cm^3)	[0, 1]	[0, 0.5]
	Violaxanthin concentration (mg/cm^3)	[0, 0.5]	[0, 0.25]
	Neoxanthin concentration (mg/cm^3)	[0, 0.5]	[0, 0.25]
	Water content (cm^3/cm^3)	[0.8, 1]	[0.1, 0.5]
	scattering coefficient (1/cm)	[3.5, 5.5]	[1000, 1100]
	Probability of direct reflection	[0.4, 0.06]	
Constants	Chloroplast diameter (cm)	0.0005	
	Chloroplast concentration (1/cm^3)	5×10^{-9}	6.7×10^{-8}
	Thickness (cm)	0.0069	0.0069

Mathematically, the Markov chain is handled as a matrix multiplication routine. The algorithm is the following:

1. Initialize the state vector: The initial state of the network of states, for example "all photons coming from above, none inside" corresponds to the following state vector:

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_k \end{bmatrix}_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

2. Matrix multiplication: The new state vector is the transition probability matrix multiplied with the old state vector:

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_k \end{bmatrix}_{n+1} = \begin{bmatrix} P_{0,0} & P_{0,1} & \dots & P_{0,k} \\ \vdots & \vdots & \dots & \vdots \\ P_{k,0} & P_{k,1} & \dots & P_{k,k} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_k \end{bmatrix}_n$$

3. Check the end condition: If the new and old state vectors are close enough to each other, end; otherwise, repeat from step 2. If

$$\left| \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_k \end{bmatrix}_{n+1} - \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_k \end{bmatrix}_n \right| < \epsilon,$$

then each x_i in the final state vector corresponds to the probability that the photon will end up in that state.

In this research, SLOP was used in making training, testing, and validation datasets for 1DCNN training. The input wavelengths depended on the validation scheme. With simulated validation data wavelengths from 400 nm to 2500 nm with 10 nm spacing were used and with empirical validation data the first 19 wavelengths from Table 2 were used. The other inputs for SLOP are described in Table 1. Example spectra produced with SLOP can be seen in Figure 2.

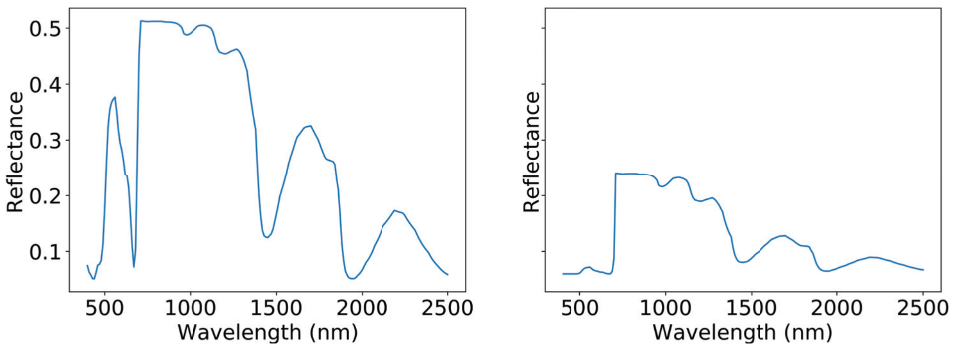


Figure 2. Two spectra produced with SLOP. The graph on the left is produced with minimum values from Table 1 while the graph on the right is produced with maximum values.

2.2. Convolutional Neural Network

Convolutional neural networks (CNN) were first introduced by LeCun and Bengio in 1995 [13]. Traditional dense neural networks with hidden layers are based on matrix multiplication [33], where a convolutional layer replaces multiplication with convolution. CNNs are considered especially powerful for image-related tasks, such as image classification [8,9], or regression problems concerning images, such as object detection [10]. Altogether, CNN has been found to work well with signals [11,12].

The used 1DCNN architecture is described in Table 3. It consisted of 1D convolution with 64 filters and a kernel size of 3, 1D pooling with a kernel size of 3, another 1D convolution with a kernel size of 3 and 128 filters, 15% dropout, a 100-filter dense layer and a dense output layer, and batch normalization [34] at the beginning and after each convolution layer. Input shape, and therefore the shape of each layer, depended on the shape of the training data. In this study, two different input data shapes were used. For the simulated validation data, the input data had a length of 210 and in the case of remotely sensed validation data it had a length of 19. Each layer that had an activation function used rectified linear unit activation [35]. The Adam optimizer was used [36], with a learning rate of 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-7}$. The loss score was mean square error (MSE) [37], and the accuracy score was the coefficient of determination or, as it is also called, the r^2-score [38]. The dropout layers and batch normalization were added to reduce overfitting and also to help transition from the simulated data to the noisy empirical data.

All neural network implementation was performed with a Keras [39] package using the TensorFlow [40] backend. The computer used in neural network training had an Nvidia GT1080 16 Gb GPU.

Training data for the 1DCNN was split from the dataset produced by SLOP. For simulated data validation these data were used as they were, and for the case with remotely sensed data some Gaussian noise (M = 0, SD = 0.025) was added to the data. In both cases chlorophyll *a* and *b* estimators were trained separately.

2.3. Empirical Dataset

The empirical test site was located in the Vesijako experimental forest area in the municipality of Padasjoki in southern Finland. The test site was covered by young to middle-aged forest dominated by birch (*Betula pendula*), with spruce (*Picea abies*) as a secondary tree species. The flight campaign was carried out on 26 June 2014 from 12:09 to 12:22 using a hexacopter UAV. The weather conditions were cloudy. The hyperspectral images were captured using a 2D frame format hyperspectral camera based on a tunable Fabry–Pérot interferometer (FPI). The flying height was 88 m from the ground level, providing an average ground sampling distance (GSD) of 8.8 cm for the FPI images at ground level; the flight height was 67 m from the tree treetops, giving a GSD of 6.7 cm at the treetops. Hyperspectral orthophoto mosaics were calculated with a 10 cm GSD and calibrated to reflectance units using the Finnish Geospatial Research Institute's (FGI) in-house mosaicking software. The wavelengths measured were from 507.6 nm to 885.6 nm, of which only 19 wavelength bands up to 671 nm were used for neural network training and validation. All wavelengths were available for index calculations. Table 2 presents wavelength and full width of the half maximum (FWHM) values in detail. Example spectra can be seen in Figure 3. For details of the datasets and post-processing, see Nevalainen et al. [41]

Table 2. Wavelength and full width of the half maximum (FWHM) values of the measured hyperspectral data [41].

Wavelength (nm):	507.60, 509.50, 514.50, 520.80, 529.00, 537.40, 545.80, 554.40, 562.70, 574.20, 583.60, 590.40, 598.80, 605.70, 617.50, 630.70, 644.20, 657.20, 670.10, 677.80, 691.10, 698.40, 705.30, 711.10, 717.90, 731.30, 738.50, 751.50, 763.70, 778.50, 794.00, 806.30, 819.70
FWHM (nm):	11.2, 13.6, 19.4, 21.8, 22.6, 20.7, 22.0, 22.2, 22.1, 21.6, 18.0, 19.8, 22.7, 27.8, 29.3, 29.9, 26.9, 30.3, 28.5, 27.8, 30.7, 28.3, 25.4, 26.6, 27.5, 28.2, 27.4, 27.5, 30.5, 29.5, 25.9, 27.3, 29.9

Table 3. The architecture of the used one-dimensional convolutional neural network (1DCNN).

Layer	**Kernel/Pool Size and Activation**	**Filters/ Units**
Batch Normalization		
Conv1D	3 ReLU	64
Batch Normalization		
MaxPooling1D	3	
Conv1D	3 ReLU	128
Batch Normalization		
Dropout (0.15)		
Flatten		
Dense	ReLU	100
Dense	ReLU	1
Optimiser:	Adam	
Loss:	Mean square error	
Accuracy:	r^2-score	

2.4. Validation

The results of the neural network training were validated in two different ways. The first corresponds to an ideal scenario: The validation data are produced by SLOP and the assessment represents a case in which the measurements are noiseless and perfectly calibrated. In the second scenario, the inversion model was validated with empirical hyperspectral UAV data from a boreal forest. The second scenario can be further divided into three subscenarios:

1. Calculation of chlorophyll *a* and *b* maps and comparison with the $\frac{\text{TCARI}}{\text{OSAVI}}$ index (TCARI = transformed chlorophyll absorption reflectance index, OSAVI = optimized soil adjusted vegetation index), a spectral index that is known for having a strong negative correlation with chlorophyll concentration [42],
2. Comparison of simulated and measured $\frac{\text{TCARI}}{\text{OSAVI}}$ indices,
3. Chlorophyll *a*/*b* ratio comparison to literature.

Figure 4 summarizes the proposed workflow and used methods.

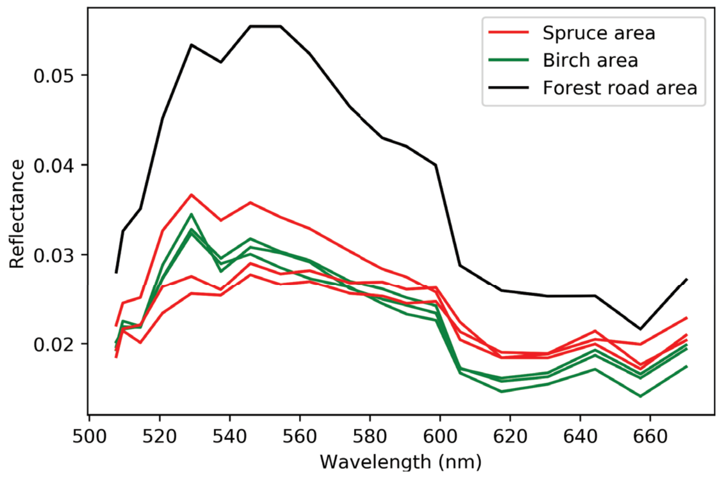


Figure 3. Median spectrum from each plot 1–7 described in Figure 5.

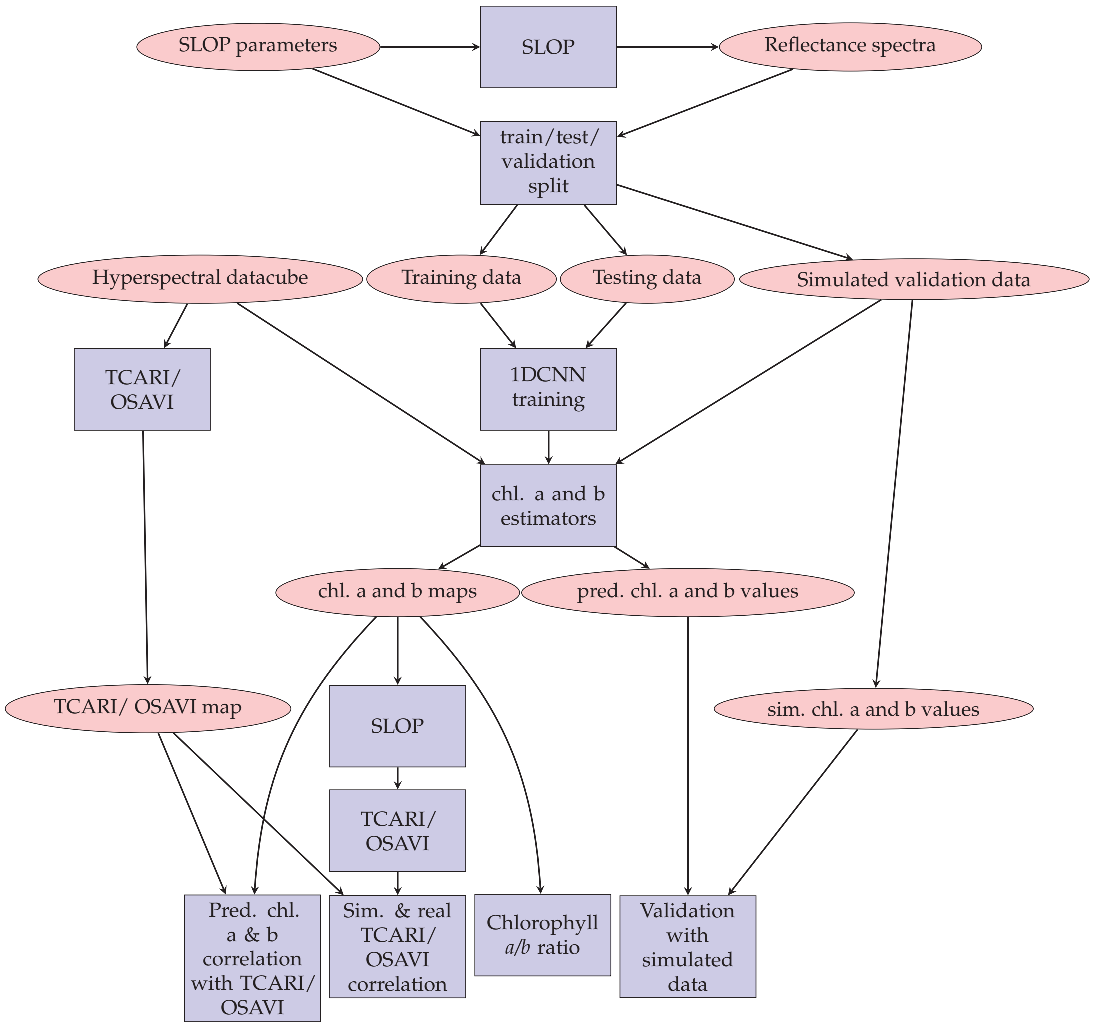


Figure 4. Flow chart of the research methodology. Ovals represent data and boxes represent methods.

2.4.1. Simulated Validation Dataset

In order to validate the method, the simulated data was divided into the training (45%), testing (22%), and validation (33%) datasets. The estimator was trained with the training and testing data and validated with the independent validation dataset. The predicted concentrations were provided by the estimator and compared to the validation data. Then the following metrics for the original and predicted concentrations were calculated:

- r^2 score between predicted and original values,
- Correlation coefficient between original and predicted values,
- MSE of their difference,
- Average difference,
- Standard deviation for the difference, and
- 95% confidence interval for the difference.

2.4.2. Empirical Validation Dataset

With the remotely sensed validation data, chlorophyll *a* and *b* maps were calculated by using the empirical data as input into the trained estimators. First, however, the data was shifted with a linear shift of 0.02 in order to get it to the same level as the training data and to eliminate negative values, that remained in the data after calibration. Since chlorophyll measurements for the data were not available, one had to acquire the concentrations using an alternative approach. Therefore, chlorophyll *a* and *b* predictions were compared to the $\frac{\text{TCARI}}{\text{OSAVI}}$ index. TCARI and OSAVI are calculated as follows:

$$\text{TCARI} = 3 \cdot (\text{sel}(700\,\text{nm}) - \text{sel}(670\,\text{nm})) - 0.2 \cdot (\text{sel}(700\,\text{nm}) - \text{sel}(500\,\text{nm})) \cdot \frac{\text{sel}(700\,\text{nm})}{\text{sel}(670\,\text{nm})} \quad (7)$$

and

$$\text{OSAVI} = (1 + 0.16) \cdot \frac{\text{sel}(800\,\text{nm}) - \text{sel}(670\,\text{nm})}{\text{sel}(800\,\text{nm}) + \text{sel}(670\,\text{nm}) + 0.16} \quad (8)$$

where the function sel chooses the wavelength band nearest to the desired band from the dataset. In this case, these were

$$\begin{aligned} \text{sel}(700\,\text{nm}) &= 698.40\,\text{nm}, \\ \text{sel}(670\,\text{nm}) &= 670.10\,\text{nm}, \\ \text{sel}(500\,\text{nm}) &= 507.60\,\text{nm}, \\ \text{sel}(800\,\text{nm}) &= 794.00\,\text{nm}. \end{aligned}$$

After calculating the different maps, each map was smoothed with a sliding 8×8 pixel averaging window to account for camera noise. The maps were compared with each other on a large scale and in more detail using the plots described in Figure 5. The plots were selected so that they represent the forest area well. In the middle of it, there is an area that is mainly birch forest, and the border areas are spruce dominated. There is also a forest road. Three plots were chosen from the spruce-dominated area (plots 1, 6, and 7), two in the birch-dominated area (plots 2, 3, and 5), and one from the forest road (plot 4). Plot 8 is a larger plot combining large areas of both spruce- and birch-dominated parts of the forest.

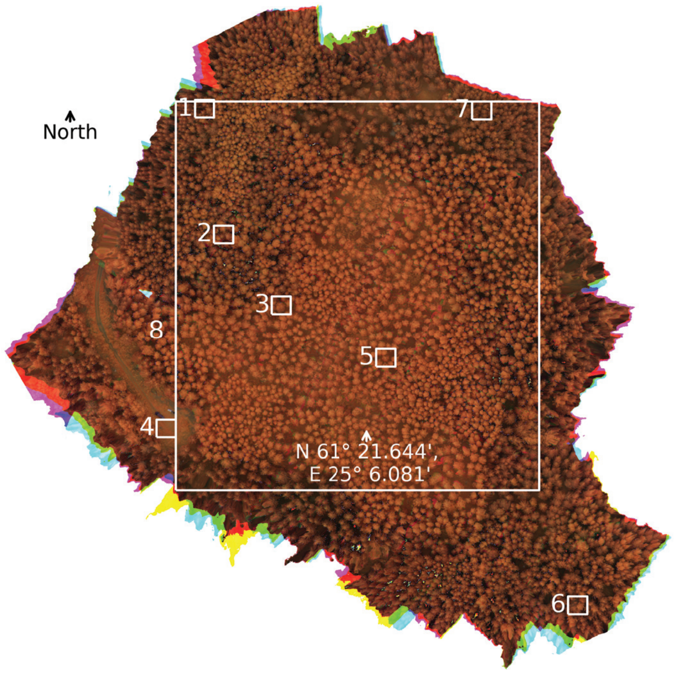


Figure 5. Plots selected for in-depth analysis plotted over a false color image of the research forest. The wavelength bands used in making the figure are approximately 800 nm, 700 nm, and 500 nm. The edges show a rainbow artefact produced by some of the bands being empty in the plot. Plots 1, 5, and 6 are in a spruce-dominated plot, plots 2, 3, and 5 are in a birch-dominated plot and plot 4 is on a forest road. Plot 8 is a larger plot that consists mainly of birch forest, while having a significant amount of spruce on the border plots.

2.4.3. Comparison of Simulated and Measured $\frac{\text{TCARI}}{\text{OSAVI}}$ Indices

$\frac{\text{TCARI}}{\text{OSAVI}}$ index was calculated with simulated data in order to compare it to the measured $\frac{\text{TCARI}}{\text{OSAVI}}$ index calculated in the previous chapter. The estimated chlorophyll *a* and *b* values for plots 1–7 were used as inputs for SLOP, while other inputs were averages of the intervals in Table 1. With SLOP, the reflectances on wavelengths 500 nm, 670 nm, 700 nm, and 800 nm were simulated and used to calculate the simulated $\frac{\text{TCARI}}{\text{OSAVI}}$ for plots 1–7. Then they were compared visually and the correlations between them and the empirical $\frac{\text{TCARI}}{\text{OSAVI}}$ were calculated. The simulated map was limited to between 2 and 98 percentiles in order to remove the outliers in the simulated data.

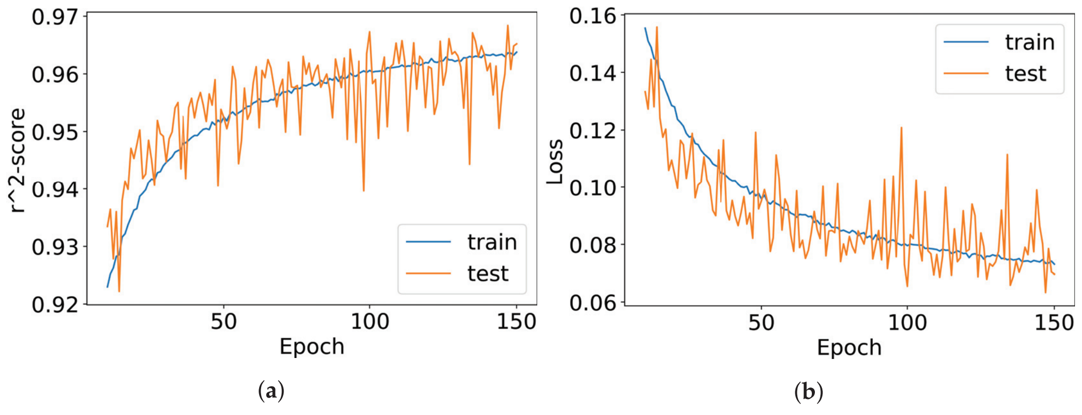
2.4.4. Chlorophyll *a/b*

The chlorophyll *a/b* ratio map was also compared to common values for birch and spruce chlorophyll ratio. The birch chlorophyll ratios during early summer found in the literature were between 2.3 [43] and 3.78 [44]. For spruce, the corresponding interval was from 1.47 [45] to 3.75 [46].

3. Results and Discussion

3.1. Simulated Validation Dataset

The MSE and r^2 scores converged during the training of the 1DCNN to 0 and 1 respectively and neither graph showed signs of over- or under-fitting (Figure 6), which holds for both the chlorophyll *a* and *b* training.



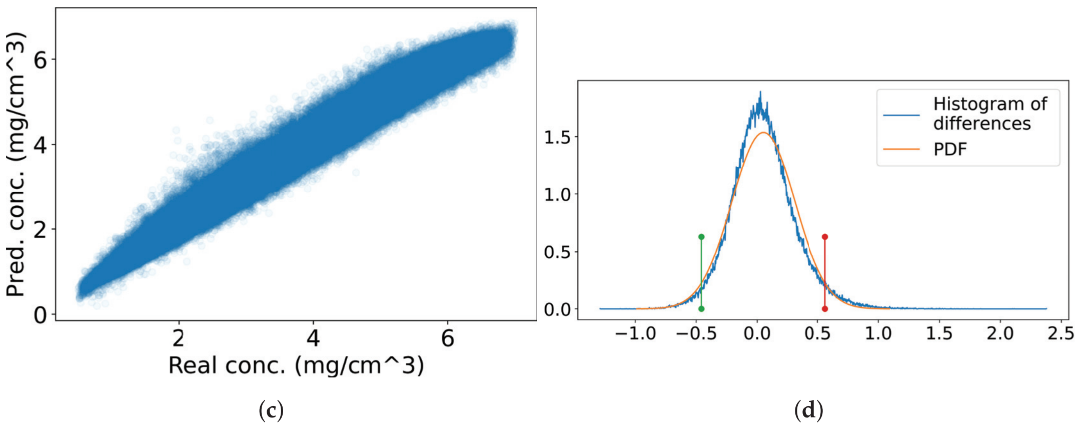
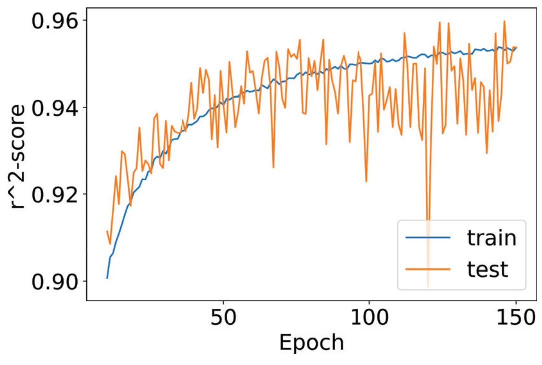
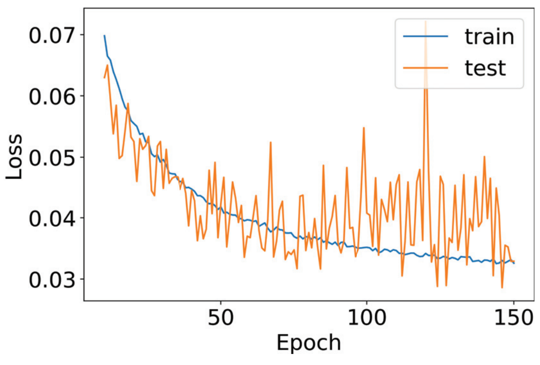


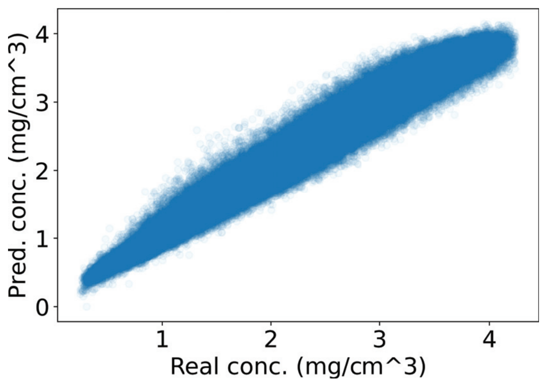
Figure 6. *Cont.*



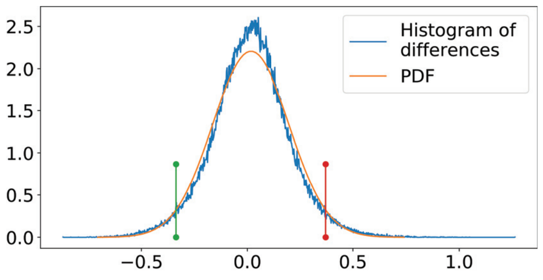
(**e**)



(**f**)



(**g**)



(**h**)

Figure 6. Convolutional neural network (CNN) training and testing results for chlorophyll *a* (**a**–**d**) and *b* (**e**–**h**). Figures (**a**,**e**) contain training and testing r^2 scores and figures (**b**,**f**) contain training and testing mean square error (MSE) scores. In figures (**c**,**g**) the estimated values are compared to the values in the validation dataset and in figures (**d**,**h**) their difference is computed and the matching normal distribution is calculated. Red and green lines represent the 95% confidence interval.

For the simulated validation dataset, the r^2 score between the original and predicted values was 0.97 for chlorophyll *a* and 0.95 for chlorophyll *b*. The correlation coefficient was 0.98 between the predicted and original chlorophyll *a* concentrations and 0.98 between the predicted and original chlorophyll *b* concentrations. The MSE was 0.07 between the original and predicted chlorophyll *a* values and 0.03 for the original and predicted chlorophyll *b* concentrations. The average difference in chlorophyll *a* predictions compared to original values was 0.05, with a standard deviation of 0.26. Thus, the difference between predicted and original values was between −0.46 and 0.56 with 95% probability. For chlorophyll *b* the corresponding values were the following: mean 0.02, standard deviation 0.18, 95% confidence interval $[-0.34, 0.37]$ (Figures 6c,d,g,h and Table 4).

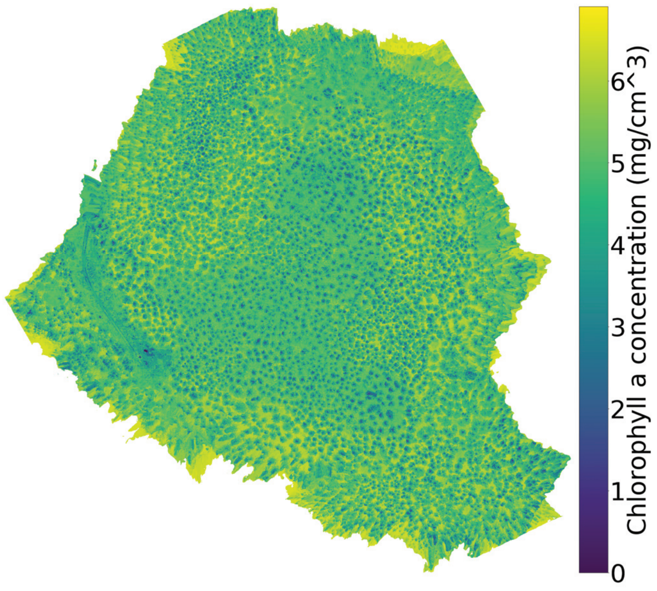
Table 4. Results of training and validating the CNN estimators for chlorophyll *a* and *b*.

	Chlorophyll *a*	**Chlorophyll *b***
r^2 score between original and predicted values	0.97	0.95
Correlation coefficient between original and predicted values	0.98	0.98
MSE	0.07	0.03
Average difference	0.05	0.02
Standard deviation	0.26	0.18
95% Confidence interval of the difference between predicted and original values	$[-0.46, 0.56]$	$[-0.34, 0.37]$

The results indicate that the estimators predict simulated chlorophyll *a* and *b* values accurately. This means that if the SLOP model is assumed to be perfect, the used hyperspectral camera is optimal with a spectral range of 400 nm to 2600 nm and illumination conditions are perfect, the chlorophyll *a* and *b* estimators give reasonably accurate results. Compared to previous research, the used network matched the results received in the past. Croft et al. simulated spectral data from multiple sensors and obtained r^2 scores of 0.96–0.99 when they compared their results to chlorophyll concentrations [14].

3.2. Empirical Validation Dataset

The results involving measured hyperspectral validation data can be seen in Figures 7–11 and Tables 5 and 6. The chlorophyll *a* and *b* and $\frac{\text{TCARI}}{\text{OSAVI}}$ maps clearly resembled each other, with the birch and spruce areas distinguishable along with the forest road (Figure 7).



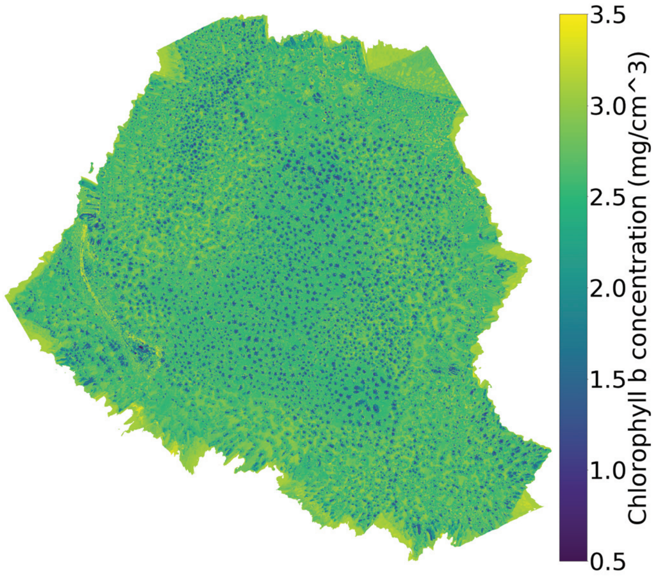


Figure 7. *Cont.*

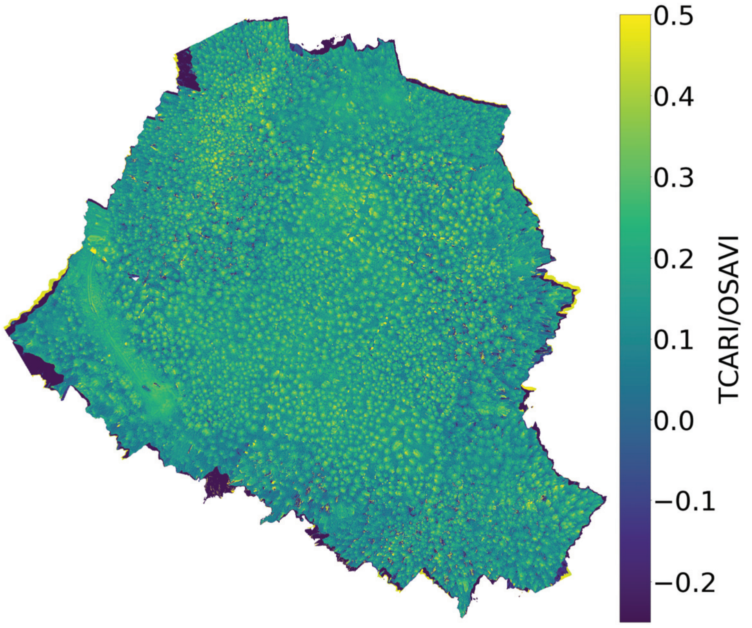
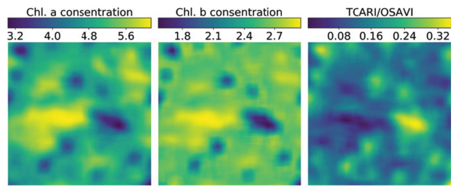
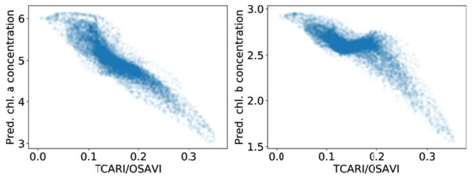


Figure 7. The chlorophyll *a* (top), *b* (middle), and $\frac{\text{TCARI}}{\text{OSAVI}}$ index (bottom) maps. The chlorophyll maps are calculated by feeding the hyperspectral data to the inverse SLOP model.

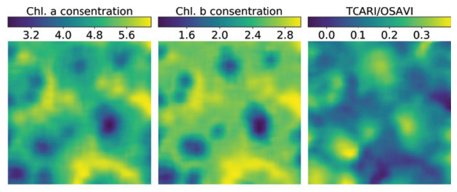
The predicted chlorophyll *a* and *b* correlated with the empirical $\frac{\text{TCARI}}{\text{OSAVI}}$ in the plots 1–7 described in Figure 5. In the spruce-dominant plots, the correlation coefficient was, on average, −0.9 for chlorophyll *a* and −0.7 for chlorophyll *b*. In the birch-dominated plots, the corresponding average correlation coefficients were −0.89 and −0.77 while in all of the plots 1–7 they were −0.87 and −0.68. The results for the plot 8 were −0.82 for chlorophyll *a* and −0.76 for chlorophyll *b* (Figures 8 and 9 and Table 5).



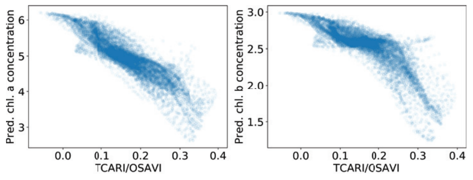
(**a**)



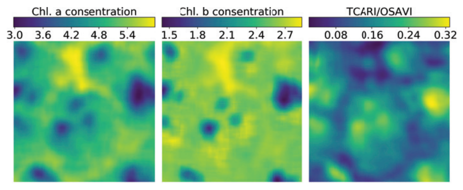
(**b**)



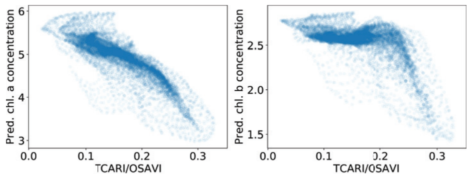
(**c**)



(**d**)

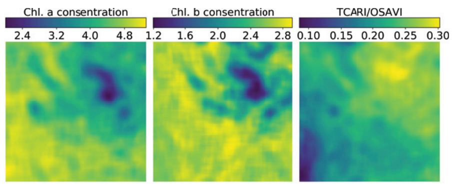


(**e**)

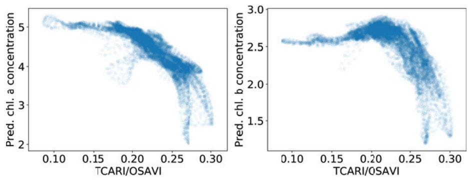


(**f**)

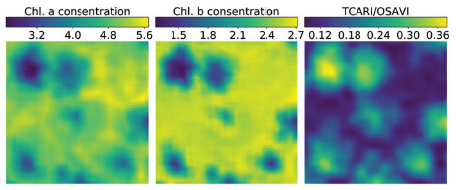
Figure 8. *Cont.*



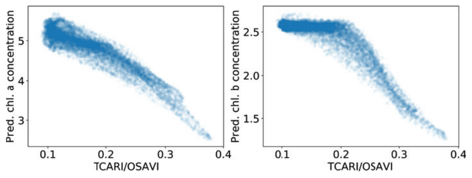
(**g**)



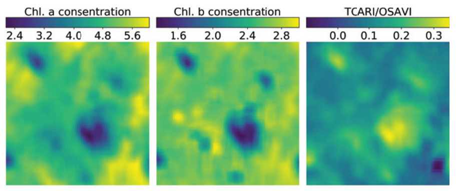
(**h**)



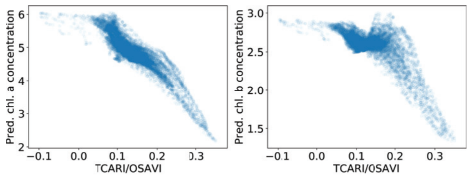
(**i**)



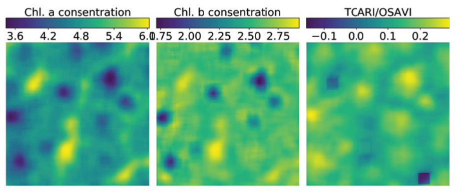
(**j**)



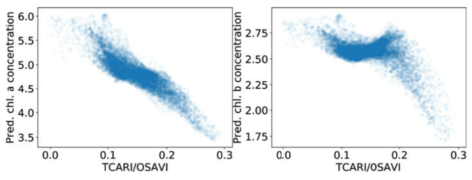
(**k**)



(**l**)



(**m**)



(**n**)

Figure 8. Chlorophyll *a*, *b* and $\frac{\text{TCARI}}{\text{OSAVI}}$ maps (left column) and correlation between the chlorophylls and the $\frac{\text{TCARI}}{\text{OSAVI}}$ index (right column) in plots 1–7. Figures (**a**,**b**) are related to plot 1, (**c**,**d**) to plot 2, (**e**,**f**) to plot 3, (**g**,**h**) to plot 4, (**i**,**j**) to plot 5, (**k**,**l**) to plot 6 and (**m**,**n**) to plot 7.

Table 5. Correlation coefficients between predicted chlorophyll values and the $\frac{\text{TCARI}}{\text{OSAVI}}$ index in the eight researched plots.

	Correlation Coefficient for Chlorophyll *a* and Index	Correlation Coefficient for Chlorophyll *b* and Index	Plot Description
Plot 1	−0.91	−0.81	Spruce
Plot 2	−0.90	−0.83	Birch
Plot 3	−0.84	−0.61	Birch
Plot 4	−0.81	−0.61	Forest road
Plot 5	−0.93	−0.88	Birch
Plot 6	−0.91	−0.74	Spruce
Plot 7	−0.88	−0.55	Spruce
Plots 1, 6 and 7	−0.90	−0.70	Spruce
Plots 2, 3 and 5	−0.89	−0.77	Birch
Plots 1–7	−0.87	−0.68	Spruce and Birch
Plot 8	−0.82	−0.76	Larger plot. Mainly birch, spruce on the border areas

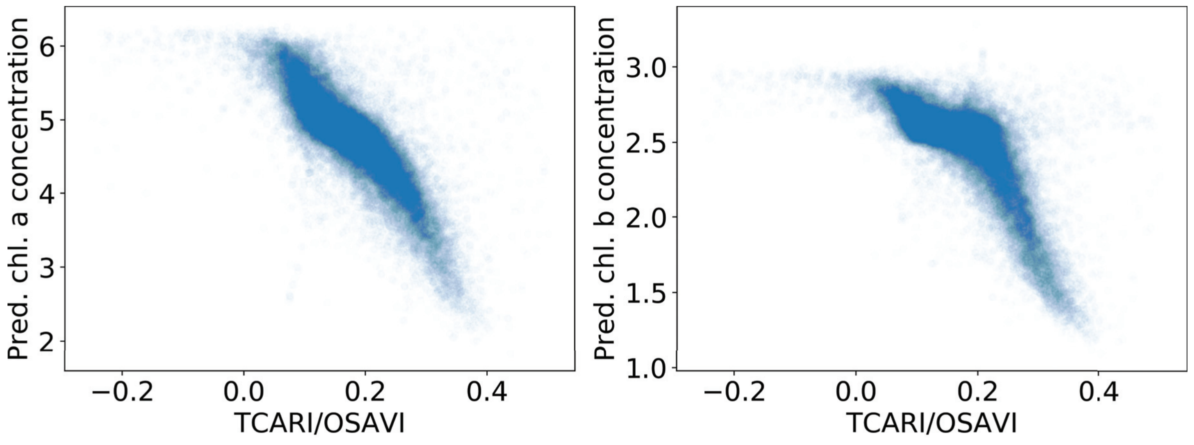


Figure 9. Correlation between chlorophyll and the $\frac{\text{TCARI}}{\text{OSAVI}}$ index in larger plot 8.

We did not have reference data to assess the absolute accuracies, but results in Figures 7, 8, and 11 proposed that the results were at least proportionally correct. The estimated chlorophyll values of different land cover classes were consistent with the expectations (Figure 7). For example, the birch dominated area in the center of the forest was clearly separable from the spruce dominated area at the area perimeters, and the forest road obtained low chlorophyll values. As expected, the chlorophyll values were high for the low $\frac{\text{TCARI}}{\text{OSAVI}}$ values. Also, the analysis of the finer details indicated that these observations were correct (Figure 8). Comparisons of the values of predicted chlorophyll concentrations (Figure 7) to simulated values (Table 1) indicated that the values were of the same order of magnitude, which confirms that our approach could produce correct results if the training data has good quality.

While for chlorophyll *a* the relation with $\frac{\text{TCARI}}{\text{OSAVI}}$ was relatively linear, the corresponding connection for chlorophyll *b* had an approximately constant part when chlorophyll *b* was about 2.7 mg cm^{-3} and $\frac{\text{TCARI}}{\text{OSAVI}}$ increased from approximately 0.1 to 0.2 (Figures 8 and 9). One possible reason is that the $\frac{\text{TCARI}}{\text{OSAVI}}$ correlation is with (combined) chlorophyll content, and its correlation with chlorophyll *a* and *b* separately has not been accounted for. In addition, chlorophyll *a* content is usually higher in plants, so it follows that chlorophyll *b* has a smaller net contribution in the correlation. To find the cause of the constant part, there is a need for further research and comparison of the inverse SLOP results to measured chlorophyll concentrations. If the constant part persists in such study, our postulation of the cause is false. At the moment we lack a proper dataset to conduct such research.

Our results are in line with previous research. Croft et al. found an r^2 score of 0.78 between modeled and measured chlorophyll concentrations, which assuming linearity would mean correlation similar to our results [14]. Results from [15] show an r^2 score of 0.87 between modeled and measured canopy chlorophyll content, which is slightly higher than what we obtained. The results are not directly comparable due to our lack of field chlorophyll measurements, and for a fair comparison of the results of this study, the method should be tested in further studies using in-situ chlorophyll measurements.

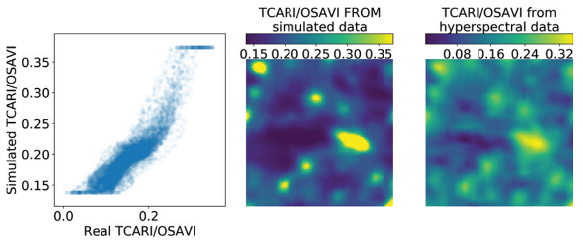
3.3. Comparison of Simulated and Measured $\frac{TCARI}{OSAVI}$ Indices

The simulated and empirical $\frac{\text{TCARI}}{\text{OSAVI}}$ indexes correlated strongly (see Figure 10 and Table 6). The correlation values ranged between 0.69 and 0.89, and the patterns were similar to each other and compared to Figure 8. This validates the SLOP model's ability to produce data similar to empirical data and reinforces our conclusion that the 1DCNN estimators are truly emulating the true inverse function of SLOP in regards of chlorophyll *a* and *b*. If either of these conclusions were false, the data produced by SLOP would be significantly different from empirical data, and the error would propagate to the $\frac{\text{TCARI}}{\text{OSAVI}}$ calculation.

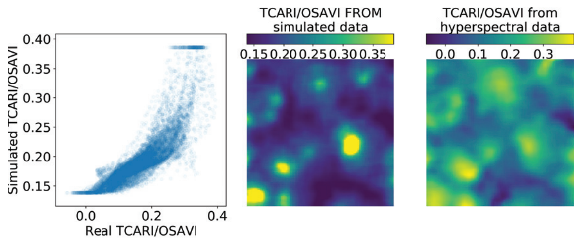
The reasoning of this assessment is the following. The $\frac{\text{TCARI}}{\text{OSAVI}}$ was calculated from the empirical validation data in two ways: by calculating directly from the empirical data and by using two different functions before calculating the $\frac{\text{TCARI}}{\text{OSAVI}}$. If the two approaches produced mutually correlating data, the two functions would act as approximate inverse function of each other. In addition, if the empirical validation data were vastly different from the data produced by SLOP, the trained inverse function would not have capability to explain the input and the output would be irrational.

Table 6. Correlation coefficients between simulated and empirical $\frac{\text{TCARI}}{\text{OSAVI}}$ indexes in the seven investigated plots.

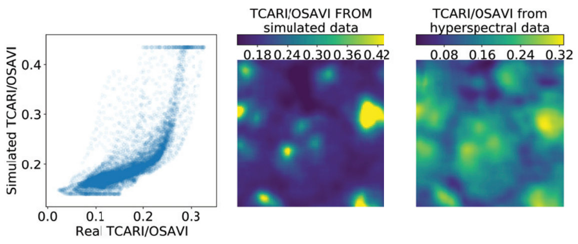
	Correlation Coefficient between Simulated and Empirical $\frac{\text{TCARI}}{\text{OSAVI}}$ Indexes	**Plot Description**
Plot 1	0.89	Spruce
Plot 2	0.83	Birch
Plot 3	0.75	Birch
Plot 4	0.68	Forest road
Plot 5	0.88	Birch
Plot 6	0.85	Spruce
Plot 7	0.81	Spruce



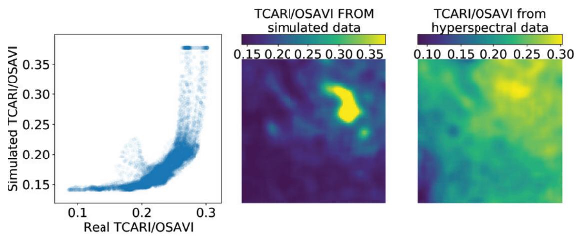
(**a**)



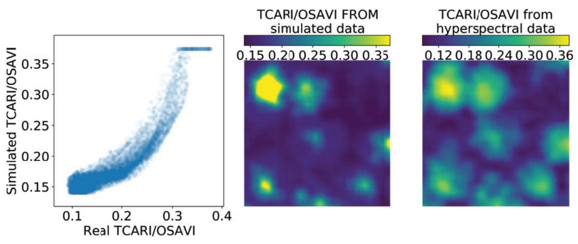
(**b**)



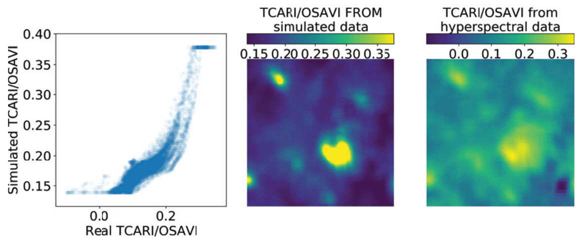
(**c**)



(**d**)

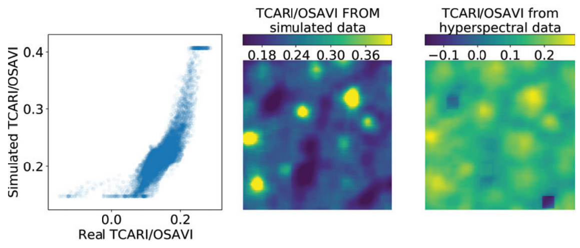


(**e**)



(**f**)

Figure 10. *Cont.*

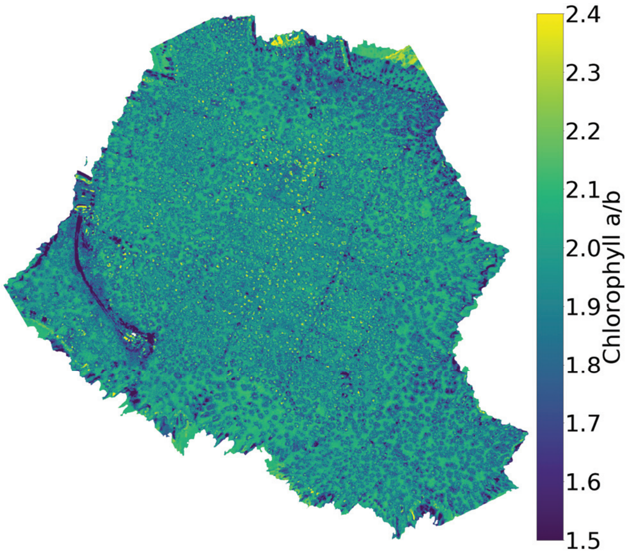


(**g**)

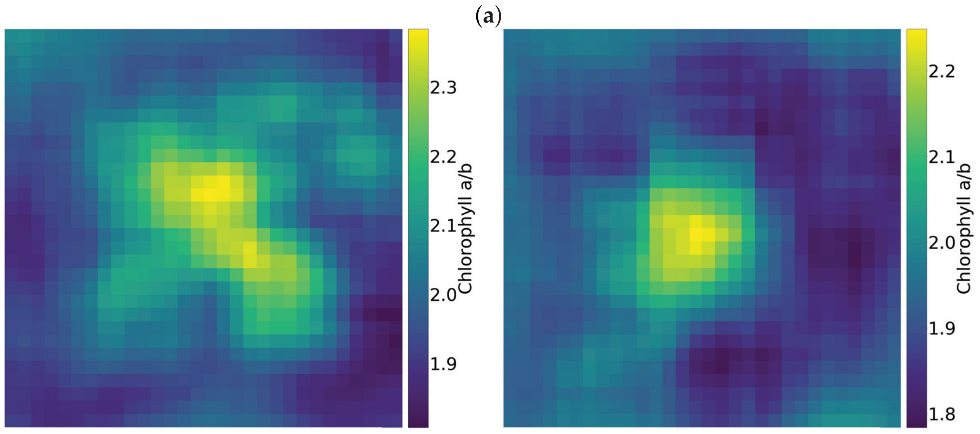
Figure 10. Comparison between simulated and empirical $\frac{\text{TCARI}}{\text{OSAVI}}$ indexes. The simulated index is calculated from data simulated using SLOP. (**a**) corresponds to the research plot 1, (**b**) to plot 2, (**c**) to 3, (**d**) to 4, (**e**) to 5, (**f**) to 6 and (**g**) to 7.

3.4. Chlorophyll a/b

Chlorophyll *a/b* values ranged from 1.5 to 2.5 (see Figure 11) which was in line with spruce and birch intervals: from 1.47 to 3.75 for spruce [45,46] and from 2.3 to 3.78 for birch [43,44]. For spruce (Figure 11c) the values ranged from 1.8 to 2.2 and in the middle of the figure where the spruce stood the values were around 2.1 to 2.2, which were in the correct range. For the birch (Figure 11b) the values ranged from 1.85 to 2.35. In the middle the values were in between 2.1 and 2.35, which means that the values were at the lower end of the correct range.



(**a**)



(**b**) (**c**)

Figure 11. Chlorophyll *a* values divided by chlorophyll *b* values for the whole forest (**a**), for a birch (**b**) and for a spruce (**c**).

3.5. Other Observations

The $\frac{\text{TCARI}}{\text{OSAVI}}$ index can also be used to estimate chlorophyll concentrations [42]. This estimation would be useful to correlate with our estimators, yet then the leaf area index (LAI) should be calculated. Unfortunately, it was not possible to estimate the LAI from the used empirical dataset.

In vegetation parameter retrieval studies, the leaf-level model is often paired to a canopy radiative transfer model (RTM) [14,15,17]. This procedure would have been advantageous in our study as well, and the fact that we used the leaf-level simulated spectral data instead of canopy-level features will cause some uncertainties in the analysis. However, we expect that these differences are smaller than in studies with conventional manned aircrafts or satellites. Firstly, our dataset had ultra-high spatial resolution with a ground sampling distance of 8 cm, thus spectral values of individual pixels had less mixing with background than the typical aircraft or satellite images with GSDs of e.g., 50 cm to 10 m. Secondly, the measurements took place in cloudy weather with diffuse illumination, thus the disturbing bidirectional reflectance (BRDF) effects were minimal in the data [41]. Thirdly, we added some Gaussian noise to the leaf-level training data, which could also account for some of the uncertainty that is introduced in transition from leaf-level to canopy-level. Even though we did not account for the canopy RTM, the results were consistent with the leaf level simulations. In future studies we will also implement the canopy RTM in our modeling. If the 1DCNN model is integrated with a radiative transfer simulator such as Librat [47], training data for various sensors and conditions could be generated.

Based on our literature review, the combination of using a simulation model for hyperspectral data and CNN for parameter retrieval is new in the field of hyperspectral data analysis. As discussed earlier, CNN is a powerful tool for signal regression and classification problems. It has been shown to perform better than traditional dense deep neural networks in image recognition and other signal-related tasks, thus it should perform better in regression problems concerning hyperspectral images. The use of simulated data in training is justified by the fact that obtaining a comprehensive dataset in field measurements requires a huge effort, whereas mathematical models can be developed with significantly less labor. Our approach of using simulated data for CNN training has the potential for a more universal and accurate regression and classification models in the context of hyperspectral imaging.

One of the difficult parts of using simulated data in parameter retrieval is accounting for error sources in hyperspectral images. The simulated data are smooth and noiseless, whereas the empirical data are vulnerable to multiple error sources. Two approaches to tackle this challenge are the development of rigorous data calibration approaches [4] and the implementation of different error sources to the canopy radiative transfer modeling [47]. In this study we attempted to take noise into account in the design of the CNN by adding batch Normalization layers to the bare-bones CNN template, by adding Gaussian noise to the training data and by averaging the resulting chlorophyll *a* and *b* and $\frac{\text{TCARI}}{\text{OSAVI}}$ maps over an 8×8 pixel sliding window. There is still a need for development in this area to identify the best ways to control noise in hyperspectral images.

While it appears that our approach works relatively well with SLOP, it would be interesting to study different other models as well, such as ray tracing models [48,49], models specialized in coniferous trees [50], or models that take the entire forest stand into account [51–54]. They could improve the results, because SLOP is tied to the composition of the basic leaf. The ray tracing would be more accurate, but it would also require more computing time. In some cases, the use of a simpler or less accurate model could be justified if computation time is the limiting factor. It is also possible to obtain more specialized training data by narrowing the range of SLOP parameters around known values of some certain tree species. This would probably make results more accurate for that species while degrading the results for other tree species.

The 1DCNN model may not be the best possible network for the task. However, it has already shown promising results as the basis for the further developments. There is much work to do on developing CNN networks for different tasks, including regression and classification tasks in hyperspectral data analysis.

4. Conclusions

Our study has presented a novel method for chlorophyll a and *b* estimation from hyperspectral image data using the stochastic model of leaf optical (SLOP) properties and a convolutional neural network. We found that when the conditions and imaging system are consistent with the SLOP model, the convolutional neural network estimators for chlorophyll *a* and *b* produce feasible results. Our results showed that even with less-than-ideal remote sensing data, the results were in the right range and correlated well with an index known to correlate strongly with chlorophyll concentration. This indicates that our method shows promising results in measuring chlorophyll, although further verification of the results is needed to ensure their correctness.

We utilized a conventional one-dimensional convolutional neural network (CNN) structure, which showed promising performance. By optimizing the CNN model, the results are expected to improve, and the prediction errors are expected to decrease. The aim of further research could be developing a model based on CNN and hyperspectral data for reliable estimation of tree health through measurements of chlorophyll *a* and *b*.

Author Contributions: Conceptualization, L.A. and I.P.; methodology, L.A. and I.P.; software, L.A.; validation, L.A., I.P., E.H., and S.T.; formal analysis, L.A.; investigation, L.A.; resources, I.P.; data curation, L.A., I.P., E.H., and S.T.; writing—original draft preparation, L.A.; writing—review and editing, E.H., S.T., I.P., and L.A.; visualization, L.A.; supervision, I.P.; project administration, I.P.; funding acquisition, I.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Finnish Funding Agency for Innovation Tekes grant number 1711/31/2016.

Acknowledgments: Niko Viljanen and Teemu Hakala from Finnish Geospatial Research Insititute, National Land Survey of Finland, are acknowledged for the UAV data capture and georeferencing. We acknowledge grants of computer capacity from the Finnish Grid and Cloud Infrastructure (persistent identifier urn:nbn:fi:research-infras-2016072533).

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

1DCNN	One Dimensional Convolutional Neural Network
CNN	Convolutional Neural Network
FGI	Finnish Geospatial Research Institute
FPI	Fabry–Pérot interferometer
FWHM	Full width of the half maximum
GPU	Graphics Processing Unit
GSD	Ground Sampling Distance
LAI	Leaf Area Index
MSE	Mean Square Error
OSAVI	Optimized Soil Adjusted Vegetation Index
RTM	Radiative Transfer Model
SLOP	Stochastic model of Leaf Optical Properties
TCARI	Transformed Chlorophyll Absorption Reflectance Index

References

1. Pan, Y.; Birdsey, R.A.; Fang, J.; Houghton, R.; Kauppi, P.E.; Kurz, W.A.; Phillips, O.L.; Shvidenko, A.; Lewis, S.L.; Canadell, J.G.; et al. A large and persistent carbon sink in the world's forests. *Science* **2011**, *333*, 988–993. [CrossRef] [PubMed]

2. Rossini, M.; Panigada, C.; Meroni, M.; Colombo, R. Assessment of oak forest condition based on leaf biochemical variables and chlorophyll fluorescence. *Tree Physiol.* **2006**, *26*, 1487–1496. [CrossRef] [PubMed]
3. Bjorkman, C.; Niemela, P. *Climate Change and Insect Pests*; CABI: Wallingford, UK, 2015; Volume 8.
4. Aasen, H.; Honkavaara, E.; Lucieer, A.; Zarco-Tejada, P.J. Quantitative remote sensing at ultra-high resolution with UAV spectroscopy: A review of sensor technology, measurement procedures, and data correction workflows. *Remote Sens.* **2018**, *10*, 1091. [CrossRef]
5. Bioucas-Dias, J.M.; Plaza, A.; Camps-Valls, G.; Scheunders, P.; Nasrabadi, N.; Chanussot, J. Hyperspectral remote sensing data analysis and future challenges. *IEEE Geosci. Remote Sens. Mag.* **2013**, *1*, 6–36. [CrossRef]
6. Verrelst, J.; Camps-Valls, G.; Muñoz-Marí, J.; Rivera, J.P.; Veroustraete, F.; Clevers, J.G.; Moreno, J. Optical remote sensing and the retrieval of terrestrial vegetation bio-geophysical properties–a review. *ISPRS J. Photogramm. Remote Sens.* **2015**, *108*, 273–290. [CrossRef]
7. Verrelst, J.; Malenovskỳ, Z.; Van der Tol, C.; Camps-Valls, G.; Gastellu-Etchegorry, J.P.; Lewis, P.; North, P.; Moreno, J. Quantifying vegetation biophysical variables from imaging spectroscopy data: A review on retrieval methods. *Surv. Geophys.* **2018**, *40*, 589–629. [CrossRef]
8. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–8 December 2012; pp. 1097–1105.
9. Karpathy, A.; Toderici, G.; Shetty, S.; Leung, T.; Sukthankar, R.; Fei-Fei, L. Large-scale video classification with convolutional neural networks. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 1725–1732.
10. Szegedy, C.; Toshev, A.; Erhan, D. Deep neural networks for object detection. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 5–10 December 2013; pp. 2553–2561.
11. Cecotti, H.; Graeser, A. Convolutional neural network with embedded Fourier transform for EEG classification. In Proceedings of the 2008 19th International Conference on Pattern Recognition, Tampa, FL, USA, 8–11 December 2008; pp. 1–4.
12. Salamon, J.; Bello, J.P. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Process. Lett.* **2017**, *24*, 279–283. [CrossRef]
13. LeCun, Y.; Bengio, Y. Convolutional networks for images, speech, and time-series. In *The Handbook of Brain Theory and Neural Networks*; MIT Press: Cambridge, MA, USA, 1995.
14. Croft, H.; Chen, J.; Zhang, Y.; Simic, A.; Noland, T.; Nesbitt, N.; Arabian, J. Evaluating leaf chlorophyll content prediction from multispectral remote sensing data within a physically-based modeling framework. *ISPRS J. Photogramm. Remote Sens.* **2015**, *102*, 85–95. [CrossRef]
15. Atzberger, C.; Jarmer, T.; Schlerf, M.; Kötz, B.; Werner, W. Retrieval of wheat bio-physical attributes from hyperspectral data and SAILH+ PROSPECT radiative transfer model. In Proceedings of the 3rd EARSeL Workshop on Imaging Spectroscopy, Herrsching, Germany, 13–16 May 2003; pp. 473–482.
16. Jacquemoud, S.; Baret, F. PROSPECT: A model of leaf optical properties spectra. *Remote Sens. Environ.* **1990**, *34*, 75–91. [CrossRef]
17. Jacquemoud, S.; Verhoef, W.; Baret, F.; Bacour, C.; Zarco-Tejada, P.J.; Asner, G.P.; François, C.; Ustin, S.L. PROSPECT+ SAIL models: A review of use for vegetation characterization. *Remote Sens. Environ.* **2009**, *113*, S56–S66. [CrossRef]
18. Tedesco, M.; Pulliainen, J.; Takala, M.; Hallikainen, M.; Pampaloni, P. Artificial neural network-based techniques for the retrieval of SWE and snow depth from SSM/I data. *Remote Sens. Environ.* **2004**, *90*, 76–85. [CrossRef]
19. Notarnicola, C.; Angiulli, M.; Posa, F. Soil moisture retrieval from remotely sensed data: Neural network approach versus Bayesian method. *IEEE Trans. Geosci. Remote Sens.* **2008**, *46*, 547–557. [CrossRef]
20. Trombetti, M.; Riaño, D.; Rubio, M.; Cheng, Y.; Ustin, S. Multi-temporal vegetation canopy water content retrieval and interpretation using artificial neural networks for the continental USA. *Remote Sens. Environ.* **2008**, *112*, 203–215. [CrossRef]
21. Cipollini, P.; Corsini, G.; Diani, M.; Grasso, R. Retrieval of sea water optically active parameters from hyperspectral data by means of generalized radial basis function neural networks. *IEEE Trans. Geosci. Remote Sens.* **2001**, *39*, 1508–1524. [CrossRef]
22. Maier, S.; Lüdeker, W.; Günther, K. SLOP: A Revised Version of the Stochastic Model for Leaf Optical Properties. *Remote Sens. Environ.* **1999**, *68*, 273–280. [CrossRef]

23. Tucker, C.; Garratt, M. Leaf optical system modeled as a stochastic process. *Appl. Opt.* **1977**, *16*, 635–642. [CrossRef]
24. Jacquemoud, S.; Ustin, L. Modeling leaf optical properties. *Photobiol. Sci. Online* **2008**. Available online http://photobiology.info/Jacq_Ustin.html (accessed on 10 January 2020).
25. Buschmann, C.; Nagel, E. Reflection Spectra Of Terrestrial Vegetation As Influenced By Pigment-protein Complexes And The Internal Optics Of The Leaf Tissue. In Proceedings of the Remote Sensing: Global Monitoring for Earth Management (IGARSS'91), Espoo, Finland, 3–6 June 1991; Volume 4, pp. 1909–1912. [CrossRef]
26. Lichtenthaler, H.K. [34] Chlorophylls and carotenoids: Pigments of photosynthetic biomembranes. In *Plant Cell Membranes*; Methods in Enzymology; Academic Press: Cambridge, MA, USA, 1987; Volume 148, pp. 350–382. [CrossRef]
27. Richter, T.; Fukshansky, L. Authentic in vivo absorption spectra for chlorophyll in leaves as derived from in situ and in vitro measurements. *Photochem. Photobiol.* **1994**, *59*, 237–247. [CrossRef]
28. Lichtenthaler, H.; Burkard, G.; Kuhn, G.; Prenzel, U. Light-Induced Accumulation and Stability of Chlorophylls and Chlorophyll-Proteins during Chloroplast Development in Radish Seedlings. *Z. Naturforschung C* **1981**, *36*, 421–430. [CrossRef]
29. Evans, J. A quantitative analysis of light distribution between the two photosystems, considering variation in both the relative amounts of the chlorophyll-protein complexes and the spectral quality of light. *Photobiochem. Photobiophys.* **1986**, *10*, 135–147.
30. Hale, G.M.; Querry, M.R. Optical Constants of Water in the 200-nm to 200-μm Wavelength Region. *Appl. Opt.* **1973**, *12*, 555–563. [CrossRef]
31. Tam, A.C.; Patel, C.K.N. Optical absorptions of light and heavy water by laser optoacoustic spectroscopy. *Appl. Opt.* **1979**, *18*, 3348–3358. [CrossRef]
32. Palmer, K.F.; Williams, D. Optical properties of water in the near infrared. *J. Opt. Soc. Am.* **1974**, *64*, 1107–1110. [CrossRef]
33. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef]
34. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv* **2015**, arXiv:1502.03167.
35. Nair, V.; Hinton, G.E. Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th International Conference on Machine Learning (ICML-10), Haifa, Israel, 21–24 June 2010; pp. 807–814.
36. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
37. Casella, G.; Lehmann, E.L. *Theory of Point Estimation*; Springer: New York, NY, USA, 1998.
38. Wright, S. Correlation and causation. *J. Agric. Res.* **1921**, *20*, 557–585.
39. Keras. 2015. Available online: keras.io (accessed on 9 January 2020).
40. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M.; et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. 2015. Available online: tensorflow.org (accessed on 9 January 2020).
41. Nevalainen, O.; Honkavaara, E.; Tuominen, S.; Viljanen, N.; Hakala, T.; Yu, X.; Hyyppä, J.; Saari, H.; Pölönen, I.; Imai, N.N.; et al. Individual Tree Detection and Classification with UAV-Based Photogrammetric Point Clouds and Hyperspectral Imaging. *Remote Sens.* **2017**, *9*, 185. [CrossRef]
42. Haboudane, D.; Miller, J.R.; Tremblay, N.; Zarco-Tejada, P.J.; Dextraze, L. Integrated narrow-band vegetation indices for prediction of crop chlorophyll content for application to precision agriculture. *Remote Sens. Environ.* **2002**, *81*, 416–426. [CrossRef]
43. Kauppi, A. Seasonal fluctuations in chlorophyll content in birch stems with special reference to bark thickness and light transmission, a comparison between sprouts and seedlings. *Flora* **1991**, *185*, 107–125. [CrossRef]
44. Possen, B.J.; Anttonen, M.J.; Oksanen, E.; Rousi, M.; Heinonen, J.; Kostiainen, K.; Kontunen-Soppela, S.; Heiskanen, J.; Vapaavuori, E.M. Variation in 13 leaf morphological and physiological traits within a silver birch (Betula pendula) stand and their relation to growth. *Can. J. For. Res.* **2014**, *44*, 657–665. [CrossRef]
45. Barnes, J.; Balaguer, L.; Manrique, E.; Elvira, S.; Davison, A. A reappraisal of the use of DMSO for the extraction and determination of chlorophylls a and b in lichens and higher plants. *Environ. Exp. Bot.* **1992**, *32*, 85–100. [CrossRef]
46. Robinson, D.C.; Wellburn, A.R. Seasonal changes in the pigments of Norway spruce, Picea abies (L.) Karst, and the influence of summer ozone exposures. *New Phytol.* **1991**, *119*, 251–259. [CrossRef]

47. Disney, M.; Lewis, P.; North, P. Monte Carlo ray tracing in optical canopy reflectance modeling. *Remote Sens. Rev.* **2000**, *18*, 163–196. [CrossRef]
48. Govaerts, Y.M.; Jacquemoud, S.; Verstraete, M.M.; Ustin, S.L. Three-dimensional radiation transfer modeling in a dicotyledon leaf. *Appl. Opt.* **1996**, *35*, 6585–6598. [CrossRef]
49. Govaerts, Y.M.; Verstraete, M.M. Raytran: A Monte Carlo ray-tracing model to compute light scattering in three-dimensional heterogeneous media. *IEEE Trans. Geosci. Remote Sens.* **1998**, *36*, 493–505. [CrossRef]
50. Dawson, T.P.; Curran, P.J.; Plummer, S.E. LIBERTY—Modeling the effects of leaf biochemical concentration on reflectance spectra. *Remote Sens. Environ.* **1998**, *65*, 50–60. [CrossRef]
51. Verhoef, W. Light scattering by leaf layers with application to canopy reflectance modeling: The SAIL model. *Remote Sens. Environ.* **1984**, *16*, 125–141. [CrossRef]
52. Kuusk, A. A two-layer canopy reflectance model. *J. Quant. Spectrosc. Radiat. Transf.* **2001**, *71*, 1–9. [CrossRef]
53. Goel, N.S. Models of vegetation canopy reflectance and their use in estimation of biophysical parameters from reflectance data. *Remote Sens. Rev.* **1988**, *4*, 1–212. [CrossRef]
54. Kuusk, A. A multispectral canopy reflectance model. *Remote Sens. Environ.* **1994**, *50*, 75–82. [CrossRef]

PV

KUBELKA-MUNK MODEL AND STOCHASTIC MODEL COMPARISON IN SKIN PHYSICAL PARAMETER RETRIEVAL

by

Leevi Annala and Ilkka Pölönen in press, 2020

Computational Sciences and Artificial Intelligence in Industry – New digital technologies for solving future societal and economical challenges, Springer

Kubelka-Munk Model and Stochastic Model Comparison in Skin Physical Parameter Retrieval

Leevi Annala and Ilkka Pölönen

Abstract In medical field there is need for non-invasive diagnostic tools. One particular research area is skin cancer diagnostics. Here we study Kubelka-Munk model and stochastic skin reflectance model, which we combined from two sources to better reflect the physical structure of the skin. Our objective is to compare the models to each other in terms of accuracy, usefulness and biophysical parameter retrieval using convolutional neural network. The results are promising. Both model are found suitable options for further research and used stochastic model is similar to Kubelka-Munk in terms of accuracy. In physical parameter retrieval both models perform moderately. Inverted models reasonably retrieve the pigment concentrations from the simulated test data set. With empirical testing data the inverted models are mutually consistent.

1 Introduction

There is a need for automated non-invasive diagnostic methods for different illnesses and diseases in the medical field. Especially in case of melanomas and other skin cancers, the accuracy of the clinical diagnostic tools are poor, resulting in unnecessary operations and re-operations [9]. A well working non-invasive detection method could decrease the number of unnecessary operations and therefore bring savings to the hospital. One potential technology is combination of hyperspectral cameras, machine learning and neural networks in skin diagnostics [21, 23, 25].

Machine learning, and particularly training of the neural networks, require large amount of training data. A way to avoid laborious data gathering process is to use

Leevi Annala (✉)
University of Jyväskylä, e-mail: leevi.a.annala@jyu.fi

Ilkka Pölönen
University of Jyväskylä, e-mail: ilkka.polonen@jyu.fi

mathematical modelling in producing such augmented data set. Mathematical models for skin reflectance can be roughly divided into two categories: deterministic and stochastic [4]. Deterministic models are models where the inputs directly determine the output. Stochastic models include randomness. Examples of deterministic models include multitude of Kubelka-Munk equation based models [12, 26, 3, 6, 17, 1, 31, 7], Boltzmann photon transport equation [10], diffusion theory models [29] and many more, while stochastic modelling is exclusively based on Monte Carlo modelling [8, 27, 32, 19]. Model, that augments training data for machine learning, should have useful input parameters for inversion. This model should be easy to understand and modify and it should be sufficiently accurate.

Examples of previous research in non-invasive methods to determinate biochemical and biophysical skin properties using hyperspectral imaging include a study where skin thickness was successfully retrieved from hyperspectral image using inverted Kubelka-Munk Model [30]. The results were verified by ultrasound imaging and the machine learning method used in inversion was support vector regressor. Jolivot *et al.* retrieved melanin and blood concentrations and skin layer thicknesses from multispectral images [12]. They inverted Kubelka-Munk Model using genetic algorithm.

In this chapter we compare the Kubelka-Munk implementation of Jolivot *et al.* [12] to our own implementation of Stochastic Model, which is based on multi-layered stochastic radiative transfer model by Maier *et al.* [19] and parameters described in [11]. Our objective is to use both models to skin reflectance modeling and compare them in terms of accuracy, usefulness and inversion with convolutional neural network [18]. Based on our knowledge this approach of stochastic modeling and convolutional neural network has not been used previously in skin physical parameter retrieval.

2 Materials and Methods

2.1 Stochastic Model

Stochastic Model (SM) is a Markov chain based model for the light propagation in layered media. The SM we use is modified from [19], by changing the pigments to those of the skin and using more general absorption and scattering coefficients. The basic principle of the SM is that the skin is seen as a network of states, and there is a certain probability of each transition between two states. The states and possible transitions are described in Figure 1. For this study, we assume that the skin has two layers: epidermis and dermis. Light that goes past these two layers is considered absorbed. The transition probabilities (P) are based on Beer's law and calculated as follows[19]. For up and down states the up/down, scattering and absorption probabilities are [19]

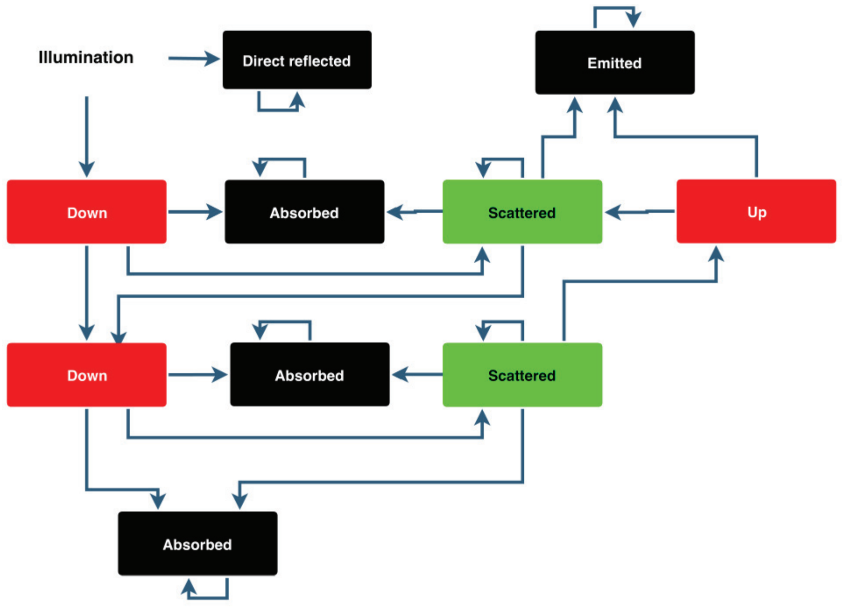


Fig. 1 Network of states and transitions in Stochastic Model.

$$P_{\text{up/down}}(\lambda) = 1 - P_{\text{absorption}} - P_{\text{scattering}}, \tag{1}$$

$$P_{\text{scattering}}(\lambda) = \frac{s(\lambda)}{a(\lambda)+s(\lambda)} \cdot (1 - e^{-(a(\lambda)+s(\lambda))\cdot L}), \text{ and} \tag{2}$$

$$P_{\text{absorption}}(\lambda) = \frac{a(\lambda)}{a(\lambda)+s(\lambda)} \cdot (1 - e^{-(a(\lambda)+s(\lambda))\cdot L}), \tag{3}$$

where $a(\lambda)$ is the absorption coefficient [11],

$$a(\lambda) = \sum_n a_i(\lambda)c_i, \tag{4}$$

$s(\lambda)$ is the reduced scattering coefficient [11],

$$s(\lambda) = s(500\,\text{nm}) \cdot \left(f_{\text{Ray}} \left(\frac{\lambda}{500\,\text{nm}} \right)^{-4} + (1 - f_{\text{Ray}}) \left(\frac{\lambda}{500\,\text{nm}} \right)^{-b_{\text{Mie}}} \right) \text{ and} \tag{5}$$

L is the length of the light path (in cm), which is assumed to be the same as the thickness of the layer. In the previous equations (4) and (5) a_i are absorption coefficients (in cm^{-1}) for each pigment, c_i are their concentrations (in fractions of total concentration), $s(500\,\text{nm})$ is the measured reduced scattering coefficient (in cm^{-1}) at 500 nm, f_{Ray} is the Rayleigh scatterings part of total scattering and b_{Mie} is Mie scattering power.

For the scattered state, the scattering and absorption probabilities are the same as for up/down states in the same layer. The up/down probabilities are

$$P_{\text{up}} = P_{\text{down}} = \frac{P_{\text{up/down}}}{2}, \tag{6}$$

as the photon can now go both ways. The probability of direct reflection is given as a parameter to the model $P_{dr} = 0.02$ and therefore transition probability to the first layers down state is $1 - P(dr) = 0.98$. Transition probabilities from absorbed, reflected or emitted state to itself is 1, and transition between states that are not connected is impossible.

The parameters needed for calculating the transition probabilities include pigment concentrations in skin layers, scattering coefficients and thicknesses of the skin layers and blood oxygenation level. These are sufficiently described in [11] and listed in Table 1. Example reflectance spectrum produced by SM can be seen in Figure 2. The values used in creating Figure 2 can be seen in Table 2.

Table 1 Input parameters and their ranges for Stochastic Model.

Input parameter	Range	Layer
Melanosome volume fraction	0 - 0.08	Epidermis
Blood volume fraction	0 - 0.01	Dermis
Blood oxygen fraction	0.2 - 0.5	Dermis
Water volume fraction	0.5 - 0.8	Dermis
Reduced scattering coefficient at 500nm (cm^{-1})	38 - 58	Both
Rayleigh scattering fraction	0.38 - 0.42	Both
Mie scattering power	0.3 - 1	Both
Thickness of epidermis (cm)	0.005 - 0.035	Epidermis
Thickness of dermis (cm)	0.1 - 0.4	Dermis

Table 2 Input parameters and their ranges for Stochastic Model and Kubelka-Munk Model in Figure 2.

Input parameter	Value for SM	Value for KM
Melanosome volume fraction	0.1	0.1
Blood volume fraction	0.02	0.02
Blood oxygen fraction	0.5	0.5
Water volume fraction	0.4	*Not applicable*
Reduced scattering coefficient at 500nm (cm^{-1})	48	48
Rayleigh scattering fraction	0.41	0.41
Mie scattering power	0.7	0.7
Thickness of epidermis	0.007 cm	0.000 07 m
Thickness of dermis	0.2 cm	0.002 m
Anisotropy	*Not applicable*	0

2.2 Kubelka-Munk Model

Kubelka-Munk Model (KM) is a special case solution to the radiative transfer equation [17]. The model consists of two differential equations for opposing light fluxes *I* and *J*:

$$\begin{cases} \frac{dI}{dx} &= -KI - SI + SJ \\ \frac{dJ}{dx} &= -KJ - SJ + SI, \end{cases} \tag{7}$$

where *K* and *S* are absorption and scattering functions, and *x* is the thickness of the media. Their reasoning and analytical solution can be found for example in [22].

Our implementation of the model follows the implementation described in [12]. Compared to that we changed the scattering coefficient to the same we used in SM, from [11]. The model takes into account the two first main layers of the skin: epidermis and dermis. Light that goes through these layers is considered absorbed. The parameters and their used ranges are adapted from [11, 12] and are described in Table 3. Details of the implementation can be found in [12]. Example spectrum produced by KM can be seen in Figure 2. The values used in creating Figure 2 can be seen in Table 2.

Table 3 Inputs and their ranges for KM.

Input parameter	Range	Layer
Melanosome volume fraction	0.01 - 3.01	Epidermis
Blood volume fraction	0.001 - 0.501	Dermis
Blood oxygen fraction	0.6 - 0.99	Dermis
Reduced scattering coefficient at 500nm (cm^{-1})	38 - 58	Both
Rayleigh scattering fraction	0.38 - 0.42	Both
Mie scattering power	0.3 - 1	Both
Thickness of epidermis (m)	0.0001 - 0.0006	Epidermis
Thickness of dermis (m)	0.001 - 0.004	Dermis
Anisotropy	0.7 - 0.8	Both

2.3 Convolutional neural network

Convolutional neural network (CNN) is a neural network where at least one of the traditional fully connected layers is replaced with convolutional layer [18]. It has been found useful in various tasks including image and signal type data [16, 13, 28, 5, 24].

Our CNN implementation can be seen in Table 4. It consists of two convolutional layers and three dense layers. Additionally there are two pooling layers and one dropout layer and the output layer. It is very conventional CNN. The optimization algorithm used is the Adam-algorithm [14] with meta parameters of learning rate =

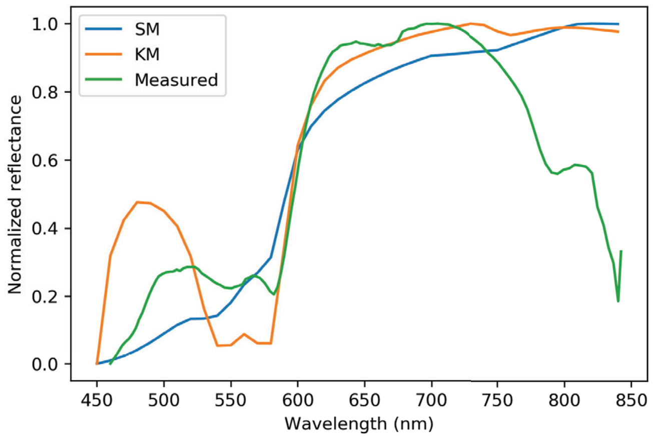


Fig. 2 Example spectra produced by stochastic and KMs using realistic values and a measured spectrum.

0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1 \cdot 10^{-7}$. As the loss function we use the mean square error. The CNN is implemented with Keras using Tensorflow backend [20]. CNN is used in inverting the KM and SM models.

Table 4 Convolutional neural network used in inversions.

Layer	Kernel / pool size or Activation	Output Shape	Parameters
Conv1D	(3) ReLU	(38, 64)	256
MaxPooling1D	(2)	(19, 64)	0
Conv1D	(3) ReLU	(17, 128)	24704
MaxPooling1D	(2)	(8, 128)	0
Flatten		(1024)	0
Dense	ReLU	(128)	131200
Dense	ReLU	(64)	8256
Dense	ReLU	(32)	2080
Dropout (0.25)		(32)	0
Dense		(9)	297
Total params:	166,793		
Trainable params:	166,793		
Non-trainable params:	0		

Table 5 Correlation coefficients of values retrieved from Stochastic Model.

Retrieved parameter	Correlation coefficient between estimated and real value
Melanosome	0.96
Blood volume fraction	0.87
Blood oxygen fraction	0.07
Water volume fraction	0.27
Reduced scattering coefficient at 500nm	0.57
Rayleigh scattering fraction	0.11
Mie scattering power	0.92
Thickness of epidermis	0.95
Thickness of dermis	0.96

2.4 Model Inversion

Both SM and KM are inversed by CNN and the inversion results are used to predict parameters from simulated and empirical data. For the inversion, the training and validation labels described in Tables 1 and 3 are scaled to range from 0 to 1 in order to receive best possible performance from CNN. Hence, the predictions are also in range from 0 to 1. For predictions using simulated data, correlation coefficients are calculated and analyzed. Predictions from empirical data are visually interpreted and their potential for further research is discussed.

Our empirical data consinsts of a hyperspectral image of human skin with a large nevus. Example spectrum can be seen in Figure 2.

3 Results and Discussion

3.1 Retrieval Results

The inversion results for SM were strongest at dermis and epidermis thicknesses and melanosome concentration, with correlations of 0.96, 0.95 and 0.96 respectively. The weakest correlations were in blood oxygenation, Rayleigh scattering fraction, and water volume fraction at 0.07, 0.11, and 0.27, respectively. Altogether three correlations could be considered weak, one moderate and five strong. (Figure 3, Table 5)

The results of the inversion of the KM had strongest retrieval correlations in blood oxygen fraction, blood volume fraction and thicknesses of epidermis and dermis at 0.99, 0.94, 0.94 and 0.92 respectively. The weakest correlations were in Rayleigh scattering fraction and Mie scattering power at 0.02 and 0.29, respectively. Altogether two correlations could be considered weak, two moderate and five strong.

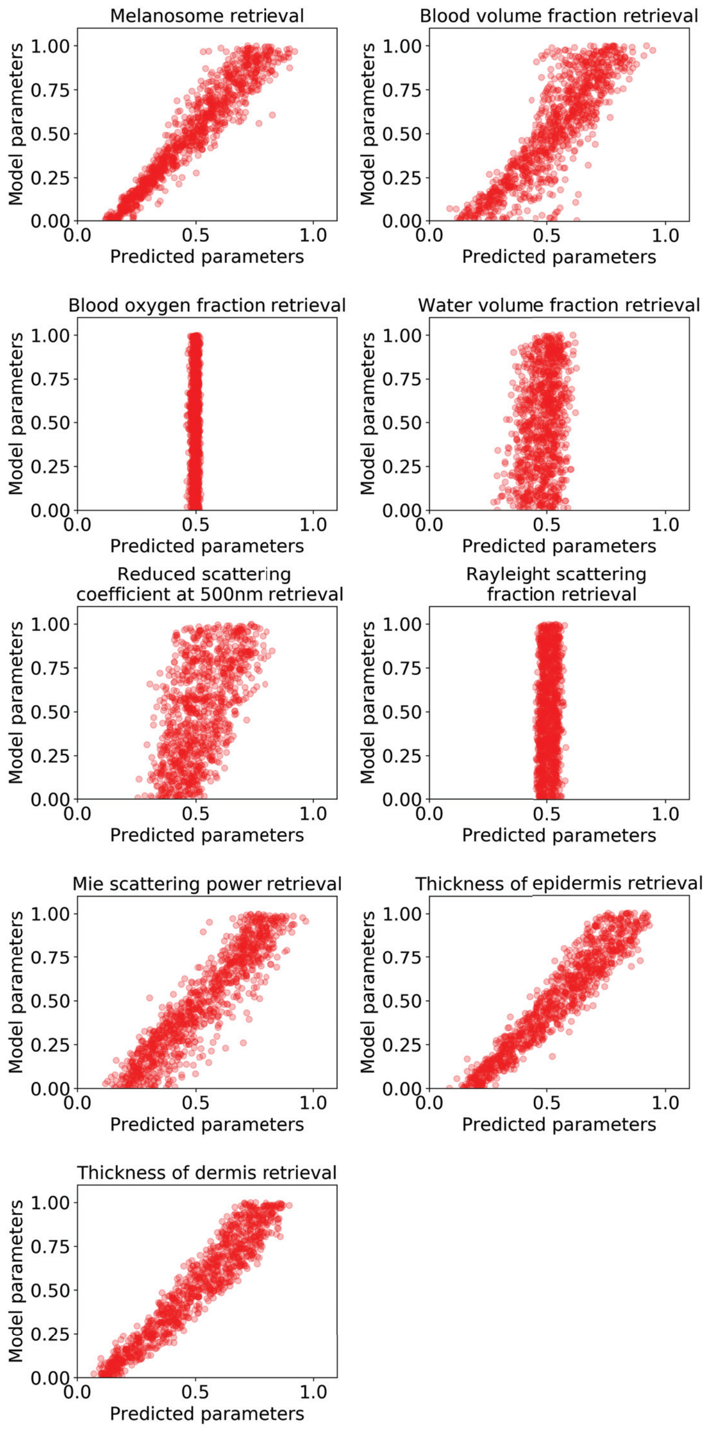


Fig. 3 Inversion of Stochastic Model.

Table 6 Correlation coefficients of values retrieved from Kubelka-Munk Model.

Retrieved parameter	Correlation coefficient between estimated and real value
Melanosome	0.89
Blood volume fraction	0.94
Blood oxygen fraction	0.99
Reduced scattering coefficient at 500nm	0.40
Rayleigh scattering fraction	0.02
Mie scattering power	0.29
Thickness of epidermis	0.94
Thickness of dermis	0.92
Anisotropy	0.39

The results for empirical data (Figure 5) showed that there is potential for further research using both models as a training data source for CNN. The models were mutually consistent in showing higher and lower values for different parameters and at least for the melanin concentration the models rightly predicted higher melanin concentrations on the area of the nevus.

3.2 Model Comparison

3.2.1 Accuracy

Both of these models seemed to produce spectra, which mimic skin reflectance, although the the effect of haemoglobin absorption (450-550 nm) in KM spectrum was suspiciously symmetrical (Figure 2). The haemoglobin absorption seems to have too much influence to the KM while the SM seems to be influenced too little. The spectrum of KM seemed similar to normal skin while the spectrum of SM appeared to be similar to pale (less blood) skin [2, 11, 15]. The fact that parameters given to the KM (Table 3) were not realistic decreases the KMs credibility. Especially melanosome volume fraction was too high. The accuracy of the KM we used has been partially verified previously by inverting the model with evolutionary algorithm and retrieving plausible pigment concentrations from living skin [12].

The accuracy of SM was verified in previous research with pigments typical to plants [19]. The modifications we made did not change the mathematical core of the model, therefore the accuracy of the model is derived from the accuracy of the pigment absorption spectrums.

The accuracy of the inversions varied for both models. From SM, predictions (Figure 3) of skin layer thicknesses and melanosome concentration were well predicted and Mie scattering power and blood volume fraction less accurately. For the rest of the parameters the accuracy was non-existent.

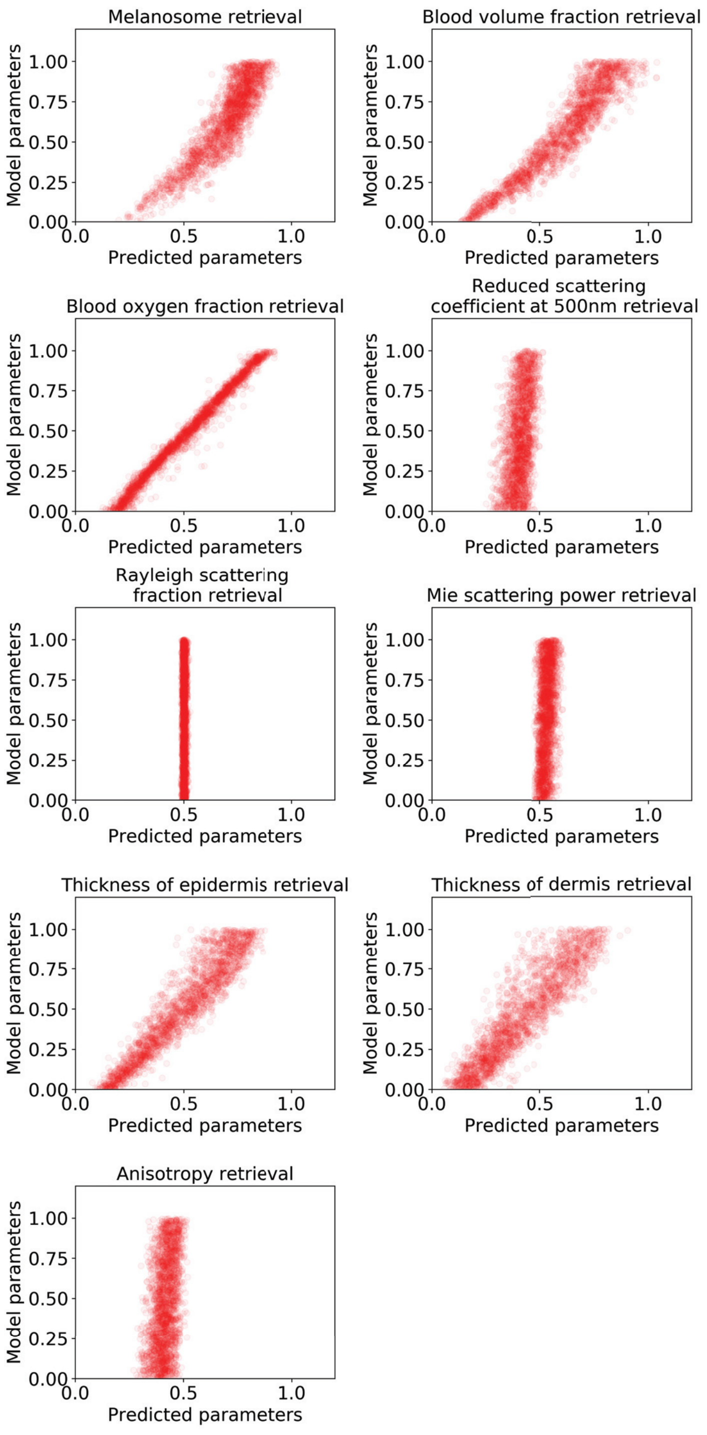


Fig. 4 Inversion of Kubelka-Munk Model.

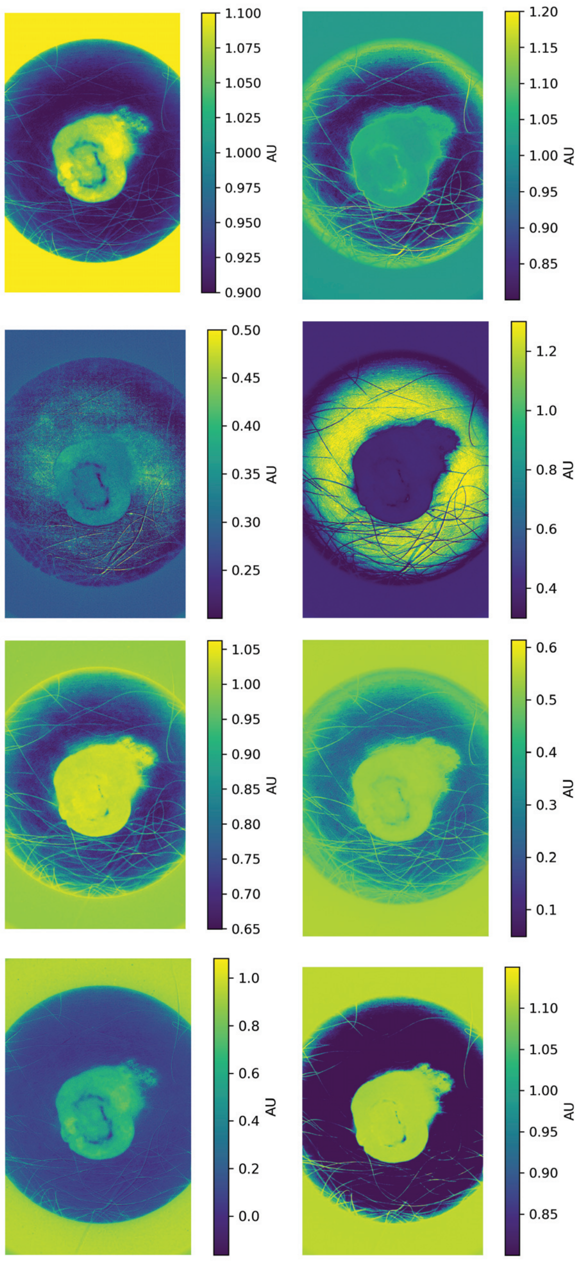


Fig. 5 Values predicted by inversed Kubelka-Munk Model (left) and Stochastic Model (right). Predicted parameters are, from top to bottom, Melanosome volume fraction, blood volume fraction, epidermis thickness and dermis thickness. The units are arbitrary.

From KM (Figure 4), the blood oxygen fraction and blood volume fraction were particularly well predicted and melanosome concentration, and skin layer thicknesses were little less accurate. Once again, for the rest of the parameters (4), the accuracy was poor.

3.2.2 Usefulness

KM is the most used in the field of colour modelling [26], and it is sufficiently accurate while being easy to understand and fast to calculate. Our implementation of SM takes a while to calculate, but it seems like it has potential to be more accurate than KM.

For physical parameter retrieval, the models were useful in different areas. For some reason the scattering parameters seemed to have little to no impact on the KM, while for the SM the result were a little bit better. KM seemed more likely to return the correct values on the actual pigment concentrations, and SM gave better results at evaluating the thicknesses of the skin layers.

3.3 Discussion

The results indicate that the both KM and SM can be used as a data augmentation source in physical parameter retrieval from skin. KMs strengths are in calculation time and its simplicity. SMs strengths are higher potential accuracy and adaptability, as the absorption and scattering coefficients are independently tunable and the layers are easily added and removed and probability calculations can be changed if needed. With more investigation the reason for the irretrievable parameters could be found. The easiest hypothesis is that they simply do not affect the spectrum enough to be retrieved, and based on Figure 2, this seems to be the case for SM. Water absorption in the studied wavelength range is quite small compared to the other parameters [11], and haemoglobin absorption has a negative peak between 450 nm and 550 nm [15], and our SM does not seem to show it. In KM retrieval, only parameters related to scattering are poorly retrieved. They are also poorly retrieved in SM and they appear to affect the spectrum very little.

When we compare the spectra in Figure 2 to previous research it is clear that the measured skin spectra can be replicated pretty closely [2]. The empirical implications are left for further research, but this research and previous research have given us reason to believe that the SM can be well adapted for accurate skin reflectance modelling [19]. If this holds, it means that we could use SM and a skin structure model to produce accurate training data for machine learning applications in medical field. This would decrease the need for data gathering for such application. However, measured data always includes noise, therefore one needs to find a way to introduce realistic noise to the model.

4 Conclusion

We have demonstrated that the Stochastic Model originally developed for leaf optical properties can be successfully transported to skin reflectance modelling by changing equations and parameters of the model to skin related equations and parameters. We have demonstrated that the model has similar accuracy with Kubelka-Munk Model, while being easier to modify. It is up for debate if the probability and transition net based Stochastic Model is easier to understand when compared to Kubelka-Munk Model, which is based on solutions of a differential equations. The most direct implications of our work are, that the model should be first verified and further adapted to skin reflectance modelling and then tested in machine learning applications in the medical field.

References

1. R. Rox Anderson and John A. Parrish. The Optics of Human Skin. *Journal of Investigative Dermatology*, 77(1):13–19, July 1981.
2. Elli Angelopoulou. Understanding the color of human skin. In *Human Vision and Electronic Imaging VI*, volume 4299, pages 243–251. International Society for Optics and Photonics, June 2001.
3. M. Gorji Bandpay, F. Ameri, K. Ansari, and S. Moradian. Mathematical and empirical evaluation of accuracy of the Kubelka–Munk model for color match prediction of opaque and translucent surface coatings. *Journal of Coatings Technology and Research*, 15(5):1117–1131, September 2018.
4. Gladimir VG Baranoski and Tenn F Chen. Multilayer modeling of skin color and translucency. 2014.
5. Hubert Cecotti and Axel Graeser. Convolutional neural network with embedded Fourier transform for EEG classification. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pages 1–4. IEEE, 2008.
6. B. L. Diffey. A mathematical model for ultraviolet optics in skin. *Physics in Medicine and Biology*, 28(6):647–657, June 1983.
7. Motonori Doi and Shoji Tominaga. Spectral estimation of human skin color using the Kubelka-Munk theory. In *Color Imaging VIII: Processing, Hardcopy, and Applications*, volume 5008, pages 221–228. International Society for Optics and Photonics, 2003.
8. John Hammersley. *Monte carlo methods*. Springer Science & Business Media, 2013.
9. Clare F Heal, Beverley A Raasch, PG Buettner, and David Weedon. Accuracy of clinical diagnosis of skin lesions. *British Journal of Dermatology*, 159(3):661–668, 2008.
10. Akira Ishimaru. *Wave propagation and scattering in random media*, volume 2. Academic press New York, 1978.
11. Steven L Jacques. Optical properties of biological tissues: a review. *Physics in Medicine & Biology*, 58(11):R37, 2013.
12. Romuald Jolivot, Yannick Benezeth, and Franck Marzani. Skin parameter map retrieval from a dedicated multispectral imaging system applied to dermatology/cosmetology. *Journal of Biomedical Imaging*, 2013:26, 2013.
13. Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.
14. Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

15. Aravind Krishnaswamy and Gladimir VG Baranoski. A study on skin optics. *Natural Phenomena Simulation Group, School of Computer Science, University of Waterloo, Canada, Technical Report*, 1:1–17, 2004.
16. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
17. Paul Kubelka. Ein Beitrag zur Optik der Farbanstriche (Contribution to the optic of paint). *Zeitschrift fur technische Physik*, 12:593–601, 1931.
18. Yann LeCun and Yoshua Bengio. Convolutional networks for images, speech, and time-series. *The handbook of brain theory and neural networks*, 1995.
19. S. W. Maier, W. Lüdeker, and K. P. Günther. SLOP: A Revised Version of the Stochastic Model for Leaf Optical Properties. *Remote Sensing of Environment*, 68(3):273 – 280, 1999.
20. Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015.
21. Noora Neittaanmäki, Mari Salmivuori, Ilkka Pölönen, L Jeskanen, A Ranki, O Saksela, E Snellman, and M Grönroos. Hyperspectral imaging in detecting dermal invasion in lentigo maligna melanoma. *British Journal of Dermatology*, 177, 2017.
22. James H. Nobbs. Kubelka—Munk Theory and the Prediction of Reflectance. *Review of Progress in Coloration and Related Topics*, 15(1):66–75, 1985.
23. Ilkka Pölönen, Samuli Rahkonen, Leevi Annala, and Noora Neittaanmäki. Convolutional neural networks in skin cancer detection using spatial and spectral domain. In *Photonics in Dermatology and Plastic Surgery 2019*, volume 10851, page 108510B. International Society for Optics and Photonics, 2019.
24. Justin Salamon and Juan Pablo Bello. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters*, 24(3):279–283, 2017.
25. Mari Salmivuori, N Neittaanmäki, Ilkka Pölönen, L Jeskanen, Erna Snellman, and Mari Grönroos. Hyperspectral imaging system in the delineation of Ill-defined basal cell carcinomas: a pilot study. *Journal of the European Academy of Dermatology and Venereology*, 33(1):71–78, 2019.
26. T. Shakespeare. *Colorant modelling for on-line paper coloring: evaluations of models and an extension to Kubelka-Munk model*. Number 304 in Tampereen teknillinen korkeakoulu. Julkaisuja. Tampere University of Technology, 2000.
27. M Shimada, Y Yamada, M Itoh, and T Yatagai. Melanin and blood concentration in a human skin model studied by multiple regression analysis: assessment by Monte Carlo simulation. *Physics in Medicine & Biology*, 46(9):2397, 2001.
28. Christian Szegedy, Alexander Toshev, and Dumitru Erhan. Deep neural networks for object detection. In *Advances in neural information processing systems*, pages 2553–2561, 2013.
29. Martin JC Van Gemert, Ashley J Welch, Willem M Star, Massoud Motamedi, and Wai-Fung Cheong. Tissue optics for a slab geometry in the diffusion approximation. *Lasers in medical Science*, 2(4):295–302, 1987.
30. Saurabh Vyas, Jon Meyerle, and Philippe Burlina. Non-invasive estimation of skin thickness from hyperspectral imaging and validation using echography. *Computers in biology and medicine*, 57:173–181, 2015.
31. San Wan, R Rox Anderson, and John A Parrish. Analytical modeling for the optical properties of the skin with in vitro and in vivo applications. *Photochemistry and Photobiology*, 34(4):493–499, 1981.

32. Lihong Wang, Steven L Jacques, and Liqiong Zheng. MCML—Monte Carlo modeling of light transport in multi-layered tissues. *Computer methods and programs in biomedicine*, 47(2):131–146, 1995.

PVI

COMPARISON OF MACHINE LEARNING METHODS IN STOCHASTIC SKIN OPTICAL MODEL INVERSION

by

Leevi Annala, Sami Äyrämö and Ilkka Pölönen 2020

Applied Sciences, 10(20): 7097

https://doi.org/10.3390/app10207097

applied sciences

Article

Comparison of Machine Learning Methods in Stochastic Skin Optical Model Inversion

Leevi Annala *, Sami Äyrämö and Ilkka Pölönen

Faculty of Information Technology, University of Jyväskylä, PL35, 40014 Jyväskylän yliopisto, Finland; sami.ayramo@jyu.fi (S.Ä.); ilkka.polonen@jyu.fi (I.P.)

* Correspondence: leevi.a.annala@jyu.fi

Received: 31 August 2020 ; Accepted: 5 October 2020; Published: 13 October 2020

Featured Application: This research can potentially be applied in improving the accuracy of clinical skin cancer diagnostics.

Abstract: In this study, we compare six different machine learning methods in the inversion of a stochastic model for light propagation in layered media, and use the inverse models to estimate four parameters of the skin from the simulated data: melanin concentration, hemoglobin volume fraction, and thicknesses of epidermis and dermis. The aim of this study is to determine the best methods for stochastic model inversion in order to improve current methods in skin related cancer diagnostics and in the future develop a non-invasive way to measure the physical parameters of the skin based partially on the results of the study. Of the compared methods, which are convolutional neural network, multi-layer perceptron, lasso, stochastic gradient descent, and linear support vector machine regressors, we find the convolutional neural network to be the most accurate in the inversion task.

Keywords: skin; physical parameter retrieval; neural networks; convolutional neural network; machine learning; model inversion

1. Introduction

Skin related diseases such as skin cancer are common [1], and non-invasive methods for diagnosing them are needed. According to Le et al. [2], the melanoma incidence is increasing. The incidence between 2009–2016 was 491.1 cases per hundred thousand people, which is nearly 64% more compared to the time period of 1999–2008. The difference in incidence at an older age is even starker, as the incidence nearly doubled from 1278.1 to 2424.9 in people older than 70 years old [2]. The cost of the melanoma for the society has the same trend and early detection and accurate treatment lowers these costs and improves the life expectancy of the patients. Hyperspectral imaging is one way for the early detection and guidance for the treatment. Skin physical parameter retrieval with hyperspectral camera or spectrometer and machine learning (ML) provide a non-invasive method of measuring the chromophore concentrations and other parameters in the skin [3].

One hinderance to developing ML models for clinical use is that the needed training datasets are large and difficult to produce. The ethical standards of using human testing make it hard to obtain data, and there needs to be a team of medical staff in addition to computer science specialists for the work. One way to avoid some of these pitfalls is to use mathematical modeling to produce training data for ML algorithms. In this study, we use the stochastic model, which is partially of our own design.

In our previous research [4,5], we have used convolutional neural networks (CNN) for stochastic model inversion. In [4], we used CNN in inverting the stochastic model for leaf optical properties (SLOP) [6], which was the inspiration for the stochastic model used in this study. We found the inversion successful. In [5], we compared the invertibility and usefulness in the physical parameter retrieval from the stochastic model used in this study and a Kubelka–Munk model [7] by inversion with CNN. It has since occurred to us that it would be useful to verify the applicability of the CNN networks in model inversion by comparing it to other ML algorithms.

The ML algorithms have been compared multiple times in different hyperspectral imaging scenarios. For example, one study compared ML algorithms in assessing strawberry foliage Anthracnose disease stage classification [8]. They compared spectral angle mapper, stepwise discriminant analysis, and their own spectral index they call simple slope measure. These algorithms did not show good performance in the task, with classification accuracy just breaking 80%. Another study found a least mean squares classifier to perform best in classifying a small batch of lamb muscles compared to six other machine learning algorithms including support vector machine (SVM) approaches, simple neural networks, nearest neighbor algorithm, and linear discriminator analysis [9]. The algorithms were also tested using principal component analysis (PCA) for dimensionality reduction in training and testing data. They found no statistical differences in the classification results between using PCA for the data or not.

Gewali et al. [10] have written a survey on machine learning methods in hyperspectral remote sensing tasks. The retrieval of (bio)physical or (bio)chemical parameters from the hyperspectral images was one of the tasks, and they found three articles where SVMs were used in retrieval, five articles that used latent linear models such as PCA, four that used ensemble learning, and five that used Gaussian processes. Surprisingly, they found no articles where deep learning was used in retrieval.

In the articles [8–10], the distinction to our work is that they do not apply mathematical modeling prior to applying machine learning. In contrast, the following articles by Vyas et al. and Liang et al. are similar to our work, only employing different mathematical and machine learning models [11–13].

Liang et al. [11] used simulated PROSAIL data to select optimal vegetation indices for leaf and canopy chlorophyll content prediction from an inverted PROSAIL model. In the inversion, they used an SVM algorithm and a random forest (RF) algorithm, of which they found RF to be better. The results were promising, as the coefficient of determination (r^2) was 0.96 between measured and predicted data. However, their usage of indices makes it rather incomparable to our research.

Vyas et al. [12,13] have done the work most similar to ours. In [12], the Kubelka–Munk (KM) model was inverted using a k nearest neighbors (k-NN) method with different distance metrics and a support vector regressor. The inversion parameters were melanosome volume fraction, collagen volume fraction, hemoglobin oxygenation percentage, and blood volume fraction. In the synthetic experiment, they found the k-NN with spectral angle distance to have the smallest mean absolute errors. In the in vivo experiment, they show that the predicted parameters produce modeled spectra strikingly similar to measured spectra. This is used as evidence of inversion success.

In their other study [13], the KM model was inverted using an SVM approach. The inversion parameter was thickness of the skin. The linear correlation coefficient (r) between inverted KM predictions and ultrasound measurements of the skin thickness was 0.999. In further experiments, they found the measured and modeled spectra nearly indistinguishable, although it is not disclosed how they chose the other parameters for the modeling. The difference to our approach in aforementioned studies is that the inverted models differ from ours, as we are trying to find useful models alternative to KM.

The objective of our study is to find a good way to invert the stochastic model for skin optical properties. Our hypothesis is that the convolutional neural network (CNN) will outperform the others as it has been shown to perform well in similar tasks [14–17].

In Section 2, we provide the reader with information of our data, the methods we use, including the stochastic model and the different machine learning algorithms, and the metrics we use in evaluating the results. In Section 3, we show the experimental results and, in Section 4, we compare our results to the previous research, discuss the strengths and weaknesses of our work, and discuss the direction of future work. Section 5 concludes the work.

2. Materials and Methods

2.1. Data

2.1.1. Stochastic Model

The training data for the machine learning methods was produced by a stochastic model for light propagation in the skin (SM). The SM was adapted from the stochastic model for leaf optical properties [6] by using skin specific parameters [18].

SM is the Markov chain based model for light propagation. The light propagation can be expressed as a network of transitions and states (Figure 1), in which there is a physically meaningful probability of transfer from one state to another. The probabilities are based on the chromophore concentrations and absorption and scattering properties of the skin. SM is used in the same way and with the same parameters as in [5].

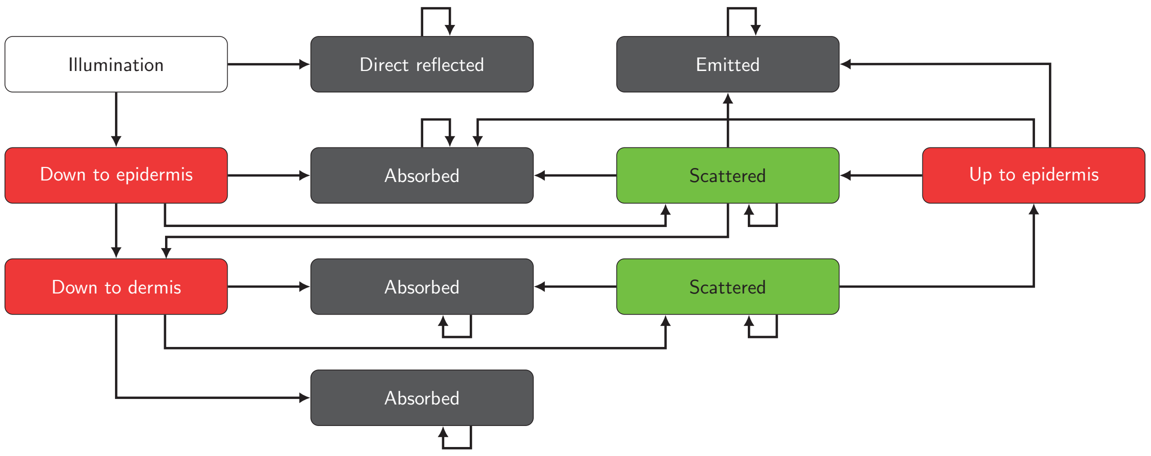


Figure 1. Network of states and transitions of light propagation in the Stochastic Model.

In this research, the first two main skin layers, epidermis and dermis, were considered. Light through dermis was considered absorbed. From the first state, illumination, the light either goes into the skin ($P = 0.98$) or is reflected ($P = 0.02$). In the epidermis and dermis layers, the transition probabilities are based on Beer's law and calculated as follows [6]:

$$P_{\text{absorption}}(\lambda) = \begin{cases} 1 & \text{if the current state is absorbed,} \\ \frac{a(\lambda)}{a(\lambda)+s(\lambda)} \cdot (1 - e^{-(a(\lambda)+s(\lambda))\cdot L}) & \text{otherwise,} \end{cases} \tag{1}$$

$$P_{\text{scattering}}(\lambda) = \begin{cases} 0 & \text{if the current state is absorbed,} \\ \frac{s(\lambda)}{a(\lambda)+s(\lambda)} \cdot (1 - e^{-(a(\lambda)+s(\lambda))\cdot L}) & \text{otherwise,} \end{cases} \tag{2}$$

$$P_{\text{up/down}}(\lambda) = \begin{cases} 1 - P_{\text{absorption}} - P_{\text{scattering}} & \text{if the current state is up or down,} \\ \frac{1-P_{\text{absorption}}-P_{\text{scattering}}}{2} & \text{if state is scattered,} \\ 0 & \text{if state is absorbed.} \end{cases} \tag{3}$$

In the previous equations, λ is the wavelength in nanometers, $a(\lambda)$ is the absorption coefficient [18], and $s(\lambda)$ is the reduced scattering coefficient [18] and L is the length of the light path, which is assumed to be the same as the thickness of the layer [18].

$$a(\lambda) = \sum_{n} a_i(\lambda)c_i, \text{ and} \tag{4}$$

$$s(\lambda) = s(500\,\text{nm}) \cdot f_{\text{Ray}} \left(\frac{\lambda}{500\,\text{nm}}\right)^{-4} + (1 - f_{\text{Ray}}) \left(\frac{\lambda}{500\,\text{nm}}\right)^{-b_{\text{Mie}}} \tag{5}$$

In Equations (4) and (5), a_i are the skin chromophore absorption coefficients and c_i their concentrations, $s(500\,\text{nm})$ the reduced scattering coefficient at 500 nm, f_{Ray} fraction of the Rayleigh scattering and b_{Mie} the Mie scattering power. The chromophores used in the study are melanin, deoxygenated and oxygenated hemoglobin and water. Their absorption coefficients are from [18], and the concentrations are in fractions of the volume they take. The deoxygenated and oxygenated hemoglobin volume fractions are calculated from blood oxygen fraction (S) and hemoglobin volume fraction (c_{HGb}) as follows [18]:

$$c_{\text{HGb, deoxy}} = c_{\text{HGb}}(1 - S) \tag{6}$$

$$c_{\text{HGb, oxy}} = c_{\text{HGb}}S \tag{7}$$

The input parameters used in SM are described in Table 1. Example spectra produced by the stochastic model are in Figure 2.

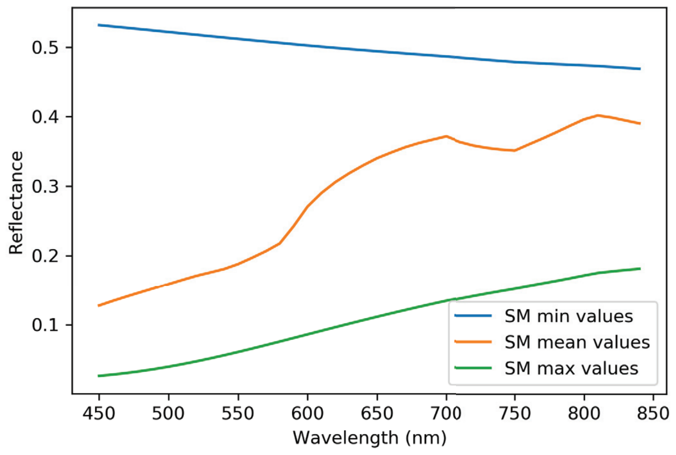


Figure 2. Example spectra produced by the used stochastic model. Minimum, mean and maximum values from Table 1 were used.

Table 1. Input parameters and their ranges for the stochastic model. The parameters that are used as prediction targets are marked with (*). The wavelengths are described in detail in Table 2 [5].

Input Parameter	Range	Layer
* Melanosome volume fraction, $c_{\text{Melanosome}}$	0–0.08	Epidermis
* Hemoglobin volume fraction, c_{HGb}	0–0.01	Dermis
Blood oxygen saturation, S	0.2–0.5	Dermis
Water volume fraction, c_{water}	0.5–0.8	Dermis
Reduced scattering coefficient at 500 nm, $s(500\,\text{nm})$	38–58	Both
Rayleigh scattering fraction, f_{Ray}	0.38–0.42	Both
Mie scattering power, b_{Mie}	0.3–1	Both
* Thickness of epidermis, $L_{\text{epidermis}}$ (cm)	0.005–0.035	Epidermis
* Thickness of dermis, L_{dermis} (cm)	0.1–0.4	Dermis
Wavelength, λ (nm)	460–843	Both

For machine learning algorithm training, the hyperspectral data simulated with SM are normalized. The prediction target parameters, melanosome, and hemoglobin volume fractions, and epidermis and dermis thicknesses, are normalized to between 0 and 1, and the spectral data are normalized to between 0 and 1 for the neural network algorithms (convolutional and regular neural networks described in Sections 2.2.2 and 2.2.3) and to between -1 and 1 for scikit-learn algorithm implementations (other algorithms, described in Sections 2.2.4–2.2.6) by the StandardScaler algorithm [19].

2.1.2. Empirical Data

The empirical data were used for visual inspection of the trained machine learning regressors. The used hyperspectral image was captured using Revenio Prototype 2016 hyperspectral imager with spatial resolution of 1920×1200 pixels and spectral resolution of 120 wavelengths (Table 2). The hyperspectral camera/imager is a device that captures multiple monochrome photographs in rapid succession, while controlling the wavelength of the light that gets through the controlling device to the imaging sensors. Each monochrome photograph represents a narrow wavelength interval that can be interpreted as single wavelength, such as 400 nm. There are usually approximately one hundred of these monochrome images in one hyperspectral image. The hyperspectral image contains the same spatial data as the normal RGB-picture, but far more data in the spectral domain, that is, the interaction between light and the subject of the image can be seen with greater precision [20].

Table 2. Wavelength of the measured hyperspectral data and the wavelengths used in training data production.

Wavelength (nm):	460,	461.84,	464.2,	466.43,	468.96,	471.16,	473.83,	476.66,	479.12,
	481.5,	483.88,	486.82,	489.01,	491.28,	494.12,	496.09,	498.67,	501.44,
	504.48,	506.89,	509.53,	512.15,	514.76,	517.53,	520.37,	523.1,	525.76,
	528.48,	531.25,	534.27,	536.97,	539.63,	542.39,	545.15,	547.8,	550.48,
	553.09,	555.93,	558.72,	561.27,	564.1,	566.55,	569.19,	571.85,	575.39,
	579.44,	582.22,	584.92,	587.49,	590.13,	592.77,	595.46,	598.37,	600.98,
	603.7,	606.47,	609.16,	612,	615.07,	617.84,	621.01,	623.53,	626.28,
	629.25,	632.06,	634.8,	637.95,	640.51,	643.59,	646.8,	649.04,	651.97,
	654.93,	657.75,	660.5,	663.63,	666.51,	669.53,	672.59,	675,	678.46,
	682.67,	687.25,	691.66,	696.16,	700.39,	704.64,	708.56,	712.78,	716.63,
	720.81,	725.04,	729.34,	733.99,	738.38,	742.65,	746.64,	751.27,	755.19,
	759.42,	763.86,	768.13,	772.48,	776.92,	781.83,	786.09,	790.58,	795.01,
	799.41,	803.62,	807.85,	812.18,	816.33,	820.69,	824.78,	829.33,	832.96,
	836.72,	840.25,	842.35						

2.2. Machine Learning Models

The inversion methods we compare are multi-layer perceptron (MLP) [21], convolutional neural network (CNN) [16], stochastic gradient descent regressor (SGD) [22], linear support vector regressor (SVR) [23], and lasso regressor [24].

The machine learning methods for inversion are selected by following criteria:

1. Applicability for special characters of hyperspectral data, such as
    - nonlinearity,
    - sparsity, i.e., some wavelengths have greater significance.
2. Applicability for the dataset and sample size

The multi-layer perceptron was chosen based on its applicability to nonlinear data [25,26], which is usually the case when light is scattering in complex media. The convolutional neural network fills both criteria, as it is proven to be good in image [14,15] and signal regression [16,17] problems, and it scales well to big datasets. It also takes the structure/shape of the data into account by use of convolution. The stochastic gradient descent regressor is also used in signal regression, and, according to scikit-learn documentation, version 0.23.2 [19], it is also a good fit for the size of our data, which is in the range of hundreds of thousands pixels. The lasso regressor was chosen based on the first criteria. It is also good for sparse problems [27], which means only some of the input parameters are important. This is the case with spectral data: the spectra will follow the same line in many places, but the differences are found in the wavelength zone of the chromophore of interest. The linear support vector regression was included based on its applicability on pattern recognition [28].

All machine learning algorithms were either implemented in scikit-learn Python library [19] version 0.20.1 or written by the authors using Python programming language and Tensorflow library [29] version 2.0.0.

2.2.1. Training and Testing Process

All the models were trained and tested with the simulated data. The data set consisted of 50,000 simulated spectra with associated target variables, which were melanosome volume fraction, hemoglobin volume fraction, epidermis thickness, and dermis thickness. All algorithms were trained for all targets simultaneously, resulting in regressors with an output size of four.

The data set was divided into training (n = 48,000) and test samples (n = 2000). For each method, the best set of hyperparameter values was found by the grid search method [19] and 5-fold cross-validation strategy on training samples. The final model was then trained by using all the 48,000 training samples and the best combinations of hyperparameter values. The performance of each model was then assessed with the test samples.

In addition to the simulated training and test data, empirical data were used in visual interpretation of the trained models.

2.2.2. Convolutional Neural Network

CNN is widely used in image related regression and classification tasks [14–17]. The state-of-the-art image classification algorithms utilize some form of CNN [30,31]. There is also a lot of research for it being used with hyperspectral images [10]. The CNN algorithms we used are described in Tables 3–6.

In the first experiment with grid search (Table 3), we varied the shape of two convolutional layers, the size of the convolution kernel, and size of the max-pooling [32] window. The resulting network (Table 4) had two one-dimensional, convolution layers with convolution window of size 12, 32, and 64 filters, respectively. After each convolution, there were one-dimensional, max-pooling with pool size of two. Both convolutional layers have rectified linear unit (ReLU) [33] activation. After the last max-pooling layer, there was a flattening layer and three dense layers (ReLU activation), dropout of 50%, and an output layer. The first experiment is referred to in the results and the discussion as CNN.

Table 3. Grid search parameters for the first convolutional neural network experiment, abbreviated as CNN in the results and discussion. The best parameters are bolded.

Parameter	Values
Convolution filters	(64 and 128), (128 and 256), **(32 and 64)**
Convolution kernel size	6, 9, **12**
MaxPooling pool size	1, **2**, 3

In the second experiment, we varied the amount of convolutional layers, the convolution kernel size, and size of the max-pooling window (Table 5). The resulting network (Table 6) consisted of a one-dimensional, convolutional neural network, max-pooling, two one-dimensional, convolution layers, another max-pooling, flattening layer three dense layers, dropout of 50% [34], and output layer. All dense and convolutional layers except output layers utilized ReLU activation. Every convolutional layer had a convolution window of size 15, and the max-pooling pool size was two. The second experiment is referred to in the results and discussion as CNNV2.

The used optimizer was an Adam optimizer [35] in both experiments. The learning rate was 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-7}$, which were not included as the optimization parameters as the focus was on the architecture of the network. The loss score was mean square error (MSE).

Table 4. The convolutional neural network used in first CNN inversion experiment. The best parameters from Table 3 were used.

Layer	Kernel/Pool Size	Activation	Output Shape	Parameters
Conv1D	(12)	ReLU	(109, 32)	416
MaxPooling1D	(2)		(54, 32)	0
Conv1D	(12)	ReLU	(43, 64)	24,640
MaxPooling1D	(2)		(21, 64)	0
Flatten			(1344)	0
Dense		ReLU	(128)	172,160
Dense		ReLU	(64)	8256
Dense		ReLU	(32)	2080
Dropout (0.5)			(32)	0
Dense			(4)	132
Total params:	207,684			
Trainable params:	207,684			
Non-trainable params:	0			

Table 5. Grid search parameters for the second CNN experiment, abbreviated as CNNV2. The best parameters are bolded.

Parameter	Values
Amount of convolution layers	0, **3**, 6
Convolution kernel size	9, 12, **15**
MaxPooling pool size	1, **2**, 4

Table 6. Convolutional neural network used in the CNNV2 experiment. The best parameters from Table 5 were used.

Layer	Kernel/Pool Size	Activation	Output Shape	Parameters
Conv1D	(15)	ReLU	(106, 64)	1020
MaxPooling1D	(2)		(53, 64)	0
Conv1D	(15)	ReLU	(39, 64)	61,504
Conv1D	(15)	ReLU	(25, 64)	61,504
MaxPooling1D	(2)		(12, 64)	0
Flatten			(768)	0
Dense		ReLU	(128)	98,432
Dense		ReLU	(64)	8256
Dense		ReLU	(32)	2080
Dropout (0.5)			(32)	0
Dense			(4)	132
Total params:	232,932			
Trainable params:	232,932			
Non-trainable params:	0			

2.2.3. Multi-Layer Perceptron

With the MLP, we varied the amount and shape of the dense layers of the network (Table 7). The resulting network (Table 8) had a flattening input layer and three dense layers with one hundred nodes each. After the dense layers, there is a dropout of 50% and the output layer. All layers except the output layer are ReLU activated and the network optimizer is the Adam optimizer with the same parameters as CNN. The loss score was MSE.

Table 7. Grid search parameters for MLP. The best parameters are bolded.

Parameter	Values
Dense layer shapes	(128, 64, 32), (64,32), (100,100), (200,200), **(100,100,100)**

Table 8. The used multi-layer perceptron. The best parameters from Table 7 were used.

Layer	Activation	Output Shape	Parameters
Flatten		(120)	0
Dense	ReLU	(100)	12,100
Dense	ReLU	(100)	10,100
Dense	ReLU	(100)	10,100
Dropout (0.5)		(100)	0
Dense		(4)	404
Total params:	32,704		
Trainable params:	32,704		
Non-trainable params:	0		

2.2.4. Lasso

Least absolute shrinkage and selection operator (More commonly known as its acronyme: Lasso) was chosen because it works well with data where only some of the features are important [24], as is the case with our data. Mathematically, Lasso is a linear model with an added regularization term, and its objective function to minimize is

$$\min \frac{1}{2n} ||X\omega - y||_2^2 + \alpha\, ||\omega||_1,$$

where α is the penalty term, ω the fitted model, n the number of samples in the training data, X the input training data, and y the target variables. In practice, we used the MultiClassLasso algorithm from scikit-learn [19], which applies Lasso regression for all target variables at the same time.

In Lasso grid search training, we varied the α, the stopping tolerance of the algorithm, and the selection method (Table 9). In the resulting estimator $\alpha = 1$, tolerance $= 10^{-4}$, and the selection method was cyclic.

Table 9. Grid search parameters for Lasso. The best parameters are bolded.

Parameter	Values
α	$10^{-4}, 10^{-3} \ldots \mathbf{10^{0}} \ldots 10^{3}$
Tolerance	$10^{-7}, 10^{-6} \ldots \mathbf{10^{-4}} \ldots 10^{-1}$
Selection	**cyclic**, random

2.2.5. Linear Support Vector Regressor

In binary classification tasks, linear support vector regressor (LSVR) means simply fitting a line or hyperplane in the space where the sum of minimum distances from each class to line is maximal. In regression, the goal is to fit a line that has the most points within a predetermined distance from the line [23].

The implementation we used was LinearSVR from scikit-learn package. It was transformed to multi-class regression by a MultiClassRegressor algorithm in scikit-learn.

With grid search (Table 10) for LSVR, we varied the loss method, the regularization term C, and whether or not the algorithm is solving primal or dual optimization problem.

Table 10. Grid search parameters for linear support vector machine regressor. The best parameters are bolded.

Parameter	Values
Loss	**epsilon insensitive**, squared epsilon insensitive
C	**1**, 10, 100, 1000
Dual	**True**, False

2.2.6. Stochastic Gradient Descent Regressor

Stochastic gradient descent (SGD) is a gradient minimizing algorithm. It takes a random initial value of the input training data and tries to adjust the initial value to a point where the gradient of the objective function is zero. The adjustment is done one random observation at a time. The norm used in optimization is $L2$-norm, as opposed to $L1$-norm used in Lasso regressor [22].

The implementation we used is SGDRegressor from scikit-learn. It was transformed to multi-class regression by MultiOutputRegressor method in scikit-learn.

In grid search, we varied the regularization term α, loss function, penalty function, and the learning rate method (Table 11). In the results and discussion, this method is referred to as a stochastic gradient descent regressor (SGDR).

Table 11. Grid search parameters for linear stochastic gradient descent regressor. The best parameters are bolded.

Parameter	Values
Alpha	$10^{-7}, 10^{-6} \ldots \mathbf{10^{-4}} \ldots 10^{-1}$
Loss	**Squared loss**, Huber, Epsilon insensitive
Penalty	**l2**, l1, elastic net
Learning rate	Constant, Optimal, **Inverse scaling**

2.3. *Accuracy Evaluation Metrics*

The following metrics were calculated from the regression results with the simulated testing data:

- Correlation coefficient (r),
- Root mean squared error (RMSE), also known as standard estimate of error (SEE),
- Mean squared error (MSE),
- Mean absolute error (MAE), and
- Saliency maps [36] were calculated only for the CNN and MLP regressors.

The correlation coefficient r was calculated to see the connection between the prediction and true target values. It is covariance (cov) of the predictions ($Y^{pred.}$) and true targets (Y^{true}) normalized by standard deviations (σ) for both groups:

$$r = \frac{\text{cov}\left(Y^{\text{pred.}}, Y^{\text{true}}\right)}{\sigma_{Y^{\text{pred.}}} \sigma_{Y^{\text{true}}}}. \tag{8}$$

RMSE, MSE, and MAE were calculated to capture the average errors in the predictions. They are calculated as follows:

$$MSE = \frac{1}{n} \sum_{k=1}^{n} (Y_k^{\text{pred.}} - Y_k^{\text{true}})^2, \tag{9}$$

$$RMSE = \sqrt{MSE}, \tag{10}$$

$$MAE = \frac{1}{n} \sum_{k=1}^{n} \left| Y_k^{\text{pred.}} - Y_k^{\text{true}} \right|. \tag{11}$$

The saliency map shows how much each place in the map contribute to the prediction. In other words, the map indicates to which locations in the input space the output is most sensitive. The saliency map is described in detail in [36]. The saliency maps are useful in determining the most useful areas of the training data. For example, if there is need to reduce the size of the training data, one can drop the features that show lower values in the saliency map. The saliency maps were calculated using a keras-vis package version 0.4.1 [37].

3. Results

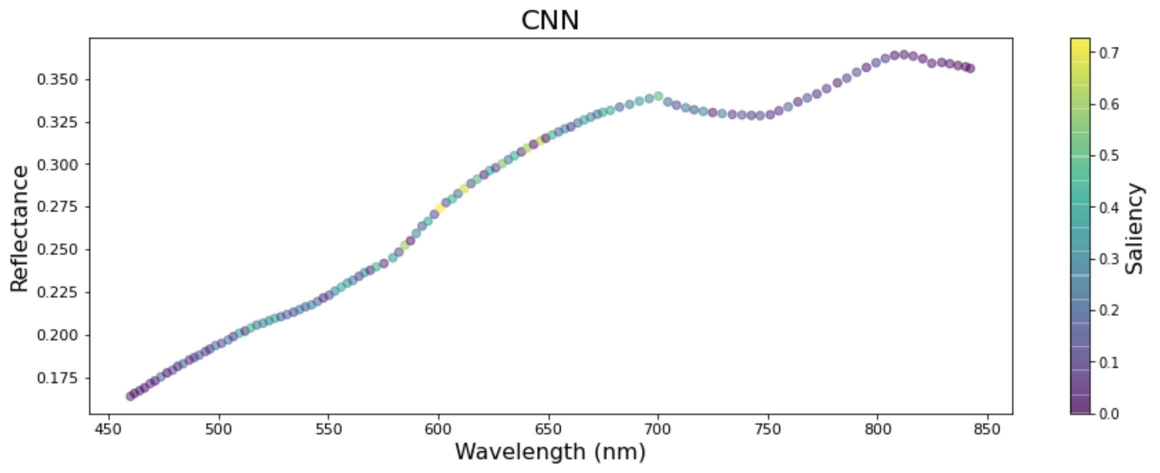
The six inversions had variable success with the simulated data (Figures 3 and 4 and Table 12). The CNN and CNNV2 experiments (Table 12) performed best with correlation coefficients being the highest between 0.93 and 0.96 (mean 0.95) and their RMSE, MSE, and MAE values being lowest 0.09–0.12 (mean 0.11) (RMSE), 0.01–0.02 (mean 0.01) (MSE), and 0.08–0.09 (mean 0.09) (MAE). The performance of the MLP was also good. It outscored CNN in some regression goals by some metrics and was a little poorer in others. Correlation coefficients were between 0.79 and 0.95 (mean 0.91), RMSEs 0.09–0.18 (mean 0.12), MSEs 0.01–0.03 (mean 0.02), and MAEs 0.07–0.14 (mean 0.09). Based on visual interpretation of Figure 4, the second experiment of CNN regressor is the best of the neural network regressors, as the correlation seems to be strongest there.

Table 12. Performance metrics for the trained inversion models on the test set.

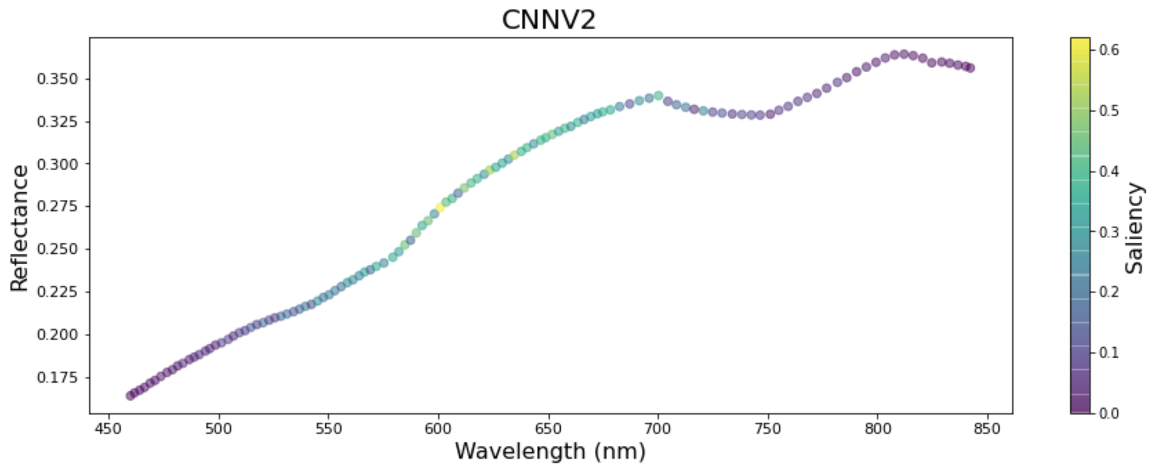
	Melanosome Volume Fraction	Hemoglobin Volume Fraction	Epidermis Thickness	Dermis Thickness
Coefficient of Correlation				
CNN	0.96	0.93	0.96	0.96
CNNV2	0.96	0.93	0.96	0.96
MLP	0.95	0.79	0.94	0.95
LASSO	0.78	0.56	0.62	0.88
LSVR	0.64	0.59	0.63	0.68
SGDR	0.78	0.55	0.62	0.88
Root Mean Squared Error				
CNN	0.10	0.12	0.09	0.10
CNNV2	0.10	0.12	0.11	0.10
MLP	0.09	0.18	0.11	0.09
LASSO	0.18	0.24	0.23	0.14
LSVR	0.23	0.24	0.23	0.22
SGDR	0.18	0.24	0.23	0.14
Mean Squared Error				
CNN	0.01	0.02	0.01	0.01
CNNV2	0.01	0.02	0.01	0.01
MLP	0.01	0.03	0.01	0.01
LASSO	0.03	0.06	0.05	0.02
LSVR	0.05	0.06	0.05	0.05
SGDR	0.03	0.06	0.05	0.02
Mean Absolute Error				
CNN	0.08	0.09	0.08	0.08
CNNV2	0.08	0.10	0.09	0.08
MLP	0.07	0.14	0.08	0.07
LASSO	0.15	0.20	0.19	0.11
LSVR	0.18	0.19	0.18	0.16
SGDR	0.15	0.20	0.18	0.11

In the saliency maps of the CNN and MLP regressors (Figure 3), we see that the CNN experiments had similar areas of interest. The interval between 550 nm and 700 nm seems to be most important in predicting the target variables. In the MLP experiment, the same interval seems to be useful, but additionally it highlights some areas in the ends of the spectrum.

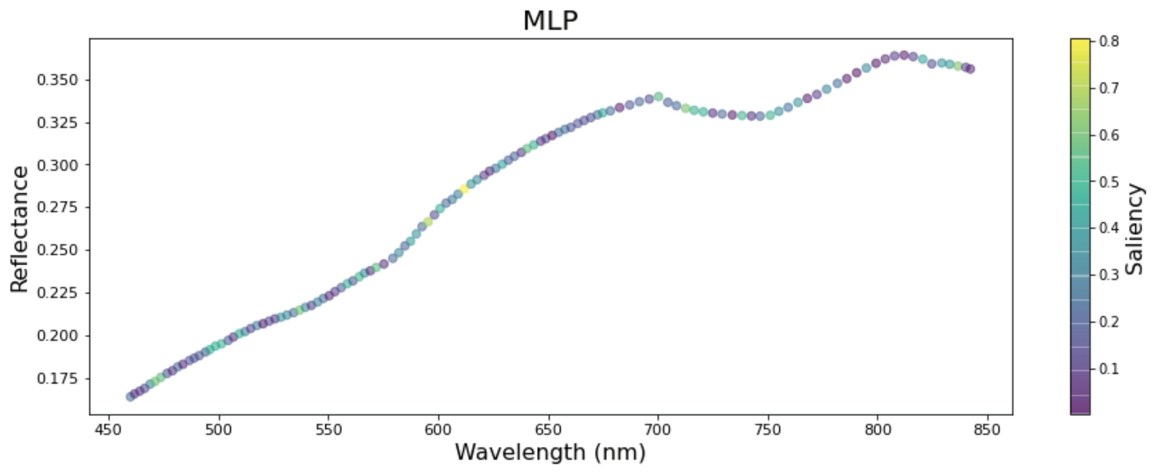
Lasso and stochastic gradient descent regressors showed similar results. As Figure 4 shows, the correlation is virtually identical, and the points scatter to nearly the same place. In fact, only looking at individual points, one can observe some differences between the lasso and SGD regressors. Lasso and SGDR had correlation coefficients between 0.55 and 0.88 (means 0.71 and 0.71, respectively), RMSEs between 0.14 and 0.24 (means 0.20 and 0.20), MSEs between 0.02 and 0.06 (means 0.04 and 0.04), and MAEs between 0.11 and 0.20 (means 0.16 and 0.16).



(**a**)

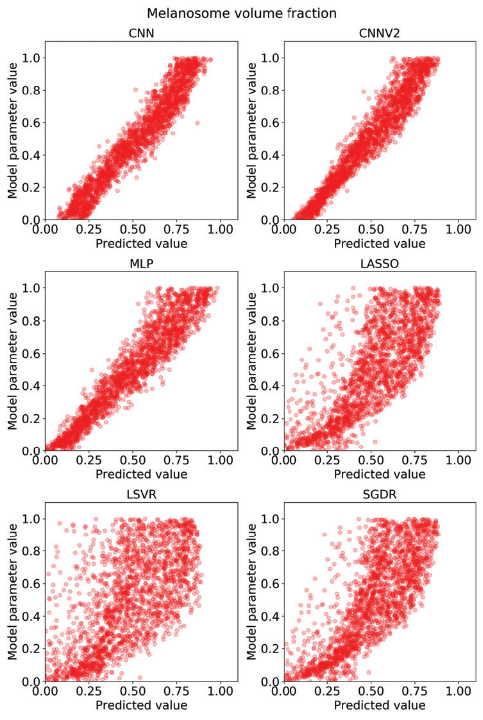


(**b**)

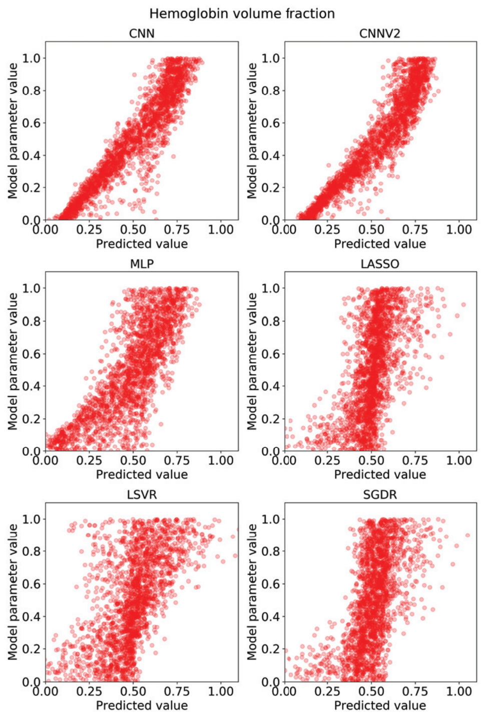


(**c**)

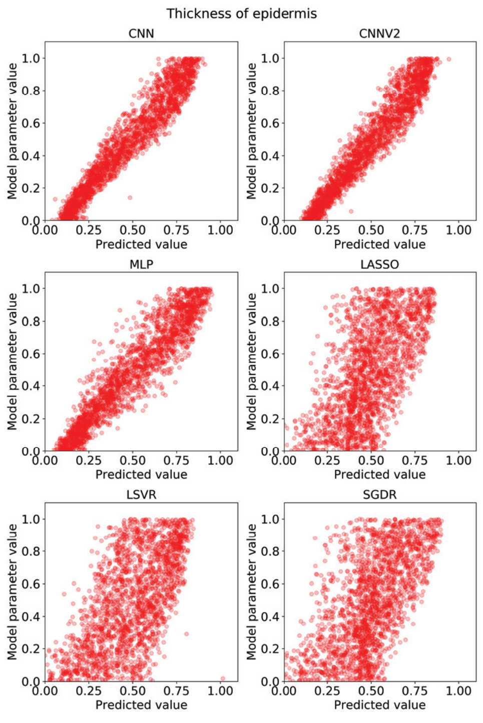
Figure 3. The saliency maps of the neural network regressors. (**a**) the first CNN model; (**b**) the second CNN model; (**c**) the MLP model.



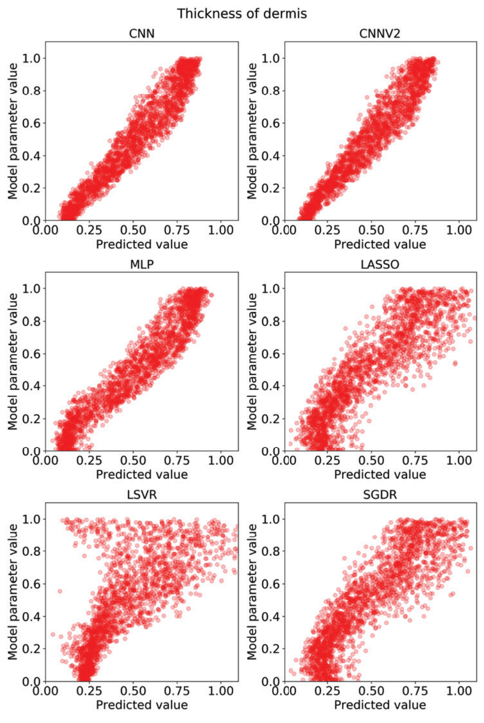
(**a**)



(**b**)



(**c**)



(**d**)

Figure 4. The scattering images of predicted values (x-axis) compared to original values (y-axis) by different regression targets: (**a**) melanosome volume fraction; (**b**) hemoglobin volume fraction; (**c**) thickness of epidermis; (**d**) thickness of dermis.

The worst predictor was the linear support vector regressor. It is worst by all metrics by average, but, in hemoglobin volume fraction and epidermis thickness predictions, it seems to be marginally better than Lasso and SGDR. The correlation coefficients were between 0.59 and 0.68 (mean 0.64), RMSEs between 0.22 and 0.24 (mean 0.23), MSEs between 0.05 and 0.06 (mean 0.05), and MAEs between 0.16 and 0.19 (mean 0.18).

Overall, the least accurate predictions were for hemoglobin volume fraction and the models were most accurate for dermis thickness prediction. Melanosome volume fraction was relatively easily learned by all regressors, while, in epidermis thickness, there is a clear divide between group of CNN and MLP regressors and other regressors. The neural network models provide accurate responses which seems almost impossible for the other types of models.

Regarding the test with the measured hyperspectral data in Figure 5, the details of the lesions are most easily visible in the CNN experiments. In MLP, lasso, and LSVR experiments, the epidermis thickness is poorly visible due to noise. Looking at Figure 5, the fourth best model seems to be the SGDR, as the details of the lesions are somewhat visible.

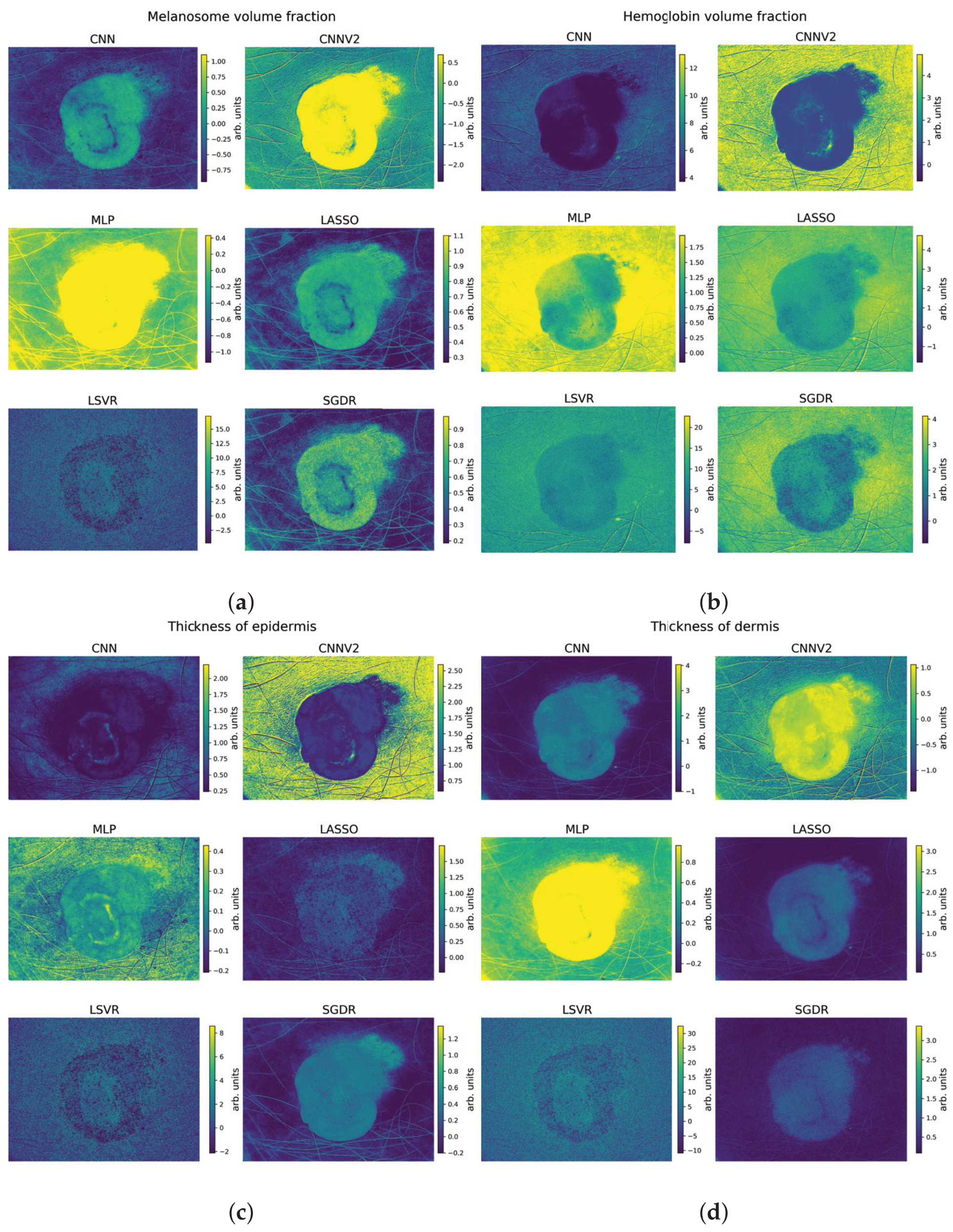


Figure 5. Comparison of the inversion algorithms used with measured hyperspectral data. Each subfigure represents a target parameter. The colorbar units are arbitrary. (**a**) melanosome volume fraction; (**b**) hemoglobin volume fraction; (**c**) thickness of epidermis; (**d**) thickness of dermis.

4. Discussion

The purpose of this study was to investigate how different machine learning approaches perform in the inversion task of predicting stochastic model parameters from the spectral input data. The present results suggests that the most accurate models can be obtained by the CNN method. It is likely that performance of all the trained models can be improved by further parameter tuning or increasing the amount of data, but it is unlikely that one model would have improvement rate vastly different from others. The results were expected as the CNN has been shown to be superior in signal processing tasks, which are essentially analogous to one-dimensional, spectral data used in this study.

The measured correlation coefficients show that the CNN is as close to perfection as can be reasonably expected without overfitting. However, we must remember that the results were achieved with simulated data, and the testing with measured biophysical parameters are yet to be carried out. The error metrics show similarly good signs, as the MAE was less than 10% of the range (all true target parameters were normalized to between zero and one), and the RMSE is less than 2%. If the errors are similar in further testing with measured data, then the model would be ready for in-vivo clinical testing.

Based on the results regarding the measured hyperspectral data, it seems that the used stochastic skin model represents the optical properties of the skin quite well. The CNN predictors in Figure 5 show the same general pattern as Figure 6, and at least the melanosome volume fraction is predicted to be higher in the area of the lesion, which is clearly correct based on Figure 6. Of the absolute correctness of the model, these experiments do not provide information, as we did not have measurements of the skin biophysical parameters to compare to Figure 5.

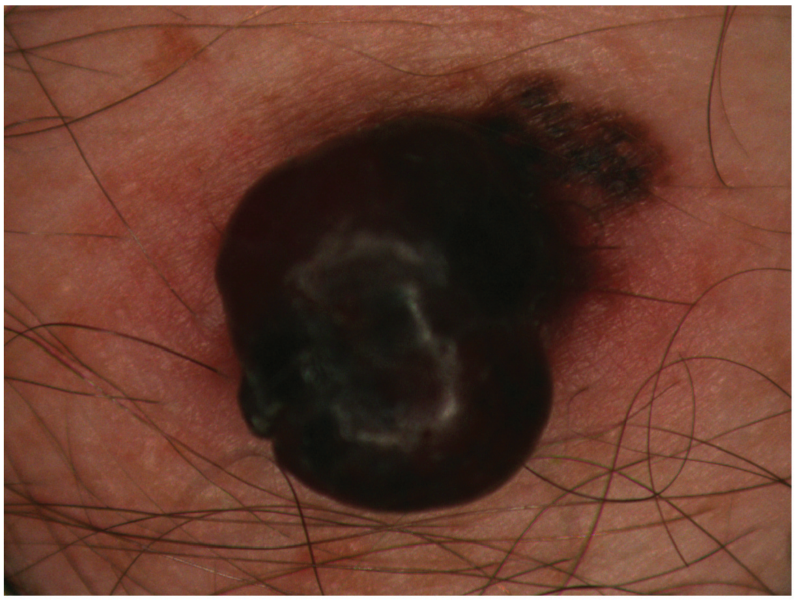


Figure 6. RGB-representation of the measured lesion.

Compared to the previous research by Vyas et al. [12,13], our correlation results are slightly inferior. They found a correlation coefficient between measured thickness and estimated thickness to be nearly perfect, while our best correlation is 0.96. However, our goal was different: we set out to find the best inversion algorithm, instead of best inversion. In future studies, we hope we can build a more robust system of parameter retrieval with a perfected CNN model.

In future research, the stochastic model and CNN combination should be thoroughly tested with measured hyperspectral data and measured biophysical parameters. The goals of the testing could include finding the best CNN estimator by optimizing the absolute and relative accuracy of the predictions with respect to the model. Our ultimate goal is to obtain a comprehensive model for predicting skin optical properties and its inverse function for predicting skin biophysical parameters.

5. Conclusions

We provided experiments that show that a convolutional neural network is a good option in skin optical model inversion and skin physical parameter retrieval. The results indicate that the most meaningful parameters of the used stochastic model were predicted accurately by the best inverted model. We also tested the inverted models while measuring the hyperspectral data, and it showed promising results to be tested in further research. It seems that the inverted model may work well with the measured data, at least when one is looking at proportional differences of skin areas instead of absolute values. Our research also suggests that the capacity of the traditional non-neural machine learning models is insufficient for accurate modeling of the hyperspectral data.

Author Contributions: Conceptualization, L.A. and I.P.; methodology, L.A.; software, L.A.; validation, L.A., I.P., and S.Ä.; formal analysis, L.A.; investigation, L.A.; resources, I.P.; data curation, L.A., I.P.; writing—original draft preparation, L.A.; writing—review and editing, L.A., I.P., and S.Ä.; visualization, L.A.; supervision, I.P. and S.Ä.; project administration, I.P.; funding acquisition, I.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Academy of Finland Grant No. 314519.

Acknowledgments: We acknowledge grants of computer capacity from the Finnish Grid and Cloud Infrastructure (persistent identifier urn:nbn:fi:research-infras-2016072533).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CNN	Convolutional neural network
k-NN	k-nearest neighbor
KM	Kubelka–Munk model
Lasso	Least absolute shrinkage and selection operator
LSVR	Linear support vector regressor
MAE	Mean average error
ML	Machine learning
MLP	Multi-layer perceptron
PCA	Principal component analysis
r	Coefficient of correlation
r^2	Coefficient of determination
RF	Random forest
(R)MSE	(Root) mean squared error
SEE	Standard error of estimation
SGD(R)	Stochastic gradient descent (regressor)
SLOP	Stochastic model for leaf optical properties
SM	Stochastic model
SVM	Support vector machine

References

1. Ferlay, J.; Colombet, M.; Soerjomataram, I.; Mathers, C.; Parkin, D.; Piñeros, M.; Znaor, A.; Bray, F. Estimating the Global Cancer Incidence and Mortality in 2018: GLOBOCAN Sources and Methods. *Int. J. Cancer* **2019**, *144*, 1941–1953. [CrossRef] [PubMed]
2. Le, H.V.; Le, C.H.H.; Le, P.H.U.; Truong, C.T.L. Incidence and Trends of Skin Cancer in the United States, 1999–2016. *J. Clin. Oncol.* **2020**. [CrossRef]

3. Neittaanmäki-Perttu, N.; Grönroos, M.; Jeskanen, L.; Pölönen, I.; Ranki, A.; Saksela, O.; Snellman, E. Delineating Margins of Lentigo Maligna Using a Hyperspectral Imaging System. *Acta Derm.-Venereol.* **2015**, *95*, 549–552. [CrossRef]
4. Annala, L.; Honkavaara, E.; Tuominen, S.; Pölönen, I. Chlorophyll Concentration Retrieval by Training Convolutional Neural Network for Stochastic Model of Leaf Optical Properties (SLOP) Inversion. *Remote Sens.* **2020**, *12*, 283. [CrossRef]
5. Annala, L.; Pölönen, I. Kubelka-Munk Model and Stochastic Model Comparison in Skin Physical Parameter Retrieval. In *Computational Sciences and Artificial Intelligence in Industry—New Digital Technologies for Solving Future Societal and Economical Challenges*; Springer: Berlin/Heidelberg, Germany, 2020; in press.
6. Maier, S.W.; Lüdeker, W.; Günther, K.P. SLOP: A Revised Version of the Stochastic Model for Leaf Optical Properties. *Remote Sens. Environ.* **1999**, *68*, 273–280. [CrossRef]
7. Jolivot, R.; Benezeth, Y.; Marzani, F. Skin Parameter Map Retrieval from a Dedicated Multispectral Imaging System Applied to Dermatology/Cosmetology. *J. Biomed. Imaging* **2013**, *2013*, 978289. [CrossRef]
8. Yeh, Y.H.F.; Chung, W.C.; Liao, J.Y.; Chung, C.L.; Kuo, Y.F.; Lin, T.T. A Comparison of Machine Learning Methods on Hyperspectral Plant Disease Assessments. *IFAC Proc. Vol.* **2013**, *46*, 361–365. [CrossRef]
9. Sanz, J.A.; Fernandes, A.M.; Barrenechea, E.; Silva, S.; Santos, V.; Gonçalves, N.; Paternain, D.; Jurio, A.; Melo-Pinto, P. Lamb Muscle Discrimination Using Hyperspectral Imaging: Comparison of Various Machine Learning Algorithms. *J. Food Eng.* **2016**, *174*, 92–100. [CrossRef]
10. Gewali, U.B.; Monteiro, S.T.; Saber, E. Machine Learning Based Hyperspectral Image Analysis: A Survey. *arXiv* **2019**, arXiv:1802.08701.
11. Liang, L.; Qin, Z.; Zhao, S.; Di, L.; Zhang, C.; Deng, M.; Lin, H.; Zhang, L.; Wang, L.; Liu, Z. Estimating Crop Chlorophyll Content with Hyperspectral Vegetation Indices and the Hybrid Inversion Method. *Int. J. Remote Sens.* **2016**, *37*, 2923–2949. [CrossRef]
12. Vyas, S.; Banerjee, A.; Burlina, P. Estimating Physiological Skin Parameters from Hyperspectral Signatures. *J. Biomed. Opt.* **2013**, *18*, 057008. [CrossRef] [PubMed]
13. Vyas, S.; Meyerle, J.; Burlina, P. Non-Invasive Estimation of Skin Thickness from Hyperspectral Imaging and Validation Using Echography. *Comput. Biol. Med.* **2015**, *57*, 173–181. [CrossRef] [PubMed]
14. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*; ACM: New York, NY, USA, 2012; pp. 1097–1105.
15. Karpathy, A.; Toderici, G.; Shetty, S.; Leung, T.; Sukthankar, R.; Fei-Fei, L. Large-Scale Video Classification with Convolutional Neural Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 1725–1732.
16. LeCun, Y.; Bengio, Y. Convolutional Networks for Images, Speech, and Time-Series. In *The Handbook of Brain Theory and Neural Networks*; MIT Press: Cambridge, MA, USA, 1995.
17. Salamon, J.; Bello, J.P. Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification. *IEEE Signal Process. Lett.* **2017**, *24*, 279–283. [CrossRef]
18. Jacques, S.L. Optical Properties of Biological Tissues: A Review. *Phys. Med. Biol.* **2013**, *58*, R37. [CrossRef]
19. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-Learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830. [CrossRef]
20. Pölönen, I.; Rahkonen, S.; Annala, L.; Neittaanmäki, N. Convolutional Neural Networks in Skin Cancer Detection Using Spatial and Spectral Domain. In Proceedings of the SPIE BiOS, San Francisco, CA, USA, 2–7 February 2019; Volume 10851, p. 108510B.
21. Specht, D.F. A General Regression Neural Network. *IEEE Trans. Neural Netw.* **1991**, *2*, 568–576. [CrossRef]
22. Bottou, L. Large-Scale Machine Learning with Stochastic Gradient Descent. In Proceedings of the COMPSTAT'2010, Paris France, 22–27 August 2010; Springer: Berlin/Heidelberg, Germany, 2010; pp. 177–186.
23. Smola, A.J.; Schölkopf, B. A Tutorial on Support Vector Regression. *Stat. Comput.* **2004**, *14*, 199–222. [CrossRef]
24. Friedman, J.; Hastie, T.; Tibshirani, R. Regularization Paths for Generalized Linear Models via Coordinate Descent. *J. Stat. Softw.* **2010**, *33*, 1. [CrossRef]

25. Almeida, J.S. Predictive Non-Linear Modeling of Complex Data by Artificial Neural Networks. *Curr. Opin. Biotechnol.* **2002**, *13*, 72–76. [CrossRef]
26. Sarle, W.S. Neural Networks and Statistical Models. In Proceedings of the 19th Annual SAS Users Group International Conference, Dallas, TX, USA, 10–13 April 1994.
27. Hastie, T.; Tibshirani, R.; Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2009.
28. Burges, C.J. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Min. Knowl. Discov.* **1998**, *2*, 121–167. [CrossRef]
29. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. Tensorflow: A System for Large-Scale Machine Learning. In Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), Savannah, GA, USA, 2–4 November 2016; pp. 265–283.
30. Touvron, H.; Vedaldi, A.; Douze, M.; Jégou, H. Fixing the Train-Test Resolution Discrepancy: FixEfficientNet. *arXiv* **2020**, arXiv:2003.08237.
31. Xie, Q.; Luong, M.T.; Hovy, E.; Le, Q.V. Self-Training with Noisy Student Improves Imagenet Classification. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 10687–10698.
32. Scherer, D.; Müller, A.; Behnke, S. Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition. In Proceedings of the International Conference on Artificial Neural Networks, Thessaloniki, Greece, 15–18 September 2010; Springer: Berlin/Heidelberg, Germany, 2010; pp. 92–101.
33. Nair, V.; Hinton, G.E. Rectified Linear Units Improve Restricted Boltzmann Machines. In Proceedings of the 27th International Conference on Machine Learning (ICML-10), Haifa, Israel, 21–24 June 2010; pp. 807–814.
34. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
35. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.
36. Simonyan, K.; Vedaldi, A.; Zisserman, A. Deep inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *arXiv* **2013**, arXiv:1312.6034.
37. Kotikalapudi, R. Keras-Vis Version 0.4.1. 2017. Available online: https://github.com/raghakot/keras-vis (accessed on 24 September 2020).

PVII

GENERATING HYPERSPECTRAL SKIN CANCER IMAGERY USING GENERATIVE ADVERSARIAL NEURAL NETWORK

by

Leevi Annala, Noora Neittaanmäki, John Paoli, Oscar Zaar and Ilkka Pölönen
2020

2020 42st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), IEEE

https://doi.org/10.1109/EMBC44109.2020.9176292

Generating Hyperspectral Skin Cancer Imagery using Generative Adversarial Neural Network

Leevi Annala[1,*], Noora Neittaanmäki[2], John Paoli[3], Oscar Zaar[3] and Ilkka Pölönen[1]

Abstract— **In this study we develop a proof of concept of using generative adversarial neural networks in hyperspectral skin cancer imagery production. Generative adversarial neural network is a neural network, where two neural network compete. The generator tries to produce data that is similar to the measured data, and the discriminator tries to classify correctly classify the data as fake or real. This is a reinforcement learning model, where both models get reinforcement based on their performance. In the training of the discriminator we use data measured from skin cancer patients. The aim for the study is to develop a generator for augmenting hyperspectral skin cancer imagery.**

I. INTRODUCTION

In the northern countries, melanoma incidence is increasing and the increase is predicted to continue [1]. Furthermore the clinical accuracy of the skin lesion classification is poor, leading to costly operations that could be avoided [2]. There is a need for reliable non-invasive method for clinical use for skin lesion diagnostics.

Hyperspectral imaging is one such potential tool, and it has been previously used successfully in classification of the lesions [3], [4]. While in both articles, the results were promising both of the datasets used were relatively small. Furthermore, if the models would be used in marginally different situation, either a lengthy data gathering process would have to take place or the existing data could be reused as is or through a data transformation/augmentation process. This transformation or augmentation could be achieved by modeling mathematically the interaction of the light and the matter and the structure of the matter, but the uncertainties are high and the computations complex, downgrading the usability.

Another way to transform or augment data is generative adversarial neural network (GAN) [5]. The basic idea of the GAN is that the generator network and the discriminator network compete in a game. Discriminator networks task is to determine whether or not the data given to it represents the training data or not. The task of the generator network is to produce data similar to the training data to trick the discriminator. The only input the generator receives during this time is yes/no confirmations from the discriminator, while the discriminator is trained with the full training data set.

GANs have been previously used in the field of medical imaging [6]. The use cases include denoising [7], image reconstruction [8], artifact removal [9], superresolution [10], image synthesis [11], classification [12] and segmentation [13]. Despite producing promising results, the field of using GANs in the medical field is still young, and there is lots of room for experimenting [6].

In the more specific field of hyperspectral (medical) imaging, GANs have been used in two different ways [14], [15]. In [14], hyperspectral images of lung samples were automatically stained using trained conditional GAN (cGAN) [16]. In [15], Wasserstein distance was used as basis for modifying GAN and the resulting networks were verified using data sets from different fields of study. There are also various different approaches to using GANs in hyperspectral classification problems (for example [17], [18]).

The objective of this study is to provide the reader with a minimal working example of a proof-of-concept level data augmentation tool for hyperspectral skin cancer image production.

II. MATERIALS AND METHODS

A. *Generative Adversarial Neural Network Architecture*

The used generative adversarial neural network (GAN) was one implementation of deep convolutional GAN (DC-GAN) [19]. The used generator architecture (Table I) consisted of input layer and three transposed convolution blocks. The first two convolutional blocks had two three dimensional convolutional layers, first of which took into account the spectral domain, and the second the spatial. The last block had one convolutional layer, that worked in all dimensions. Next in the blocks were batch normalisation [20]. The last two blocks utilized striding, which in the case of transposed convolution increases the output size which in the end will be $40 \times 40 \times 40$. The activation function used in the blocks was leakyReLU [21] and the loss function is cross entropy between a data cube full of ones and the generated data cube. Both the generator and discriminator used the Adam-optimizer [22], with a learning rate of 10^{-4}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-7}$.

The discriminator architecture (Table II) consisted of two blocks that contained a 3-dimensional convolution and dropout of 0.3 [23] and an output block that contained flattening and output layers. The activation function used

* Corresponding author

This research was partly funded by Academy of Finland (grant: 314519).

[1] Leevi Annala and Ilkka Pölönen are with the Faculty of Information Technology, University of Jyväskylä, Jyväskylä, Finland. leevi.a.annala@jyu.fi

[2] Noora Neittaanmäki is with Departments of Pathology and Dermatology, Institutes of Biomedicine and Clinical Sciences, Sahlgrenska Academy, University of Gothenburg, Gothenburg, Sweden.

[3]John Paoli and Oscar Zaar are with the Department of Dermatology and Venereology, Institute of Clinical Sciences, Sahlgrenska Academy, University of Gothenburg, Gothenburg, Sweden.

TABLE I

THE ARCHITECTURE OF THE USED GENERATOR NETWORK.

Layer	**Kernel/Pool Size, Strides and Activation**	**Filters/ Units**
Dense	LeakyReLU	100
Batch Normalization		
Conv3DTranspose	(5,1,1), (1,1,1), LeakyReLU	128
Batch Normalization		
Conv3DTranspose	(1,5,5), (1,1,1), LeakyReLU	128
Batch Normalization		
Conv3DTranspose	(5,1,1), (2,1,1), LeakyReLU	64
Batch Normalization		
Conv3DTranspose	(1,5,5), (1,2,2), LeakyReLU	64
Batch Normalization		
Conv3DTranspose	(5,5,5), (2,2,2), LeakyReLU	1
Optimiser:	Adam	
Loss:	Cross entropy	

in the convolutional layers was leakyReLU and the loss function was a sum of generator loss and cross-entropy between data cube full of ones and training data.

TABLE II

THE ARCHITECTURE OF THE USED DISCRIMINATOR NETWORK.

Layer	**Kernel/Pool Size, Strides and Activation**	**Filters/ Units**
Conv3D	(5,5,5), (2,2,2), LeakyReLU	64
Dropout (0.3)		
Conv3D	(5,5,5), (2,2,2), LeakyReLU	128
Dropout (0.3)		
Flatten		
Dense		1
Optimiser:	Adam	
Loss:	Cross entropy	

B. Data Capturing and Preprosessing

Raw hyperspectral measurement data was captured at Sahlgrenska University Hospital during 2018 – 2019, using Revenio Prototype 2016 hyperspectral imager with spatial resolution of 1920×1200 and spectral resolution of 120. The camera uses Fabry-Pérot interferometer technology [24]. All patients have volunteered to participate in the study. The study protocol has followed the Declaration of Helsinki and it was approved by the local ethics committee. The captured dataset consists of 316 hyperspectral images of skin. The type and existence of lesion vary; a portion of the images contain no lesion.

The captured raw measurement data cubes were then processed to radiance using a fpipy Python package [25] and downsampled to $40 \times 40 \times 40$ data cubes by averaging. Dataset size is increased by mirroring images by its three spatial symmetry axes, increasing the amount of images eight-fold to 2528. This data was used in training the generator and discriminator networks to produce similar data and to classify the data as real or fake. Various selected wavelength bands from the training data can be seen in Figure 1 and a typical radiance spectrum from data can be seen in Figure 2.

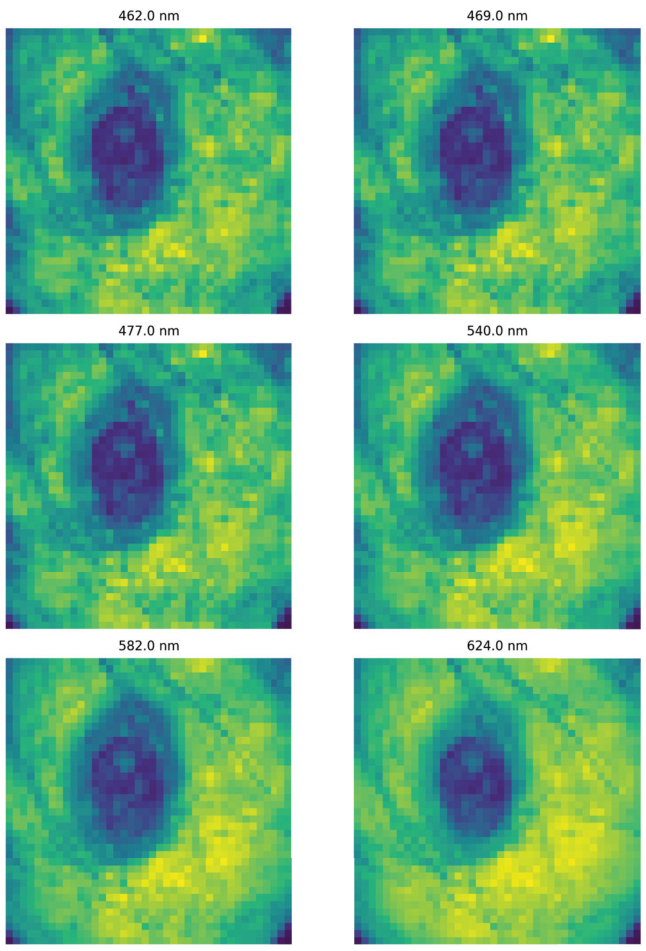


Fig. 1. Selected bands from training data hyperspectral image

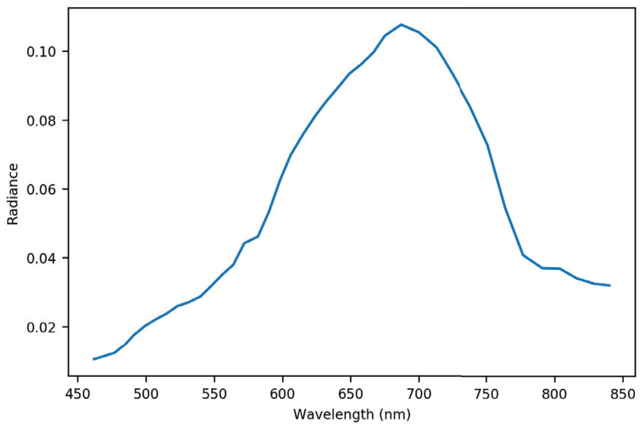


Fig. 2. Typical radiance in the training data

III. Results and Discussion

Generated hyperspectral image (Figure 3) shows positive and negative patterns. All bands, except the first band at 462 nm, look similar to the measured data in texture. They are obviously not random, and the patterns persist from band to band. However, the lesions in the generated data tend to have slight gradual shift, as can be witnessed in the Figure 3 right column. When one looks at the bottom right corner of each of the three images one sees that the lesion moves significantly to the right, in contrast to Figure 1, where the lesion becomes a little less clear, but does not move. The first band (Figure 3, 462 nm) shows some artefact.

Both of these problems likely have their roots in the convolution layers. There is no data before the first layer to convolute with therefore the training affects it less. The shift of the lesions/features may have a root in the depth of the convolution. The convolution that addresses the layers 32 – 36 has little information on the layers 1 – 5.

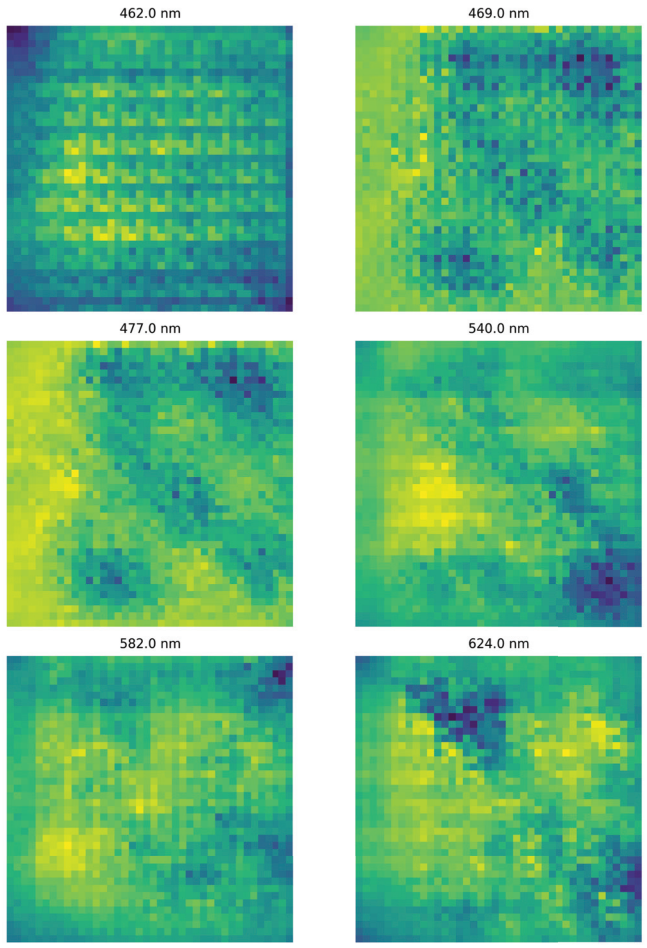


Fig. 3. Selected bands from generated hyperspectral image

The typical radiance spectrum (Figure 4) from generated data looks similar to the typical radiance of the measured data (Figure 2), except for the artefact that causes the spectrum to fluctuate. This means that in the spectral domain the orders of magnitude are correct and the wave bands are correctly generated in that sense.

Further research of the use of GANs in the field of hyperspectral imaging are needed. The used DCGAN could be made deeper or wider, and different conditional GANs [16] could be experimented with. The future training should aim for using larger hyperspectral images to increase the accuracy of the training data. A U-net network could be tried [26]. A larger dataset should also be gathered to increase the likelihood of succesful training. When the GAN produced data is judged accurate enough repression of measured data, it can be used for example in the training of other machine learning algorithms for tasks such as model inversion (such as in [27]) and hyperspectral image classification (example in the medical field [3]). There is also need to compare the method for more traditional data augmentation methods.

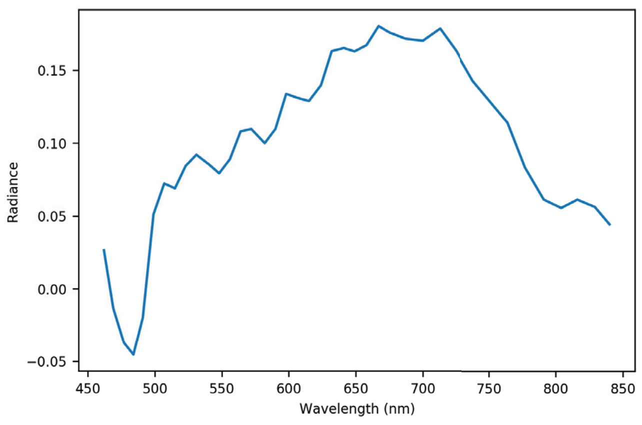


Fig. 4. Typical radiance in the generated data

IV. Conclusion

This study presented a proof of concept for producing augmented hyperspectral data using generative adversarial neural network (GAN). The used generator network was not fine tuned to it's full potential and in all likelihood there is much room for improvement in the network developement as well, as the first passable iteration was presented.

An aim for further research could be to utilize data produced by generative adversarial neural networks in further machine learning applications for skin cancer diagnostics by hyperspectral data to see how useful the generated data is. In further research there is a call for using GAN augmented data in machine learning applications to improve the quality of the algorithm training.

References

[1] B. Møller, H. Weedon-Fekjær, T. Hakulinen, L. Tryggvadottir, H. Storm, M. Talbäck, and T. Haldorsen, "Prediction of cancer incidence in the Nordic countries up to the year 2020," *European journal of cancer prevention : the official journal of the European Cancer Prevention Organisation (ECP)*, vol. 11 Suppl 1, pp. S1–96, 2002.

[2] C. F. Heal, B. A. Raasch, P. Buettner, and D. Weedon, "Accuracy of clinical diagnosis of skin lesions," *British Journal of Dermatology*, vol. 159, no. 3, pp. 661–668, 2008.

[3] I. Pölönen, S. Rahkonen, L. Annala, and N. Neittaanmäki, "Convolutional neural networks in skin cancer detection using spatial and spectral domain," in *Photonics in Dermatology and Plastic Surgery 2019*, vol. 10851. International Society for Optics and Photonics, 2019, p. 108510B.

[4] U.-O. Dorj, K.-K. Lee, J.-Y. Choi, and M. Lee, "The skin cancer classification using deep convolutional neural network," *Multimedia Tools and Applications*, vol. 77, no. 8, pp. 9909–9924, 2018.

[5] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *arXiv preprint arXiv:1406.2661*, 2014.

[6] X. Yi, E. Walia, and P. Babyn, "Generative adversarial network in medical imaging: A review," *Medical Image Analysis*, vol. 58, p. 101552, Dec. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1361841518308430

[7] J. M. Wolterink, T. Leiner, M. A. Viergever, and I. Išgum, "Generative adversarial networks for noise reduction in low-dose CT," *IEEE transactions on medical imaging*, vol. 36, no. 12, pp. 2536–2545, 2017.

[8] S. U. H. Dar, M. Yurt, M. Shahdloo, M. E. Ildız, and T. Çukur, "Synergistic reconstruction and synthesis via generative adversarial networks for accelerated multi-contrast mri," *arXiv preprint arXiv:1805.10704*, 2018.

[9] J. Wang, Y. Zhao, J. H. Noble, and B. M. Dawant, "Conditional generative adversarial networks for metal artifact reduction in ct images of the ear," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2018, pp. 3–11.

[10] C. You, G. Li, Y. Zhang, X. Zhang, H. Shan, M. Li, S. Ju, Z. Zhao, Z. Zhang, W. Cong, and others, "CT super-resolution GAN constrained by the identical, residual, and cycle learning ensemble (GAN-CIRCLE)," *IEEE Transactions on Medical Imaging*, vol. 39, no. 1, pp. 188–203, 2019.

[11] M. J. Chuquicusma, S. Hussein, J. Burt, and U. Bagci, "How to fool radiologists with generative adversarial networks? a visual turing test for lung cancer diagnosis," in *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*. IEEE, 2018, pp. 240–244.

[12] X. Yi, E. Walia, and P. Babyn, "Unsupervised and semi-supervised learning with Categorical Generative Adversarial Networks assisted by Wasserstein distance for dermoscopy image Classification," 2018.

[13] Z. Zhang, L. Yang, and Y. Zheng, "Translating and segmenting multimodal medical volumes with cycle-and shape-consistency generative adversarial network," in *Proceedings of the IEEE conference on computer vision and pattern Recognition*, 2018, pp. 9242–9251.

[14] N. Bayramoglu, M. Kaakinen, L. Eklund, and J. Heikkila, "Towards virtual h&e staining of hyperspectral lung histology images using conditional generative adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2017, pp. 64–71.

[15] M. Zhang, M. Gong, Y. Mao, J. Li, and Y. Wu, "Unsupervised Feature Extraction in Hyperspectral Images Based on Wasserstein Generative Adversarial Network," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 5, pp. 2669–2688, May 2019.

[16] J. Gauthier, "Conditional generative adversarial nets for convolutional face generation," *Class Project for Stanford CS231N: Convolutional Neural Networks for Visual Recognition, Winter semester*, vol. 2014, no. 5, p. 2, 2014.

[17] H. Wang, C. Tao, J. Qi, H. Li, and Y. Tang, "Semi-Supervised Variational Generative Adversarial Networks for Hyperspectral Image Classification," in *IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2019, pp. 9792–9794.

[18] L. Zhu, Y. Chen, P. Ghamisi, and J. A. Benediktsson, "Generative adversarial networks for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 9, pp. 5046–5063, 2018.

[19] A. Radford, L. Metz, and S. Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks," *arXiv:1511.06434 [cs]*, Jan. 2016, arXiv: 1511.06434. [Online]. Available: http://arxiv.org/abs/1511.06434

[20] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[21] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. icml*, vol. 30, 2013, p. 3.

[22] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[23] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html

[24] M. Blomberg, H. Kattelus, and A. Miranto, "Electrically tunable surface micromachined Fabry–Perot interferometer for visible light," *Sensors and Actuators A: Physical*, vol. 162, no. 2, pp. 184–188, 2010.

[25] M. Eskelinen, "Computational methods for hyperspectral imaging using Fabry–Perot interferometers and colour cameras," *JYU dissertations*, 2019.

[26] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[27] L. Annala, E. Honkavaara, S. Tuominen, and I. Pölönen, "Chlorophyll Concentration Retrieval by Training Convolutional Neural Network for Stochastic Model of Leaf Optical Properties (SLOP) Inversion," *Remote Sensing*, vol. 12, no. 2, p. 283, Jan. 2020. [Online]. Available: https://www.mdpi.com/2072-4292/12/2/283